FACULTAD DE INGENIERÍA, ARQUITECTURA Y URBANISMO

ESCUELA PROFESIONAL DE INGENIERÍA DE SISTEMAS

TESIS

DETECCIÓN AUTOMÁTICA DE ATAQUES DE INYECCIÓN MEDIANTE APRENDIZAJE AUTOMÁTICO EN BASES DE DATOS NOSQL

PARA OPTAR EL TÍTULO PROFESIONAL DE INGENIERO DE SISTEMAS

Autor (es):

Bach. Paico Chileno Daniel

ORCID: https://orcid.org/0000-0002-5734-3915

Bach. Valdera Contreras Jhon Harry

ORCID: https://orcid.org/0000-0003-4785-9253

Asesor:

Mg. Bravo Ruiz Jaime Arturo

ORCID: https://orcid.org/0000-0003-1929-3969

Línea de Investigación:

Infraestructura, Tecnología y Medio Ambiente

Pimentel – Perú

2024

DETECCIÓN AUTOMÁTICA DE ATAQUES DE INYECCIÓN MEDIANTE APRENDIZAJE AUTOMÁTICO EN BASES DE DATOS NOSQL

Aprobación del jurado

Mg. BRAVO RUIZ JAIME ARTURO

Presidente del Jurado de Tesis

Mg. ALVA ZAPATA JULIANA DEL PILAR

Secretario del Jurado de Tesis

Mg. BANCES SAAVEDRA DAVID ENRIQUE

Vocal del Jurado de Tesis

DECLARACIÓN JURADA DE ORIGINALIDAD

Quien(es) suscribe(imos) la DECLARACIÓN JURADA, soy(somos) Paico Chileno Daniel y Valdera Contreras Jhon Harry del Programa de Estudios de Ingeniería de Sistemas de la Universidad Señor de Sipán S.A.C, declaro (amos) bajo juramento que soy (somos) autor (es) del trabajo titulado:

DETECCIÓN AUTOMÁTICA DE ATAQUES DE INYECCIÓN MEDIANTE APRENDIZAJE AUTOMÁTICO EN BASES DE DATOS NOSQL

El texto de mi trabajo de investigación responde y respeta lo indicado en el Código de Ética del Comité Institucional de Ética en Investigación de la Universidad Señor de Sipan (CIEI USS) conforme a los principios y lineamientos detallados en dicho documento, en relación a las citas y referencias bibliográficas, respetando al derecho de propiedad intelectual, por lo cual informo que la investigación cumple ser inédito, original y autentico.

En virtud de lo antes mencionado, firman:

Paico Chileno Daniel	DNI: 75577344	
Valdera Contreras Jhon Harry	DNI: 46863809	

Pimentel, 09 de marzo de 2024

Dedicatoria

A DIOS Todopoderoso, Padre Eterno, Jesucristo, Virgen María Santísima.

Por ser guía y luz inmensa en nuestro camino, prestarnos la vida para seguir adelante, luchando por cumplir los sueños, permitir llegar a este gran momento tan especial en la vida, de ver cumplido los objetivos, anhelos y muchos sentimientos que nos llenan de satisfacción como profesionales; gracias padre celestial por ser mi guía día a día.

A nuestros Padres

A nuestros padres que desde muy pequeños nos enseñaron los valores y deberes para ser personas buenas y correctas, desde la vida inicial, primaria y secundaria, nos brindaron todo su apoyo incondicional para formarnos como personas con educación y valor, y además por su confianza en seguir dándonos una carrera profesional obteniendo más conocimientos para brindar solucionas para la sociedad, muy grato decir gracias padres queridos por ser nuestro brazo derecho en la vida universitaria y así cumplir los sueños y anhelos que buscamos como seres humanos.

A los Docentes

A todos los docentes de la Escuela Profesional de Ingeniería de Sistemas, el cual en el trascurso de nuestra carrera fueron guiándonos a ser cada día mejores, verificando nuestros errores y apoyándonos siempre, como docentes nos sentimos orgullosos de ser sus alumnos porque gracias a ellos hemos logrado muchas cosas, cosas que nos han permitido tener oportunidades laborales en diferentes empresas, a nuestro asesor Mg. Ing. Bravo Ruiz Jaime Arturo y una dedicatoria muy grata al Mg. Ing. Mejia Cabrera Heber Ivan por su excelente calidad como profesional y docente en el curso de Investigación, el cual nos guio, motivo e hizo posible que el estudio fuera más preciso y significativo.

A la Universidad

A la Universidad Señor de Sipán por formarnos como profesionales de calidad y alta excelencia educativa y así brindar soluciones tecnológicas para las organizaciones y la sociedad.

Agradecimientos

En primer lugar, agradecer a dios todo poderoso por ser guía y darnos la vida para seguir luchando por nuestros seres queridos, a nuestros padres por el constante apoyo para hacer cumplir los objetivos y sueños establecidos, a todos los docentes de la Escuela Académico Profesional de Ingeniería de Sistemas de la Universidad Señor de Sipán por haber compartido todos sus conocimientos durante el desarrollo de nuestra formación profesional y a nuestro asesor el Mg. Ing. Mejia Cabrera Heber Ivan, por su apoyo incondicional durante todo el ciclo en el logro de esta investigación.

Los Autores.

ÍNDICE

Resumen

Las bases de datos NoSQL están orientadas en varias categorías como documentos, clave-valor, columnas, gráficos y memoria, dependiendo del problema a resolver. Los gestores de datos NoSQL se han convertido en los últimos años en la solución ideal para disponer de información en tiempo real, ya que tecnologías como Big Data e Internet de las Cosas están revolucionando la sociedad. Como resultado, empresas como Facebook, Google y Amazon, entre otras, han tenido que crear sus propias bases de datos NoSQL para satisfacer mejor a sus clientes. Debido al uso generalizado de SQL, existe una idea errónea sobre la existencia de ataques de inyección en los almacenes de datos NoSQL. Organizaciones como la fundación The Open Application Security Project (OWASP) sostienen que los ataques de inyección se encuentran entre las amenazas más graves para las aplicaciones web, razón por la cual tanto la plataforma PARSE SERVER como la API de autenticación de Fortnite fueron vulnerables a ataques de inyección causados por dominios mal configurados y el uso de expresiones regulares (regex). En consecuencia, los aspectos de seguridad de NoSQL son cruciales porque, lamentablemente, los diseñadores del sistema no dieron prioridad a las preocupaciones de seguridad al crear el sistema, lo que ha dado lugar a problemas de rendimiento con las medidas defensivas actuales contra los ataques de inyección. Por ello, se presenta un estudio que utiliza el aprendizaje automático en bases de datos NoSQL para detectar ataques de inyección. Para ello, se identificaron los mejores algoritmos de clasificación en base a criterios de precisión y casos de ataques de inyección. Además, se creó un conjunto de datos orientado a un patrón que se adhiere a la estructura de la Java Script Object Notation (JSON), el cual está compuesto por 509 ejemplos, 19 atributos, 2 etiquetas, e inicio y fin de documento, clave, valor, separador y operador. Los resultados demuestran que los algoritmos de clasificación de red neuronal, perceptrón multicapa, árbol de decisión, bosque aleatorio, máquina de vectores soporte y vecinos más cercanos a K cumplieron los objetivos del estudio y clasificaron con éxito los datos en muy poco tiempo con una precisión extremadamente alta (98,7%, 98,7%, 96,3%, 89,6%, 84,8% y 84,8%, respectivamente). Se sugiere que, en un estudio posterior, se cree un conjunto de datos basado en un único patrón orientado a cualquier tipo de base de datos NoSQL, ya que los modelos de clasificación sugeridos en este trabajo son capaces de identificar ataques de inyección en bases de datos NoSQL orientadas a documentos.

PALABARAS CLAVE: Algoritmos de Clasificación, Aprendizaje automático, Ataques de Inyección, Data Set, JSON, NoSQL.

Abstract

NoSQL databases are oriented in various categories such as documents, key-value, columns, graphs and memory, depending on the problem to be solved. NoSQL data managers have become in recent years the ideal solution for real-time information, as technologies such as Big Data and the Internet of Things are revolutionizing society. As a result, companies such as Facebook, Google and Amazon, among others, have had to create their own NoSQL databases to better satisfy their customers. Due to the widespread use of SQL, there is a misconception about the existence of injection attacks on NoSQL data stores. Organizations such as The Open Application Security Project (OWASP) foundation claim that injection attacks are among the most serious threats to web applications, which is why both the PARSE SERVER platform and the Fortnite authentication API were vulnerable to injection attacks caused by misconfigured domains and the use of regular expressions (regex). Consequently, the security aspects of NoSQL are crucial because, unfortunately, the system designers did not prioritize security concerns when creating the system, which has resulted in performance issues with current defensive measures against injection attacks. Therefore, a study using machine learning on NoSQL databases to detect injection attacks is presented. For this purpose, the best classification algorithms were identified based on accuracy criteria and cases of injection attacks. In addition, a pattern-oriented dataset was created that adheres to the Java Script Object Notation (JSON) structure, which is composed of 509 examples, 19 attributes, 2 tags, and document start and end, key, value, separator, and operator. The results show that the neural network, multilayer perceptron, decision tree, random forest, support vector machine, and K-nearest neighbor classification algorithms met the study objectives and successfully classified the data in a very short time with extremely high accuracy (98.7%, 98.7%, 96.3%, 89.6%, 84.8%, and 84.8%, respectively). It is suggested that, in a further study, a dataset based on a single pattern targeting any type of NoSQL database be created, since the classification models suggested in this work are capable of identifying injection attacks in document-oriented NoSQL databases.

KEYWORD: Classification Algorithms, Machine Learning, Injection Attacks, Data Set, JSON, NoSQL.

I. INTRODUCCIÓN

1.1. Realidad Problemática.

Las organizaciones están adoptando los almacenes de datos NoSQL (Not Only SQL) a un ritmo significativamente mayor debido a su suficiencia para gestionar volúmenes masivos de datos con exactitud. Pero hoy en día, la seguridad es una gran preocupación en este campo. [1]

Los almacenes de datos NoSQL presentan deficiencias en diversas áreas de seguridad, incluyendo la integridad, autorización y autenticación. Además, se enfrentan a una variedad de amenazas, como las inyecciones de código y las sucesiones de comandos entre sitios (XSS), que han sido heredadas de banco de datos tradicionales. Estas vulnerabilidades están generando importantes daños en las organizaciones. [1]

Un ataque muy conocido contra las bases de datos relacionales es el denominado inyección SQL, en el que un usuario hábil intenta alterar datos a través de formularios en línea. Este tipo de ataque pretende cambiar los parámetros de entrada de la consulta utilizando cadenas que pretenden engañar a los operadores lógicos para que la consulta se ejecute siempre como verdadera. [1]

Cuando se trata de asaltos de inyección, los motores de datos NoSQL y SQL (motores de datos relacionales) tienen dificultades similares. Este tipo de asaltos son habituales en los motores de datos NoSQL más populares como CouchDB, MongoDB, Hbase y Cassandra. Por ello, los atacantes

han creado y empleado estrategias de vanguardia que se inspiran en métodos empleados por primera vez en SQL con la intención de perjudicar gravemente a las empresas que utilizan NoSQL. [1]

Belmer (2019) informa que después de hablar a un cierto grupo de alumnos de BOOTCAMP ya graduados, y revisar sus proyectos finales de software, se encontró que estos programas eran débiles en términos de seguridad y abiertos a inyecciones NoSQL. Además, se observó que la universidad no ofrecía ninguna instrucción sobre codificación segura.

El sitio web oficial DB ENGINES de las bases de datos más famosas afirma que de los 10 almacenes de datos más populares, el 84% SQL y el 16% son NoSQL. Esta disparidad en popularidad indica, hay mayor probabilidad que los alumnos universitarios estén familiarizados con las inyecciones SQL porque son una amenaza más conocida. Muchos desconocen la existencia de la inyección NoSQL, que altera el código lógico de consulta SQL para atacar estos nuevos motores de datos. [2]

La revista OWASP aborda las 10 inseguridades más significativas en las tecnologías web, los cuales continúan planteando interrogantes sobre cómo mitigar estos riesgos para muchas organizaciones. Las inyecciones son uno de los riesgos prevalentes en las tecnologías web que aún mantienen su relevancia. Según el ranking, desde el año 2033 las inyecciones han ganado aún más prominencia, cuando ocupaban el sexto lugar, hasta llegar al primer lugar en el año 2017. [3]

Un grupo de expertos en seguridad informó a FORNITE de que podrían explotarse ataques de inyección NoSQL contra su API de autenticación,

que se utiliza para iniciar sesión con credenciales de Google y Facebook. Se reveló que los hackers habían accedido y robado los datos del perfil de varios cientos de personas utilizando esta vulnerabilidad en la URL. [4]

Muchas aplicaciones Node.js utilizan la plataforma de código abierto PARSE SERVER. Como esta plataforma hace uso de expresiones regulares (regex), los usuarios pueden buscar en muchos modelos dentro de las aplicaciones. Sin embargo, las expresiones regulares son vulnerables a los ataques de inyección NoSQL, ya que tienen un gran parecido con las consultas en los motores de datos centrados a documentos MongoDB. Dado que esta vulnerabilidad podría proporcionar a un atacante acceso a muchas cuentas de usuario sin autorización, se informó de ella a todos los usuarios y se les dieron recomendaciones sobre cómo prevenir tales ataques. [5]

En 2015 se llevaron a cabo evaluaciones de seguridad para determinar la susceptibilidad de las bases de datos que se empleaban con administradores de motores de datos NoSQL. Se descubrió disponibilidad en el control de acceso efectivo de casi 40.000 bases de datos en línea, lo que permitía vistas y escrituras ilegales. Esto demostró lo sencillo que era para los atacantes robar datos e información teniendo el fácil acceso a los motores de datos. [6]

Encontrar fallos de seguridad en un sistema es el primer paso para iniciar un ataque contra su almacén de datos. Estos fallos pueden deberse a diversas causas, como prácticas ineficaces de gestión de permisos, autenticación descuidada y falta de cifrado de datos por parte de los desarrolladores. [7]

Los desafíos relacionados con la protección de información en almacenes de datos NoSQL son significativos. En la actualidad, la exploración se centra en regenerar la seguridad, por medio del desarrollo de diversos métodos y la implementación de herramientas especializadas para detectar consultas maliciosas.

