Universidad Cenfotec

Maestría en Ciberseguridad

Documento final de Proyecto de Investigación Aplicada 2

UTILIZACIÓN DE MÉTODOS DISCRETOS Y PROBABILÍSTICOS EN LA IDENTIFICACIÓN DE HIPERVÍNCULOS MALICIOSOS EN SITIOS WEB POR MEDIO DE UNA EXTENSIÓN DE NAVEGADOR

Chaves González Jorge

Diciembre del 2017

Derechos de autor

Dedicatoria

Deseo dedicar el presente trabajo a aquellos amigos y familiares que de una manera u otra a partir de importantes esfuerzos o palabras gentiles me impulsaron a seguir adelante, desde el inicio de esta carrera hasta este momento de culminación. Siempre agradeceré su apoyo.

Agradecimientos

Un agradecimiento especial a mi tutor de tesis Juan Ignacio Zamora por su colaboración, consejo y palabras de motivación a lo largo del desarrollo y presentación de esta investigación. A la decanatura de la Universidad por su colaboración y disposición de ayudarme en temas administrativos durante momentos difíciles. Por último, para aquellas personas que han tenido palabras de apoyo en la culminación de este largo proceso.

San José, 27 de enero de 2018

Señores
Universidad CENFOTEC

Estimados señores:

Por medio de la presente hago constar que mi persona, Henry Rivera Morales, profesional en enseñanza del español y filología española, colegiado bajo el número 036633, he revisado el Trabajo Final de Graduación del estudiante **Jorge Chaves González,** denominado **Utilización de Métodos Discretos y Probabilísticos en la Identificación de Hipervínculos Maliciosos en Sitios Web por Medio de una Extensión de Navegador**, presentado como parte de los requisitos para optar por el grado de Maestría en Ciberseguridad.

He revisado la gramática, puntuación, ortografía y estructuras idiomáticas del documento escrito, y he verificado que los mismos fueran corregidos por el autor.

Agradeciendo su atención,

Lic. Henry Rivera Morales
Nº 036633
Colegio de Licenciados y Profesores

ÍNDICE DE CONTENIDO

1. RESUMEN 1

1. CAPÍTULO I: INTRODUCCIÓN 2

1.1. Generalidades 2

1.2. Antecedentes del problema 2

1.3. Definición y descripción del problema 3

1.4. Justificación 3

1.5. Objetivos del proyecto 4

1.5.1. Objetivo general 4

1.5.2. Objetivos específicos 4

1.6. Viabilidad 4

1.6.1. Punto de vista técnico 4

1.6.2. Punto de vista operativo 5

1.6.3. Punto de vista económico 5

1.7. Alcances y limitaciones del proyecto 6

1.7.1. Alcance 6

1.7.2. Limitaciones 7

1.8. Estado del Arte 8

2. MARCO TEÓRICO 11

3. MARCO METODOLÓGICO 27

3.1. Tipo de Investigación 27

3.2. Alcance Investigativo 27

3.3. Enfoque 27

3.4. Diseño 28
3.5. Instrumentos de Recolección de Datos 28
3.6. Técnicas de Análisis de la Información 28
3.7. Estrategia de Desarrollo de la Propuesta 29
4. DESARROLLO DE LA SOLUCIÓN 32
4.1. Selección y diseño del servicio discreto 32
4.2.1. Identificación de mecanismos disponibles 32
4.2.2. Selección y diseño de servicio discreto 34
4.3. Diseño e implementación del servicio probabilístico 35
4.3.1. Selección del modelo predictivo 35
4.3.2. Diseño de los datos de aprendizaje y entrenamiento 36
4.3.3. Preprocesamiento y transformación de datos. 37
4.3.4. Entrenamiento y validación del modelo predictivo 40
4.3.5. Implementación del servicio 42
4.4. Arquitectura de la solución 43
4.4.1. Arquitectura de alto nivel 44
4.4.2. Proceso de clasificación de hipervínculos 49
4.4.3. Diseño lógico 52
5. RESULTADOS 56
5.1. Resultados de entrenamiento y validación del modelo predictivo 56
5.2. Comparación de los resultados con n-gramas de distinto valor 57
5.3. Resultados de clasificación de hipervínculos 58

6. CONCLUSIONES 61

7. RECOMENDACIONES 63

5. REFERENCIAS 66

6. ANEXOS 70

ÍNDICE DE FIGURAS

Figura 1. Árbol de clasificación. 17

Figura 2. Curva S. Elaboración propia 20

Figura 3. Clasificación SVM. 24

Figura 4. SVM con kernel polynomial de grado 3. 25

Figura 5. Plan de trabajo de la investigación. 29

Figura 6. Muestra #1 del conjunto de datos inicial 38

Figura 7. Muestra #2 del conjunto de datos inicial 38

Figura 8. Muestra del diccionario de datos generado con trigramas 39

Figura 9. Proceso de implementación del servicio probabilístico 42

Figura 10. Arquitectura de alto nivel de la solución 45

Figura 11. Proceso de clasificación de la solución 50

Figura 12. Diseño lógico de la solución 53

Figura 13. Muestra de tabla de validación 59

ÍNDICE DE TABLAS

Tabla 1: Resumen de costos de Microsoft Azure 6

Tabla 2: Datos del modelo. 21

Tabla 3: Instrumento de selección de datos. 28

Tabla 4: Tabla comparativa entre Google Safe Browsing y PhishTank 34

Tabla 5: Resultados sobre datos de entrenamiento 56

Tabla 6: Resultados sobre datos de validación 57

Tabla 7: Grado de confianza del modelo predictivo según categoría de n-grama 57

Tabla 8. Resumen de resultados 60

GLOSARIO

- **Código malicioso:** Es un tipo de código de computadora dañino diseñado para explotar vulnerabilidades en los sistemas que conlleven a la creación de puertas traseras, brechas de seguridad, robo de datos e información, y otros daños potenciales a archivos y sistemas computacionales. https://usa.kaspersky.com/resource-center/definitions/malicious-code
- **Cross-Site Request Forgery (CSRF):** Es un tipo de ataque que fuerza a un usuario a ejecutar acciones indeseadas sobre una aplicación web en la cual ya se encuentra previamente autenticado https://www.owasp.org/index.php/Cross-Site_Request_Forgery_(CSRF)
- **Cross-Site Scripting (XSS):** Corresponden a un tipo de ataque de inyección de código, en el cual códigos maliciosos son inyectados en sitios web benignos y de confianza del usuario. https://www.owasp.org/index.php/Cross-site_Scripting_(XSS)
- **Dominio de Nivel Superior (TLD):** Es el dominio de más alto nivel en el sistema de nombres de dominio que utiliza Internet. Ejemplos de estos corresponden a los sufijos .com, .net, .org, etc., utilizados a través de Internet. http://www.pcmag.com/encyclopedia/term/52942/tld
- **HTML:** Es un lenguaje de marcado utilizado para describir la estructura de una página web. Cada uno de los elementos de esta estructura son representados mediante etiquetas. https://www.w3schools.com/html/html_intro.asp
- **Hojas de Estilo en Cascada (CSS):** Lenguaje que describe la forma en que los elementos de HTML son desplegados en una pantalla, papel u otro medio. https://www.w3schools.com/css/css_intro.asp
- **Internet de las Cosas:** Red de objetos físicos conectada a Internet, los cuales contienen tecnología embebida para comunicarse o interactuar con sus estados internos o el entorno externo https://www.gartner.com/it-glossary/internet-of-things/
- **Javascript:** Es un lenguaje de programación ampliamente utilizado en páginas web. Habilita la adición de funciones interactivas a dichas páginas, que de otro modo serían únicamente estáticas. http://www.pcmag.com/encyclopedia/term/45585/javascript
- **Lenguaje de Modelado Unificado (UML):** Es un lenguaje gráfico para visualizar, especificar, construir y documentar un sistema. http://www.uml.org/

- **MatLab:** Herramienta de software matemático que ofrece un entorno de desarrollo integrado y que incluye su propio lenguaje de programación. https://es.mathworks.com/products/matlab.html
- **Microsoft Azure:** Plataforma de Microsoft que permite ofrecer una gran variedad de servicios en la nube, soportada por múltiples centros de datos administrados por esta compañía. https://azure.microsoft.com/es-es/overview/what-is-azure/
- **Modelo de Objetos del Documento (DOM):** Corresponde a una interface estándar que describe la forma en que un sistema o aplicación de software puede manipular elementos representados en documentos HTML, XHTML y XML. http://www.w3.org/TR/WD-DOM/introduction.html
- **R:** Es un lenguaje y entorno de programación para gráficos y computación estadística. Es desarrollado como un GNU Project y distribuido bajo licencia GPL. https://www.r-project.org/about.html
- **Protocolo Simple de Acceso a Objetos (SOAP):** Es un protocolo de comunicación por Internet entre aplicaciones y es un componente clave dentro de la arquitectura de un servicio web. Su estructura está basada en un formato XML. https://www.w3schools.com/xml/xml_soap.asp
- **Transferencia de Estado Representacional (REST):** Corresponde a un estilo arquitectural para la comunicación de sistemas distribuidos en la web. Su uso más común es la utilización del protocolo HTTP como interface para la manipulación u obtención de datos. http://www.ics.uci.edu/~fielding/pubs/dissertation/rest_arch_style.htm
- **Superficie de ataque:** El conjunto total de vulnerabilidades alcanzables y explotables de un servicio o sistema computacional. https://www.sans.edu/cyber-research/security-laboratory/article/did-attack-surface
- **Suplantación de identidad (phishing):** Técnica utilizada en un ambiente informático y que hace uso de ingeniería social para adquirir información confidencial de una manera fraudulenta, en donde el atacante se hace pasar por una persona o entidad de confianza para la víctima. https://www.avast.com/es-es/c-phishing
- **Whois:** Protocolo TCP utilizado para la realización de consultas en una base de datos que permite determinar el propietario de un nombre de dominio o dirección IP en Internet. https://whois.icann.org/en/technical-overview

1. RESUMEN

La presente investigación expone un mecanismo desarrollado en la forma de una extensión de navegador, la cual emplea métodos discretos y probabilísticos para indicarle al usuario final cuáles hipervínculos potencialmente maliciosos están presentes en la página web que se encuentra visualizando. De esta manera se puede prevenir el acceso a sitios destinados al robo de información personal y confidencial, o bien propagar código malicioso en el dispositivo utilizado por el usuario.

La generación de conciencia y recomendaciones de seguridad sobre el uso adecuado de computadoras y dispositivos móviles es una ardua labor que ha sido impulsada desde muchos frentes, desde los programas de entrenamiento y políticas de seguridad en las empresas hasta las pautas y consejos brindadas por expertos en el área, proveedores de hardware y software, medios de comunicación dirigidos al consumidor en general, etc. (SANS Institute, 2016). No obstante, el seguimiento y cumplimiento de las recomendaciones de seguridad por parte de los expertos no garantiza que se vayan encontrar completamente seguros a la hora de navegar en la web. Hackers y desarrolladores de software con fines maliciosos pueden utilizar alternativas técnicas o de ingeniería social que puedan engañar a los usuarios a la hora de ingresar a los hipervínculos mostrados en un sitio web, incluso aunque este sitio sea considerado “seguro” ya que se encargan de enmascararlos o incluir recursos adicionales no autorizados con el propósito de que los usuarios finales no se percaten del contenido malicioso subyacente.

Palabras clave: seguridad en navegación web, algoritmos probabilísticos, detección de hipervínculos maliciosos, clasificación binaria de hipervínculos.

1. CAPÍTULO I: INTRODUCCIÓN

1.1. Generalidades

Dado el panorama actual en lo que respecta a las amenazas de seguridad existentes a la hora de navegar en la WWW (Symantec, 2017), no basta únicamente con seguir las mejores prácticas y recomendaciones de seguridad respectivas, ya que el atacante puede valerse de sus habilidades técnicas y/o de ingeniería social para forzar o inducir al usuario final a ingresar a un sitio web malicioso.

Por otro lado, el área de la seguridad de la información se ha visto beneficiada de las principales tendencias tecnológicas de los últimos años, tales como Big Data, aprendizaje automático y la computación en la nube. Así, se ha logrado incorporar los beneficios de estas tendencias en la creación de nuevos mecanismos de seguridad y en el desarrollo de productos de software de seguridad automatizados (Trend Micro, 2017).

La extensión de navegador propuesta en esta investigación se encuentra dirigida a generar un control de seguridad preventivo a la hora de navegar en la web, y de esta manera evitar el acceso a sitios web potencialmente maliciosos. Este mecanismo presenta un enfoque basado en métodos discretos y probabilísticos que pueden ser optimizados en trabajos o investigaciones posteriores para mejorar la probabilidad de detección de sitios maliciosos y reducir el porcentaje de falsos positivos.

1.2. Antecedentes del problema

La ciberdelincuencia en todas sus formas conlleva a un sinnúmero de riesgos, ante los cuales constantemente nos vemos expuestos debido a la ubicuidad del Internet y la proliferación de dispositivos que utilizamos de manera diaria, ya sea en el ámbito personal o laboral. Miles de sitios web con fines maliciosos alojados en todo el mundo se encuentran en funcionamiento y son accedidos por cientos de miles de personas sin percatarse del peligro al que se encuentran expuestas. Estos sitios comprometen la confidencialidad, integridad y disponibilidad de la información, de los dispositivos y de la infraestructura subyacente de las personas y/o organizaciones (Symantec, 2017).

El uso de listas negras para impedir el acceso a sitios de suplantación de identidad u otro contenido malicioso mitiga de cierta medida este riesgo, sin embargo su efectividad depende del

hecho de que un determinado dominio dentro de un hipervínculo se encuentre dentro de la base de datos de la fuente que proporciona el servicio de listas negras.

Los antivirus han utilizado técnicas heurísticas desde hace muchos años para compensar este tipo de carencias específicas, que se reflejan de manera análoga a nivel de las firmas presentes en las bases de datos de este tipo de software (Harley & Lee, s.f). Por otro lado las mejoras considerables en los últimos años a nivel de algoritmos de aprendizaje automático, y la expansión de esta área de conocimiento en general, han propiciado el empleo de inteligencia artificial en la seguridad de la información, mejorando las técnicas para detectar sitios potencialmente maliciosos de manera ad hoc (Trend Micro, 2017).

Por otro lado, se han desarrollado herramientas tales como bloqueadores de anuncios no deseados, conocidos como Ad-blockers (Corpus, 2017), extensiones de navegador que advierten sobre sitios web catalogados como maliciosos a partir de retroalimentación de usuarios finales y otros que obtienen sus resultados a partir de bases de datos de las compañías productoras de antivirus, etc. (Wordpress, 2017). La eficacia de estos métodos yace en la calidad y frecuencia de actualización de las bases de datos.

1.3. Definición y descripción del problema

Se ha identificado un área de mejora en la experiencia del usuario al navegar en la web, específicamente la necesidad de contar con un control de seguridad preventivo que permita mejorar la seguridad de la información de los usuarios cuando navegan en WWW para evitar el acceso a sitios web potencialmente maliciosos por medio de hipervínculos.

El mecanismo empleado como control de seguridad debe ser capaz de identificar patrones ya conocidos dentro de los sitios web asociados a los hipervínculos que se muestran en una determinada página, de tal manera que pueda identificar hipervínculos que representen una amenaza potencial para el usuario y le indiquen el peligro que corren si deciden ingresar a ellos.

1.4. Justificación

El mejoramiento en las tecnologías empleadas para proteger a las personas y organizaciones es un requerimiento indispensable en una era marcada por la reducción de la brecha digital, proliferación de dispositivos móviles, penetración del acceso a Internet, el Internet de las Cosas, así como el almacenamiento y procesamiento de enormes cantidades de datos.

La protección del usuario a la hora de navegar en la WWW requiere el empleo de mecanismos que puedan brindar, de una manera costo-efectiva, un nivel de seguridad aceptable que permita establecer un balance entre la seguridad, funcionalidad y usabilidad en su experiencia diaria a la hora de utilizar un navegador web. Debe ser extensible a otras tecnologías conectadas a la red como el caso de teléfonos inteligentes e Internet de las Cosas.

1.5. Objetivos del proyecto

Los objetivos del proyecto expuestos a continuación han sido redactados tomando como base la taxonomía de Bloom, para reflejar de manera más precisa el nivel cognoscitivo de cada objetivo planteado (Armstrong, 2017).

1.5.1. Objetivo general

Evaluar los resultados generados por una extensión de navegador que permita detectar hipervínculos potencialmente maliciosos dentro de una página web por medio de métodos discretos y probabilísticos para proporcionar seguridad en la navegación web.

1.5.2. Objetivos específicos

- Identificar y seleccionar métodos discretos y probabilísticos que permitan derivar metadatos sobre un hipervínculo web para su correspondiente clasificación.
- Identificar y seleccionar tecnologías costo-efectivas que den soporte a la implementación de los métodos seleccionados.
- Diseñar la arquitectura subyacente de la extensión de navegador con base en los métodos y tecnologías definidas.
- Desarrollar la extensión de navegador a partir de la integración de los métodos, servicios y tecnologías seleccionadas.

1.6. Viabilidad

1.6.1. Punto de vista técnico

Se utilizarán métodos probabilísticos para la clasificación de hipervínculos maliciosos utilizados por la industria y la academia, por lo que este componente se limita únicamente a su

utilización y no a su diseño y desarrollo como parte del mecanismo de seguridad que será elaborado.

Por otro lado, se considerarán tecnologías no propietarias tales como lenguajes de programación especializados en operaciones probabilísticas y estadísticas para el desarrollo del mecanismo estocástico, fuentes de datos de listas negras que cuenten con librerías gratuitas para consultar en tiempo real si un hipervínculo es o no malicioso, etc. El interés es de propiciar el desarrollo e investigaciones futuras sobre la extensión de navegador a construir sin que esto implique altos costos por motivos de adquisición de licencias.