Los almacenes de datos NoSQL se enfrentan actualmente a una serie de problemas, como la incapacidad de cifrar los datos de forma segura al conectarse al servidor y la susceptibilidad o falta de soporte de comunicación en varios gestores entre el cliente y el nodo. Para garantizar una configuración correcta, es necesario adoptar más medidas. Además, dado que los permisos y la autenticación suelen estar desactivados en las bases de datos NoSQL, ciertos gestores conceden acceso inmediato a las bases de datos por defecto, lo que provoca problemas de autenticación. La visualización de la colección ordenada de información sobre el sistema o los clientes vinculados presenta otro problema, ya que, a pesar de que cada instancia del gestor dispone de una consola HTTP (Hypertext Transfer Protocol), la seguridad no está habilitada en ella. [7]

Se ha criticado duramente la seguridad que ofrecen los gestores de datos NoSQL en relación con la creación de aplicaciones de éxito basadas en estas soluciones, tanto en sistemas informáticos como en entornos del mundo real. [8]

De acuerdo con Olusola, A. y Temidayo, A, para regenerar la técnica de encubrimiento de datos, es crucial adherirse a un conjunto de principios. Estos principios establecen que los datos, manteniendo su estructura original, deben ser modificados de manera que se asemejen lo más posible

a los datos originales, evitando así problemas de lectura por parte del servidor. Es importante recalcar que el enmascaramiento y la mezcla de datos solo deben aplicarse a la información sensible, con el fin de resguardar la defender de los datos.

La aplicación de estos principios ha adquirido una significativa relevancia en el ámbito de la seguridad, ya que una de sus características principales es proporcionar una mayor protección para prevenir la divulgación de información confidencial. Asimismo, estos principios permiten ocultar datos sensibles, como códigos, claves y nombres, contribuyendo así a la invulnerabilidad de la información.

La replicación y la alta consistencia son dos capacidades que prometen los motores de datos NoSQL, garantizando una alta disponibilidad en el almacenaje de datos. A pesar de estas ventajas, son infamemente susceptibles de sufrir ataques de inyección, que permiten el robo de información para obtener ventajas competitivas o personales.

Al utilizar un asalto para cambiar la condición de la consulta original, la inyección de PHP Array en el motor de datos crea una consulta maliciosa que puede recuperar datos que la consulta original dice que no deberían estar disponibles. [6]

En el caso de la inyección de JavaScript, el delincuente utiliza la manipulación de este lenguaje de programación para alterar consultas complejas. Esto es especialmente relevante ya que varios gestores NoSQL emplean JavaScript con el mismo propósito. [9]

Cuando una página web no proporciona advertencias de error al intentar consultar la base de datos, es necesario emplear la técnica de inyección NoSQL Ciega (Blind). Dicho de otro modo, sólo se recibe una respuesta si el resultado de la consulta es correcto. Encontrar diferencias entre, contestación del servidor a una índole genuina y a una índole falsa es la base de esta técnica. [10]

Dependiendo de la condición establecida, las distintas consultas realizadas a través de un tipo de inyección pueden resultar en la obtención de todas las credenciales del usuario. Esto se debe a la utilización de TIPAR, el cual es un controlador que concede la manipulación de objetos en un motor de base de datos NoSQL, ofreciendo una solución que puede ser explotada para este propósito. [11]

La seguridad de los datos es de cuestión vital, y las organizaciones deben recurrir a herramientas externas o de terceros para reforzar la seguridad de los almacenes de datos frente a posibles ataques de inyección. [11]

Por ello, se creó un sistema llamado Persistent Pentesting Service (Faast) para encontrar y poner de relieve fallos y vulnerabilidades en los motores de datos NoSQL. Para encontrar este tipo de vulnerabilidades, Faast ofrece una serie de complementos especializados.
Hay diversos controladores disponibles para varios sistemas de almacenamiento de data, como MongoDB, quiwn utilizan el complemento de Faast. Este complemento se especializa en identificar y señalar inyecciones en Mongo Data Base, aprovechando JavaScript y la técnica de descubrimiento situada en el tiempo para reconocer la vulnerabilidad que puede ser explotada a través de una URL. [11]

El procedimiento de Faast implica dividir la tarea en dos segmentos. El primero se efectua con un retraso de 10s (segundos), mientras que el segundo constituye el resto del proceso. Esta división busca identificar si existe una discrepancia significativa en los tiempos de ejecución que supere un cierto umbral. Si esto ocurre, se considera que la prueba ha detectado una vulnerabilidad y se repite según sea necesario para confirmar los hallazgos. [11]

La dificultad radica en las actualizaciones de los controladores. Para que Faast funcione de manera efectiva, es fundamental mantener actualizada tanto la base de datos como los controladores correspondientes de Faast. De lo contrario, existe el riesgo de obtener respuestas incorrectas, ya sean positivas o negativas. [1]

1.2. Formulación del Problema.

¿Cómo identificar ataques de inyección NoSQL utilizando técnicas de aprendizaje automático?

1.3. Hipótesis.

La detección eficaz de ataques de inyección en almacenes de datos NoSQL será posible gracias a la aplicación de métodos clasificadores de aprendizaje automático.

1.4. Objetivos.

1.4.1. Objetivo General.

Identificar amenazas de inyección en bases de datos NoSQL, utilizando técnicas de aprendizaje automático.

1.4.2. Objetivos Específicos.

a) Elegir los algoritmos de aprendizaje automático que ofrezcan los mejores resultados entre los distintos enfoques de clasificación.

b) Describir las consultas buenas y malas en bases de datos NoSQL.

c) Preparar el conjunto de datos (DataSet) para la inteligencia de máquina.

d) Poner en práctica técnicas o algoritmos para identificar ataques de inyección NoSQL.

1.5. Teorías relacionadas al tema.

Las tecnologías de almacenamiento NoSQL desafiaron el paradigma entidad-relación RDBMS (Sistema Motriz de Base de Datos Relacionales) con la finalidad de gestionar de forma ágil enormes cantidades de datos. [12]

1.5.1. Evolución de NoSQL

Hoy en día, las aplicaciones web acumulan una cantidad considerable de datos, lo que plantea a los motores de datos relacionales dos grandes retos: la escalabilidad y el rendimiento. [13]

> "Se dice que un sistema es escalable si puede acomodar un aumento del número de objetos o elementos que lo componen sin necesidad de modificar su estructura original". [14]

> "Toda la velocidad y capacidad del sistema, o rendimiento, es lo que permite completar la mayor parte del trabajo, minimiza los conflictos de acceso a la RAM y maximiza el uso de los recursos". [4]

El aumento en la preocupación por los problemas de escalabilidad ha dado lugar no solo a la popularización del NoSQL (Not Only SQL), que se centra en mejorar la búsqueda y lectura mediante una perspectiva en la fiabilidad y el rendimiento. [13]

Los almacenes de datos en lenguaje de marcado extensible (XML) permiten al SGBD (Sistema Administrador de Almacenes de Datos) almacenar enormes volúmenes de datos, como documentos, imágenes y objetos diversos. [13]

1.5.2. Tipo de almacenamiento de datos NoSQL

1.5.2.1. Clave-Valor

Se refiere al almacenamiento de pares clave-valor, donde el valor puede ser singular o múltiple, y cada par constituye un conjunto asociativo con su respectivo valor dentro de una colección. [15]

La clave puede tomar la forma de un nombre de fichero, una cadena parcial, un hash o un URL. Este valor puede llegar a ser algún, un archivo de preferencias de usuario, un tipo de dato una imagen o un documento. [15]

A continuación, se presenta una figura de tipo de almacenamiento clave-valor.

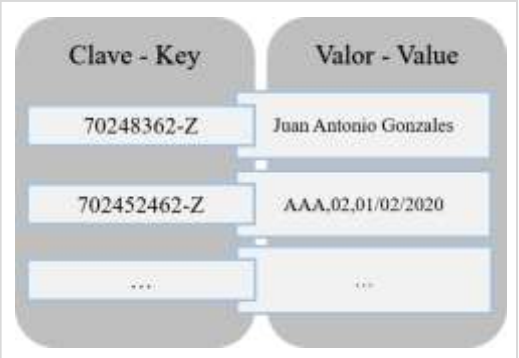


Fig. 1. Almacenaje basado en Clave - Valor.

1.5.2.2. Columnas

A diferencia del almacenamiento clave-valor, este tipo de almacenamiento se organiza por columnas, lo que significa que todos los valores de un dato se almacenan juntos para que puedan ser accedidos como una unidad. Este enfoque de almacenamiento es altamente eficiente para realizar consultas analíticas. [15]

A continuación, se presenta una figura de tipo de almacenamiento basado en columnas.

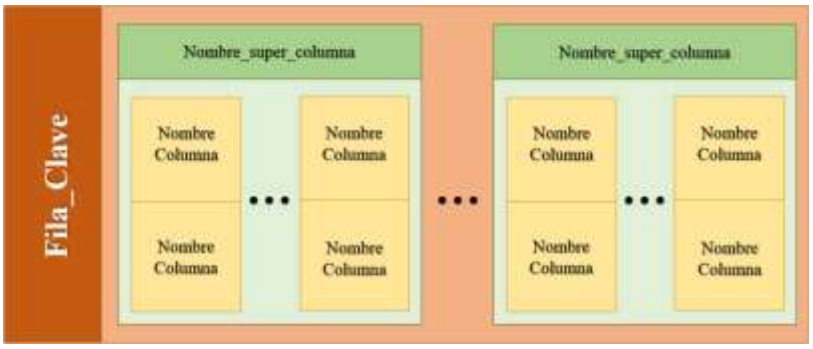


Fig. 2. Almacenaje basado en columnas.

1.5.2.3. Grafos

Los motores de datos de este mecanismo de almacenamiento pueden navegarse utilizando la teoría de grafos. Los enlaces binarios entre las piezas de la colección pueden representarse mediante un grafo, que está compuesto por una agrupación de nodos u objetos conectados por aristas. [16]

Los datos dentro de un motor de datos NoSQL, se presentan como nodos que se conectan entre sí mediante aristas. Para establecer una estructura adecuada en este tipo de almacén de datos, es importante que esté normalizada, lo que significa que cada nodo almacena información en una única columna, y cada relación se limita a tener únicamente dos conjuntos de datos. [15]

A continuación, se presenta una figura de tipo de almacenamiento basado en grafos.

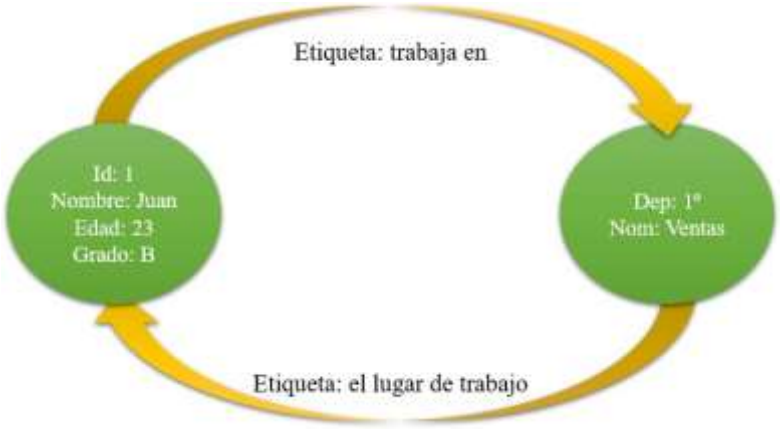


Fig. 3. Almacenaje basado en grafos.

1.5.2.4. Documentos

Dado que los datos semiestructurados pueden guardarse en formatos de archivo como XML, JSON o BSON, los motores de datos que supervisan este tipo de almacenaje son bastante adaptables. [15]

Estos documentos presentan similitudes significativas con los de clave-valor, ya que necesitan de un esquema definido y, al mismo tiempo, necesitan de características de consistencia y confiabilidad. [13]

A continuación, se presenta una figura de tipo de almacenamiento basado en documentos.

Fig. 4. Almacenaje basado en documentos.

Con una demanda de datos masiva y en constante expansión, empresas como Facebook, Amazon y Google han desarrollado bases de datos según sus necesidades. [13]

1.5.2.5. Sistemas de Administración de Base de Datos NoSQL

1.5.2.5.1. Cassandra

Es un motor de datos de código libre, que se destaca por combinar las características de 2 almacenes de datos: BigTable y Dynamo. [17]

> “Dynamo es un motor de datos NoSQL desarrollado por Amazon, elaborado para almacenar datos en formato clave-valor y documento, lo que posibilita un rendimiento rápido y una escalabilidad impecable” [18]

> “BigTable es un motor de almacenamiento de datos NoSQL creado por Google con énfasis en el almacenamiento de datos en columnas, lo que posibilita la gestión eficaz de cantidades masivas de datos.” [12]

Utilizando el lenguaje de programación Java, Facebook desarrolló Cassandra y lo lanzó como una forma de mejorar la interacción con el usuario y la eficiencia de las búsquedas. También intentaron sacar provecho de las opciones de precio razonable y escalabilidad. Facebook convirtió Cassandra en un sistema motor de datos de código abierto en 2008 al liberar el código fuente. Actualmente, Apache se encarga de su desarrollo y mantenimiento. [17]

1.5.2.5.2. CouchDB

Al igual que Cassandra, CouchDB es un motor de código libre para el almacenamiento de datos, centrado en documentos y gobernado por Apache. Para construirlo se utilizó Erlang, un lenguaje informático que funciona en una máquina virtual extremadamente potente. [17]

Este sistema motor de datos se destaca por su capacidad para crear vistas, las cuales facilitan la recuperación de valores de múltiples documentos. Aunque es conocido por su enfoque en documentos, sorprendentemente, también permite realizar operaciones JOIN, una característica común en SQL. Además, al no tener un esquema definido, permite la inclusión de datos semiestructurados y la introducción de nuevos tipos de documentos de forma flexible. [17]

CouchDB es un sistema distribuido que admite transacciones y garantiza la disponibilidad continua para la lectura de documentos. En su estructura interna, los documentos se guardan en un árbol binario identificados por un número de secuencia único. [17]

1.5.2.5.3. MongoDB

Un sistema motor de datos NoSQL de código abierto llamado MongoDB fue creado específicamente para documentos JSON. Combina las ventajas del almacenamiento en bases de datos relacionales (RDBMS) con [clave][valor], y está construido en C++. MongoDB puede ser potencialmente contenido binario ya que opera sobre documentos que son serializados binarios en formato JSON en su núcleo. [17]

En este sistema motor de datos, las operaciones de modificación pueden realizarse enviando solo los cambios de datos, y estas modificaciones son manejadas por el servidor en lugar del cliente, con un límite de tiempo de 2 segundos para su ejecución. Si la dimensión del documento aumenta, se traslada a un área vacía dentro del fichero de datos. [17]

1.5.3. Inteligencia Artificial. (IA)

La IA está cambiando la crónica del mundo y se considera una de las tecnologías más avanzadas hasta la fecha. Se puede dividir en dos ramas: la IA como conocimiento y la IA como ingeniería. [19]

La Inteligencia Artificial en su vertiente científica implica un análisis exhaustivo que abarca diversos campos como la neurología y la cognición. Se centra en comprender y desarrollar subrutinas generales relacionadas con la el lenguaje, percepción, la memoria, las emociones, la enseñanza y, la toma de decisiones. [19]

La Inteligencia Artificial en su aspecto ingenieril que estudia la aplicación práctica del conocimiento. Su objetivo es lograr resultados concretos en un plazo breve o moderado. Este conocimiento es examinado, modelado, formalizado y modificado mediante procesos de aprendizaje que pueden ser similares o superiores a los procesos humanos. [19]

1.5.3.1. Aprendizaje Automático

El proceso de enseñar a una máquina a generar predicciones se conoce como aprendizaje automático. Estas predicciones pueden fallar al principio, lo que podría deberse a que el DataSet no tiene suficientes datos. [20]

En el pasado, realizar predicciones era prácticamente inaccesible debido al limitado almacenamiento de datos y la disponibilidad de conocimiento. Sin embargo, en el presente momento, gracias a los nuevos avances en tecnología, tanto en almacenamiento y el procesamiento de datos, esta situación ha cambiado drásticamente. Ahora, las máquinas pueden procesar grandes cantidades de datos y, además, actualizar sus propios algoritmos para mejorar continuamente sus capacidades predictivas. [20]

1.5.3.2. Tipos de Aprendizaje Automático:

A. Aprendizaje Supervisado

Se trata de un modelo que aprende y busca reducir al mínimo la discrepancia entre los resultados y la función de error. Utiliza un conjunto de ejemplos, junto con las soluciones esperadas, denominadas etiquetas, para llevar a cabo la clasificación de una tarea típica. Cuando se enfrenta a nuevos ejemplos, el sistema debe aprender a clasificarlos adecuadamente. [21]

B. Aprendizaje No Supervisado

El componente fundamental de este tipo de aprendizaje es normalizar el aspecto en que se presentan al sistema los datos de acceso para encontrar características o patrones pertinentes. Como en este método no se dan ni etiquetas ni datos de entrada, los datos de entrada son la única información accesible. [19]

1.5.3.3. Mediciones de Rendimiento de Clasificación

A. Matriz de Confusión

Conocida como matriz binaria, se emplea en el ámbito de la clasificación para resumir el rendimiento de un clasificador. Se representa como una tabla que muestra los resultados de distribución en términos de falso positivo (FP), falso negativo (FN), verdadero negativo (VN) y verdadero positivo (VP).

Cada celda de la tabla indica una medida probabilística, a menudo representada utilizando términos como consonantes del alfabeto. [22]

En caso de que los valores sean recuentos, se pueden emplear cifras numéricas; en contraste, si son valores probabilísticos, se recurre a letras para su representación. [22]

Verdadero y Falsos Positivos

Se describe a la cantidad de positivos que sean correctos / incorrectos.

FP: Valores Falsos Positivos.

VP: Valores Verdaderos Positivos.

Verdadero y Falsos Negativos

Se describe a la cantidad de falsos que sean correctos / incorrectos.

FN: Valores Falsos Negativos.

VN: Valores Verdaderos Negativos.

A continuación, se presenta una figura de ejemplo en matriz de confusión

		Resultados de Clasificación	
		SEGURO	INSEGURO
Resultados de Reales	SEGURO	Verdadero Positivo (VP)	Falso Negativo (FN)
	INSEGURO	Falso Positivo (FP)	Verdadero Negativo (VN)

Fig. 5. Confusión Matrix.

B. Recall o Sensibilidad, Exhaustividad

La exhaustividad, que se evalúa mediante el análisis de la curva ROC y descubre todos los casos verdaderos positivos, es una métrica básica en el campo del proceso del lenguaje natural. La completitud, por tanto, es el porcentaje de casos verdaderos positivos que se anticiparon correctamente. [22]

En la siguiente expresión matemática, se determina la exhaustividad:

$$Recall = \frac{VP}{VP + FN}$$

Donde,

FN: Falso Negativo

VP: Verdadero Positivo

C. Precisión o Confianza

Esta métrica estima la productividad del modelo de aprendizaje automático comparando el número de casos positivos predichos con la cantidad real de sucesos positivos. Esta estadística es importante en la minería de datos y el AA (Aprendizaje Automatizado), aunque el estudio de la curva ROC suele ignorarla. [22]

La siguiente fórmula matemática, determina la precisión:

$$Precisión = \frac{VP}{VP + FP}$$

Donde,

VP: Verdadero Positivo

FP: Falso Positivo

D. Exactitud

Evalúa la dimensión de sucesos verdaderos que el modelo ha asociado correctamente. Si las clases están desequilibradas, utilizar la exactitud como métrica puede ser engañoso, ya que puede dar la impresión de que el modelo es más preciso de lo que realmente es. [22]

La fórmula siguiente se maneja para determinar la exactitud:

$$Exactitud = \frac{VN + VP}{VP + VN + FP + FN}$$

Donde,

FN: Falso Negativo

FP: Falso Positivo

VP: Verdadero Positivo

VN: Verdadero Negativo

E. F-Score o F-Measure

Esta medida, que oscila entre 0 y 1, combina exhaustividad y precisión. Esta se obtiene como el promedio armónico de ambos valores determinados. [23]

La siguiente fórmula matemática determina el F-Score:

$$F - Score = 2 * \frac{Precisión * Recall}{Precisión + Recall}$$

F. Curva de Receiver Operating Characteristic (ROC)

Se trata de un gráfico que traza la probabilidad de descubrimiento frente a la precisión realizando varios puntos de corte en una escala constante para evaluar el rendimiento de un sistema. También ayuda a determinar las capacidades de la colección de datos. [24]

Se visualiza 2 factores en la curva de ROC:

TPR (Tasa de Verdaderos Positivos)

$$TPR = \frac{VP}{VP + FN}$$

Donde,

FN: Falso Negativo

VP: Verdadero Positivo

FPR (Tasa de Falsos Positivos)

$$TPR = \frac{FP}{FP + VN}$$

Donde,

FP: Falso Positivo

VN: Verdadero Negativo

La técnica AUC (Area Under the Curve) puede utilizarse para determinar los puntos de la Característica Operativa del Receptor (ROC). Al trazar la tasa de falsos positivos (FPR) frente a la tasa de verdaderos positivos (TPR) en distintos niveles de categorización se crea la curva ROC. [25]

G. AUC (Zona Bajo la Curva)

Toda la superficie bajo la curva ROC en los niveles de distribución se evalúa mediante el AUC (Area Under the Curve). Con valores que van de 1 a 0, puede entenderse como la posibilidad de que un clasificador aleatorio clasifique mejor un ejemplo positivo que uno negativo. [25]

El AUC es útil porque es insensible a la escala y al umbral de clasificación, lo que lo hace una medida consistente de la calidad de los pronósticos del modelo. [25]

1.5.3.4. Modelos de Aprendizaje Automatizado

A. Árbol de decisión

Es toda representación gráfica de una técnica de clasificación o agrupación de datos. Genera una predicción utilizando las características encontradas en nuestra colección de datos. Como tal, este algoritmo pertenece a la categoría de SA (Aprendizaje Supervisado). [19]

El proceso para clasificar comienza en el nodo raíz, que evalúa un atributo y sigue la rama correspondiente al valor obtenido. A medida que avanza, el nodo raíz se ramifica hacia otros nodos, dividiendo la colección de datos en grupos más pequeños. Esto permite recorrer todos los nodos y establecer un modelo predictivo completo. [19]

Si la variable objetivo es categórica, el algoritmo se denomina árbol de clasificación; de lo contrario, se le llama árbol de regresión.

A continuación, se presenta una figura de ejemplo basada en Árbol de decisión.

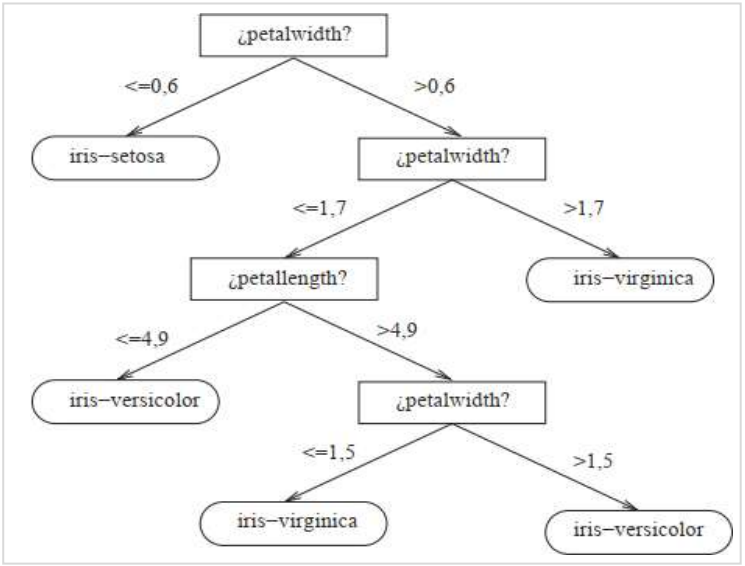


Fig. 6. Planta iris catalogado por Árbol de Decisión [19].

Un ejemplo de cómo utilizar este algoritmo para resolver el tipo de planta de iris basándose en características de anchura y longitud de los pétalos o sépalos es la distribución de las plantas de iris. [19]

Para clasificar el tipo de planta de iris utilizando el ejemplo del árbol de decisión, primero se toman medidas de la anchura y la longitud del sépalo y los pétalos de la planta. A continuación, continua la rama correspondiente hacia la derecha y se toman decisiones utilizando las expresiones condicionales "si... entonces... entonces... si no" hasta determinar que la planta es un ejemplar de iris virginica. [19]

La estrategia Top-Down Decision Tree Construction (TDIDT) es la principal técnica para construir un árbol de decisión. Al principio, todas las muestras están disponibles. Para asegurarnos de que cada nodo elegido es descendiente de su padre, continuamos eligiendo el más

conveniente atributo y lo establecemos como nodo descendiente. El árbol se desarrolla gradualmente de este modo. [19]

Metodo TDIDT

(colección de ejemplos: E)

[Construir raíz *R*]

[Construir-Árbol (*R*, *E*)]

Todos los datos deben pertenecer a la misma clase porque esta técnica se fundamenta en el concepto de que no hay datos inexactos. No obstante, esto puede dar lugar a problemas de sobreajuste con el conjunto de entrenamiento, lo que puede dificultar la correcta categorización de muestras frescas de plantas de iris. [19]

Crear particiones viables y seleccionar la mejor partición son las dos tareas principales.

A continuación, se presenta una figura de ejemplo basada en la creación de un arbol.

```
Procedimiento Crea-Árbol(N: nodo , E: conjunto de ejemplos)
si todos los ejemplos E son de la misma clase c entonces
  Asignar la clase c al nodo N;
  SALIR; {Esta rama es pura, ya no hay que seguir partiendo. N es hoja}
si no
  Particiones = Generar-particiones(E);
  MejorParticion := Seleccionar-mejor-partición(Particiones);
  para cada condicion i de MejorParticion hacer
    Añadir un nodo hijo i a N y asignar los ejemplos consistentes (Ei) a cada hijo;
    Crea-Arbol (i, Ei). {Realizar el mismo procedimiento global con cada hijo};
  fin para
fin si
```

Fig. 7. Procedimiento para Crear-Árbol [19].

Su relevancia radica en que, una vez seleccionada la partición, independientemente de si es subóptima o no, no se puede modificar. Además, permitir más particiones aumenta la probabilidad de generar árboles con datos más precisos. [19]

Por consiguiente, las divisiones basadas en atributos nominales y las basadas en atributos numéricos son los dos tipos de divisiones que ofrecen la gran mayoría de algoritmos de árboles de decisión.