1.6.2. Punto de vista operativo

El mecanismo que será desarrollado utilizará tecnología y servicios disponibles en la nube, tales como los ofrecidos por Microsoft Azure, etc. Su implementación en un dispositivo implica únicamente la utilización del navegador web Google Chrome y la instalación de la extensión en modo desarrollador, la cual no requiere de la aprobación de Google al no residir en su Tienda de Aplicaciones. Adicionalmente se utilizarán plataformas de programación gratuitas, ampliamente utilizadas en el mercado, como el lenguaje de programación estadístico “R” y lenguaje del lado de servidor llamado Node.js, para gestionar la capa lógica de la solución a desarrollar.

1.6.3. Punto de vista económico

Se utilizarán tecnologías y métodos que no poseen costos asociados, con excepción del servicio en la nube ofrecido por Microsoft Azure para alojar parte de la funcionalidad de la extensión de navegador. Específicamente la lógica de clasificación que se alojará a nivel de servidor y el servicio de aprendizaje automático. La intención final es que el mecanismo desarrollado pueda ser probado y utilizado de manera costo-efectiva por parte organizaciones y el público general.

La siguiente tabla resume una estimación de costos asociados a los servicios en la nube de Microsoft Azure:

Servicio	Descripción	Costo mensual aproximado
Servicio de alojamiento web	Servicio de pago por consumo del servicio y recursos computacionales asociados	$45 (Cuenta básica)
Servicio de aprendizaje automático	Servicio para el desarrollo, ejecución y publicación de programas de aprendizaje automático	$0 (cuenta gratuita)
Total aproximado		**$45**

Tabla 1: Resumen de costos de Microsoft Azure

Estos servicios serán consumidos durante el último mes del desarrollo del trabajo de investigación, durante su diseño y programación serán trabajos locales, es decir, en la computadora de trabajo.

1.7. Alcances y limitaciones del proyecto

Este apartado se dirige a definir de manera clara el marco de trabajo del presente proyecto de investigación, de manera que facilite el entendimiento del alcance de esta investigación.

1.7.1. Alcance

A continuación, se listan los elementos que conforman el alcance del presente trabajo de investigación:

- Se utilizará un algoritmo probabilístico ya desarrollado y publicado en la literatura académica para la clasificación binaria de hipervínculos con etiquetas de maliciosos o no maliciosos.
- Los métodos discretos seleccionados se basarán en la comparación de los hipervínculos a partir de una fuente de datos de sitios web previamente identificados como maliciosos, la cual sea mantenida por un tercero y que pueda ser accedida de manera gratuita mediante Internet.
- Se diseñará y codificará la extensión de navegador de tal manera que consuma o utilice los métodos probabilísticos y discretos seleccionados. Los hipervínculos potencialmente maliciosos serán resaltados por medio de la manipulación del Modelo de Objetos del Documento (DOM) utilizando el metalenguaje Javascript y Hojas de Estilo en Cascada (CSS).

- Se desarrollará una capa intermedia entre la extensión de navegador y los servicios de clasificación, la cual será alojada en un servidor en la nube.

1.7.2. Limitaciones

Se presentan los elementos que limitan el alcance detallado anteriormente:

- La extensión de navegador a desarrollar será implementada para el navegador web Google Chrome únicamente; esto debido a que la extensión representa un componente que comparte las características de extensiones genéricas que pueden ser reproducidas en otros navegadores disponibles en el mercado.
- La extensión no será publicada en la Tienda de Aplicaciones de Google, sino que se instalará en modo desarrollador en una computadora local; no obstante el código fuente será alojado en un repositorio público en GitHub para que pueda ser consultado a conveniencia. El enlace a la raíz de este repositorio puede encontrarse en el *Anexo I.*
- El análisis de hipervínculos minificados se excluye dentro del alcance de la extensión de navegador a desarrollar debido a que requieren de la elaboración de análisis de mayor complejidad. En este sentido, todo hipervínculo minificado será resaltado dentro la página web con un indicador que explique que esta no fue analizado por la extensión de navegador desarrollada.
- El análisis de hipervínculos se restringe a todos aquellos que se encuentran incorporados dentro de elementos HTML de tipo anchor (“<a>”), imagen (“<img>”) y script (“<script>”). De esta manera se excluyen hipervínculos presentes o generados de manera dinámica mediante javascript y que sean accedidos por medio de eventos ligados a elementos presentes dentro del Modelo de Objetos del Documento (DOM, por sus siglas en inglés) como por ejemplo etiquetas HTML de tipo input (“<input>”) y botones (“<button>”), o bien que se encuentren presentes en formato de texto o imagen sin un hipervínculo asociado.
- Las fuentes de datos a consultar y recopilar para construir el conjunto de datos de entrenamiento se restringen a aquellas que puedan obtenerse de manera gratuita por medio de Internet.

- El desarrollo técnico e implementación de la solución no tomará como prioridad aspectos no funcionales tales como tiempos de respuesta o rendimiento de la extensión de navegador. Si bien se contemplarán tecnologías web con buenas prestaciones en lo que respecta su características de procesamiento interno y transmisión de mensajes a través de Internet, la optimización de la solución en estos aspectos no está contemplada.

1.8. Estado del Arte

Durante la presente década se han realizado distintos esfuerzos, tanto teóricos como experimentales, en lo que respecta a la identificación de hipervínculos o URLs maliciosos por medio de técnicas y algoritmos comúnmente asociados a aprendizaje automático. En este sentido se mencionan algunos de estos trabajos, seleccionados según su relevancia y similitudes en lo que respecta la presente temática de investigación.

En la investigación realizada por Lawrence et al. (2011), se realiza un análisis por medio de un componente discreto y otro probabilístico. El enfoque probabilístico utiliza un algoritmo de clasificación binaria, al cual se le suministra ciertas características del URL que se desea analizar, incluyendo el Dominio de Nivel Superior (TLD), dominio primario, nombre del computador, ruta completa del recurso que se intenta acceder, así como algunas palabras claves dentro de la semántica del hipervínculo.

A partir de ello se logró construir una fuente de datos de entrenamiento y una fuente de datos de validación del algoritmo utilizado. Se toma como fuente de los datos un listado de sitios web “benignos” obtenidos del ya descontinuado servicio de Yahoo Directory (Rooney, 2014), así como un listado de sitios “maliciosos” de tipo spam, obtenidos de un proveedor de servicios de Internet estadounidense.

La implementación de la propuesta descrita anteriormente se realizó por medio de la creación de scripts desarrollados en distintos lenguajes de programación, los cuales se encargan de obtener las características mencionadas anteriormente del hipervínculo, así como otras características adicionales tales como la localización geográfica de la dirección IP subyacente, información obtenida por medio del protocolo whois, datos de los registros de DNS, etc. Todos estos atributos son recolectados y enviados al algoritmo de aprendizaje en tiempo real,

desarrollado sobre el lenguaje “R”, el cual realiza el cálculo probabilístico y obtiene un resultado casi en tiempo real, aproximadamente 3.5 segundos por solicitud.

Un trabajo con características similares fue desarrollado por Sahoo Liu et al. (2017) en donde utilizan mecanismos para recolectar características asociadas al hipervínculo potencialmente malicioso, particularmente elemento del léxico del hipervínculo, información asociada al protocolo whois, consultas a bases de datos con listas negras de sitios clasificados como maliciosos y elementos inherentes al código HTML subyacente. Sahoo Liu et al. (2017) recomiendan el uso del algoritmo predictor de máquinas de vectores de soporte (SVM, por sus siglas en inglés), el cual es un algoritmo supervisado y uno de los más populares en lo que respecta a la clasificación de hipervínculos maliciosos.

Darling (2015) realiza su propia implementación de una clasificación de hipervínculos maliciosos por medio de un algoritmo de árbol de clasificación llamado J48. No obstante, señala que un atacante o desarrollador malicioso podría utilizar técnicas de evasión que le podrían permitir “disfrazar” el hipervínculo malicioso.

El uso de ciertos atributos dentro del hipervínculo podría hacer creer tanto a la persona como al algoritmo que se trata de dominios seguros, por ejemplo, un hipervínculo que contenga la siguiente forma*: http://www.facebook.xyz.com*. Este hipervínculo podría hacer creer a un usuario no atento que se encuentra dentro del dominio legítimo de Facebook.com, sin embargo el algoritmo podría detectar que el subdomio *xyz* presenta una irregularidad en términos de una sobrecarga de los subdominios de un sitio tal como el de Facebook.

Otros mecanismos que un atacante podría utilizar es la conversión del hipervínculo malicioso en uno acortado o minificado mediante el servicio del mismo nombre Google u otros proveedores de este servicio de manera gratuita accesibles en Internet, por lo cual es importante la revisión de otras características que son recolectadas de otras fuentes adicionales a la semántica del hipervínculo.

Por otro lado, se han desarrollado extensiones de navegador que permiten realizar una detección de hipervínculos maliciosos, un ejemplo de esto es el *“Malware & URL Scanner”*, el cual se encuentra disponible para descargar desde la Tienda de Aplicaciones de Google para el navegador Google Chrome (Chrome Web Store, 2017).

No obstante este tipo de soluciones se basan en la presencia de un determinado dominio o sitio web, incorporado dentro de un hipervínculo, en una o múltiples bases de datos o listas

negras populares en Internet, tales como el servicio de *Google Safe Browsing*, o el denominado *Google Wayback Machine*, el cual mantiene una caché histórica de una gran cantidad de sitios web, registros de servidores de dominio (DNS, por sus siglas en inglés), etc.

Otras extensiones desarrolladas por empresas desarrolladoras de programas antivirus y de servicios de seguridad se basan en la realización de escaneos de vulnerabilidades en sitios web, ya sea para plataformas genéricas o específicas, tales como las listadas por Instantshift (2016) y Wordpress (2017). Sin embargo basan sus resultados en el reconocimiento de patrones existentes en sus bases de datos de firmas para detectar algún tipo malware, y no son capaces de identificar sitios web maliciosos presentes en el Modelo de Objetos del Documento (DOM) si estos sitios no se encuentran presentes en ninguna base de datos o listas negras de sitios maliciosos; o bien, no exponen código malicioso presente o inyectado en una página web en particular. Además muchas de estas extensiones se encuentran dirigidas a administradores de sitios web o desarrolladores web, y no así a usuarios finales no técnicos.

A partir de los casos presentados anteriormente, la extensión de navegador a desarrollar busca mitigar el riesgo presente, al momento de navegar por Internet, de que los usuarios ingresen a sitios potencialmente maliciosos. Esto se buscará lograr por medio de un método discreto y otro probabilístico, que puedan ser utilizados para identificar con un alto grado de certeza una potencial amenaza, a partir de un análisis de los elementos presentes en un hipervínculo determinado; además esta clasificación reside en el análisis del texto del hipervínculo, evitando tener que ingresar o consultar el sitio web subyacente.

2. MARCO TEÓRICO

De acuerdo con Biolchini et al (2005) se siguió el proceso elaborado en su trabajo investigativo para la recopilación de las fuentes de literatura referidas en la presente sección, la cual incorpora un proceso de revisión que abarca tres subprocesos fundamentales:

1. Planeación de la revisión

2. Ejecución de la revisión

3. Análisis de resultados

Los estudios y literatura utilizados a continuación corresponden a los todos aquellos que pasaron los filtros de selección y evaluación dispuestos en el proceso anteriormente mencionado. Estos pueden encontrase en el *Anexo II*.

Seguridad en navegación web

El uso de los navegadores web para ingresar a sitios accedidos principalmente mediante Internet ha crecido vertiginosamente en las últimas dos décadas. Según datos estadísticos del año 2016, existían aproximadamente 966 millones de sitios web a nivel mundial, adicionalmente al mes de diciembre del 2015 se tenían contabilizados 3.26 billones de usuarios de Internet (Stevens, 2016).

La tendencia constantemente en alza de esta estadística conlleva también a un incremento en las amenazas existentes para los usuarios a la hora de navegar en Internet, en donde se registraban para el mismo año un promedio de 37 mil sitios web que eran comprometidos de manera diaria, sin contar los sitios creados con único propósito de vulnerar la seguridad de los usuarios (Stevens, 2016).

En el año 2013, de acuerdo con la última revisión de riesgos de seguridad en aplicaciones web por parte de OWASP (2013), estos fueron categorizados de la siguiente manera:

1. Inyección de código malicioso
2. Autenticación comprometida y gestión de sesiones
3. Cross-Site Scripting (XSS)

4. Referencia insegura a objetos
5. Mala configuración de seguridad
6. Exposición de datos sensibles
7. Ausencia de control de acceso a nivel de función
8. Cross-Site Request Forgery (CSRF)
9. Utilización de componentes vulnerables
10. Redirecciones no validadas

Si bien muchos de estos riesgos se valen en las destrezas y habilidades técnicas del atacante para comprometer los navegadores web y los sistemas subyacentes, no se puede dejar de lado la falta de entrenamiento y concientización del usuario en términos de prácticas de seguridad a la hora de navegar por Internet, lo cual es aprovechado por personas con intereses maliciosos.

Un caso particular en donde se emplea ingeniería social junto con mecanismos técnicos reside en técnicas como la suplantación de identidad, cuya principal contramedida es el uso de listas negras de servicios tales como Windows SmartScreen o Google Safe Browsing (Virvilis et al., 2015).

Listas negras

Dentro del contexto las tecnologías de información y comunicación (TICs), las organizaciones manejan listas que incluyen usualmente direcciones IP, nombres de dominio o hipervínculos que son considerados potencialmente maliciosos y ante los cuales se aconseja a los usuarios no visitarlos. Estas listas son conocidas como listas negras (Metcalf & Spring, 2013).

En un estudio realizado por Metcalf & Spring (2013), los autores obtuvieron 28 listas negras de distintas fuentes presentes en Internet entre el 2012 y 2013 con el objetivo primario de comparar el nivel de intersección presente entre cada una de las listas. Sus resultados indicaron que estas listas no se intersecaban en un alto grado, y recomiendan no valerse de una única fuente, sino utilizar la mayor cantidad de fuentes posibles.

Aprendizaje automático

Como área de conocimiento, el aprendizaje automático surge como una intersección entre los campos de ciencias de la computación y la estadística. Según Mitchell (2006), esta área

se dirige a responder *"Cómo podemos construir sistemas computacionales que mejoren automáticamente con experiencia, y cuáles son las leyes fundamentales que gobiernan todo proceso de aprendizaje?"*

El uso de aprendizaje automático se ha esparcido desde el área de ciencias de la computación y ha incursionado de manera exitosa en la ingeniería de software, en donde se utilizan algoritmos de aprendizaje para personalizar el software según el contexto en el que es empleado en un rango de aplicación que abarca desde software de reconocimiento de voz hasta su empleo en robots autónomos diseñados para cumplir una tarea o conjunto de tareas específicas (Mitchell, 2016).

El aprendizaje automático se refiere a un conjunto de herramientas para el entendimiento de los datos. Estas pueden ser clasificadas como supervisadas o no supervisadas. El aprendizaje supervisado involucra la construcción de un modelo estadístico para predecir, dictar o estimar una salida basada en una o más entradas.

Entre los métodos supervisados destacan los modelos de regresión lineal, modelos de clasificación tales como la regresión logística, análisis de discriminantes, modelos basados en árboles de decisión y los métodos de separación geométrica de datos por medio de hiperplanos denominados Vectores de Soporte (o SVM, por sus siglas en inglés).

Modos de entrenamiento

Al hablar de tipos de entrenamiento de un algoritmo de aprendizaje, básicamente se piensa en dos categorías bien definidas (Heaton, 2013):

- Entrenamiento supervisado: Este tipo de entrenamiento se realiza cuando se tiene claro el tipo de resultado o salida (etiqueta) que se desea obtener del algoritmo de aprendizaje. A manera de ejemplo de este modelo de entrenamiento, se puede referenciar el uso de regresión lineal para la predicción de precios de viviendas a partir de distintos atributos tales como el número de cuartos, baños, garaje, ubicación de la vivienda, etc. En este caso en particular, el método de regresión producirá una tendencia predictiva del valor de mercado de tales viviendas tomando en cuenta datos históricos existentes sobre su valorización a partir de todos los atributos incluidos, esto puede ser utilizado por compradores potenciales a la hora de buscar una vivienda como en el caso de la propuesta de

implementación realizada por Ng (2015) para la creación de una aplicación de dispositivo móvil para compradores en Londres.

- Entrenamiento no supervisado: Se presenta cuando no se proporciona al algoritmo de aprendizaje las salidas esperadas. Un caso común de este tipo de entrenamiento se refleja mediante el uso de métodos de clasificación tales como el algoritmo de los k-vecinos más cercanos (KNN, por sus siglas en inglés). Jabbar et al. (2013) utilizan este algoritmo junto con un algoritmo genético para la clasificación de enfermedades de corazón que asista a médicos en la India en la definición de los pasos clínicos a seguir como parte de un tratamiento específico al tipo de enfermedad del paciente. El algoritmo toma en consideración distintos atributos presentes en los datos de entrada suministrados e intenta clasificar cada caso con base en los casos más cercanos; esto partir de funciones matemáticas utilizadas para medir la distancia existente con otros elementos previamente clasificados.

Por otro lado, el entrenamiento del algoritmo de aprendizaje puede realizarse por lotes o en línea. La variante por lotes implica acumular aprendizaje de un cierto número de elementos y actualizar el algoritmo de manera acorde. Mientras tanto, en el entrenamiento en línea, el aprendizaje se lleva a cabo por cada elemento de la fuente de datos de entrenamiento; este es útil cuando se requiere que el algoritmo aprenda y entrene al mismo momento (Heaton, 2013).

Modelado de problemas

Según Heaton (2013), existen cuatro enfoques principales sobre los cuales se puede modelar un problema de aprendizaje automático:

- Clasificación de datos: El objetivo es la determinación de la clase sobre la cual deben categorizarse los datos de entrada, tomando en cuenta que la resolución de este tipo de problemas resulta en un valor cualitativo y no así cuantitativo, como por ejemplo el género de una persona, su color de ojos, etc. Métodos de predicción tales como la regresión logística, los k-vecinos más cercanos (KNN), análisis del discriminante lineal (LDA), árboles de decisión, etc., son todos ejemplos de algoritmos basados en modelos de clasificación (James et al., 2013). La realización de un análisis crediticio de una persona, el diagnóstico médico y selección de un tratamiento a partir de una serie de síntomas presentes en un

paciente o la detección de transacciones bancarias fraudulentas son problemas que pueden resolverse mediante clasificación de datos.