Las divisiones de atributos nominales pueden representar decisiones binarias, como, por ejemplo, si un atributo es "desposado", las contestaciones podrían ser [Sí] o [No]. Por lo tanto, si un atributo posee n posibles valores, se crearían n divisiones binarias, abarcando el rango de $1 \leq j \leq n$. [19]

En las divisiones de atributos numéricos, el rango se fracciona en dos intervalos: $x_i \leq v_c$, $x_i > v_c$ donde v_c [punto de corte] es un valor mediano entre el mínimo y el máximo observado del atributo x_i en la constatación E. Asi mismo, una vez que se encuentran estos valores de v_c , se pueden determinar los puntos óptimos para reducir el tamaño del árbol. [19]

La selección de la mejor partición para un nodo concreto viene después de identificar las particiones viables. Para ello, las divisiones se ordenan utilizando funciones heurísticas que reducen la entropía. [19]

La entropía evalúa qué tan homogéneos son los ejemplos dentro de un conjunto de entrenamiento E, considerando las diferentes clasificaciones de C clases en el problema. [19]

$$Entropía\ (E) = \sum_{i=1}^{c} -p_i * Log_2 p_i$$

Donde: p_i determina probabilidad de que un ejemplar sea de la clase i

El criterio de ganancia de información se utiliza para cuantificar el impacto de la división de los datos al calcular la disminución de entropía. [19]

$$Gain(E, x) = Entropía(E) - \sum_{v \in V(x)} p_v * Entropía\ (E_v)$$

Donde: E_v es el subconjunto de E cuyo atributo [x] tiene el valor [v], $V(x)$ es la colección de valores de la propiedad [x] y p_v es la probabilidad de que la propiedad [x] capture el valor de $v\left(\frac{|E_v|}{|E|}\right)$.

La ganancia de información muestra cómo las muestras se distribuyen equitativamente por la variable [x]. Para estimar esta ganancia se aplica la norma Gain Ratio (Ratio de Ganancia). [19]

$$GainRatio(E, x) = \frac{Gain(E, x)}{SplitInformation(E, x)}$$

En la siguiente expresión, donde E es el conjunto de ejemplos, [x] es el atributo seleccionado y la entropía de E con respecto a [x] está representada por $SplitInformation(E, x)$.

B. Random Forest

El algoritmo está pensado para aprender bajo supervisión. Su principal idea es combinar muchos modelos de árboles de decisión para conseguir conclusiones basadas en patrones. Se emplea en muchos ámbitos diferentes, como la detección, el procesamiento de datos, el reconocimiento de caras y la clasificación de textos. [26]

A continuación, se presenta una figura de ejemplo basada en Random Forest.

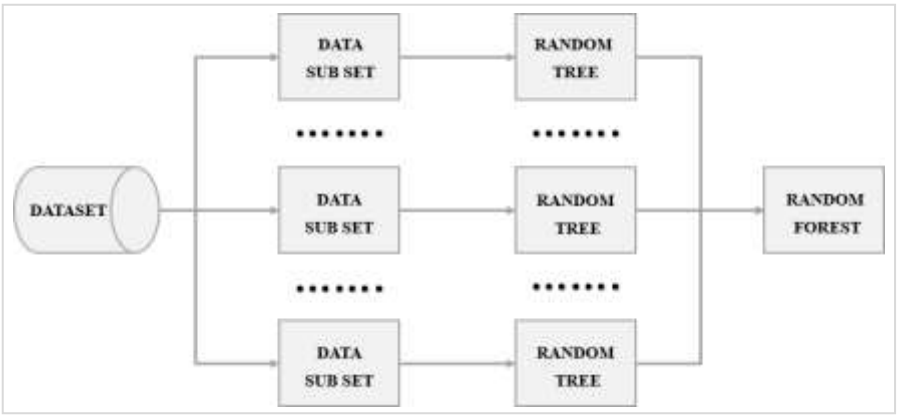


Fig. 8. Mapeo estructural de Random Forest.

La creación de los árboles de votación y de decisión son las dos fases principales del algoritmo Random Forest (RF). Tres etapas componen el proceso de desarrollo: la selección aleatoria de la colección de entrenamiento, la formación de árboles aleatorios y la subdivisión de nodos. [26]

Durante el desarrollo de subdivisión de nodos, la característica clave es elegir la función de división que tenga el menor coeficiente $Gini$. [26]

La siguiente fórmula matemática determina el coeficiente de $Gini$:

$$Gini(S) = 1 - \sum_{i=1}^{m} P_i^2$$

Donde: P_i describe la probabilidad de la categoría C_j en el conjunto S.

Por lo tanto, la siguiente fórmula matemática determina el coeficiente del nodo dividido:

$$Gini_{split}(S) = \frac{|S_1|}{|S|} Gini(S_1) + \frac{|S_2|}{|S|} Gini(S_2)$$

En este contexto donde $|S|$ representa el número total de muestras en el conjunto S que se divide en los conjuntos de S_1 & S_2.

También es posible manifestar RF como:

$$h(x; \theta_k)_1 \; K = 1, \ldots., K$$

Donde: x describe el vector de acceso (covariable) de longitud p con el vector aleatorio asociado [X] y [θ_k] son independientes e idénticamente distribuidas (iid).

El ajuste de regresión suele dar como resultado un efecto numérico. No obstante, pueden obtenerse resultados categóricos definiendo umbrales, lo que plantea problemas de categorización. [27]

Los datos creados trazan la distribución conjunta de (X, Y) y abarca $n(p + 1)$. En el contexto de la regresión, RF representa la media sin ponderar de la colección. [27]

$$h(x) = (\frac{1}{K}) \sum_{k=1}^{K} h(x; \theta_k)$$

Como [$K \to \infty$] La ley de amplias cantidades asegura

$$E_{X,Y} = (Y - \bar{h}(X))^2 \to E_{X,Y}(Y - E_\theta h(X; \theta))^2$$

La cifra a la derecha representa el error de predicción para el RF nominado, denotado como PE_f^i . La coincidencia indica que el RF no está sufriendo sobreajuste.

Para establecer el promedio del error de predicción de un solo árbol $h\,(X; \theta)$:

$$PE_i^* = E_\theta E_{X,Y}(Y - h(X; \theta))^2$$

Supongamos que el árbol es imparcial. $EY = E_X h(X; \theta)$ luego

$$PE_f^* \leq \bar{\rho} PE_i^*$$

Donde $\bar{\rho}$ es la correlación ponderada entre residuos

$Y - h(X; \theta)$ y $Y - h(X; \theta')$

Para lograr una regresión precisa con Bosques Aleatorios, es fundamental que haya una caída correlación entre los árboles (Decision Tree) individuales en el Bosque Aleatorio, así como un bajo error de predicción. [27]

C. K- Vecinos más próximos (K-NN)

Basado en el aprendizaje basado en instancias, o aprendizaje no generalizado, funciona este algoritmo no paramétrico. Su objetivo principal es pronosticar la etiqueta de clase del conjunto de muestras utilizando las etiquetas de clase de los k vecinos más próximos. Una medida métrica de la distancia entre las muestras determina el número exacto de vecinos. [28]

El objetivo de este algoritmo es guardar instancias de muestra en lugar de intentar construir un modelo interno genérico. Para asignar a cada punto los datos más representativos dentro de sus vecinos más próximos, la clasificación se decide por mayoría de votos de los k vecinos más próximos de cada punto. [28]

Luego, el conjunto de prueba se clasifica mediante el voto de los K-NN en el conjunto de entrenamiento.

Si consideramos $X = \{X_1, X_2, X_3, \dots, X_n\}$ como el conjunto de preparación, donde $x_i \epsilon R^n$ representa un punto de preparación con n características dimensionales, y $Y = \{Y_1, Y_2, Y_3, \dots, Y_n\}$ representa las etiquetas de clase, entonces, para un punto X cuya etiqueta de clase se desconoce, el proceso es el siguiente: Principalmente, se calculan las medidas de afinidad entre las muestras de prueba y de entrenamiento; luego, se encuentran los K-NN según la distancia de afinidad, y finalmente, se determina la etiqueta de clase mediante el voto predominante. [29]

Un conjunto de valores preparados X asociado a una clase C se define mediante una matriz de valores binarios $\{0,1\}$. Si el conjunto de preparación X corresponde a la clase C_1 entonces $U_{c_1}(x) = 1$ y $U_{c_2}(x) = 0$, dónde $C = \{c_1, c_2\}$. [29]

No obstante, en el enfoque difuso, el algoritmo K vecinos más próximos emplea un rango constante de pertenencia a causa a la teoría difusa. Esto se calcula utilizando la siguiente fórmula matemática:

$$U_G(x) = \begin{cases} 0.51 + 0.49\frac{k_c}{K} \; si \; c = c_1 \\ 0.49\frac{k_c}{K} \; de \; otra \; manera \end{cases}$$

Donde: k_{c_1} exhibe el número de instancias que compete a la clase c_1 hallado en los k vecinos de X y K es un valor entero entre $\{3,9\}$.

Después de este cálculo, se determina la etiqueta de clase de la muestra de la siguiente manera: En primer lugar, se utiliza la distancia euclídea para determinar los k vecinos más próximos de la muestra. A continuación, la norma euclídea y la pertenencia a una clase se utilizan para realizar una votación final para cada clase. [29]

$$V(k_j, c) = \frac{U_c(k_j) \Big/ \left(\| \acute{x} - k_j \|\right)^{2/m-1}}{\sum_{i=1}^{k} 1 \Big/ \left(\| \acute{x} - k_i \|\right)^{2/m-1}}$$

Donde: k_j es el j vecino más próximo y $m = 2$ es un parámetro.

Una de las principales limitaciones del método K-NN es su capacidad para dar el mismo peso a cada etiqueta del conjunto de datos de muestra. Para evitar esta restricción, utilizamos membresías neutrosóficas, que indican la importancia de un punto de datos dentro de su clase. [29]

Las pertenencias que representan verdadero (T), falso (F) o indeterminado (I) son establecidas por el conjunto neutrosófico, y pueden ser cuantificadas utilizando los siguientes criterios:

$$T_{ij} = \frac{(x_i - c_j)^{-(\frac{2}{m-1})}}{\sum_{j=1}^{c}(x_i - c_j)^{-(\frac{2}{m-1})} + (x_i - \bar{c_i}_{max})^{-(\frac{2}{m-1})} + \delta^{-(\frac{2}{m-1})}}$$

$$F_i = \frac{(\delta)^{-(\frac{2}{m-1})}}{\sum_{j=1}^{c}(x_i - c_j)^{-(\frac{2}{m-1})} + (x_i - \bar{c_i}_{max})^{-(\frac{2}{m-1})} + \delta^{-(\frac{2}{m-1})}}$$

$$I_i = \frac{(x_i - \bar{c_i}_{max})^{-(\frac{2}{m-1})}}{\sum_{j=1}^{c}(x_i - c_j)^{-(\frac{2}{m-1})} + (x_i - \bar{c_i}_{max})^{-(\frac{2}{m-1})} + \delta^{-(\frac{2}{m-1})}}$$

Donde: m es una constante, δ es un parámetro de normalización y c_j muestra el centro de grupo j. Por cada punto i, el $\bar{c_i}_{max}$ es la media de dos valores mayores. El valor T_{ij} muestra el auténtico valor del punto i para la clase j. F_i muestra el número de inexactitud de los puntos i y j, especifica el valor para i. T_{ij} significa el punto i está próximo de un grupo y da excepción de oportunidad que sea ruido.

El valor de pertenencia falso se resta del valor de pertenencia final para un punto dado, y se añade el valor de pertenencia indeterminado. [29]
El siguiente cálculo se realiza para demostrar que, con respecto a la etiqueta de clase j la pertenencia neutrosófica se triplica para una muestra desconocida x_u.

$$\mu_{ju} = \frac{\sum_{i=1}^{k} d_j(T_{ij} + I_i - F_i)}{\sum_{i=1}^{k} d_i}$$

$$d_j = \frac{1}{\|x_u - x_i\|^{\frac{2}{q-1}}}$$

Donde: d_j es toda función de longitud para calcular x_i y x_u, k muestra el número de K-NN y q es un número entero. x_u son los grados de muestra incógnita a todas las etiquetas de clase.

D. Redes neuronales

Con un enfoque distinto al de la inteligencia artificial estándar, Las redes neuronales artificiales (RNA) se desarrollaron en un intento de imitar el proceso de pensamiento del cerebro humano. [30]

Un modelo de red neuronal artificial consta de tres tipos diferentes de neuronas que se utilizan como unidades computacionales: el perceptrón multicapa (MLP), la red neuronal de pulsos (SNN) y el McCulloch-Pitts (MP). [31]

El estado de una neurona depende de sus conexiones, ya que, si está aislada, tanto su utilidad como su capacidad de procesamiento se ven reducidas. [30]

Las redes neuronales, en opinión de Haykin, son capaces de retener información basada en la experiencia para poder aplicarla posteriormente. Esto es similar al funcionamiento del cerebro, que adquiere inteligencia mediante un proceso de aprendizaje y la almacena en las conexiones entre neuronas. [30]

Según Caicedo & López. [30], existen tres niveles de organización en una red neuronal.

a) **Entrada:** Grupo de neuronas que reciben directamente datos de fuentes externas.

b) **Oculta:** Un grupo de neuronas interiores que no están directamente conectadas con el mundo exterior. Estas neuronas, a menudo denominadas capas ocultas, están conectadas en red y comparten y procesan la información de distintas maneras.

c) **Salida:** Grupo de neuronas que se comunica con el exterior.

Término utilizado a menudo para modelar cómo se produce el aprendizaje en las redes neuronales:

$$w(t+1) = w(t) + \triangle w(t)$$

Donde,

$w(t+1)$: Valor renovado del peso sináptico

$\triangle w(t)$: Alteración del peso sináptico

$w(t)$: Valor vigente del peso sináptico

En una red neuronal, la adaptación se realiza globalmente calculando una deriva basada en el error cuadrático medio, en lugar de localmente mediante ajustes. Durante el proceso de entrenamiento, la red genera errores en sus numerosas neuronas de salida, que se representan mediante esta deriva global, también conocida como E_p. [30]

$$E_p = \frac{1}{2P} \sum_{p=1}^{P} \sum_{j=1}^{M} (d_{pj} - y_{pj})^2$$

Donde,

P: Numero de modelos de entrenamiento

M: Numero de neuronas en la capa saliente

A continuación, se presenta una figura de ejemplo en acoplamiento entre neuronas.