- Análisis de regresión: La intención principal es entrenar al algoritmo con datos de entrada de tal manera que permita generar una respuesta calculada sobre dichos datos, la cual es de tipo numérica. Para la generación de esta respuesta el modelo buscar encontrar una relación entre dos variables y ajustarla de tal manera que se produzca una ecuación que pueda ser utilizada para predecir una variable cuantitativa a partir de uno o múltiples datos de entrada. El método más conocido dentro de este modelo es lo que se domina regresión lineal y busca encontrar una ecuación que produzca una recta lineal entre las variables dependientes (James et al., 2013).
- Segmentación: Este tipo de modelo se dirige a realizar una identificación y segmentación de grupos con características similares y a "etiquetar" observaciones con base en el grupo al cual pertenecen; este modelo es común dentro del modo de aprendizaje no supervisado.

  Para definir cada segmento, los métodos existentes dentro de este modelo buscan medir las distancias entre las distintas observaciones y entre los distintos segmentos de tal manera que a la hora de realizar una clasificación de una nueva observación, se realice para que se logre minimizar la distancia con el resto de observaciones que pertenezcan al mismo segmento (IBM, 2012).

  El algoritmo denominado k-medios (K-Means, su nombre en inglés) es uno de los más ampliamente utilizados y dentro de su funcionamiento asigna un número fijo de segmentos sobre los cuales realiza la clasificación de observaciones de forma iterativa, y además reajusta la distancia con los centros de cada uno de los segmentos para refinar la distribución de cada segmento hasta que ya no se pueda optimizar más el modelo generado (IBM, 2012).
- Series temporales: A partir de una secuencia de observaciones ordenadas a través del tiempo se realiza un entrenamiento tal que el algoritmo permita realizar una predicción basada en la tendencia producida por esta secuencia. Ejemplos típicos del uso de análisis de series temporales corresponden a pronósticos del tiempo,

proyecciones de ventas de una compañía, valor de las acciones de una empresa en bolsas de valores, etc.

Métodos de clasificación

Un modelo de clasificación es ampliamente utilizado cuando la variable dependiente, es decir, el resultado, es una variable de tipo cualitativa más que de tipo cuantitativa. A partir de una observación, la obtención de una respuesta cualitativa es referida como la clasificación de dicha observación, debido a que requiere ser asignada a una categoría o clase (James et al., 2013).

Dentro de este subdominio, se hará una breve introducción de los métodos de análisis discriminante lineal (LDA, por sus siglas en inglés), árboles de decisión y regresión logística, los cuales son utilizados para la realización de clasificaciones binarias, técnica que puede ser usada para la clasificación de hipervínculos, como ya se ha descrito anteriormente en trabajos e investigaciones realizadas sobre el problema de clasificación de hipervínculos maliciosos. Por otro lado, se presenta de manera adicional el método de Máquinas de vectores de soporte (SVM, por sus siglas en inglés), ya que a pesar de no ser un método probabilístico, es también utilizado para resolver problemas de clasificación.

Árboles de decisión

Este tipo de métodos son utilizados para el modelado de problemas de clasificación o regresión y sus algoritmos son conocidos actualmente como árboles de clasificación y regresión, (CART, por sus siglas en inglés) (James et al., 2013). La creación de un modelo CART involucra la selección de variables de entrada y de puntos de división en dichas variables hasta que se logre generar un árbol optimizado que cumpla con el propósito particular. A continuación se presenta una estructura simple de un árbol de decisión para la clasificación del género de una persona:

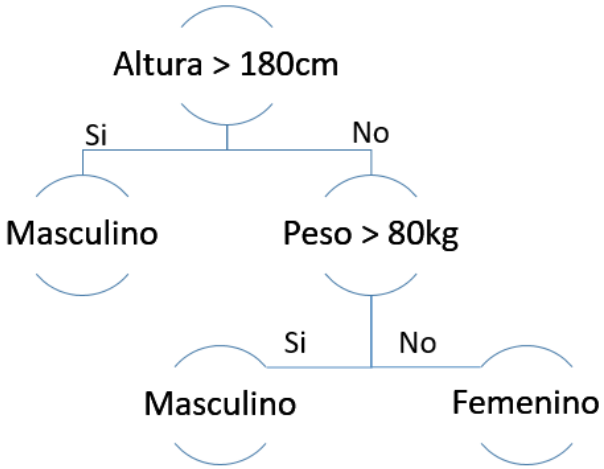

Figura 1. Árbol de clasificación.
Fuente: Elaboración propia.

De esta manera la clasificación de una nueva entrada requiere simplemente atravesar el árbol evaluando cada una de las condiciones establecidas comenzando desde el nodo raíz.

Creación del árbol

La creación del árbol decisión involucra la utilización de un algoritmo voraz que permita definir los puntos de división y finaliza mediante la definición de un criterio predefinido de detención de la ramificación del árbol.

Este algoritmo voraz utiliza lo que se denomina como división binaria recursiva, la cual consiste en la alineación de los valores y la creación y prueba de distintos puntos de división hasta que se logren seleccionar aquellos puntos con menor costo. Esto se realiza por cada ramificación, de ahí su naturaleza recursiva. Una de las funciones existentes que determina los mejores puntos de división en un problema de clasificación es la función Gini, cuya ecuación se presenta a continuación, en donde *p1 y p2* representan los porcentajes de los resultados ante la selección de una determinada variable (James Witten et al., 2013):

$$G = 1 - (p_1{}^2 + p_2{}^2)$$

El índice Gini provee entonces un indicador de la "pureza" de los nodos hoja del árbol, es decir, qué tan mezclados están los datos de entrenamiento asignados al nodo. Un índice igual a cero representa una pureza perfecta, mientras que uno que obtenga una división de clases de 50-50, o bien G = 0.5 representa la peor pureza (James et al., 2013).

Ejemplo

En una clase de 30 estudiantes de secundaria, se tienen las siguientes estadísticas respecto a los alumnos que les gusta jugar baloncesto:

- Del total de alumnos de la clase, 20 son hombres y son 10 mujeres, de estos, 2 mujeres y 13 hombres juegan al baloncesto.
- Del total de alumnos de la clase 14 superan 1.65m de altura y 16 no sobrepasan una altura de 1.65m, de estos, 9 personas que superan una altura de 1.65m y 6 que no la superan juegan al baloncesto.

Ante la llegada de un alumno nuevo a la clase, se quiere determinar cuál de estas dos variables puede presentar mejores resultados como punto de división para decidir si la persona jugaría o no al baloncesto. Para ello se utilizaría la función Gini de la siguiente manera:

Variable de género

1. **Se calcula el índice Gini para el nodo de mujeres**

   0.2: Porcentaje de mujeres que juega baloncesto respecto al total de mujeres de la clase

   0.8: 1- 0.2

   $(0.2^2 + 0.8^2) = 0.68$

2. **Se calcula el índice Gini para el nodo de hombres**

   0.65: Porcentaje de hombres que juega baloncesto respecto al total de mujeres de la clase

   0.35: 1- 0.65

   $(0.65^2 + 0.35^2) = 0.55$

3. **Se calcula el ponderado Gini para la división de género**

   10: Total de mujeres en la clase

   20: Total de hombres en la clase

   $\left(\frac{10}{30}\right) * 0.68 + \frac{20}{30} * 0.55) =$ **0.59**

Variable de altura

1. **Se calcula el índice Gini para el nodo mayor a 1.65m**

   $(0.43^2 + 0.57^2) = 0.51$

2. **Se calcula el índice Gini para el nodo menor a 1.65m**

   $(0.56^2 + 0.44^2) = 0.51$

3. **Se calcula el ponderado Gini para la división de género**

$\left(\frac{14}{30}\right) * \ 0.51 + \frac{16}{30} * 0.51)$ = **0.51**

Debido a que 0.59 es un índice mayor a 0.51, la división utilizando la variable de género toma precedencia sobre la variable de la altura dentro del árbol de clasificación.

Criterio de detención

El procedimiento más común en lo que respecta a la definición del criterio de detención es mediante la asignación de un conteo mínimo sobre el número de instancias de entrenamiento asignadas a cada nodo hoja. Si el conteo no cumple con dicho número, entonces el algoritmo no acepta la división y el nodo es tomado como un nodo hoja final (James et al., 2013).

Regresión logística

La regresión logística es un método de clasificación binaria empleado cuando se requiere obtener la probabilidad de una observación a partir de más de una variables explicativas, y restringiendo a su vez una respuesta o variable de salida binomial (Sperandei, 2014). Debido a que la probabilidad de un resultado es una tasa, el modelado corresponde el logaritmo de la probabilidad dada por:

$$\ln\left(\frac{p(X)}{1 - \ p(X)}\right) = \ \mathrm{B}_0 + \ \mathrm{B}_1\mathrm{x}_1$$

En donde $p(X)$ corresponde a la probabilidad de un evento, y cada $\mathrm{B}_i$corresponde a un coeficiente de regresión asociado a la variable explicativa x_i.

Para obtener los coeficientes de regresión, el método más utilizado es conocido como *estimación por máxima verosimilitud*, cuyo objetivo es estimar valores para B_0 y B_1, tales que la probabilidad predicha para cada observación esté tan cerca como sea posible al estatus observado de cada observación, ya sea un número cercano a uno para las observaciones que representan un caso positivo de la variable medida y uno cercano a cero para todas las que no lo presentan (James et al., 2013).

Esto se traduce matemáticamente en la función de probabilidad:

$$p(\mathrm{B}_0, \mathrm{B}_1) = \prod_{i:y_1} p(x_i) \ \prod_{i':y'_1} \left(1 - \ p(x_{i'})\right)$$

Las variables estimadas $\widehat{\mathrm{B}}_0$ y $\widehat{\mathrm{B}}_1$son seleccionadas de tal manera que logren maximizar esta función de probabilidad. Al graficar este método, se obtiene la siguiente curva en donde se observa que el eje funciona como una tasa probabilística con rango [0,1]:

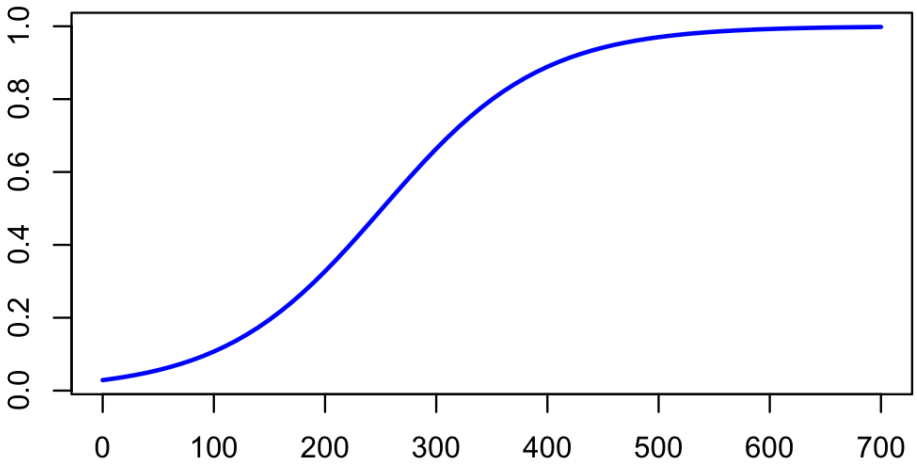

Figura 2. Curva S. Elaboración propia

Debido a que los resultados del modelo deben encontrarse dentro del rango [0,1], ya que deben cumplir con una respuesta binomial o dicotómica, uno de los métodos empleados más comunes es la utilización de lo que se denomina como función logística (James et al., 2013):

$$\hat{p} = \frac{e^{B_0 + B_1 x_1}}{1 + B_0 + B_1 x_1}$$

Una vez obtenidos los coeficientes de regresión, se procede con la realización de predicción por medio de la sustitución de los coeficientes en la función logística junto con valor de la variable independiente.

El siguiente ejemplo permitirá comprender a un mayor nivel la forma en que opera este método de regresión:

Problema

Se requiere conocer la probabilidad de recibir un crédito hipotecario a partir de una calificación crediticia de 585. El rango de las calificaciones de crédito emitidas por el banco se extiende desde los 300 a 600 puntos.

Datos del modelo

La siguiente tabla contiene 79 registros históricos de las aprobaciones realizadas por el banco junto con la calificación crediticia del cliente. Un valor de uno representa que el crédito fue aprobado mientras que un valor de cero indica que fue rechazado:

Calificación	Aprobado
300	0
302	0
305	0
315	0
317	0
320	0
320	0
325	0
335	0
337	0
339	0
341	0
342	0
342	0
345	0
350	0
362	0
367	0
371	0
373	0
377	0
389	0
400	0
403	0
405	0
409	0
409	0
415	0
420	0
425	0
430	0
430	0
438	0
441	0
442	0
444	0
448	0
456	0
458	0
467	0
478	0
481	0
485	0
489	0
500	0
505	0
510	0
511	0
516	0
516	0
521	1
533	0
535	0
539	0
541	1
541	1
550	1
555	0
560	1
566	0
567	0
568	1
570	1
575	1
575	1
580	1
581	1
583	0
584	0
585	1
589	0
590	1
591	1
595	1
597	1
598	1
599	1
600	1
600	1

Tabla 2: Datos del modelo.
Fuente: Elaboración propia.

Estimación de los coeficientes

Luego de realizar los cálculos se obtiene los siguientes valores para los coeficientes:

$\widehat{B}_0$ = -26,79600838

$\widehat{B}_1$ = 0,04832335891

Cálculo del predictor

El último paso consiste en incorporar los valores de los coeficientes y la variable independiente por la nota de crédito 585 en la ecuación de regresión:

$$\hat{p} = \frac{e^{B_0 + B_1 x_1}}{1 + B_0 + B_1 x_1}$$

$$\hat{p} = \frac{e^{-26{,}79600838 + 0{,}04832335891 \text{ x } 585}}{1 + -26{,}79600838 + 0{,}04832335891 \text{ x } 585}$$

$$\hat{p} = 0.814$$

Esto indica que obteniendo una calificación crediticia de 585 se tiene un 81.4% de probabilidades de obtener el préstamo hipotecario del banco.

Análisis discriminante lineal (LDA)

Este método es comúnmente utilizado en estadística, reconocimiento de patrones y aprendizaje automático, y es sumamente conveniente cuando el problema de clasificación involucra más de dos categorías (James et al., 2013). El objetivo de este método es encontrar una combinación lineal de variables óptima que permita realizar la separación de dos o más clases.

Dentro del ámbito del aprendizaje automático, esta alternativa es considerada superior al método de regresión logística cuando se presentan las siguientes condiciones:

- El problema requiere la definición de más de dos clases.
- Se cuenta con pocas observaciones que se puedan utilizar para la estimación de los coeficientes.
- Cuando las clases se encuentran muy separadas, el método de regresión logística se vuelve inestable.

Para la realización de predicciones, este método utiliza el Teorema de Bayes, el cual le permite estimar la probabilidad, para una entrada determinada, de pertenecer a una clase específica. A continuación se muestra el postulado de este teorema (James et al., 2013):

$$\Pr(Y = k | X = x) = \frac{\pi_k f_k(x)}{\sum_{l=1}^{K} \pi_l f_l(x)}$$

En el postulado anterior, π_k representa la probabilidad previa de que una observación escogida al azar provenga de la k-ésima clase, es decir, que la observación esté asociada con la k-ésima clase de la variable de respuesta Y. $f_k(x)$ corresponde a lo que se denomina como la función de densidad de X, para una observación que provenga de la k-ésima clase; esta es

relativamente grande si hay una alta probabilidad de que una observación en la k-ésima clase tenga $X \approx$ x. En la práctica la estimación de esta función toma dos supuestos (James et al., 2013):

La función de densidad sigue una distribución normal o gaussiana.

- Todos los grupos cuentan con la misma matriz de covarianza.

Dadas las condiciones anteriores y a partir de diversas derivaciones matemáticas se tiene como resultado la siguiente función:

$$\delta_k(x) = \log \pi_k - \frac{1}{2}\, \mu_k^T\, \Sigma^{-1}\, \mu_k + \, x^T \Sigma^{-1} \mu_k$$

Debido a que la función anterior es lineal en x, se le conoce como función discriminante lineal; es por esto que el método es conocido como análisis discriminante lineal. Como no se conocen los valores para π_k, μ_k y Σ, estos son estimados de la siguiente manera:

- $\pi_k = \frac{n_k}{n}$, en donde *n* es el número de observaciones y n_k es el número de observaciones en la k-ésima clase.
- $\mu_k = \frac{1}{n_k} \sum_{\{i:g_i=k\}} x_i$, este es el promedio de todas las observaciones de entrenamiento en a k-ésima clase.
- $\hat{\Sigma} = \frac{1}{n-K} \sum_k \sum_{\{i:g_i=k\}} (x_i - \widehat{\mu_k})(x_i - \widehat{\mu_k})^T$, este es el promedio ponderado de las varianzas de muestra para cada una de las K clases.

A partir de esto, el método requiere la inserción de todos estos estimados en la función discriminante lineal postulada anteriormente y asigna una observación $X = x$ a la clase que sea la más grande (James et al., 2013).

<u>Máquinas de vectores de soporte (SVM)</u>

Este método fue desarrollado en el año de 1995 para ser utilizado en problemas de clasificación y de regresión; un algoritmo de este tipo debe cumplir con las siguientes tareas en términos de la resolución de un problema de clasificación (Meyer & Wien, 2017):

- Separación de clases: Para ello se busca encontrar el híper-plano de separación óptimo para la las dos clases en cuestión, tratando de maximizar la distancia entre los puntos más cercanos de las clases; a los puntos que se encuentran en el límite se les conoce como vectores de soporte.

- Superposición de clases: Los puntos que se encuentren en el lado equivocado del margen de discriminación son desvalorizados para reducir su influencia.
- No linealidad: Si no se encuentra una separación lineal, se proyectan los puntos en el espacio de una dimensión más alta, en donde estos logren ser separables de forma lineal.
- Solución del problema: La resolución del problema se convierte en una optimización cuadrática.