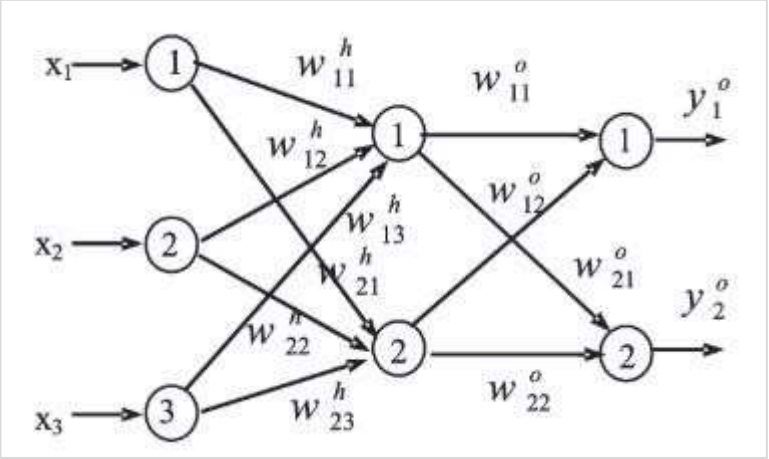


Fig. 9. Estructura de acoplamiento entre neuronas [30].

E. Perceptrón Multicapa

En 1958 Frank Rosenblatt presentó el perceptrón como el primer modelo de red neuronal artificial (RNA) capaz de reconocer patrones sencillos. La entrada de este modelo tenía muchas neuronas lineales, pero luchaba con una función de activación bipolar, que daba lugar a dos posibles estimaciones para su salida. [30]

Un perceptrón es un modelo de red neuronal de una capa en la que todas las neuronas de entrada están acopladas a cada neurona de salida, creando una única capa de procesamiento. La neurona de salida puede tener una función de activación binaria escalonada $[0, +1]$ o bipolar$[-1, +1]$. [30]

A continuación, se presenta una figura de ejemplo en Arquitectura Perceptron Multicapa.

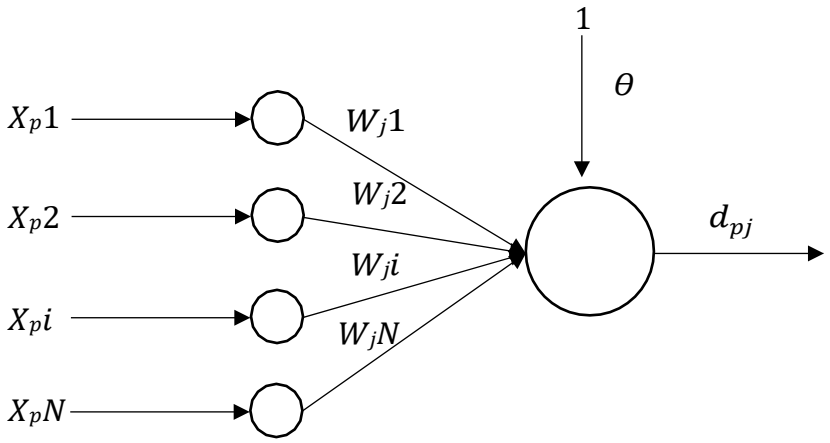


Fig. 10. Arquitectura de un perceptrón *[30]*.

El perceptrón multicapa (MLP) se compone de varias capas ocultas, aunque la mayoría de las aplicaciones sólo requieren una capa oculta. El MLP tiene dos capas: una de entrada y otra de partida, además de la capa oculta. [32]

Como la arquitectura de red neuronal de un Multilayer Perceptron presentaba capas de entrada, ocultas y de salida, Rosenblatt observó que podía manejar problemas de categorización más difíciles. [30]

A continuación, se presenta una figura de ejemplo en Perceptron Multicapa.

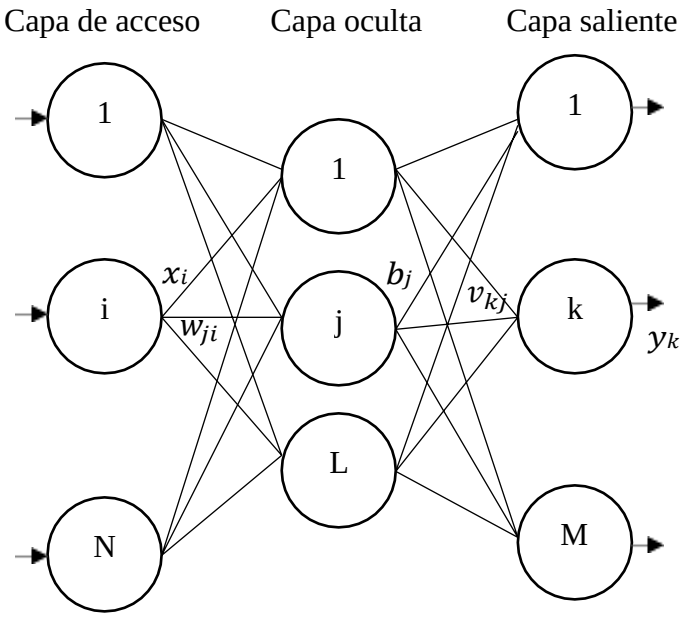


Fig. 11. Perceptrón multicapa [32].

Donde,

v_{kj}: Peso del enlace entre la neurona oculta j con la neurona saliente k

w_{ji}: Peso de la conexión entre la neurona de acceso i con la neurona oculta j

Según la arquitectura, las conexiones que transfieren información entre neuronas siempre van de la neurona de entrada a la de salida a través de las neuronas ocultas. [32]

Paul Werbos desarrolló la técnica de retro propagación en los años setenta y David Rumelhart la popularizó en 1989 como método de entrenamiento del perceptrón multicapa. Este método permite la resolución de una gran diversidad de problemas muy complejos mediante el entrenamiento del perceptrón multicapa. [30]

Al contener una o varias capas ocultas que pueden almacenar funciones entre un conjunto de variables de entrada y salida con entidades no lineales, el método de retro propagación destaca por su versatilidad y no linealidad. [32]

Los pesos de la capa de acceso a la de partida se modifican gradualmente durante la fase de entrenamiento del método de retro propagación para garantizar que la salida final coincida con la prevista. [32]

Durante la fase de actuación de la capa oculta, se le envían distintos pesos w_{ji} en un patrón de entrada x_p: x_1 ... x_n. Esta capa genera un valor de salida mediante una función de activación y lo envía a la capa de salida a través de los pesos v_{kj}. [32]

La aproximación absoluta o net que recibe una neurona encubierta j, se representa como net_{pj}:

$$net_{pj} = \sum_{i=!}^{N} w_{ji}\, x_{pi} + \Theta_j$$

Donde,

$\boldsymbol{\theta}$: Es el umbral de una neurona típica con un 1 que se calcula como peso agrupado.

La tasación saliente de la neurona oculta j, b_{pj}, se adquiere superponiendo una subrutina f(.) sobre su acceso net.

$$b_{pj} = f(net_{pj})$$

Asi mismo, el acceso net alberga una neurona saliente k, net_{pk}, es:

$$net_{pk} = \sum_{j=1}^{L} v_{kj}\, b_{pj} + \Theta_k$$

Finalmente, de la neurona saliente k, y_{pk}, el valor de salida es:

$$y_{pk} = f(net_{pk})$$

A partir de un patrón de entrada preestablecido, el usuario define la salida deseada en el proceso de retro propagación. Dado que su objetivo es reducir la diferencia entre la partida deseada y la alcanzada durante la fase de entrenamiento, este algoritmo se clasifica como de tipo supervisado. [32]

La subrutina de desvió que pretende desestimar para cada muestra p viene presunto por:

$$E_p = \frac{1}{2} \sum_{k=1}^{M} (d_{pk} - y_{pk})^2$$

Donde,

d_{pk}: Este término se refiere al valor destacado de la neurona de salida k para el modelo p. A partir de este término, es posible obtener una regla para el desvío mediante:

$$E = \sum_{P=1}^{p} E_p$$

El descenso gradiente es el fundamento matemático de la técnica de retro propagación utilizada para cambiar los pesos. El gradiente en este proceso acelera el fallo si se translada en dirección positiva y lo desacelera si se mueve en dirección negativa. Como resultado, al mover cada peso en la dirección adecuada, se puede disminuir la deriva. La función de los pesos está representada por E_p, que es igual a la derivada parcial de E_p para cada valor de peso.

$$-\sum_{p=1}^{P} \frac{\partial E_p}{\partial W_{ji}}$$

A continuación, se presenta una figura de ejemplo basada en la superficie del error.

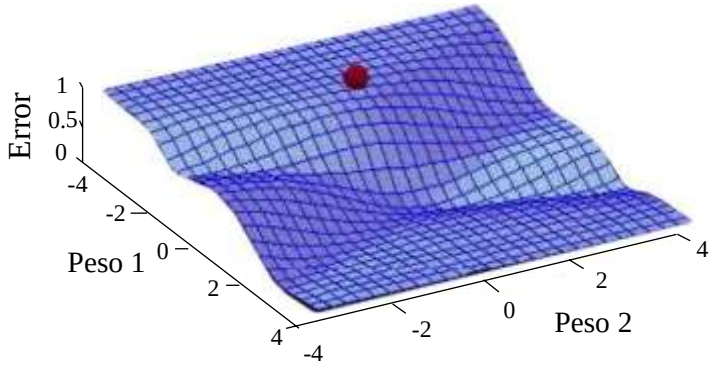


Fig. 12. Superficie del error [32].

F. Support Vector Machine

Vapnik y sus colegas desarrollaron la metodología de clasificación estadística SVM (método de vectores de soporte) [33] en la década de 1990 para discutir problemas vinculados con la regresión, la selección de características y la clasificación multicategoría. [34]

Las SVM han argumentado su eficacia en numerosos campos, como el reconocimiento de caracteres, la visión por ordenador y el procesamiento del lenguaje natural. Los sólidos fundamentos teóricos de este algoritmo han merecido elogios. Dado que crea hiperplanos que se dividen de forma lineal o casi lineal tanto en el espacio original como en el espacio de características, entra en la categoría de clasificadores lineales. [35]

Ye, Lin & Li, [33] explican que con un conjunto de entrenamiento de m muestras etiquetadas, representadas como $\{(\bar{x},\ y_i)|\bar{x}_i \epsilon R^n,\ y_i \epsilon \{+1,\ -1\},\ i = 1,\ \dots\ ,\ m\}$, SVM puede generar una hiper superficie de separación que maximiza la capacidad de propagación. La función 58ecisión se formula matemáticamente de la siguiente manera:

$$d(\bar{x}) = \sum_{i=1}^{m} \alpha_i y_i K(\bar{x}_i, \bar{x}) + b$$

Donde,

α_i: Parámetros establecidos por el algoritmo de aprendizaje de SVM.

$K(\bar{x}_i, \bar{x})$: Es la función del centro que mapea implícitamente las muestras a un contorno dimensional extraordinario.

Las muestras $\bar{x_i}$ con parámetros α_i distintos que cero se consideran "Vectores de Soporte (SVs)".

A continuación, se presenta una figura de ejemplo en clasificación lineal de SVM.

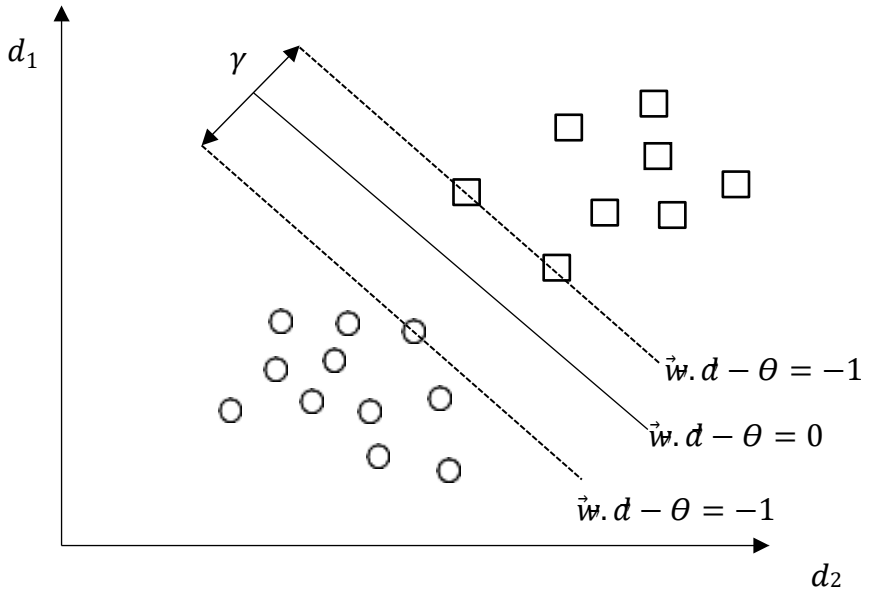


Fig. 13. El clasificador lineal de SVM con margen máximo [33].

Mediante su técnica discriminante, la SVM resuelve una serie de problemas de optimización convexa y obtiene el mismo parámetro de hiperplano óptimo. Para convertir los datos del espacio de entrada al espacio de características, utiliza un núcleo específico. Como resultado, sólo se mantiene el óptimo durante la fase de entrenamiento. [34]

A continuación, se presenta una figura de ejemplo en grafico bidimensional, perceptrón e hiperplanos.

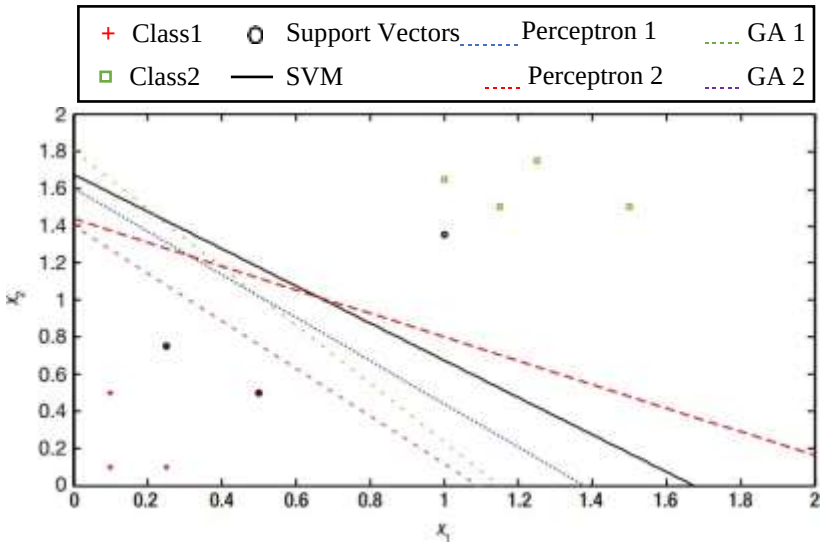


Fig. 14. Gráfico bidimensional de 2 clases para SVM, perceptrón e hiperplanos GA [34].

Una de las características de SVM es su capacidad para prevenir el sobreajuste al buscar minimizar el error de entrenamiento hacia cero. Esto es particularmente relevante en modelos de aprendizaje automático. SVM no requiere ajustes manuales para controlar la complejidad del modelo; en su lugar, automáticamente determina esta complejidad seleccionando el número de vectores de soporte adecuado. [34]

Desde su debut en 1992, las máquinas de vectores soporte (SVM) han adquirido un uso generalizado como algoritmo para tratar diversas cuestiones, como el procesamiento de imágenes y audio, la categorización de defectos, el análisis de vídeo y el diagnóstico médico. SVM es un algoritmo clasificador potente en el campo del aprendizaje automático debido a su robustez y capacidad para ofrecer una resolución global óptima y distintiva, como demuestran las comparaciones con otros 16 clasificadores en 21 conjuntos de datos. [34]

Características de SVM:

a) **SVM es una técnica eficiente en cuanto a recursos:** La SVM debe conservar en memoria todos los datos durante la fase de entrenamiento. La máquina de vectores soporte, sin embargo, sólo se basa en una fracción elegida de estas ocurrencias una vez establecidos los parámetros del modelo. Por ello, la SVM puede considerarse como la sumatoria ponderada de los vectores de soporte. [34]

b) **SVM es un enfoque basado en núcleos:** La SVM maneja un núcleo para proyectar los datos en un nuevo espacio dimensional para su clasificación. A partir de ahí, realiza tareas de aprendizaje automático analizando y optimizando convexamente los datos. [34]

   La SVM utiliza un núcleo predeterminado para asignar los datos y un separador lineal para diferenciar las clases. En la optimización de SVM, la elección y el ajuste son componentes esenciales. [34]

c) **SVM opera como un separador de margen máximo:** La SVM pretende maximizar el hiperplano minimizando la desviación o una función de coste situándose lo más lejos posible entre las distintas clases. Esto sugiere que, mientras que los subconjuntos se utilizan para hacer predicciones en casos que aún no se han visto, las muestras de población se utilizan durante el entrenamiento. [34]

Para cumplir las restricciones de dualidad y convexidad, en SVM se utiliza la minimización del riesgo estructural (SRM). Se trata de un refinamiento convexo con m restricciones y n variables en la función de coste. SRM ordena una colección de patrones en un orden determinado en función de lo desafiantes que resulten. [34]

A continuación, se presenta una figura de ejemplo en vínculos de error e índice de modelo.

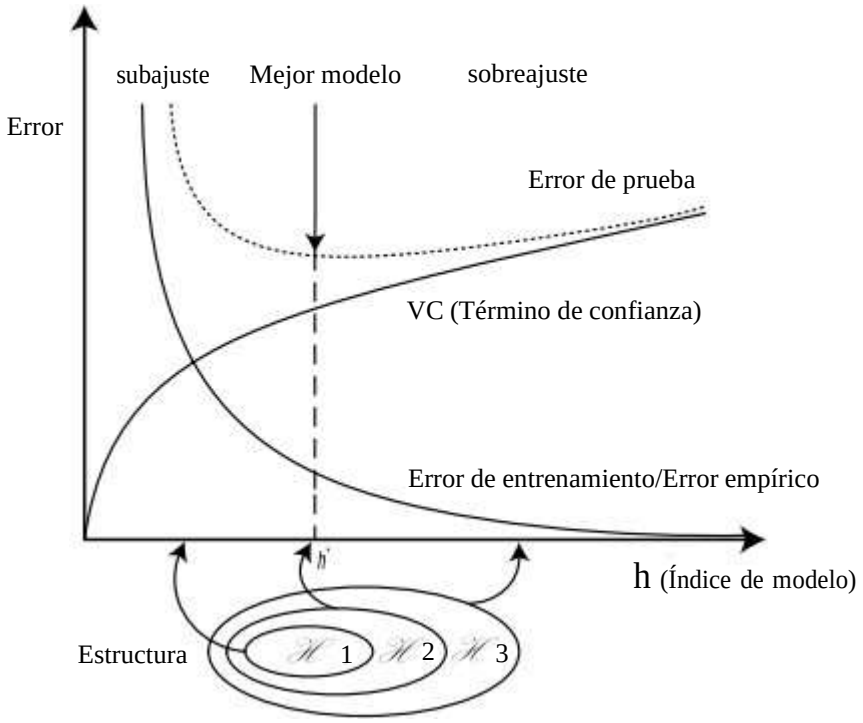


Fig. 15. Vínculo entre tendencias de error e índice de modelo [34].

II. MATERIAL Y MÉTODO

2.1. Tipo y Diseño de Investigación.

Dado que los resultados de la investigación pueden obtenerse por medios estadísticos, matemáticos o computacionales, el tipo de estudio será cuantitativo.

Los objetivos de la investigación determinarán la selección de nuestra muestra, por lo que se empleará un diseño de estudio cuasiexperimental.

2.2. Variables, Operacionalización.

Tabla I.

Variables y Operacionalización.

Variables	Dimensión	Indicador	Ítem	Técnica e instrumentos de recolección de datos
Algoritmos de clasificación Detección de ataques de inyección	Uso de recursos	Grado de uso de CPU	$Cc = \sum_{j}^{n} \frac{cc_j}{n}$	Registro Electrónico
		Grado de uso de memoria	$Cm = \sum_{j}^{n} \frac{cm_j}{n}$	
		Promedio de tiempo de respuesta	$Tr = \sum_{j}^{n} \frac{tf_j - tf_i}{n}$	
	Rendimiento	Exactitud	$E = \frac{VP + VN}{VP + VN + FP + FN}$	
		Precisión	$P = \frac{VP}{VP + FP}$	
		Recall	$R = \frac{VP}{VP + FN}$	
		F-Score	$F = (2)\frac{P * R}{P + R}$	
		AUC	$AUC = \sum_{ig(P+N)} \frac{(TPR_i + TPR_{i-1})(FPR_i - FPR_{i-1})}{2}$	

2.3. Población de estudio, muestra, muestreo y criterios de selección.

Árbol de Decisión, Detector Automático de Interacción Chi-Cuadrado (CHAID), Dicotomizador Iterativo (ID3), Árbol de Clasificación y Regresión (CART), Redes neuronales, Análisis de Elementos Principales (PCA), Análisis Principal Independiente (ICA), Regresión Logística, K-Means, K-Nearest Neighbors (KNN), regresión por mínimos cuadrados regulares, algoritmos basados en la densidad, Xgboost, Adaboost, Markov, Random Forest, Super Vector Machine (SVM) y Perceptron multicapa son los dieciocho modelos de clasificación que componen la membresía.

Para crear una muestra de conveniencia no estadística, se creó una clasificación basada en el rendimiento de la exactitud del modelo; ésta se muestra en el Apéndice 6. Redes neuronales, Random Forest, K-NN, SVM, Perceptron multicapa y Árbol de decisión fueron los seis algoritmos de muestra que se seleccionaron como los modelos con mayor precisión tras la investigación.

2.4. Técnicas e instrumentos de recolección de datos, validez y confiabilidad.

Hernández (2013) destaca la necesidad de abordar continuamente la validez, objetividad y fiabilidad de las metodologías de extracción de datos en protocolos y actuaciones.

La fiabilidad es el grado de consistencia de los resultados, la objetividad es la opinión de los expertos y la validez evalúa el grado en exactitud del parámetro a medir.

Según Romero, F. (2020), existen seis tipos diferentes de procedimientos de recolección de datos: los instrumentos específicos de la disciplina, la observación, las pruebas e inventarios estandarizados, los instrumentos mecánicos o electrónicos y el análisis de contenido cuantitativo.

En este estudio se utilizará el formato de la tabla de confusión Anexo 5 para registrar los resultados de la clasificación de las consultas benignas y nocivas y los resultados reales. Las métricas precisa, exacta, completa, AUC y puntuación F se registrarán utilizando los métodos indicados en la Tabla 1, junto con el número de falsos positivos, verdaderos positivos, verdaderos negativos y falsos negativos. Para analizar el empleo de recursos se utilizarán métricas como el tiempo, el consumo, la velocidad, la disponibilidad y el uso. Por este motivo, en los Anexos 1, 2 y 3 se adjuntan los formatos necesarios para registrar la CPU, la memoria y el tiempo medio de reacción que producirá el ordenador cuando la herramienta Waikato Environment for Knowledge Analysis (WEKA) cargue la colección de datos y produzca los resultados del estudio.

2.5. Procedimientos de análisis de datos.

El parámetro de registro electrónico ayuda a medir el consumo de recursos para los algoritmos de clasificación evaluando medidas como el tiempo medio de reacción, el empleo de la memoria, CPU y el grado de consumo, todas ellas obtenidas a partir de las siguientes fórmulas matemáticas:

Nivel de uso de la CPU:

$$Cc = \sum_{j}^{n} \frac{cc_j}{n}$$

Donde,

Cc: Nivel de uso de CPU

Cc_j: Nivel de uso de CPU en la verificación j

n: Total, de verificaciones

Nivel de uso de la memoria:

$$Cm = \sum_{j}^{n} \frac{cm_j}{n}$$

Donde,

Cm: Nivel de uso de memoria

Cm_j: nivel de uso de memoria en la prueba j

n: Total, de pruebas

Media de tiempo de contestación:

$$Tr = \sum_{j}^{n} \frac{tf_j - tf_i}{n}$$

Donde,

Tf_j: Tiempo de reacción final

Tr: Tiempo de reacción

Tf_i: Tiempo de reacción inicial

n: Total, de pruebas

Para evaluar el rendimiento en términos de detección de ataques de inyección, la variable de registro electrónico evalúa métricas como precisión, exactitud, exhaustividad, AUC y F-Score, todas ellas dependientes de la tabla de confusión.

La exactitud es la tasa de pronósticos de consultas NoSQL genuinamente dañinas verdaderas que el modelo ha presentado:

$$Exactitud = \frac{VP + VN}{VP + VN + FP + FN}$$

Donde,

VP: Verdadero positivo

VN: Verdadero negativo

FP: Falso positivo

FN: Falso negativo

La precisión se define como la relación entre los resultados efectivos proyectados de una consulta NoSQL y los resultados positivos reales:

$$Precision = \frac{VP}{VP + FP}$$

Donde,

VP: Verdadero positivo

FP: Falso positivo

El recall es el número de pronósticos exactos y veraces en consultas NoSQL perjudiciales:

$$Recall = \frac{VP}{VP + FN}$$

Donde,

VN: Verdadero negativo

VP: Verdadero positivo

FP: Falso positivo

FN: Falso negativo

Para determinar si existe la posibilidad de identificar consultas NoSQL fraudulentas, la puntuación F es la que encuentra el término medio entre exhaustividad y precisión:

$$F - Score = 2 * \frac{Precisión * Recall}{Precisión + Recall}$$

Donde,

P: Precisión

R: Recall

La curva de ROC, evalúa la exactitud y precisión, calculando el desempeño del modelo. Este se representa en dos parámetros:

Tasa de Verdaderos Positivos (TPR)

$$TPR = \frac{VP}{VP + FN}$$

Donde,

FN: Total de Falsos Negativos

VP: Total de Verdaderos Positivos

Tasa de (FP) Falsos Positivos (FPR)

$$FPR = \frac{FP}{FP + VN}$$

Donde,

FP: Total de Falsos Positivo

VN: Total de Verdaderos Negativos

El AUC determina toda el área bajo el ROC de la distribución en todos los niveles de categorización. Se adapta tanto a clasificaciones aleatorias positivas como negativas.

$$AUC = \sum_{ig(P+N)} \frac{(TPR_i + TPR_{i-1})(FPR_i - FPR_{i-1})}{2}$$

Los indicadores de la tabla de confusión para demostrar la eficacia del algoritmo de aprendizaje automático, se recopilarán a través de la herramienta de registro electrónico que se utiliza cuando se ejecuta la herramienta WEKA. La interfaz de usuario dinámica de esta herramienta facilita el modelado predictivo y el análisis de datos. La matriz de confusión, que incluye los resultados de la F-Score, curva ROC, Área bajo la curva, integridad, exactitud y precisión, se genera al finalizar la ejecución.

2.6. Criterios éticos.

Confidencialidad: Los autores, participantes y otras personas físicas o jurídicas implicadas en este estudio, junto con las fuentes de información de redes externas utilizadas para la creación y compilación del conjunto de datos, permanecerán en el anonimato.

Derechos de autor: Nos adherimos a la necesidad ética de citar a los creadores de las fuentes analizadas, incluyendo tesis, libros y revistas científicas, ya que el plagio no es una opción en esta investigación.

Además de centrarnos en la reducción de los ciberataques para promover el bienestar general, nuestro estudio también ayudará en el análisis de la seguridad de la información, dado que los delitos cibernéticos continúan siendo un problema para la comunidad moderna.