La siguiente figura grafica los principales elementos mencionados anteriormente:

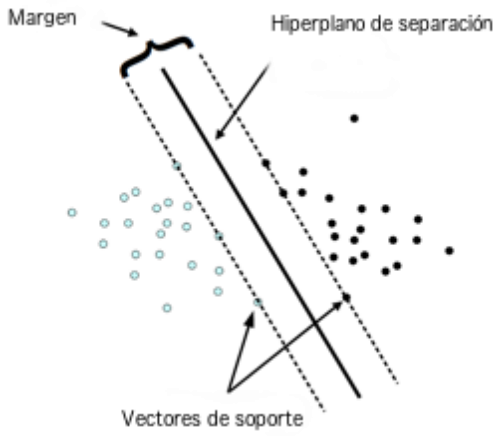

Figura 3. Clasificación SVM.
Fuente: Meyer & Wien (2017).

El denominado híper-plano es un subespacio de dimensión *p-1* para un determinado espacio de dimensión *p*, por ejemplo, en un espacio de dos dimensiones con ejes *x,y* este representa una línea, mientras que en un espacio de tres dimensiones con ejes *x,y,z*, representa precisamente un plano (James et al., 2013).

La identificación del híper-plano dentro de este modelo se realiza mediante el uso de los denominados clasificadores de vectores de soporte, y este método de identificación se le conoce como margen suave. La idea detrás de la definición del margen suave consiste en obviar que ciertas observaciones individuales puedan quedar mal clasificadas según la traza del híper-plano con tal de proveer una mejor y más robusta clasificación para la mayoría de las observaciones de entrenamiento (James et al., 2013).

Esto es particularmente útil cuando nos enfrentamos a un problema en donde la definición de un límite lineal provocaría que el algoritmo de clasificación se desempeñe una

manera pobre. Un modelo basado en SVM logra esto mediante la utilización de *kernels*, los cuales son funciones que tienen como objetivo cuantificar la similitud entre una pareja de observaciones.

SVM utiliza funciones kernel polinomiales de grado *d*, siendo *d* un entero positivo, dicha función puede representarse de la siguiente manera:

$$K(x_i, x_{i'}) = 1 + (\sum_{j=1}^{p} x_{ij} x_{i'j})^d$$

La utilización de esta función con *d > 1* permite obtener límites más flexibles que el que se puede obtener mediante una función lineal, adicionalmente cuando un clasificador de soporte de vectores es combinado con una función kernel no lineal el clasificador que se obtiene como resultado es llamado un SVM que toma la siguiente forma:

$$f(x) = B_0 + \sum_{i \in \delta} \alpha_i K(x, x_i)$$

La siguiente figura muestra una representación al utilizar un algoritmo SVM con un kernel polinomial de grado tres:

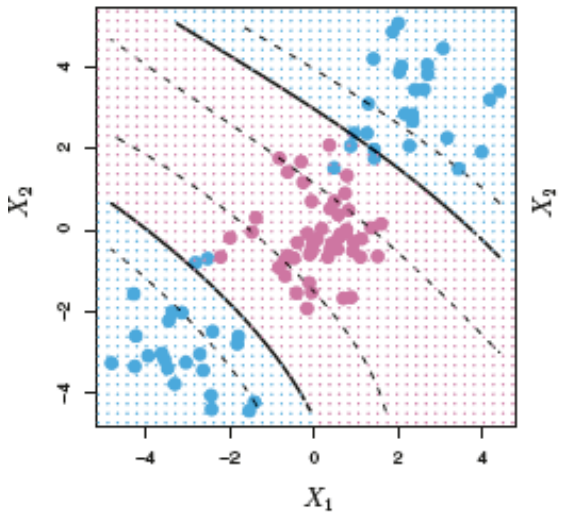

Figura 4. SVM con kernel polynomial de grado 3.
Fuente: (James, Witten, et. al., 2013).

Conjuntos de datos

Los datos utilizados para alimentar el algoritmo de aprendizaje automático según el modelo utilizado son un aspecto vital, y su preparación contribuye en gran medida al grado de fiabilidad del algoritmo. Para ello se requiere pasar por tres etapas esenciales, a saber: seleccionar los datos, preprocesar los datos y transformar los datos (Brownlee, 2013).

La selección de datos consiste en seleccionar un subconjunto de todos los datos disponibles de tal modo que se logren incluir preferiblemente los atributos que se consideren que realmente tengan un impacto directo en modelo de aprendizaje. El preprocesamiento implica realizar un formateo, limpieza, y en los casos que sea necesario, tomar una muestra del conjunto total de datos en caso de que dicha colección sea muy extensa y esto pueda perjudicar los tiempos de ejecución del algoritmo.

En cuanto a la transformación de los datos, este paso representa cualquier tipo de conversión de datos a un conjunto de valores que sean entendibles por el algoritmo de aprendizaje, ya sea por medio de descomposición o agregación de atributos, normalización de los datos, etc. Por otro lado, en el área de aprendizaje automático es común segmentar la colección de datos en tres subconjuntos distintos (Ripley, 1996):

- Datos de entrenamiento: Es un conjunto de datos de ejemplo utilizado para el aprendizaje del algoritmo, es decir, para la adecuación de los parámetros del clasificador.
- Datos de validación: Es un conjunto de ejemplos utilizados para ajustar los parámetros de un clasificador.
- Datos de prueba: Utilizados para evaluar el desempeño de un clasificador completamente especificado.

En la práctica se tiende a segmentar en términos generales el conjunto de datos total en un 60% dedicado al entrenamiento del algoritmo, un 20% para la validación y el restante 20% para la realización de pruebas y evaluación del desempeño (Ng, 2015).

3. MARCO METODOLÓGICO

3.1.Tipo de Investigación

Se realizará una investigación aplicada para el diseño y desarrollo de una extensión de navegador web, que emplee métodos discretos y probabilísticos en la identificación de hipervínculos potencialmente maliciosos.

3.2.Alcance Investigativo

El alcance investigativo del presente trabajo es de tipo correlacional. La extensión del navegador contiene un componente discreto que se empleará como un servicio para determinar si un hipervínculo presente en una página web ha sido previamente clasificado como malicioso por un tercero de confianza.

No obstante, el diseño mencionado también contiene un componente probabilístico, en donde se identificará y recolectará un conjunto de variables asociadas a un hipervínculo presente en una página web, y a partir de ellas se intentará predecir la naturaleza del sitio web relacionado al hipervínculo, es decir, una clasificación binaria que indique si el sitio es o no malicioso, en caso de que el hipervínculo no se encuentre listado en una de las listas negras.

Para ello se analizarán y seleccionarán un conjunto de variables que serán empleadas dentro del modelo de predicción, según la correlación que se presume que estas posean con respecto a la naturaleza del sitio asociado, y se convertirá este modelo en un servicio disponible por la extensión para identificar hipervínculos potencialmente maliciosos.

3.3.Enfoque

El enfoque de la investigación es de tipo cuantitativo. Las bases arquitectónicas de la extensión de navegador web que se desarrollará para detectar hipervínculos potencialmente maliciosos estarán constituidas por métodos matemáticos que serán traducidos en la recolección y procesamiento de una serie de variables numéricas que podrán determinar o predecir la naturaleza de los hipervínculos que serán recibidos como entrada. Para ello se estudiarán diversos modelos existentes así como su base estadística, de tal manera que se logre seleccionar e implementar a nivel de software los algoritmos necesarios para formar el modelo de predicción deseado.

Además, se construirán servicios web sobre la infraestructura en la nube de Microsoft Azure, para la recepción de los hipervínculos capturados por medio de la extensión de navegador para su subsecuente inspección.

3.4.Diseño

En el presente trabajo se utilizará un diseño investigativo estadístico, sobre el cual estará basada la extensión de navegador web que se desarrollará para la detección de hipervínculos potencialmente maliciosos en un sitio web.

3.5.Instrumentos de Recolección de Datos

El siguiente cuadro resumen indica los tipos de instrumentos de recolección de datos que se utilizarán durante la investigación:

Instrumento	Utilización
Fichas bibliográficas	Se recopilará la literatura académica disponible para el entendimiento e implementación del método probabilístico, tanto a nivel de algoritmos como a nivel de implementación en lenguajes de programación matemáticos.
Bases de datos de listas negras	Se accederá a bases de datos disponibles en Internet de sitios web clasificados como maliciosos, tanto para su utilización en el fuente de datos de entrenamiento y verificación, como para su utilización en el método discreto a diseñar e implementar

Tabla 3: Instrumento de selección de datos.

3.6.Técnicas de Análisis de la Información

En lo correspondiente a las técnicas que se utilizarán para el análisis de los datos recolectados, se describen las siguientes:

- Análisis de regresión logística: Se empleará un método que tenga como objetivo normalizar la información para el establecimiento de una distribución binomial que, a su vez, permita generar un modelo de clasificación de dos categorías: {1,0}, en donde el clasificador 1 represente un hipervínculo malicioso y 0 represente uno no malicioso.

- Análisis documental: A partir del material recopilado de la literatura académica consultada, se analizará la implementación de algoritmos de regresión logística en un lenguaje de programación matemático, que puedan ser utilizados como mecanismos de entrenamiento y validación de las clasificaciones. Por otro lado, se revisarán librerías disponibles en forma de interfaces de comunicación con respecto a sitios que manejen listas negras de hipervínculos, direcciones IP y nombres de dominio potencialmente maliciosos.

3.7.Estrategia de Desarrollo de la Propuesta

El plan de trabajo a realizar se compone de las siguientes etapas secuenciales:

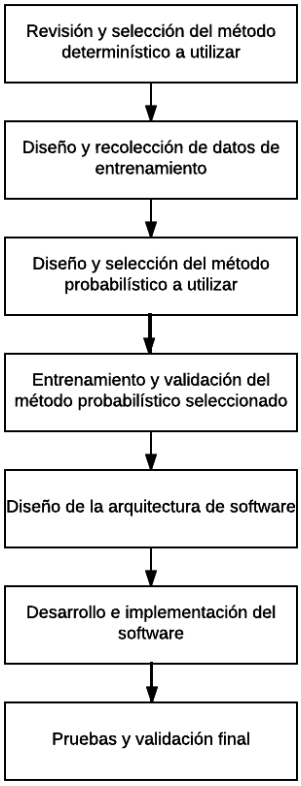

Figura 5. Plan de trabajo de la investigación.

Revisión y selección del método discreto a utilizar

Durante esta etapa se investigarán y analizarán posibles estrategias que puedan ser utilizadas para validar si un sitio web es o no malicioso. Comúnmente este tipo de servicios se presenta en la forma de listas negras o repositorios mantenidos por organizaciones de seguridad

informática, proveedores de servicios de Internet o comunidades sin ánimos de lucro sobre sitios de suplantación de identidad, spam, etc. Para lograr esto se analizará la viabilidad técnica, económica y operativa de la incorporación del servicio dentro del diseño de la extensión.

Diseño y recolección de datos

Esta etapa corresponde al diseño de la fuente de datos que se utilizará para entrenar al algoritmo respectivo, así como el subconjunto de datos dentro de esta fuente de que se utilizarán para validar el nivel de confiabilidad del algoritmo posterior a su entrenamiento, es decir, los datos de validación.

Revisión y selección del método probabilístico a utilizar

En esta etapa se identificarán algoritmos y métodos probabilísticos ya empleados por la industria y academia para el análisis y predicción de hipervínculos potencialmente maliciosos que sean ingresados como entrada al modelo seleccionado. Para ello se analizará la viabilidad técnica, económica y operativa de la incorporación del modelo seleccionado como un servicio dentro del diseño de la extensión, ampliando de esta manera la capacidad y confiabilidad de la extensión de navegador que será desarrollada.

Por otro lado, se identificarán y seleccionarán las herramientas que serán utilizadas para implementar el método probabilístico seleccionado. De esta manera se definirá el modo de entrenamiento, ya sea en línea o por lotes, y se identificarán las características (variables) que se recolectarán de cada hipervínculo para que puedan ser incorporadas dentro de la fuente de datos, además se recopilará el repositorio de datos a utilizar como parte de la fuente de datos.

Entrenamiento y validación del método probabilístico seleccionado

Durante esta etapa se realizan las actividades relacionadas al entrenamiento del método probabilístico seleccionado a partir de la fuente de datos y su posterior validación. Estas actividades corresponden a la definición y ajuste iterativo de un modelo predictivo que permita realizar la clasificación de hipervínculos con un alto grado de confiabilidad, la alimentación de dicho modelo con base en los datos de entrenamiento y ajustes adicionales realizados posterior al uso de los datos de validación para calibrar los resultados de predicción.

Diseño de la arquitectura del software

Durante esta etapa se diseñará la arquitectura de cada uno de los componentes que integrarán la solución de software a desarrollar y sus relaciones entre sí. Se identificarán las tecnologías y lenguajes de programación a utilizar en la siguiente etapa.

Desarrollo e implementación del software

Corresponde a la programación (codificación), configuración e implementación de la arquitectura diseñada para solución final.

Esto incluye los siguientes elementos:

- Lógica presente en la extensión del navegador (lado del cliente).
- Lógica de clasificación que será incorporada de forma externa a la extensión (lado del servidor).
- Interfaces de comunicación entre la lógica de clasificación con el método discreto (servicio de un tercero) y con el método probabilístico diseñado en etapas anteriores.

Pruebas y validación final

En esta última etapa se realizan pruebas del funcionamiento integral de la solución de software desarrollada e implementada. Estas pruebas incluyen la correcta operación de la extensión del navegador según el alcance funcional descrito, la validación de los resultados de clasificación obtenidos del método probabilístico y la correcta interacción entre todos los componentes de la arquitectura.

4. DESARROLLO DE LA SOLUCIÓN

4.1.Selección y diseño del servicio discreto

El servicio discreto que se incorpora dentro de solución de software tiene como objetivo brindar el primer nivel de consulta para la clasificación de hipervínculos maliciosos, es decir, responde a una mecanismo de clasificación no predictivo, con base en la consulta a una base de datos o lista negra de sitios previamente catalogados como dañinos según su naturaleza (malware, ingeniería social, contenido nocivo, etc.).

Por tanto, el valor de este servicio como un componente dentro de la arquitectura de la solución depende de la calidad de la base de datos subyacente, en términos de la amplitud de registros con los que cuente, así como el grado de fidelidad de los datos almacenados, es decir, qué tan fidedignos realmente son a la hora de indicar que una determinada página o sitio web es realmente malicioso.

4.2.1. Identificación de mecanismos disponibles

Como parte de la arquitectura de la solución desarrollada y tomando en cuenta el objetivo de utilizar tecnologías y servicios costo-efectivos, se ha decidido optar por la utilización de un servicio de listas negras (repositorio de datos) que sea gratuito y se encuentre disponible en Internet, el cual a su vez proporcione un API consumible vía servicios web de tipo SOAP o REST en lo que respecta a la consulta de hipervínculos y que cuente con un repertorio amplio y cotidianamente mantenido de registros almacenados.

A continuación, se describen dos servicios actuales disponibles en Internet que permiten identificar sitios web potencialmente maliciosos y que cumplen con las especificaciones de diseño mencionadas anteriormente:

PhishTank

PhishTank es un sitio web mantenido por una comunidad libre en donde sus miembros pueden solicitar el registro de sitios web potencialmente maliciosos en términos de suplantación de identidad, verificarlos y rastrearlos, ya que se mantiene una base de datos de libre acceso, la cual puede ser consultada en línea por medio de APIs publicados o bien, descargar una copia local del repositorio completo (PhishTank, 2017a). Este sitio es mantenido por OpenDNS, el cual ofrece servicios gratuitos de DNS a nivel mundial que son altamente utilizados en el mercado.

Para poder hacer uso de los servicios web publicados se requiere la creación de una cuenta de usuario de manera gratuita, ante la cual se permite hacer uso gratuito pero restringido de consultas diarias al API para obtener información sobre los sitios almacenados dentro de su base de datos. Por otro lado, se puede descargar la base de datos en el caso de requerir un gran número de consultas, la cual se puede obtener en distintos formatos tales como JSON, XML, CSV o PHP serializado.

Según estadísticas obtenidas del sitio web oficial, al 3 de octubre del 2017, la base de datos cuenta con 2,193,299 sitios validados como maliciosos y 5,227,927 clasificados como potencialmente maliciosos, además se tiene como media para el mes de mayo del 2017 (el cual corresponde al último registro de estadísticas publicado) 13 horas y 28 minutos para la validación de cada uno de los sitios que han sido postulados por la comunidad como potencialmente dañinos (PhishTank, 2017b).

Mantiene un API que puede ser consumido por medio de solicitudes HTTP REST.

Google Safe Browsing

Este servicio mantenido por la compañía Google permite mostrar mensajes de advertencia a los usuarios finales sobre sitios web maliciosos que se encuentran catalogados en su repositorio de datos dentro de algunas de las siguientes tres categorías: malware, ingeniería social y software no deseado (Google, 2017a).

Los usuarios finales obtienen este servicio de manera gratuita mientras navegan por Internet mediante un explorador web, sin embargo cuenta con servicios para desarrolladores de software que les permite utilizar un API publicado por la compañía para realizar consultar sobre sitios o páginas web y determinar si estos han sido clasificados como dañinos.

Dentro de la documentación disponible, se especifica que el servicio para desarrolladores es ofrecido en dos modalidades, una modalidad de consulta en línea y otra en la que por medio del API se permite descargar una copia de la base de datos en donde cada hipervínculo registrado y sus atributos de clasificación son representados por medio de hashes; posteriormente las consultas puedes ser realizadas sobre la copia local de la base de datos (Google, 2017b).

Para poder utilizar este servicio, el desarrollador debe registrar una cuenta de usuario en los servicios en la nube de Google y habilitar el uso del API de Google Safe Browsing. Este proceso requiere también la generación de una llave que es utilizada en cada una de las llamadas a las funciones del API para poder realizar consultar y rastrear el consumo del servicio a la

cuenta respectiva. De manera estándar el servicio permite el consumo de un máximo de 10 000 consultas al API de manera diaria y un límite de 3 000 consultas por segundo.

Mantiene un API que puede ser consumido por medio de solicitudes HTTP REST.

4.2.2. Selección y diseño de servicio discreto

A continuación, se presenta una tabla resumen de las características ofrecidas por las opciones descritas en el apartado anterior:

Tópico	Google Safe Browsing	PhishTank
Costo del servicio	Gratuito	Gratuito
Confianza en la veracidad de los datos mantenidos	Mantenido por Google con base en retroalimentación de usuarios	Mantenido por comunidad y retroalimentación de usuarios
Mecanismo de acceso a base de datos	Consultas HTTP REST al API y opción de descarga de una copia local	Consultas HTTP REST al API y opción de descarga de una copia local
Categorías de sitios registrados	Mantiene tres categorías: Ingeniería social, malware y software no deseado	Sitios asociados a suplantación de identidad
Límites en el consumo del API	10 000 consultas diarias; 3 000 consultas máximas por segundo	1 000 consultas diarias

Tabla 4: Tabla comparativa entre Google Safe Browsing y PhishTank

Con base en las opciones identificadas, se ha definido el servicio de Google Safe Browsing como mecanismo discreto a utilizar dentro de la solución desarrollada. Se describe a continuación los elementos considerados en la toma de decisión:

- Posee una base de datos más amplia en términos de cantidad de registros y categorías de los mismos.
- Mantenida y respaldada por Google, lo cual genera una mayor confianza en las prácticas asociadas al mantenimiento del repositorio y calidad de la información registrada.
- Amplia documentación del API, soporte y actualización del mismo.
- Consultas realizadas a la infraestructura en la nube de Google, la cual proporciona alta disponibilidad en sus servicios.

En términos del diseño del servicio, se utilizará la modalidad de consultas en línea mediante el API por encima de la opción del mantenimiento de una copia local de la base de

datos, esto responde a un tema de no aumentar el nivel de complejidad en términos de diseño e implementación de la solución de software desarrollada según el alcance ya establecido. Por otro lado, se hará uso de la versión del API más reciente de Google, que al mes de setiembre del 2017 corresponde a la versión 4.0 de su API (Google, 2017b).

4.3. Diseño e implementación del servicio probabilístico

El servicio probabilístico incorporado dentro de la arquitectura asume un rol de segundo nivel de consulta para la clasificación de hipervínculos. Como se mencionó en la sección del diseño del servicio discreto, el flujo normal de la solución desarrollada acude primero a este para identificar si un hipervínculo se encuentra presente en la base de datos de Google Safe Browsing. En caso que un hipervínculo no haya sido encontrado, este será validado por parte del servicio probabilístico en donde se obtendrá una respuesta predictiva de si es o no potencialmente malicioso.

Es importante enfatizar que la predicción realizada por el modelo predictivo no garantiza un resultado cien por ciento confiable, es decir, existe la posibilidad de que se generen falsos positivos o falsos negativos. Un falso positivo implica la clasificación de un hipervínculo como malicioso cuando en realidad no lo es, mientras que un falso negativo implica la no clasificación de un hipervínculo como malicioso cuando sí lo es. El modelo de entrenamiento a generar es capaz de detectar, mediante el set de entrenamiento, los elementos que han sido incorrectamente clasificados.

4.3.1. Selección del modelo predictivo

Con base en la investigación realizada sobre los algoritmos estocásticos utilizados actualmente para la formulación de modelos predictivos, así como con los otros trabajos realizados relacionados a la problemática de la presente investigación como los presentados por Darling (2015), Sahoo Liu et al. (2017) y principalmente por Lawrence et al. (2011), se ha considerado el uso de un algoritmo de regresión logística como base para la construcción del modelo.

A continuación se mencionan las ventajas asociadas a la utilización de este algoritmo:

- Es adecuado para la resolución de problemas de clasificación binaria, lo cual se ajusta perfectamente a las necesidades de la solución ya que se requiere que esta pueda clasificar a un hipervínculo en una de dos categorías: malicioso o no malicioso.
- Existen diversas implementaciones de algoritmos de regresión logística en lenguajes de programación de orientación estadística, por ejemplo, el lenguaje R lo incorpora dentro de su librería estándar. Esto es importante ya que, como se especificó en el alcance de la presente investigación, se tenía ideado utilizar un mecanismo ya existente e implementado para resolver el problema de clasificación predictiva.
- Estudios encontrados en la academia, Pohar et al. (2004), Salazar et al. (2012) y Al-Jazzar (2012), indican una eficiencia y grado de confiabilidad muy similar entre métodos de clasificación tales como Análisis del Discriminante Lineal (LDA), regresión logística y el método geométrico de Máquina de Vectores de Soporte (SVM). Regresión logística es apto cuando se utiliza un número muy pequeño de categorías, por ejemplo 2 o 3.

4.3.2. Diseño de los datos de aprendizaje y entrenamiento

Recolección de datos

Para la obtención de los datos de aprendizaje y entrenamiento se requirió la búsqueda y recolección de fuentes de datos que lograran proveer un conjunto de datos de sitios web calificados como maliciosos, así como un conjunto de datos de sitios benignos o no maliciosos. En lo que respecta a los datos de sitios web no maliciosos, se descargó la versión 4.1 de un archivo de valores separados por coma preprocesado por Sood (2016), el cual es de libre descarga y se encuentra mantenido en un sitio web administrado por la Universidad de Harvard.

Los datos preprocesados en este archivo fueron a su vez obtenidos de un listado de directorio que era mantenido hasta el 17 de marzo del 2017, según el foro Resource Zone (2017), por la comunidad denominada DMOZ. Dicho listado proporcionaba la categorización de miles sitios web no maliciosos más grande que se podía encontrar de manera pública en Internet.

Por otro lado, la obtención de un listado de sitios web maliciosos se descargaron los listados de sitios web que eran mantenidos por la comunidad de urlblacklist.com, la cual cerró sus operaciones el 25 de julio del 2017.

Selección de datos

A partir de los conjuntos de datos obtenidos de los sitios mencionados con anterioridad, se trabajó con un conjunto inicial de datos conformado por cuatro mil hipervínculos maliciosos (80%) y mil hipervínculos no maliciosos (20%).

La selección de los registros de estas dos categorías se realizó de manera aleatoria utilizando Microsoft Excel, buscando únicamente recopilar la proporción de registros indicada anteriormente. El producto final de esto consistió en un archivo de valores separados por coma con dos columnas que incluían el hipervínculo o URL, y una columna con valores de unos y ceros respectivamente; en esta última columna, el valor uno representaba un sitio malicioso mientras que el valor cero un sitio no malicioso.

La segmentación de este conjunto de datos inicial se realizó de la siguiente manera:

- Datos de entrenamiento (50%)
- Datos de validación (25%)
- Datos de pruebas (25%)

En el *Anexo I* se pueden consultar los enlaces directos al repositorio de la solución desarrollada para consultar las fuentes de datos de sitios maliciosos y no maliciosos, así como el conjunto de datos inicial utilizado para el entrenamiento del algoritmo de regresión logística.

4.3.3. Preprocesamiento y transformación de datos

Generación de diccionario de tokens

Como se mencionó en la subsección anterior, el proceso de recolección y selección de datos culminó con la generación de un conjunto de datos sin procesar, que incluye una columna que especifica con valores de unos y ceros si un hipervínculo es malicioso o no, y una segunda columna con el URL completo de dicho hipervínculo. Las siguientes figuras presentan muestras de dicho conjunto de datos:

y	url
0	www.gemeasurement.com
0	www.delasletrashotel.com
0	www.penguin.cz
0	www.cs-interiors.co.uk
0	www.sheffschefs.co.uk
0	www.gomungoseo.co.uk
0	www.colarulloauto.it
0	www.knight-group.co.uk
0	www.missionvet.com

Figura 6. Muestra #1 del conjunto de datos inicial

y	url
1	luther95.net/aslc-gmx
1	damnn.com/1/teen
1	angelfire.com/pa2/passover/
1	topsexphotos.com/top-sex-photos
1	atschool.eduweb.co.uk/stjosephscgs/

Figura 7. Muestra #2 del conjunto de datos inicial

El siguiente paso consistió en la generación de un diccionario de tokens, en donde cada token corresponde a un n-grama obtenido a partir de la segmentación de cada uno de los hipervínculos maliciosos en un número de letras especificado, a partir de ellos se obtuvo un ranking de los n-gramas que más veces fueron encontrados durante el análisis.

A continuación, se detallan cada uno de los pasos seguidos dentro de este proceso:

1. Selección del 50% de los registros almacenados dentro del conjunto de datos inicial. Lo anterior es equivalente al conjunto de datos de entrenamiento.
2. Selección de todos los registros asociados a sitios web maliciosos y combinación del hipervínculo de todos ellos en una única cadena de texto.
3. Ejecución de proceso de “tokenización”:
    a. Eliminación de caracteres no alfanuméricos presentes, tales como puntos “.”, guiones “-“, rayas abajo “_” barras inclinadas “/”, etc.
    b. Exclusión de palabras o “tokens” que comúnmente son encontrados hipervínculos tales como “com”, “net”, “org”, “www”, “html”, etc.
    c. Separación de la cadena de texto en n-gramas de un número de letras especificado.

4. Generación de una matriz de frecuencias sobre cada uno de los tokens obtenidos del proceso anterior. La obtención de las frecuencias se obtiene mediante el uso de una función de estándar disponible en el lenguaje R.
5. Creación del diccionario de tokens, el cual consiste en un archivo de valores separados por comas, en donde se registraron todos los tokens o palabras con una frecuencia mayor a 30.

La generación del diccionario solo requiere ser ejecutada una vez, con base en la selección del número de letras de los n-gramas sobre los cuales se desea trabajar como datos de entrada para el entrenamiento del modelo predictivo. La siguiente figura presenta una muestra de las primeras entradas del diccionario de datos generado:

	x
sex	370
umb	257
mbl	252
blr	231
tum	231
pos	221
ost	218
por	205
fre	161

Figura 8. Muestra del diccionario de datos generado con trigramas

La utilización de la técnica de "tokenización" a partir de n-gramas se basó en los trabajos de Cavnar & Trenkle (1994) y Abdallah &de La Iglesia (2015), en donde presentan las ventajas y formas de implementación de estas técnicas para la categorización de texto, en esto caso específicamente, de hipervínculos.

Generación de conjuntos de datos

Posterior a la generación del diccionario de tokens, el siguiente paso consistió en la creación de los conjuntos de datos que se utilizarán para el entrenamiento, validación y realización de pruebas que serán utilizados como entrada para el modelo predictivo. Esta segmentación requiere también que estos distintos conjuntos de datos se encuentren normalizados, ya que el algoritmo de regresión logística requiere que los datos de entrada le sean suministrados de esta manera para poder funcionar de manera correcta.

Para ello se realizaron los siguientes pasos:

1. Se cargan en memoria tanto el conjunto de datos inicial como cada uno de los tokens presentes en el diccionario.
2. Se genera una matriz de variables en donde cada uno de estas corresponde a la presencia o no de cada uno de los tokens como parte de cada uno de los hipervínculos que fueron obtenidos del conjunto de datos inicial. Cada variable es representada por una columna y contiene valores de unos y ceros, mostrando con el número uno la presencia de un token como parte del hipervínculo que representa una fila dentro de la matriz, o bien un número cero si no fue encontrado.
3. Esta matriz es segmentada en los tres conjuntos de datos indicados con anterioridad, a saber: datos de entrenamiento, datos de validación y datos de prueba con base en las proporciones ya mencionadas. Cada una de estas matrices se almacena como un archivo de valores separados por coma.

Con estos resultados obtenidos se finaliza la etapa de pre procesamiento del conjunto de datos inicial y se encuentran listos para ser utilizados por el algoritmo de regresión logística como parte del modelo predictivo.

4.3.4. Entrenamiento y validación del modelo predictivo

Creación del modelo predictivo

Esta subsección se dirige a explicar el proceso llevado a cabo como parte del entrenamiento y validación del modelo predictivo, el cual es sumamente sencillo a partir de la utilización de las librerías disponibles en el lenguaje R para la construcción de dicho modelo:

1. Se carga en memoria los datos de entrenamiento.
2. Se inicializa la función "*glm" (modelo lineal generalizado, Quick R (2017))* que permite construir el modelo de regresión mediante una librería existente. Esta función recibe como entrada los datos cargados, así como el tipo de algoritmo a utilizar, en este caso el tipo binomial implica el uso de regresión logística.
3. Internamente el algoritmo analizará los hipervínculos incluidos dentro de los datos de prueba y los resultados de unos y ceros obtenidos a partir de las variables para realizar el cálculo de los coeficientes de regresión. Esto producirá los

coeficientes logísticos a utilizar y que permiten realizar las labores de clasificación.

4. Se hace uso de la función "*predict*" del lenguaje R. Esta función recibe como entradas la función predictiva generada anteriormente y el conjunto de datos de entrenamiento. El objetivo de esto es realizar una comparación de los resultados de la clasificación obtenidos a partir de la función predictiva contra los valores reales de clasificación con los que fue generado el conjunto de datos inicial, y así determinar la precisión o confiabilidad del algoritmo a la hora de predecir los resultados.
5. Los resultados son almacenados en una matriz con valores dentro del rango [0,1]. Para cada uno de los resultados se realiza un redondeo, en donde si el resultado es mayor o igual al valor 0.5, es transformado en un número uno, es decir, corresponde a un hipervínculo malicioso. Caso contrario, si el resultado es menor al valor 0.5, es transformado a un número cero, que representa un hipervínculo no malicioso.

Validación del modelo predictivo

La validación del modelo implica un proceso simple muy similar al utilizado para la creación del mismo:

1. Se carga en memoria los datos de validación.
2. Se hace uso de la función "*predict*" del lenguaje R. Esta función recibe como entradas la función predictiva generada anteriormente y el conjunto de datos de validación. La importancia de realizar este ejercicio consiste en validar la precisión de la predicción del modelo contra datos que no formaron parte del entrenamiento de la función de regresión logística y así determinar si esta función resultante no ha sido "sobre ajustada" a los datos de entrenamiento a la hora de realizar clasificaciones predictivas.
3. Los resultados son almacenados en una matriz con valores dentro del rango [0,1]. Para cada uno de los resultados se realiza un redondeo, en donde si el resultado es mayor o igual al valor 0.5, es transformado en un número uno, es decir, corresponde a un hipervínculo malicioso. Caso contrario, si el resultado es menor

al valor 0.5, es transformado a un número cero, que representa un hipervínculo no malicioso.

4.3.5. Implementación del servicio

La implementación del servicio probabilístico basado en scripts desarrollado en el lenguaje de programación "R" se llevó a cabo mediante el servicio en la nube de Microsoft denominado *"Machine Learning Studio"*, el cual provee una interface gráfica para que los usuarios puedan generar un modelo predictivo, o bien, cargar un modelo existente y utilizarlo para realizar las predicciones. La adquisición de este servicio no requirió costo adicional ya que se utilizó una membresía gratuita cuyas especificaciones técnicas que forman parte de la oferta resultan suficientes para la puesta en marcha del servicio.

La siguiente figura muestra de manera general los pasos llevados a cabo para implementar el servicio probabilístico en esta plataforma:

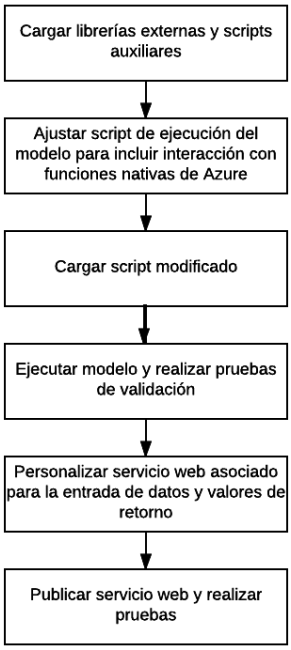

Figura 9. Proceso de implementación del servicio probabilístico

1. El primer paso consistió en cargar los scripts desarrollados en "R" que son utilizados de manera auxiliar por parte del script de ejecución principal del modelo. Estos auxiliares a su vez hacen uso de librerías externas que no se

encuentran cargadas de manera nativa en el servicio de Azure, por lo que estos paquetes debieron ser importados.

2. El script de ejecución del modelo requiere incorporar ciertas llamadas a funciones nativas del servicio de Azure para recibir los datos de entrada y generar los valores de salida, por lo que este fue ajustado en estos términos.
3. Teniendo el script ajustado se procedió a cargar dentro de la solución, lo cual se realiza de manera sencilla por medio de la interface gráfica de la plataforma.
4. Una vez teniendo cargado en la plataforma de Azure un equivalente de la solución que se desarrolló de manera local, se procedió a ejecutar el modelo con el conjunto de datos de entrenamiento y validación para asegurase de que el servicio esté reproduciendo los mismos resultados que se obtuvieron de manera local.
5. El siguiente paso consistió en la configuración del servicio web que permite la comunicación entre el modelo predictivo y sistemas externos, en este caso el servidor de clasificación. Dicha configuración incluye la parametrización de los datos de entrada así como de los datos de salida que produce el modelo.
6. Por último se realizó la publicación del servicio web y se efectuaron pruebas de comunicación y validación de resultados obtenidos al transmitirle información sobre hipervínculos maliciosos y no maliciosos.

4.4. Arquitectura de la solución

La presente sección describe el diseño arquitectural de la extensión de navegador desarrollada, para lo cual se ha tomado como base el modelo de arquitectura denominado “Vistas 4+1”, publicado por Kutchen (1995).

Este modelo arquitectural presenta la estructura y organización de un sistema de software en una serie de vistas dirigidas a distintas audiencias, a saber:

- Vista lógica: Representa una abstracción de los requerimientos del sistema comúnmente representada en objetos o clases de objetos.
- Vista física: Involucra los requerimientos no funcionales del sistema, mostrando su ubicación física e incorporados elementos tales como redes de datos, servidores, nodos de red, etc.

- Vista de implementación: Se concentra en la organización modular del sistema de software en términos de componentes, subsistemas, interfaces, paquetes y librerías de software.
- Vista de proceso: Muestra aspectos dinámicos del sistema en tiempo de ejecución, es decir, el comportamiento de los procesos internos y la comunicación entre los mismos.
- Escenarios de uso: Representan una abstracción de los requerimientos más importantes y muestran la interacción entre el usuario y/o elementos del sistema ante un flujo particular de inicio a fin para lograr cumplir con uno o varios de los requerimientos. Corresponde al "+1" del modelo, ya que complementa a los anteriores.