III. RESULTADOS Y DISCUSIÓN

3.1. Resultados

Utilizando métricas como el uso de memoria, uso de CPU y el tiempo de respuesta promedio, se pudo sacar las siguientes conclusiones sobre el consumo de recursos del algoritmo de clasificación mientras se entrena la colección de datos.

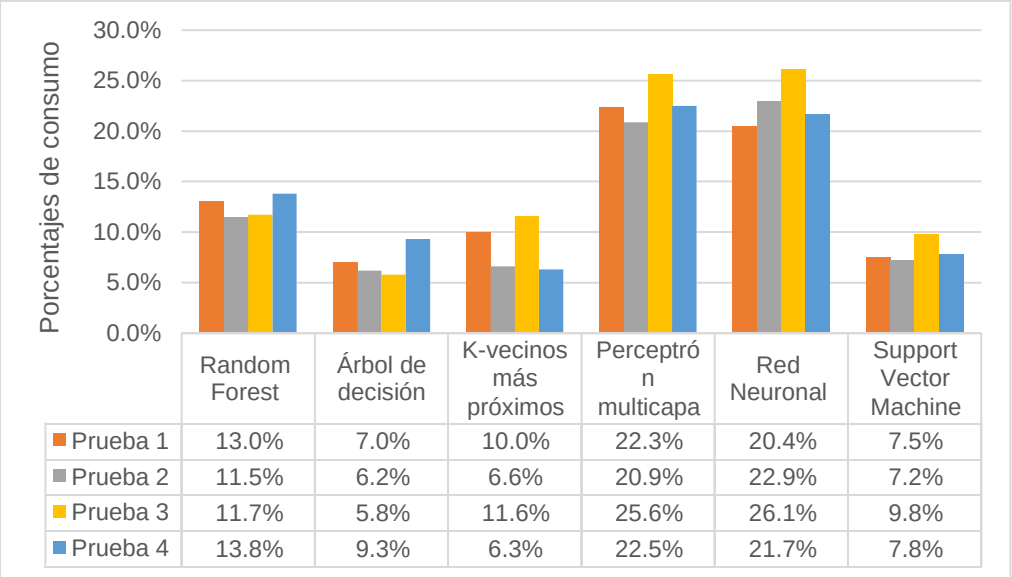
A continuación, se presenta una figura de ejemplo en tiempo promedio de respuesta.

Fig. 16. Tiempo promedio de respuesta según algoritmo de clasificación

En la Figura 16 se observa el tiempo empleado en utilizar los datos de entrenamiento por cada método de clasificación. Utilizando su técnica de propagación hacia delante, el algoritmo de clasificación de la red neuronal consiguió un tiempo de 0,96 s. El perceptrón multicapa, otro miembro de la familia de las redes neuronales, consiguió un tiempo de 1,06 segundos utilizando los métodos de propagación hacia delante y hacia atrás. El método de clasificación Support Vector Machine, que transfiere los datos a un lugar en el que se puede llevar a cabo una clasificación lineal, logró un tiempo de 0,04 segundos empleando la función kernel conocida como base radial.

Utilizando la minería de información y el índice de Gini para seleccionar parámetros, el método de clasificación del árbol de decisión puede producir los resultados deseados en 0,02 s. Random Forest es un clasificador que mantiene la base del algoritmo del árbol de decisión y logra un tiempo de 0,10 s. Su objetivo principal es crear múltiples árboles de decisión, examinar los resultados de cada árbol de decisión y determinar si una consulta es benigna o maliciosa en función de la mayoría de las respuestas emparejadas. El algoritmo K-Nearest Neighbors encontró finalmente la mejor distancia entre los k vecinos más próximos en 0,13 segundos. Para mejorar el rendimiento del modelo, esta estrategia divide las consultas en benignas y maliciosas. Además, k se representa con un número impar.

A continuación, se presenta una figura de ejemplo en consumo de CPU.



	Random Forest	Árbol de decisión	K-vecinos más próximos	Perceptrón multicapa	Red Neuronal	Support Vector Machine
Prueba 1	13.0%	7.0%	10.0%	22.3%	20.4%	7.5%
Prueba 2	11.5%	6.2%	6.6%	20.9%	22.9%	7.2%
Prueba 3	11.7%	5.8%	11.6%	25.6%	26.1%	9.8%
Prueba 4	13.8%	9.3%	6.3%	22.5%	21.7%	7.8%

Fig. 17. Consumo de CPU según algoritmo de clasificación

Según las estadísticas de la Figura 17, los algoritmos Perceptrón multicapa y Red neuronal de la familia de redes neuronales son los que más CPU utilizan. En cambio, el Perceptrón Multicapa modifica los pesos sinápticos de cada capa de la red aplicando el método BackPropagation. Estas técnicas mantienen un porcentaje diminutivo del 20,4% y un máximo porcentaje del 26,1% al entrenar los datos. En cuanto al consumo de grados de CPU, Random Forest obtuvo mejores resultados que Support Vector Machine, con una distinción del 4,3% en los valores mínimos; K-Nearest Neighbors obtuvo peores resultados que Support Vector Machine en un 3,6%. Por el contrario, tras entrenar el conjunto de datos con un grado diminutivo del 6,2%, Árbol de decisión se reveló como uno de los preferibles algoritmos y fue el enfoque con menor consumo de CPU.

A continuación, se presenta una figura de ejemplo en consumo de memoria.

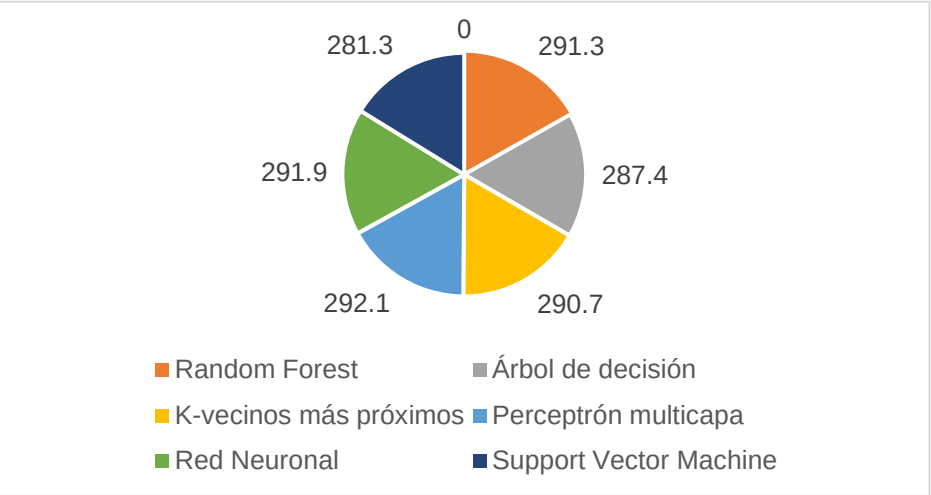


Fig. 18. Consumo de memoria en Megabytes según algoritmo de clasificación

En la Figura 18 se muestran los megabytes de RAM utilizados por los algoritmos de clasificación. Esto muestra qué algoritmo, cuando se entrena con el conjunto de datos proporcionado, requiere menos memoria.

Los algoritmos Red neuronal y Perceptrón multicapa arrojan los mayores valores de consumo de memoria, que oscilan entre 291,9 MB y 292,1 MB. Random Forest y K-Nearest Neighbors quedan por detrás de Multilayer Perceptron en 0,6 MB. Support Vector Machine fue el método que menos memoria utilizó, con un consumo de sólo 281,3 MB.

Se evaluó la eficacia de la detección de ataques de inyección NoSQL utilizando métricas como precisión, exactitud, recuperación, F-Score y AUC. A partir de aquí, se utiliza un algoritmo de clasificación para preparar la colección de datos, y para cada indicador, se obtienen los porcentajes y la matriz de confusión siguientes.

Algoritmos de clasificación y sus resultados en la matriz de confusión:

Tabla II.

Matriz de confusión de árbol de decisión.

		Clasificación	
		Malicioso	Benigno
Reales	Benigno	44	261
	Malicioso	194	10

Entre los 509 elementos de la colección de datos, el algoritmo del árbol de decisión encontró 238 consultas maliciosas y 271 consultas benignas.

Por consiguiente, de las consultas benignas que se encontraron, 261 fueron predichas precisamente por el modelo, y 10 fueron proyectadas erróneamente como benignas cuando eran claramente hostiles.

De las consultas peligrosas que se encontraron, 44 fueron predichas incorrectamente por el modelo, que las creyó hostiles cuando en realidad eran benignas, y 194 fueron anticipadas correctamente.

Tabla III.

Matriz de confusión del algoritmo Random Forest.

		Clasificación	
		Malicioso	Benigno
Reales	Benigno	11	294
	Malicioso	162	42

Entre las 509 entradas de la colección de datos, Random Forest encontró 173 consultas maliciosas y 336 búsquedas benignas.

En consecuencia, de las consultas benignas identificadas, 294 fueron predichas correctamente por el modelo, mientras que 42 fueron pronosticadas erróneamente como benignas cuando en realidad eran perjudiciales.

De las consultas peligrosas que se observaron, 11 fueron pronosticadas incorrectamente por el modelo, que las creyó hostiles a pesar de ser inocuas, por lo que 162 de ellas fueron pronosticadas correctamente.

Tabla IIIV.

Matriz de confusión de Neural Networks y Multilayer Perceptron.

		Clasificación	
		Malicioso	Benigno
Reales	Benigno	8	297
	Malicioso	200	4

Entre las 509 entradas del conjunto de datos, la red neuronal multicapa y el algoritmo perceptron detectaron 208 consultas maliciosas y 301 consultas benignas.

En consecuencia, de las búsquedas benignas descubiertas, 297 consultas fueron predichas correctamente por el modelo, y 4 consultas se proyectaron erróneamente como benignas cuando en realidad eran perjudiciales.

De las consultas peligrosas que se vieron, ocho fueron predichas incorrectamente por el modelo, que las creyó hostiles, aunque eran inocuas, con lo que 200 de ellas fueron previstas correctamente.

Tabla IV.

Matriz de confusión de Support Vector Machine.

		Clasificación	
		Malicioso	Benigno
Reales	Benigno	27	278
	Malicioso	154	50

Entre las 509 entradas del DataSet, Support Vector Machine (SVM) encontró 181 consultas maliciosas y 384 consultas benignas.

En consecuencia, de las consultas benignas identificadas, 278 consultas fueron predichas correctamente por el modelo, y 50 consultas fueron pronosticadas erróneamente como benignas cuando en realidad eran perjudiciales.

Sin embargo, de las consultas peligrosas encontradas, 154 fueron previstas correctamente y 27 fueron pronosticadas incorrectamente debido a que el modelo interpretó erróneamente algunas de las consultas como maliciosas cuando sin duda eran benignas.

Tabla VI.

Matriz de confusión de KNN.

		Clasificación	
		Malicioso	Benigno
Reales	Benigno	10	295
	Malicioso	151	53

Entre las 509 entradas del conjunto de datos, el algoritmo K-NN encontró 161 búsquedas maliciosas y 348 consultas benignas.

En consecuencia, de las consultas benignas identificadas, 295 consultas fueron predichas correctamente por el modelo, mientras que 53 consultas fueron predichas erróneamente como benignas cuando en realidad eran perjudiciales.

Pero de todas las consultas peligrosas que se encontraron, 151 de ellas fueron anticipadas correctamente, y 10 de ellas fueron pronosticadas incorrectamente ya que el modelo confundió su benignidad con su malicia.

Resultados del entrenamiento para cada indicador y algoritmo de clasificación en la base de datos:

Tabla VII.

Métricas de rendimiento basadas en indicadores de varios clasificadores.

Algoritmo Clasificador	**Exactitud**	**Precisión**	**Recall**	**F-Score**	**AUC**
Support Vector Machine	84.9%	84.8%	91.1%	87.6%	83.3%
Neuronal Network	97.6%	98.7%	97.4%	98.0%	98.9%
K-NN	87.6%	84.8%	96.7%	90.4%	96.6%
Multilayer Perceptron	97.6%	98.7%	97.4%	98.0%	98.9%
Random Forest	89.6%	87.5%	96.4%	91.7%	97.4%
Decision Tree	89.4%	96.3%	85.6%	90.6%	94.6%

Los resultados manifiestan que las redes neuronales y los perceptrones multicapa funcionan mejor en las métricas de rendimiento de detección de asaltos de inyección. Esto se debe a que suele haber pronósticos más precisos que incorrectos. Esto es posible gracias a una red neuronal y sus capas ocultas: cuantas más capas ocultas, más precisas son las predicciones. El umbral de clasificación es otra consideración importante; cuanto más alto sea el umbral y más estricta sea la clasificación, más precisa será la estimación de si una consulta es maliciosa o benigna

El núcleo del algoritmo Support Vector Machine intentó añadir una nueva dimensión a un hiperplano para distinguir entre consultas benignas y maliciosas, pero fue incapaz de dar con una solución viable. Como resultado, predijo incorrectamente un número significativo de consultas. Por eso el algoritmo produjo resultados bajos, pero no terribles

Random Forest conserva un porcentaje de AUC más alto que el de árbol de decisión porque, si bien las predicciones precisas pueden tener errores, estos errores pueden aumentar o disminuir según el umbral de clasificación. Esto se debe a que el modelo Random Forest tiene un umbral ideal de separación entre consultas maliciosas y benignas; un AUC del 50% indica que la técnica no es capaz de diferenciar entre consultas benignas y maliciosas, lo cual es muy preocupante.

Dentro del conjunto de datos, el Recall cuantifica la proporción de búsquedas benignas anticipadas con éxito y la precisión cuantifica el porcentaje de consultas perjudiciales predichas correctamente. Analizar el Recall y la precisión es necesario para estimar correctamente la eficacia del modelo; sin embargo, suele haber un conflicto en este punto, ya que aumentar el Recall reduciría la precisión y aumentar la precisión reduciría el Recall. La puntuación F, que es el promedio armónico de la precisión y el Recall, puede utilizarse para resolver este problema. Su peor valor es 0, que indica que la precisión y el Recall son perfectas, y su preferible valor es 1, que representa que la puntuación F no puede ser superior a la precisión.