La selección de este modelo se debe a su capacidad de representar la arquitectura de un sistema de software a partir de vistas fundamentales que permiten reflejar aspectos clave de estructura física y lógica, así como los flujos de información; por otro lado es lo suficientemente genérico para definir la notación y mecanismos de diseño a utilizar.

La notación seleccionada para representar las vistas arquitectónicas corresponde al lenguaje UML 2.0, debido a su amplio uso en la industria y academia.

4.4.1. Arquitectura de alto nivel

La arquitectura de alto nivel presentada a continuación se dirige a cubrir las vistas de implementación y física del modelo "4 +1" en un único diagrama, que representan los principales componentes y nodos físicos involucrados en la solución desarrollada. En términos de la notación de UML 2.0, se procedió a unificar los elementos de un diagrama de componentes y un diagrama de implementación para facilitar una vista consolidada de alto nivel:

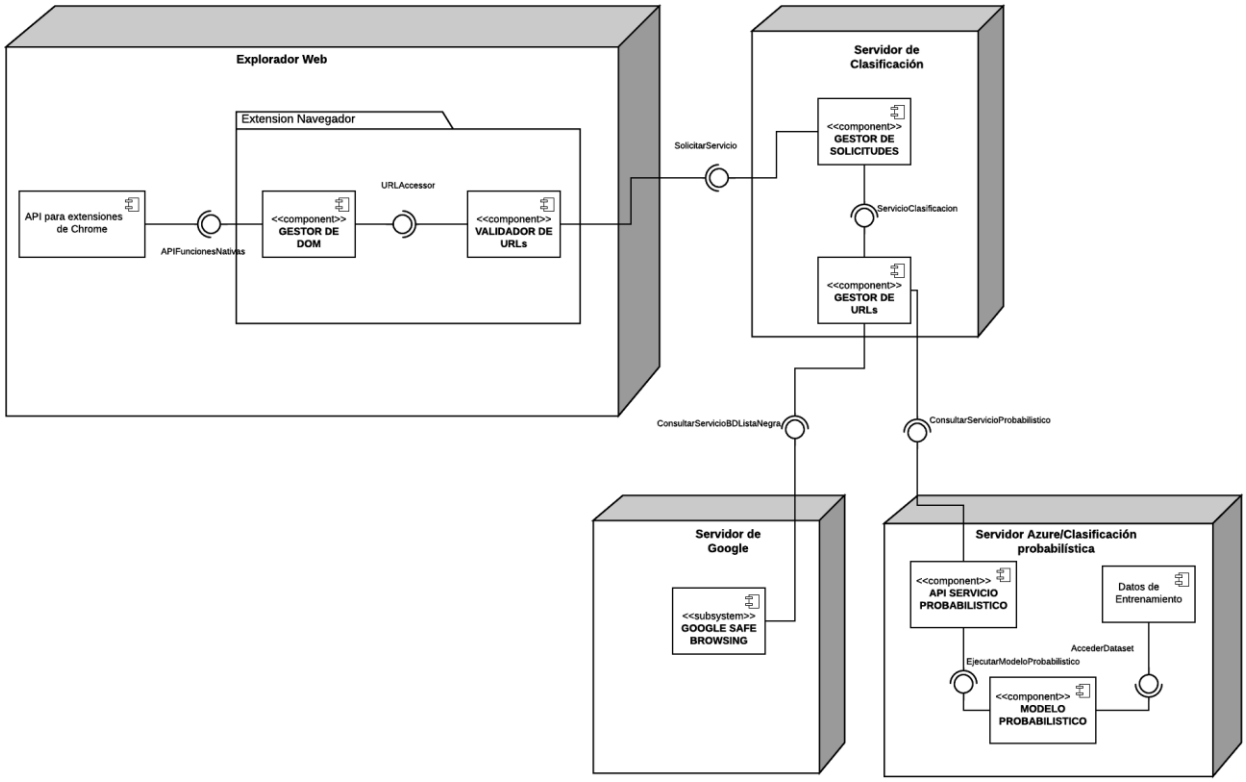

Figura 10. Arquitectura de alto nivel de la solución

Descripción de la arquitectura lógica (componentes)

La solución desarrollada se encuentra modularizada de la siguiente manera:

- API para extensiones de Google Chrome: Corresponde al API estándar para desarrolladores soportado por parte de Google. Este API permite la interacción con los componentes internos del navegador.
- Gestor de DOM: Este componente permite la manipulación del DOM de cada página web solicitada por el usuario final. Cumple dos objetivos principales:
    - Leer los elementos del DOM para obtener el conjunto de hipervínculos presentes en imágenes, scripts externos y enlaces web.
    - Manipular los atributos de los elementos que contienen hipervínculos de tal manera que puedan inyectar instrucciones CSS y resaltar los elementos maliciosos presentes.

- Validador de URLs: Punto centralizado de comunicación entre el cliente y la infraestructura de clasificación. Recibe una estructura de datos que contiene todos los hipervínculos presentes en una determinada página web y los envía al componente de gestión de solicitudes del lado del servidor. Por otro lado, se encarga de recibir las respuestas de clasificación y transmitirlas al Gestor de DOM.
- Gestor de Solicitudes: Componente que recibe solicitudes de clasificación por parte de la extensión de navegador y envía las respuestas procesadas por parte del Gestor de URLs. Punto de comunicación del lado de la infraestructura de clasificación.
- Gestor de URLs: Este componente se encarga de procesar las solicitudes de tal manera que cumplan los requisitos de los APIs de ambos servicios, y recibe procesa las respuestas de vuelta para ser transmitidas a la extensión de navegador. Por otro lado, determina si uno o múltiples hipervínculos recibidos se encuentran minificados, ante lo cual los categoriza como *"No analizados"* y los excluye de las consultas realizadas a los servicios de clasificación.
- Servicio de Google Safe Browsing: Servicio ofrecido por parte de Google que permite consultar en su base de datos sitios web que clasifica como maliciosos, y que es alojado en su propia infraestructura en la nube. Es consumible mediante APIs y ofrece dos modalidades de uso: 1) Envío de solicitudes de clasificación por medio del API, para lo cual se restringe a una cuota de uso diaria y mensual. 2) Uso del API para descargar una base de datos local de hashes de sitios maliciosos.
- API Servicio Probabilístico: Punto de contacto que permite consumir el servicio de clasificación probabilística de hipervínculos maliciosos. Este recibe el listado de hipervínculos y solicita la ejecución del modelo predictivo para categorizarlos en maliciosos o no maliciosos.
- Modelo Probabilístico: Comprende la funcionalidad de predicción de hipervínculos y los clasifica en maliciosos o no maliciosos. Para ello utiliza un modelo basado en regresión logística basado en algoritmos estándar del lenguaje

de programación R, y previamente entrenado con fuentes de datos de sitios dañinos y benignos.

Decisiones de diseño de la arquitectura lógica

- Se ha diseñado una arquitectura estilo cliente-servidor en la cual se distribuyen las funciones generales de la siguiente manera:
    - Cliente: Representado por el navegador Google Chrome y la extensión desarrollada, se encarga de identificar cuando una página web es cargada en el explorador web, procesar el DOM asociado, realizar consultas al servicio de clasificación respecto a los hipervínculos identificados, procesar las respuestas e inyectar código HTML y CSS en los elementos respectivos del DOM para resaltar hipervínculos detectados como maliciosos.
    - Servidor: Representada por el servidor de clasificación, el servicio discreto (Google Safe Browsing) y probabilístico. Recibe solicitudes por parte de la extensión de navegador y realiza las consultas respectivas a los servicios anteriormente mencionados con el fin de detectar hipervínculos de sitios web potencialmente maliciosos y categorizarlos respectivamente en las respuestas enviadas hacia el cliente.
- Se ha seleccionado el servicio de Google Safe Browsing como método discreto para la identificación de hipervínculos maliciosos. Esto debido a que Google posee la base de datos más extensa de sitios web potencialmente dañinos, alimentada a su vez por usuarios que diariamente reportan este tipo de sitios. Este repositorio es actualizado también diariamente por el proveedor.
- Para reducir complejidad en el mecanismo de consulta de la base de datos de Google Safe Browsing, se ha optado por utilizar el método de consulta en línea mediante el consumo del API proporcionado por Google en vez de la utilización de una base de datos local de hashes de los sitios maliciosos. Esto limita el número de llamadas de API que se pueden realizar de manera diaria y mensual a partir de una cuenta registrada en Google para ello.
- El servicio de clasificación probabilística está implementado sobre un servidor virtual R soportado por la plataforma de Azure. Este servicio recibe de manera

individual las solicitudes para hipervínculo que se requiere clasificar y retorna como respuesta la clasificación respectiva.

- Tecnologías de desarrollo:
  - Node.js: Para el manejo de la funcionalidad del servidor de clasificación se ha seleccionado esta tecnología, ya que cuenta con librerías que permiten habilitar un programa de servidor sin requerir la escritura extensa de código fuente para alcanzar dicho objetivo. Además utiliza el mismo lenguaje de programación (javascript) que se utiliza en la programación de la extensión de navegador del lado del cliente.
  - Lenguaje R: La principal opción disponible en el mercado junto a Matlab, con la particularidad de no ser propietaria. Cuenta con funciones estadísticas predictivas incorporadas dentro de su librería estándar, lo cual facilita el proceso de implementación del método probabilístico.
- Se han seleccionado las siguientes librerías de software externas:
  - jQuery: Utilizada en el lado del cliente de la arquitectura para la manipulación de elementos del DOM, así como para la interfaz gráfica de la extensión del navegador.
  - Express: Para un manejo simplificado de la recepción y envío de solicitudes HTTP en la lógica del servidor.
  - Json-parser: Librería que permite simplificar la manipulación de las estructuras de datos de tipo JSON que son recibidas por el servidor.

Descripción de la arquitectura física

La solución desarrollada se encuentra distribuida físicamente de la siguiente manera:

- Explorador web: Mantiene la funcionalidad de la solución alojada en el lado del cliente representada por la extensión de navegador. Requiere la habilitación de la extensión en Google Chrome.
- Servidor de clasificación: Servidor en la nube que aloja la lógica de la aplicación encargada de recibir y procesar solicitudes provenientes de la extensión de navegador y consumir los servicios de Google Safe Browsing y/o del servidor de clasificación probabilística para retornar una respuesta con la clasificación de los hipervínculos recibidos.

- Servidor de Google Safe Browsing: Servidor administrador por Google que ofrece el servicio de Google Safe Browsing, dicho servicio es consumido mediante las APIs publicadas por Google.
- Servidor de clasificación probabilística: Servidor alojado en la nube que publica el servicio de clasificación probabilística de hipervínculos. Utiliza las facilidades del lenguaje de programación R que ya es ofrecido como parte de la plataforma.

Decisiones de diseño de la arquitectura física

- Los servidores que soportan los servicios de clasificación se encuentran alojados sobre la plataforma de Microsoft Azure. Experiencias anteriores en la utilización de máquinas virtuales optimizadas para ejecutar programas desarrollados en lenguaje de programación R, así como la utilización de servicios de plataforma como un servicio, han tenido peso al momento de seleccionar los servicios de Azure. No obstante podrían considerarse otros proveedores de servicios en la nube tales como Google o Amazon.
- Se ha segmentado la funcionalidad encargada de realizar la clasificación (servidor de Google Safe Browsing, servidor de clasificación probabilística) con respecto a la funcionalidad encargada de recibir solicitudes de la extensión de navegador, procesarla, enrutarla y enviar las respuestas. Esto mejora la escalabilidad de la solución así como la agilidad para realizar un cambio de servicios, como por ejemplo, un cambio de proveedor de servicios en la nube para el servicio de clasificación probabilística, etc.

4.4.2. Proceso de clasificación de hipervínculos

El presente apartado se dirige a describir de manera breve el flujo de proceso principal que sigue la solución desarrollada. En este sentido se muestra la vista de proceso de la arquitectura por medio de un diagrama de actividades según la especificación de UML 2.0:

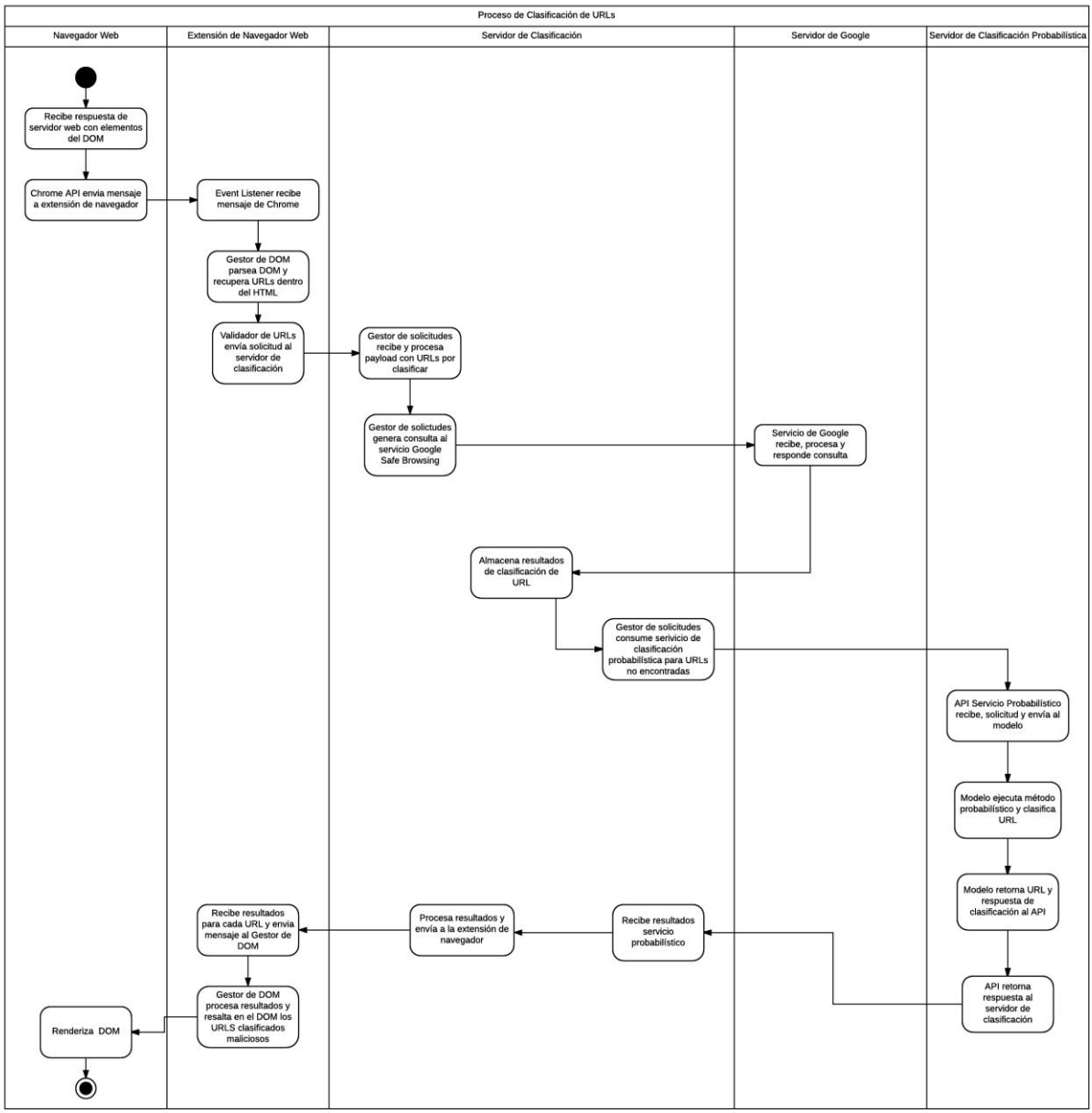

Figura 11. Proceso de clasificación de la solución

Descripción general del flujo de proceso

El proceso inicia cada en el momento en que el usuario final abre una nueva ventana o pestaña dentro del navegador y realiza una petición HTTP para obtener y desplegar los elementos visuales de la página web consultada.

1. El navegador web envía un mensaje a la extensión de navegador, la cual se encuentra configurada para iniciar la ejecución del proceso de análisis siempre y cuando esté habilitada para realizar este escaneo a nivel de configuración.
2. El componente denominado *Gestor de DOM* realiza una revisión del Modelo de Objetos del Documento (DOM) de la página web una vez que todos sus elementos hayan sido cargados en el navegador. Identifica todas las etiquetas HTML de tipo “<a>” (anchor), “<img>” y “<script>” y obtiene el hipervínculo asociado a cada uno de ellos; a su vez les asigna un identificador único por medio de un atributo que es inyectado a nivel de HTML.
3. El *Gestor de DOM* construye una estructura de datos tipo JSON, la cual corresponde a la solicitud de clasificación de hipervínculos y la envía al componente denominado *Gestor de Solicitudes* que reside a nivel del servidor de clasificación.
4. El *Gestor de Solicitudes* recibe la solicitud, la preprocesa y la transmite al componente *Gestor de URLs*.
5. El *Gestor de URLs* se encarga de realizar las consultas respectivas al servicio discreto y al probabilístico. Para ello construye una solicitud de clasificación que será enviada al servicio de Google Safe Browsing por medio de su API publicado.

   Una vez enviada la solicitud, se encarga de recibir y procesar la respuesta, identifica todas los hipervínculos identificados como maliciosos y los mantiene en una estructura de datos interna, luego de esto identifica todos los hipervínculos que no fueron encontrados en la base de datos de Google y los incorpora en una solicitud que es construida y enviada al servicio probabilístico.
6. El *API del servicio probabilístico* recibe la solicitud y la redirige al componente denominado *Modelo Probabilístico.*
7. El *Modelo Probabilístico* ejecuta la lógica de predicción por cada uno de los hipervínculos suministrados y retorna como resultado su clasificación

de 0 o 1, que determina si es considerada no maliciosa o maliciosa respectivamente.

8. El *API del servicio probabilístico* recibe los resultados, los asocia a cada uno de los hipervínculos suministrados y envía las respuestas al *Gestor de URLs.*
9. El *Gestor de URLs* incorpora los resultados junto con los resultados previamente obtenidos del servicio discreto y se los transmite al *Gestor de Solicitudes.*
10. *El Gestor de Solicitudes* prepara la estructura JSON con los resultados y retorna una respuesta al *Validador de URLs* en el lado del cliente.
11. El *Validador de URLs* transmite los resultados al *Gestor de DOM.*
12. Este último componente interpreta los resultados, y para todos aquellos que fueron catalogados como maliciosos por el servicio discreto o probabilístico, consulta el identificador o identificadores asociados a cada hipervínculo dentro del DOM, y por cada uno de ellos inyecta código CSS para resaltar el enlace, imagen, o indicar por medio de una ventana popup en el caso de una etiqueta script, que el hipervínculo asociado ha sido catalogado como malicioso.

4.4.3. Diseño lógico

A continuación se muestra un diagrama de clases que se dirige a presentar la vista lógica de la arquitectura. El objetivo de ello es presentar un segundo nivel de mayor detalle con respecto a la estructura interna de la solución desarrollada tanto a nivel del cliente como del servidor, los servicios incorporados y la utilización de librerías externas:

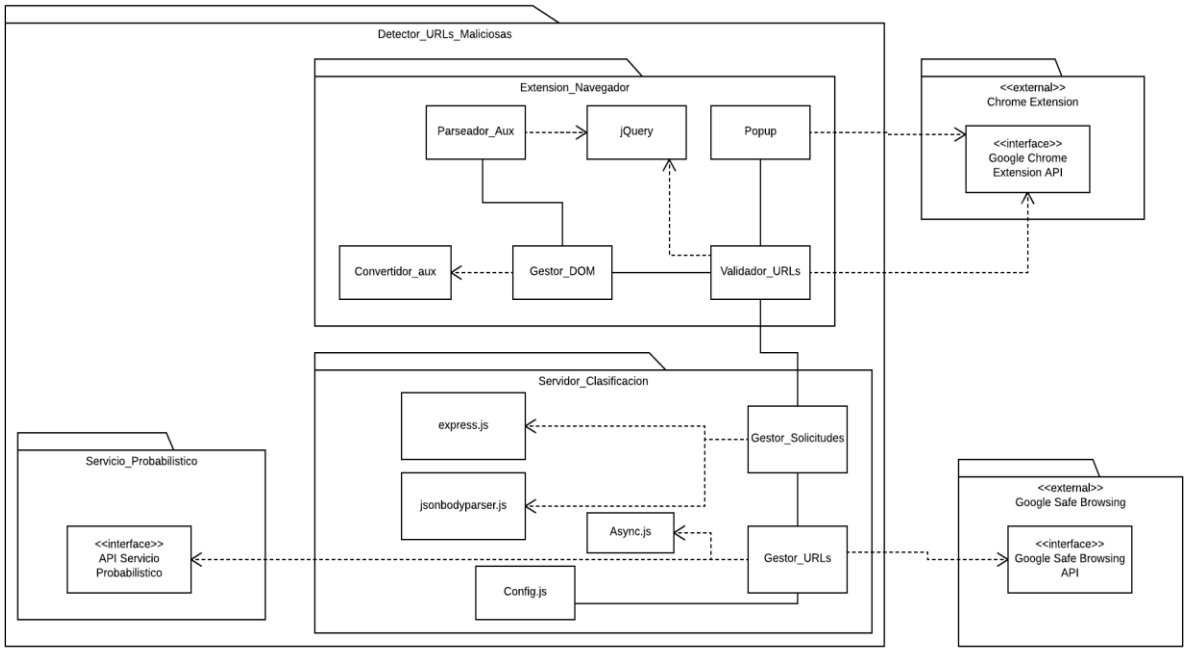

Figura 12. Diseño lógico de la solución

Descripción de la estructura lógica

- Gestor_DOM: Se activa cada vez que el navegador obtiene la respuesta de un servidor web sobre el cual el usuario final haya solicitado una página web en particular. Solicita la ejecución del parseo del DOM de dicha página web y se comunica con el *Validador_URLs* para realizar la consulta al servidor de clasificación. Por otro lado, al recibir la respuesta, se vuelve a comunicar con *Parseador_Aux* para manipular el DOM con base en los resultados de clasificación obtenidos.
- Validador_URLs: Punto de contacto entre el *Gestor_DOM* con el servidor de clasificación. Para ello utiliza el API de Google Chrome para enviar y recibir mensajes con el componente interno y utiliza jQuery para realizar solicitudes HTTP POST y recibir las respuestas respectivas.
- Gestor_Solicitudes: Encargado de establecer un servidor que reciba solicitudes de clasificación por parte del *Validador_URLs* y enviar la respuesta respectiva.
- Gestor_URLs: Se encarga de comunicarse con el servicio discreto y el probabilístico, mantiene una estructura de datos internos que mapea los

resultados obtenidos para cada uno de los hipervínculos enviados como parte de las consultas.

- Popup: Cubre la lógica de la interfaz gráfica de usuario de la extensión de navegador.
- Google Safe Browsing: API externo gestionado por Google para la identificación de hipervínculos maliciosos con respecto a los registros mantenidos en su base de datos.
- Chrome Extension API: API del navegador Google Chrome utilizado para poder realizar operaciones sobre el DOM de cada una de las pestañas activas, así como el envío y recepción de mensajes entre los componentes de la extensión de navegador desarrollada.
- Parseador_Aux: Se encarga de la manipulación del DOM cumpliendo las siguientes funciones escenciales:
    - Parsear el DOM al momento de cargar la página y generar identificadores para cada uno de los elementos que contienen hipervínculos.
    - Manipular los elementos que incluyen hipervínculos maliciosos para resaltarlos y agregar datos informativos por medio de inyección de código HTML y CSS.
- Convertidor_aux: Permite realizar conversiones en dos sentidos entre estructuras de datos de tipo mapas, objetos y JSONs.
- Config.js: Incorpora un listado de proveedores de servicios de minificado de hipervínculos, de tal manera que estos puedan ser utilizado por el *Gestor de URLs* para autoclasificar todo hipervínculo que posea el mismo dominio como *"No clasificado"*.
- Librerías externas
    - Express.js: Utilizado para facilitar el manejo de las operaciones HTTP (POST, GET) del servidor implementado sobre *Gestor_Solicitudes.*
    - jQuery: Esta librería es utilizada en la lógica del lado del cliente para facilitar la manipulación de los elementos del DOM. Además es

utilizada para realizar solicitudes HTTP POST hacia el servidor de clasificación

- Jsonbodyparser.js: Esta librería es utilizada por *Gestor_Solicitudes* para facilitar la manipulación de las estructuras JSON recibidas.
- Async.js: Utilizada para la serialización de solicitudes web que Node.js maneja de manera asíncrona inherentemente en su arquitectura.

5. RESULTADOS

La presente sección se dirige a indicar los resultados obtenidos a partir de la solución desarrollada, esto tanto en términos del grado de precisión que se obtiene del modelo predictivo que compone el servicio probabilístico así como del funcionamiento de la extensión de navegador respecto a la clasificación de hipervínculos y la latencia asociada en la presentación de resultados.

5.1. Resultados de entrenamiento y validación del modelo predictivo

A continuación se muestran los resultados obtenidos a partir del proceso de entrenamiento y validación del modelo predictivo construido. Estos resultados surgen a partir de la selección de trigramas en lo que respecta al mecanismo de clasificación de texto para la creación del diccionario de tokens; la sección 5.2 mostrará la comparación de los resultados con otras combinaciones de n-gramas para validar la selección de trigramas como componente dentro del modelo.

Categoría	Cantidad
Registros clasificados correctamente	**2 075**
Registros clasificados incorrectamente	**425**
Total de registros analizados	**2 500**
Grado de confianza alcanzado	**83%**

Tabla 5: Resultados sobre datos de entrenamiento

La tabla anterior muestra que el grado de precisión del modelo predictivo tomando como base los resultados reales de los datos de entrenamiento es de aproximadamente un 83%, es decir, de cada diez hipervínculos maliciosos suministrados, el modelo logró clasificar aproximadamente 8 de ellos de manera adecuada.

Categoría	Cantidad
Registros clasificados correctamente	**1 025**
Registros clasificados incorrectamente	**225**
Total de registros analizados	**1 250**
Grado de confianza alcanzado	**82%**

Tabla 6: Resultados sobre datos de validación

De manera equivalente, la tabla anterior muestra los resultados de clasificación obtenidos con respecto a los datos de validación. Es importante recordar que estos datos no fueron utilizados en el entrenamiento del modelo predictivo, por lo que se utilizan para presentar el grado de confianza del modelo ante datos completamente nuevos.

Se logra apreciar que se obtuvo aproximadamente un 82% de acierto en las predicciones de clasificación realizadas, la cual es una cifra que presenta muy poca variación con respecto a los resultados obtenidos con base en los datos de entrenamiento.

5.2. Comparación de los resultados con n-gramas de distinto valor

La siguiente tabla muestra un comparativo entre la precisión o grado de confianza del modelo predictivo a partir del uso de n-gramas de distinto valor:

Categoría n-grama	Datos de entrenamiento	Datos de validación
Trigrama	~83%	~82%
Cuatrigrama	~80%	~80%
Pentagrama	~80%	~81%

Tabla 7: Grado de confianza del modelo predictivo según categoría de n-grama

Como se observa en los resultados de la tabla anterior, el uso de trigramas para la generación del diccionario de tokens conlleva a una mayor precisión del modelo predictivo a la hora de ejecutar el proceso de validación. Es importante indicar que el incremento en el valor de cada uno de los n-gramas implica una reducción en la cantidad de registros presentes en el diccionario de tokens, ya que se reduce la cantidad de tokens que cumplen con el criterio de poseer una ocurrencia de treinta veces o más a nivel de los datos de entrenamiento.

5.3. Resultados de clasificación de hipervínculos

La presente sección muestra los resultados obtenidos a nivel funcional por parte de la extensión de navegador desarrollada.

Para ello se generó una plantilla en lenguaje HTML que contiene una tabla con cien registros de hipervínculos tanto maliciosos como no maliciosos; la segunda columna en esta tabla presenta un indicador de la clasificación real de cada una de las entradas. El objetivo de esta plantilla es la realización de una comparación entre la clasificación real con respecto a la categorización dada por la solución desarrollada.

Se incluyeron cien hipervínculos dentro de la plantilla, distribuidos de la siguiente manera:

- 25 hipervínculos no maliciosos obtenidos del ranquin de Alexa.com, sitios web locales, escogidos al azar, etc.
- 25 hipervínculos maliciosos obtenidos del conjunto de datos de prueba.

Listado de hipervínculos

La siguiente tabla muestra una serie de hipervínculos tanto maliciosos como no maliciosos. La segunda columna muestra la categoría de cada enlace mientras que la tercera presenta el resultado obtenido de acuerdo al análisis realizado por la extensión:

Hipervínculo	Categoría
http://images.live.com/videos/thumbnail.aspx	
http://4erta.ru	
http://sfu.ca/community/sexwork.htm	
http://soc.ucsb.edu/sexinfo/	
melen.t35.net/warez	
http://sociosite.org/pornography.php	
http://perso.wanadoo.fr/jean-paul.cecchini	
http://playersodds.com/new-warez	
http://qksrv.net/click-974931-5528027	
http://home.v3.com/error/404/come.to/bah	
http://imbris.net/~fourteenwords	
http://members.ozemail.com.au/~drumbeat	
http://cycad.com/cgi-bin/upstream	
http://ddc.net/ygg	
http://demon.co.uk/natofeur	
http://pornhub.fumblr.com	

Figura 13. Muestra de tabla de validación

La *Figura 13* presenta una muestra de la plantilla desarrollada para la validación de hipervínculos. En este caso particular se visualiza un subconjunto de registros catalogados como maliciosos, y todos aquellos que se encuentran resaltados en rojo han sido los que la extensión de navegador logró clasificar de tal manera.

La siguiente tabla resume las estadísticas obtenidas a partir de evaluación funcional de la solución desarrollada:

Categoría	Resultado
Total de hipervínculos analizados	50
Total de hipervínculos clasificados correctamente	32
Total de falsos positivos	10
Total de falsos negativos	8

Tabla 8. Resumen de resultados

Como se indica en la tabla anterior, la solución logró clasificar de manera correcta un total de treinta y dos hipervínculos, esto implica que categorizó de manera correcta tanto los maliciosos como los no maliciosos. Por otro lado, se presentaron un total de diez falsos positivos, esto implica que clasificó como maliciosos diez hipervínculos que realmente no lo eran; por contraparte ocurrieron ocho falsos negativos, esto quiere decir que no se clasificaron como maliciosos ocho hipervínculos que sí lo eran.

6. CONCLUSIONES

El presente apartado se dirige a documentar las principales conclusiones obtenidas a partir del diseño y desarrollo de la solución propuesta, así como de los resultados obtenidos de ella:

- La solución desarrollada provee un valor agregado a los mecanismos de seguridad vigentes para el uso de Internet por parte de usuarios finales, ya que emplea técnicas de detección de sitios potencialmente maliciosos aprovechando bases de datos existentes tales como Google Safe Browsing, pero a su vez incorpora un mecanismo probabilístico para clasificar hipervínculos potencialmente maliciosos. Esto se ha desarrollado a nivel de academia, pero no se logró identificar una extensión de navegador disponible en el mercado que tome este enfoque.
- Las tecnologías seleccionadas para la implementación de la arquitectura diseñada responden al criterio y habilidades técnicas del autor, no obstante, la arquitectura es lo suficientemente flexible para que pueda ser implementada en muchas otras tecnologías de desarrollo presentes en la industria tales como Java, .NET, PHP, Ruby on Rails, Python, etc. Por otro lado, su lógica puede ser migrada para su implementación en otros navegadores web populares en el mercado, tales como Mozilla Firefox, Internet Explorer, Opera, etc.
- La aplicación de técnicas de aprendizaje automático permiten llevar en la seguridad informática a un nuevo nivel de protección y detección de amenazas presentes en las distintas esferas de las tecnologías de información y comunicación. A su vez, la capacidad de aplicar modelos estadísticos eficientes y optimizados para el reconocimiento de patrones y correlación de información sobre grandes fuentes de datos (Big Data) permitirán la producción de software y hardware más robustos para la detección automática de vulnerabilidades y toma de decisiones técnicas para la mitigación de riesgos potenciales que puedan explotarlas.
- Si bien el porcentaje de fiabilidad de los resultados generados por este servicio no es del cien por ciento, este mecanismo genera mejores resultados que los

que se podría obtener a partir del juicio de valor de un usuario final ante la decisión de determinar si un hipervínculo conduce a un sitio malicioso, lo cual puede incluso ser de un cincuenta por ciento.

- La segmentación del texto de los hipervínculos en trigramas y la generación de un diccionario de datos a partir de sus frecuencias de aparición permite la generación de variables de entradas para el entrenamiento del modelo predictivo. No obstante la comunalidad de estos segmentos puede inducir al modelo a generar falsos positivos, por lo que la elaboración de un conjunto de datos de entrenamiento óptimo representa el principal reto en el área de aprendizaje automático.

7. RECOMENDACIONES

El presente apartado se dirige a documentar las principales recomendaciones en términos de alcance, diseño e implementación de la solución desarrollada, con la intención de enunciar distintos aspectos que pueden mejorar la eficacia, eficiencia, calidad y la utilidad de la solución:

- La eficiencia del servicio puede mejorarse sustancialmente si se utiliza el modo de consulta local ofrecido por Google Safe Browsing, que permite descargar una base datos de los hashes del repositorio mantenido. Esto ya que no requeriría un uso intensivo de llamadas al API de la compañía y por tanto se pueden consumir únicamente solicitudes de actualizaciones de la copia y local, y responder a todas las consultas de la extensión con base en dicha copia.
- El servicio discreto puede mejorarse ampliando las fuentes de datos que se consultan a la hora de buscar si un hipervínculo ha sido clasificado como malicioso. Bajo este contexto, puede incorporarse una consulta paralela al servicio ofrecido por PhishTank para contar con una base de datos de consulta total más amplia. No obstante, es importante tomar en consideración los tiempos de respuesta a la experiencia de uso del usuario final, por lo que se recomienda realizar pruebas de rendimiento al intentar ampliar el número de fuentes de datos ligadas al servicio discreto de la solución.
- La seguridad brindada por la presente solución no debe considerarse como un mecanismo de defensa autosuficiente al navegar por Internet mediante el uso de un explorador web. Se recomienda que el usuario mantenga instalado y actualizado un software antivirus, además de seguir las recomendaciones de seguridad de sitios como SANS Institute, Infosec, CERT, etc. En un entorno corporativo se recomienda mantener programas de entrenamiento y concientización de seguridad para los colaboradores, que incluyan las mejores prácticas para navegar por la web.
- El desempeño de la solución puede mejorarse si se incluye un repositorio intermedio o caché alojado en el servidor de clasificación. Este caché mantendría los resultados de un determinado número de hipervínculos

previamente clasificados, o bien los resultados de los hipervínculos cuya clasificación se ha solicitado en más ocasiones. Esto permitiría responder con mayor rapidez al no tener que realizar solicitudes a los servicios de clasificación ni procesar sus resultados para todos aquellos sitios web de los cuales ya se obtuvo una categorización.

- Es posible que el servicio probabilístico pueda ser optimizado si se utiliza una fuente de datos de mayor tamaño, por ejemplo diez mil o veinte mil registros de hipervínculos para que sean incorporados dentro del entrenamiento y validación del modelo predictivo.
- A pesar de que los datos transmitidos entre la extensión de navegador y el servidor de clasificación no involucran información sensible o confidencial, se podría implementar el uso de certificados digitales para encriptar la comunicación transmitida entre ambos puntos. En caso de que un atacante esté en una posición de *"Hombre en el medio"* (MitM, por sus siglas en inglés), al estar los datos encriptados se mitigaría el riesgo de que este pueda falsificar los resultados de respuesta por parte del servidor de clasificación.
- La reducción de falsos positivos en lo que respecta a la categorización de hipervínculos puede mejorarse si se adiciona una funcionalidad de "Listas blancas", es decir, incorporando un mecanismo de retroalimentación de la solución desarrollada para que pueda contar con un conjunto de hipervínculos que el usuario considere no maliciosos.
- Existen servicios de pago disponibles en Internet que permiten enviar solicitudes de hipervínculos minificados y recibir como respuesta el hipervínculo original en su forma expandida. Se puede incorporar dentro de la arquitectura un consumo de este tipo de servicios de terceros para incluir dentro del alcance funcional el análisis y clasificación de este tipo de hipervínculos.

8. TRABAJOS FUTUROS

La investigación desarrollada abre espacios para posibles trabajos e investigaciones futuras que puedan ser elaborados tomando como base los resultados y la solución generada, de tal manera que permita ampliar su alcance, o bien incorporar nuevas ideas que permitan generar una solución más completa.

Posibles trabajos futuros pueden incluir pero no restringirse a los siguientes tópicos:

- Tratamiento de la accesibilidad: Incorporar elementos que permitan ofrecer facilidades para personas con alguna discapacidad visual. Particularmente en el señalamiento de hipervínculos potencialmente maliciosos.
- Personalización de reglas de señalamiento: Mejoramiento de la funcionalidad de señalización de la solución, de tal manera que el usuario pueda ajustar de manera dinámica la forma de señalización (manipulación del DOM) de acuerdo con sus preferencias.
- Incorporación de distancia Damerau-Leveshtein: Posible incorporación de los algoritmos de medición de distancia Damerau-Leveshtein para la evaluación de una cadena de texto (hipervínculo) durante la evaluación de los tokens.
- Evaluación de técnicas de predicción: Realizar una análisis y evaluación de los resultados de predicción generados por el servicio probabilístico de la solución mediante la aplicación de distintos modelos predictivos (regresión logística, máquinas de vectores de soporte, análisis del discriminante lineal)
- Optimización de esquema de segmentación (“tokenización”): Proponer un esquema alternativo de segmentación de las cadenas de texto de tal manera que se logre optimizar las variables de entradas del modelo predictivo, es decir, el diccionario de tokens.

5. REFRENCIAS

Armstrong, P. (2017). Bloom's Taxonomy. Vanderbilt University. Recuperado el día 14 de enero del 2017 desde el sitio https://cft.vanderbilt.edu/guides-sub-pages/blooms-taxonomy/

Abdallah, T.A., de La Iglesia, B. (2015). URL-Based Web Page Classification: With n-Gram Language Models. *Knowledge Discovery, Knowledge Engineering and Knowledge Management. Communications in Computer and Information Science*. Springer. 553.

Al-Jazzar, M. (2012). A Comparative Study Between Linear Discriminant Analysis and Multinomial Logistic Regression in Classification and Predictive Modeling. Faculty of economics and Administrative Sciences Department of Applied Statistics.

Barrantes, R. (2013). Investigación: Un camino al conocimiento. Un enfoque cualitativo, cuantitativo y mixto. pp. 221-227. San José: EUNED.

Biolchini, J.,Gomes, P., Cruz, A.C., y Horta, G. (2005). Systematic Review in Software Engineering. Technical Report (RT-ES 679/05) Río de Janeiro, Brasil: Systems Engineering and Computer Department COPPE/UFRJ

Brownlee, J. (2013). How to prepare data for machine learning. Recuperado el 06 de julio del 2017 desde el sitio http://machinelearningmastery.com/how-to-prepare-data-for-machine-learning/

Corpus, J. (2017). Best ad blockers and privacy extensions. Recuperado el día 20 de octubre del 2017 desde el sitio https://www.tomsguide.com/us/pictures-story/565-best-adblockers-privacy-extensions.html#s2

Cavnar, W., Trenkle, J. (1994). N-gram Based Text Categorization. *In Proceedings of SDAIR-94, 3rd Annual Symposium on Document Analysis and Information Retrieval*. pp. 161-175

Chrome Web Store (2017). Malware & URL Scanner. Recuperado el 15 de agosto del 2017 desde el sitio https://chrome.google.com/webstore/detail/malware-url-scanner

Darling, M. (2015). A Lexical Approach for Classifying Malicious URLs

Google. (2017a). Google Safe Browsing. Obtenido el día 2 septiembre del 2017 desde el sitio https://safebrowsing.google.com/

Google. (2017b). Google Safe Browsing. Obtenido el día 2 septiembre del 2017 desde el sitio https://developers.google.com/safe-browsing/v4/

Harley, D.; Lee, A. (n.d.). Heuristic Analysis. Detecting Unknown Viruses. Recuperado el 15 de abril del 2017 desde el sitio https://pdfs.semanticscholar.org/2523/86382e2214be46b61526adf236004cb40a70.pdf

Heaton, J. (2013). Artificial Intelligence for Humans, Volume 1: *Fundamental Algorithms. Heaton Research Inc.* Primera Edición, Saint Louis, USA.

IBM (2012). Clustering Models. Recuperado el día 10 de octubre del 2017 desde el sitio https://www.ibm.com/support/knowledgecenter/en/SS3RA7_15.0.0/com.ibm.spss.modeler.help/nodes_clusteringmodels.htm

Instantshift. (2016). 13 Best Wordpress Plugins to Detect Malicious Code in Your Site. Recuperado el día 20 de setiembre del 2017 desde el sitio http://www.instantshift.com/2016/09/09/wordpress-plugins-detect-malicious-code/

James, G., Witten, D., Hastie, T., Tibshirani, R. (2013). An Introduction to Statistical Learning with Applications in R. Springer. Nueva York, EU.

Jabbar, M., Deekshatulu, B.L., Chandra, P. (2013). Classification of Heart Disease Using K-Nearest Neighbor and Genetic Algorithm. *International Conference on Computational Intelligence: Modeling Techniques and Applications (CIMTA) 2013.* pp. 85-94

Kruchten, P. (1995). Architectural Blueprints. The "4+1" View Model of Software Architecture. IEEE Software 12 (6). pp. 45-50.

Ma, J., Lawrence, K.S., Savage S., Voelker, G. (2011). Learning to Detect Malicious URLs. ACM Transactions on Intelligent Systems and Technology. 2 (3). DOI: 10.1145/1961189.1961202.

Metcalf, L. Spring, J. (2013). Everything you Wanted to Know About Blacklists But Were Afraid to Ask. A Case Study. CERT Network Situational Awareness Group. Recuperado el 23 de marzo del 2017 desde el sitio https://insights.sei.cmu.edu/cert/2015/06/domain-blacklist-ecosystem---a-case-study.html

Meyer, D., & Wien, F.T. (2017). Support Vector Machines. The interface to libsvm in package e1071. *R News.*

Mitchell, T. (2006). The Discipline of Machine Learning. Carnegie Mellon University.

Ng, A. (2015). Machine Learning for a London Housing Price Prediction Mobile Application. Department of Computing. Imperial College London.

OWASP (2013). OWASP Top 10 – 2013. The Ten most Critical Web Application Security Risks. Recuperado el 20 de marzo del 2017 desde el sitio https://www.owasp.org/images/f/f8/OWASP_Top_10_-_2013.pdf

PhishTank. (2017a). FAQ. Obtenido el día 2 de octubre del 2017 desde el sitio https://www.phishtank.com/faq.php#whatisphishtank

PhishTank. (2017b). FAQ. Obtenido el día 2 de octubre del 2017 desde el sitio https://www.phishtank.com/stats/2017/05/

Pohar, M., Blas, M., Turk, S. (2004). Comparison of Logistic Regression and Linear Discriminant Analysis: A Simulation Study. Metodolosky zvezki pp 1(1).143-161

Quick R (2017). Generalized Linear Models. Recuperado el día 8 de setiembre del 2017 desde el sitio https://www.statmethods.net/advstats/glm.html

Resource Zone (2017). Important DMOZ Closure. Recuperado el día 10 de octubre del 2017 desde el sitio https://www.resource-zone.com/forum/t/dmoz-closure.53420/

Ripley, B.D. (1996) Pattern Recognition and Neural Networks. Cambridge: Cambridge University Press.

Rooney, B (2014). Internet Icon Dies as Yahoo Directory Goes Dark. Recuperado el 01 de mayo del 2017 desde el sitio http://money.cnn.com/2014/12/29/technology/yahoo-directory/

Sahoo, D., Liu, C., Steven, C.H., (2017). Malicious URL Detection Using Machine Learning.

Sperandei, S. (2014). Understanding Logistic Regression Analysis. Biochemia Medica. 24(1). pp 12-18. DOI: 10.11613/BM.2014.003.

Stevens, J. (2016). Internet Stats and Facts for 2016. Recuperado el 25 de marzo del 2017 desde el sitio https://hostingfacts.com/internet-facts-stats-2016/

Salazar, D., Vélez, J., Salazar, J.C. (2012). Comparison Between SVM and Logistic Regression: Which One is Better to Discriminate? Revista Colombiana de Estadística. 35(2). pp. 223-237.

SANS Institute (2016). Security Awareness Report. Recuperado el 15 de abril del 2017 desde el sitio https://securingthehuman.sans.org/media/resources/STH-SecurityAwarenessReport-2016.pdf

Sood, G. (2016). Parsed DMOZ data. Recuperado el 3 de marzo del 2017 desde el sitio https://dataverse.harvard.edu/dataset.xhtml?persistentId=doi:10.7910/DVN/OMV93V

Symantec (2017). Internet Security Threat Report 2017. Recuperado el 15 de abril del 2017 desde el sitio http://www.symantec.com/content/en/us/enterprise/other_resources/b-istr_main_report_v19_21291018.en-us.pdf

Trend Micro (2017). Big Data and Machine Learning: A Perfect Pair for Cybersecurity? Recuperado el 01 de mayo del 2017 desde el sitio http://blog.trendmicro.com/big-data-and-machine-learning-a-perfect-pair-for-cyber-security/

Virvilis, N. Mylonas, A., Tsali, N., Gritzalis, D. (2015). Security Busters: Web Browser Security vs Rogue Sites. *Computers and Security* .52. pp. 90-105

Wordpress. (2017). Security Antivirus Scanner – CWIS. Recuperado el día 20 de setiembre del 2017 desde el sitio https://wordpress.org/plugins/cwis-antivirus-malware-detected/

6. ANEXOS

Anexo 1: Listado de direcciones del repositorio de datos del trabajo investigativo

Contenido	Enlace
Raíz del repositorio de la investigación	https://github.com/jchaves310/detector_urls_maliciosas
Fuente de datos sitios maliciosos	https://github.com/jchaves310/detector_urls_maliciosas/blob/master/Sitios%20maliciosos.zip
Fuente de datos sitios no maliciosos	https://dataverse.harvard.edu/dataset.xhtml?persistentId=doi:10.7910/DVN/OMV93V
Conjunto de datos inicial	https://github.com/jchaves310/detector_urls_maliciosas/blob/master/Servicio%20probabil%C3%ADstico/Datasets/DataSetRaw.xlsx
Diccionario de datos utilizado	https://github.com/jchaves310/detector_urls_maliciosas/blob/master/Servicio%20probabil%C3%ADstico/Dictionary.csv
Base del servidor de clasificación	https://github.com/jchaves310/detector_urls_maliciosas/tree/master/Servidor%20Clasificacion
Base del servicio probabilístico	https://github.com/jchaves310/detector_urls_maliciosas/tree/master/Servicio%20probabil%C3%ADstico
Base de extensión de navegador	https://github.com/jchaves310/detector_urls_maliciosas/tree/master/Ext

Anexo 2: Revisión sistemática de literatura

La presente sección se dirige a indicar el proceso llevado a cabo para la realización de una revisión sistemática de la literatura referenciada a través de la investigación.

Palabras clave

La siguiente tabla muestra las palabras clave utilizadas, catalogadas según el área de investigación referenciada:

Área	Palabras clave
Aprendizaje automático	Machine Learning Logistic Regression Linear Discriminant Analysis Support Vector Machines Data Classification Models
Clasificación de URLs	Malicious URL Classification URL Categorization
Seguridad en la web y extensiones de navegador	Security Plugins Web Browser Security
Listas negras y URLs maliciosas	URL Blacklists Malicious URL Database
Amenazas y vulnerabilidades	Web Security Web Vulnerabilities
Lenguaje de programación R	R Programming Language
Arquitectura de software	Software Architecture Architecture Frameworks

Lista de fuentes de información preseleccionadas

- Google Scholar
- Open DOAJ
- Google
- CiteSeer X

Cadenas de búsqueda

Área	Cadenas utilizadas
Aprendizaje automático	Machine Learning OR Statistical Learning Logistic Regression AND Machine Learning (Linear Discriminant Analysis OR LDA) AND Machine Learning (Support Vector Machines OR SVM) AND Machine Learning Data Classification Models
Clasificación de URLs	Malicious URL Classification OR Malicious URL Detection URL Classification Plugin
Seguridad en la web y extensiones de navegador	Web Browser Security Plugin OR Web Browser Security Extension
Listas negras y URLs maliciosas	Malicious URL Database OR Malicious URL Blacklists OR Domains Blacklist
Amenazas y vulnerabilidades	Security Vulnerabilities Trends OR Web Security Vulnerabilities Web Security Threats OR Web Browsing Threats
Lenguaje de programación R	R Programming Language OR R Language
Arquitectura de software	Software architectures Architecture frameworks

Criterios de inclusión de literatura

- Fecha de publicación posterior al año 1990
- Tipos de documento: artículo de investigación, reportes técnicos, artículos web de fuentes fidedignas, documento de estándares técnicos.
- Lenguaje del documento: español o inglés
- Propiedad del documento: Documento gratuito o de referencia pública

La siguiente tabla muestra las referencias utilizadas dentro del presente documento junto:

Referencia	Indicador
Abdallah, T.A., de La Iglesia, B. (2015). URL-Based Web Page Classification: With n-Gram Language Models.	[1]
Al-Jazzar, M. (2012). A Comparative Study Between Linear Discriminant Analysis and Multinomial Logistic Regression in Classification and Predictive Modeling	[2]
Brownlee, J. (2013). How to prepare data for machine learning	[3]
Cavnar, W., Trenkle, J. (1994). N-gram Based Text Categorization.	[4]
Chrome Web Store (2017). Malware & URL Scanner	[5]
Darling, M. (2015). A Lexical Approach for Classifying Malicious URLs	[6]
Harley, D., Lee, A. (n.d.). Heuristic Analysis. Detecting Unknown	[7]
Google. (2017a). Google Safe Browsing. Google. (2017b). Google Safe Browsing.	[8]
Heaton, J. (2013). Artificial Intelligence for Humans, Volume 1: Fundamental Algorithms.	[9]
IBM (2012). Clustering Models	[10]
Instantshift. (2016). 13 Best Wordpress Plugins to Detect Malicious Code in Your Site	[11]
James, G., Witten, D., Hastie, T., Tibshirani, R. (2013). An Introduction to Statistical Learning with Applications in R	[12]
Jabbar, M., Deekshatulu, B.L., Chandra, P. (2013). Classification of Heart Disease Using K-Nearest Neighbor and Genetic Algorithm	[13]
Kruchten, P. (1995). Architectural Blueprints. The “4+1” View Model of Software Architecture	[14]
Ma, J., Lawrence, K.S., Savage S., Voelker, G. (2011). Learning to Detect Malicious URLs	[15]
Metcalf, L. Spring, J. (2013). Everything you Wanted to Know About Blacklists But Were Afraid to Ask. A Case Study.	[16]
Meyer, D., & Wien, F.T. (2017). Support Vector Machines. The interface to libsvm in package e1071	[17]
Mitchell, T. (2006). The Discipline of Machine Learning**.**	[18]
Ng, A. (2015). Machine Learning for a London Housing Price Prediction Mobile Application.	[19]
OWASP (2013). OWASP Top 10 – 2013. The Ten most Critical Web Application Security Risks	[20]

Referencia	Indicador
PhishTank. (2017a). FAQ PhishTank. (2017b). FAQ	[21]
Pohar, M., Blas, M., Turk, S. (2004). Comparison of Logistic Regression and Linear Discriminant Analysis: A Simulation Study	[22]
Ripley, B.D. (1996) Pattern Recognition and Neural Networks	[23]
Sperandei, S. (2014). Understanding Logistic Regression Analysis	[24]
Stevens, J. (2016). Internet Stats and Facts for 2016	[25]
Salazar, D., Vélez, J., Salazar, J.C. (2012). Comparison Between SVM and Logistic Regression: Which One is Better to Discriminate?	[26]
SANS Institute (2016). Security Awareness Report	[27]
Symantec (2017). Internet Security Threat Report 2017	[28]
Trend Micro (2017). Big Data and Machine Learning: A Perfect Pair for Cybersecurity?	[29]
Virvilis, N. Mylonas, A., Tsali, N., Gritzalis, D. (2015). Security Busters: Web Browser Security vs Rogue Sites	[30]

Las siguientes tablas muestran la distribución de las referencias seleccionadas según la temática o áreas específicas tratadas:

Temática/Área	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]	[9]	[10]	[11]
Aprendizaje automático		X	X				X		X	X	
Clasificación de URLs	X			X		X					
Seguridad en la web y extensiones de navegador					X						X
Listas negras y URLs maliciosas								X			
Amenazas y vulnerabilidades											
Lenguaje de programación R											
Arquitectura de software											

Temática/Área	[12]	[13]	[14]	[15]	[16]	[17]	[18]	[19]	[20]	[21]	[22]
Aprendizaje automático		X				X	X	X			X
Clasificación de URLs				X							
Seguridad en la web y extensiones de navegador											
Listas negras y URLs maliciosas					X					X	
Amenazas y vulnerabilidades									X		
Lenguaje de programación R	X										
Arquitectura de software			X								

Temática/Área	[23]	[24]	[25]	[26]	[27]	[28]	[29]	[30]
Aprendizaje automático	X	X		X				
Clasificación de URLs								
Seguridad en la web y extensiones de navegador								X
Listas negras y URLs maliciosas								
Amenazas y vulnerabilidades			X		X	X	X	
Lenguaje de programación R								
Arquitectura de software								