A continuación, se presenta una figura de ejemplo en evaluación de efectividad.

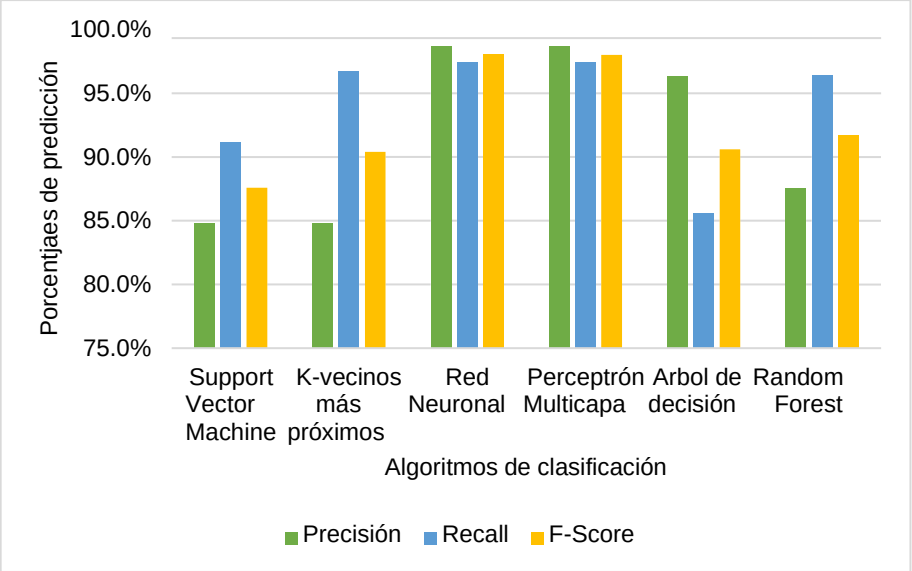


Fig. 19. Evaluación de la efectividad del modelo con F-Score

Las estadísticas muestran que las redes neuronales, los perceptrones multicapa y los árboles de decisión tienen una mejor eficiencia del modelo porque el promedio armónico o la puntuación F no supera la precisión.

Hay una distinción del 0,7% en la puntuación F y la precisión entre los algoritmos perceptrón multicapa y red neuronal. En el método del árbol de decisión, se descubrió una discrepancia del 5,7% entre estas indicaciones.

3.2. Discusión

El estudio "Automatic Detection of NoSQL Injection Using Supervised Learning" de Rafid, Saiful, Iqbal y Shahriyar se enfocó en el uso de algoritmos para detectar ataques de inyección de bases de datos NoSQL durante el inicio de sesión. Las redes neuronales obtuvieron los mejores resultados en términos de recuerdo (92,93%), exactitud (91,87%) y precisión (93,55%). Las conclusiones de Saiful , Rafid, Iqbal y Shahriyar, según las cuales el método de redes neuronales es el más eficaz para prever los ataques de inyección, se ven corroboradas por este estudio. El perceptrón multicapa y las redes neuronales produjeron puntuaciones de exactitud, precisión y recuperación del 97,6%, 98,7% y 97,4%, respectivamente. Por ello, se cree que el método de redes neuronales es el que mejor funciona para detectar riesgos de inyección en bases de datos NoSQL.

En la investigación "Detection of SQL Injection Attacks: A Machine Learning Approach" de Hasan, Balbahaith y Tarique. La técnica del árbol de decisión arrojó una precisión del 93,8%, mientras que el algoritmo SVM produjo una precisión del 93,5%. La diferencia entre ambos resultados -que atribuimos al número de registros utilizados para preparar los algoritmos- es del 8,7%; la precisión aumenta con el número de registros. Las conclusiones del estudio muestran que un porcentaje considerable de los datos se identificaron correctamente, lo que subraya la necesidad de elegir variables de alta calidad al principio del procedimiento de entrenamiento. Además, se subraya que la técnica del árbol de decisión es una de las mejores técnicas para identificar los riesgos de inyección de almacén de datos.

Weidong Qiu, Peng Tang, Zheng Huang, Guozhen Liu y Huijuan Lian utilizaron redes neuronales como algoritmo de clasificación en su investigación "Detection of SQL Injection Based on Artificial Neural Network" (Detección de inyecciones SQL basada en redes neuronales artificiales) para descubrir asaltos de inyección en bases de datos SQL. Alcanzaron una tasa de precisión del 99,86%, del 99,95% y del 98,8%. Utilizando sólo 509 de las 10.000 instancias que utilizaron para su estudio, fueron capaces de obtener índices de precisión del 97,6%, 98,7% y 97,4% de integridad. Esto implica que nuestro conjunto de datos fue capaz de alcanzar un porcentaje comparable en las medidas de clasificación descritas anteriormente, aunque con un número menor de muestras. Es importante destacar que no se ha revelado la cantidad de atributos incluidos en su conjunto de datos.

En 509 muestras de la colección de datos, la duración de procesamiento del estudio actual fue de 0,0000 segundos por consulta. Por otra parte, se emplearon árboles de decisión como clasificadores para detener inyecciones SQL en el estudio "Machine Learning for SQL Injection Prevention on Server-Side Scripting" de Krit Kamtuo y Chitsutha Soomlek. La duración de procesamiento del modelo en su colección de datos fue de 0,00334 segundos para cada consulta de 500 muestras. Comparativamente, los 500 ejemplos del conjunto de datos del estudio requirieron un tiempo de procesamiento de 0,00334 segundos por consulta, pero los 509 ejemplos del conjunto de datos del presente estudio sólo requirieron 0,0000 segundos. Aunque este estudio utilizó el mismo algoritmo de clasificación para mejorar el tiempo de ejecución, es vital tener en cuenta que los almacenes de datos no son iguales porque una es relacional y la otra no.

3.3. Aporte de la investigación

Mediante el examen de la bibliografía académica hasta 2020, el presente estudio pretendía determinar los mejores algoritmos de aprendizaje automático para la categorización. Los resultados revelaron lo siguiente:

Tabla VIII

Los algoritmos de aprendizaje automatizado más efectivos empleados en clasificación.

N°	Clasificador	Exactitud	Caso de Estudio	Autor(es)
1	Naive Bayes	78.00%	Identificar el estado de salud en pacientes patológicos	[39]
2	Detector Automático de Interacción Chi-cuadrado (CHAID)	73.20 %	Padecimiento de pie diabético en pacientes con diabetes mellitus tipo II.	[37]
3	Decision Tree	93.40%	Inyección SQL	[38]
4	Dicotomizador Iterativo (ID3)	91.66%	Clasificación de imágenes de textura Brodatz	[36]
5	Regresión Ordinaria Por Mínimos Cuadrados	90.00%	Detección de teofilina en la sangre	[40]
6	K-Medios	91.61%	Segmentación de imágenes por medio del color	[41]
7	K Vecinos más próximos (KNN)	92.00%	Detección de inyección NoSQL	[9]

N°	Clasificador	Exactitud	Caso de Estudio	Autor(es)
8	Algoritmos Basados en la Densidad	74.00%	Detección de inyección SQL	[42]
9	Neuronal Network	95.30%	Inyección SQL	[38]
10	Análisis de elementos principales (ACP)	83.30%	Padecimiento de pie diabético en pacientes con diabetes mellitus tipo II.	[37]
11	Análisis de elementos independientes (ACI)	84.00%		
12	Regresión Logística	91.50%	Inyección SQL	[38]
13	Adaboost	91.78%	Detección de inyección NoSQL	[9]
14	Modelo oculto de Markov	73.87%	Reconocimiento de entidades con nombre para él español	[43]
15	Support Vector Machine (SVM)	95.40%	Inyección SQL	[38]
16	Random Forest	93.60%	Inyección SQL	[38]
17	Xgboost	89.51%	Detección de inyección NoSQL	[6]
18	Multilayer Perceptron	95.30%	Inyección SQL	[38]

Los 18 principales algoritmos de clasificación de aprendizaje automático desarrollados en los últimos cinco años se utilizan en casos de uso como inyección SQL, inyección NoSQL, clasificación de imágenes, afecciones del pie diabético, detección de teofilina en sangre, identificación de pacientes patológicos y reconocimiento de entidades.

Como resultado, en el estudio de clasificación de imágenes, K-NN alcanzó una precisión de la métrica de rendimiento de clasificación del 92,00%; en el caso de la identificación de pacientes patológicos, Naive Bayes alcanzó una precisión del 78,00%; y en el algoritmo Independent Feature Analysis (ICA), una precisión del 84,00%. Las métricas de clasificación obtenidas por Red neuronal, Perceptrón multicapa, Árbol de decisión, Bosque aleatorio y Máquina de vectores de apoyo fueron, por este orden, del 95,30%, 95,30%, 93,60%, 93,40% y 95,40% de precisión.

Dado que la precisión es un parámetro de rendimiento de la clasificación que indica la proporción de predicciones verdaderas, se utilizó para elegir los algoritmos que obtuvieron mejores resultados en los enfoques de clasificación del estudio de caso. Como resultado, los algoritmos más precisos se muestran en la Tabla 10.

Tabla VIIIX.

Algoritmos con mayor prioridad en exactitud.

N°	Clasificador	Exactitud	Caso de Estudio
1	Neuronal Network	95.30%	Inyección SQL
2	Random Forest	93.60%	Inyección SQL
3	Support Vector Machine	95.40%	Inyección SQL
4	K-NN	92.00%	Segmentación de imágenes por medio del color
5	Decision Tree	93.40%	Inyección SQL
6	Multilayer Perceptron	95.30%	Inyección SQL

Las siguientes conclusiones se extrajeron del estudio realizado en 2020 para categorizar y describir las distintas consultas benignas y perjudiciales en las bases de datos NoSQL:

Tabla IX.

Descripción de consultas seguras e inseguras en bases de datos NoSQL.

N°	Consulta	Tipo	Operador	Reseña	Almacen de datos	Caso	Fuente
1	db.usuarios.find({ usuario:valor_1, contrasenia:valor_2})	Benigna	Ninguno	Dependiendo de los parámetros y valores dados, podemos extraer datos de la colección de usuarios utilizando este tipo de consulta.	MongoDB	Sesiones	https://blog.websecurify.com/2014/08/hacking-nodejs-and-mongodb.html
2	db.usuarios.find({ usuario:{$gt:””}, contraseña: {$gt:””})	Maliciosa	$gt	Cuando este tipo de consulta se modifica utilizando el operador de similitud «$gt», que significa mayor que y se entiende que se compara con una cadena vacía y	MongoDB	Sesiones	https://blog.websecurify.com/2014/08/hacking-nodejs-and-mongodb.html

				devuelve una afirmación verdadera, se convierte en una consulta peligrosa.			
3	db.logins.find({ username:valor_1, password: valor_2	Benigna	Ninguno	Con este tipo de consulta, podemos extraer datos de la colección de inicios de sesión basándonos en los valores y condiciones que se recogieron.	MongoDB	Sesiones	https://www.infoq.com/articles/nosql-injections-analysis/
4	db.logins.find({ username: {$ne:1}, password: {$ne: 1 })	Maliciosa	$ne	Si cambia este tipo de consulta utilizando el operador de comparación «$ne», que significa «no igual a», se vuelve peligroso. Dado que el operador $ne busca en la colección los inicios de sesión cuyo campo Nombre de usuario no	MongoDB	Sesiones	https://www.infoq.com/articles/nosql-injections-analysis/

				sea igual a 1 y cuya contraseña no sea igual a 1, la consulta devuelve todos los inicios de sesión de la colección.			
5	"var query = { $where: ""this.name === '''' + req.body.name + ''''''''" }"	Benigna	$where	Con este tipo de consulta, podemos obtener datos del usuario pasando una cadena con una expresión JavaScript a través del operador de evaluación $where.	MongoDB	Sesiones	https://zanon.io/posts/nosql-injection-in-mongodb
6	req.body.name = '\'; return \'\' == \''	Maliciosa	$where	Cuando se cambia el valor utilizando el operador de evaluación $where, la consulta se vuelve peligrosa. Esto provocaría que el valor introducido por la cadena cambiara a	MongoDB	Sesiones	https://zanon.io/posts/nosql-injection-in-mongodb

				(this.name ===''; return'' =''), lo que devolvería todos los usuarios.			
7	"db.collection.find({ $where: function() { return (this.name == $userData) } });"	Benigna	$where	Con este tipo de consulta, podemos obtener datos pasando una cadena con una expresión JavaScript a través del operador de evaluación **$where**.	MongoDB	Consulta en el Servidor	https://www.netsparker.com/blog/web-security/what-is-nosql-injection/
8	db.collection.find({ $where: function() { return (this.name == 'a'; sleep (5000))}});	Maliciosa	$where	Cuando un atacante es capaz de inyectar un exploit 'a'; sleep (5000) en la variable $userData, la consulta se vuelve hostil y provoca que el servidor se detenga durante cinco segundos.	MongoDB	Consulta en el Servidor	https://www.netsparker.com/blog/web-security/what-is-nosql-injection/
9	db.food.find({ $or : [(campo_a : 1 }	Benigna	$or y $and	Con este tipo de consulta, podemos utilizar los operadores		Búsqueda	http://oa.upm.es/43331/1/TFM_FRANCO_OL

	, { campo_b : 2 }] })			lógicos $or, que significa «o», y $and, que significa «y», para obtener datos de la colección.			IVIO_GUAMAN_BASTIDAS.pdf
10	db.food.find({ $or : [(a : 1 } , { b : 2 } , { c : /.*/] })	Maliciosa	$or y $and	Cuando se añade un carácter como /.*/ a la consulta, que siempre devuelve un resultado TRUE, el tipo de consulta se vuelve perjudicial. Los operadores lógicos $or y and están incluidos en esto.		Búsqueda	http://oa.upm.es/43331/1/TFM_FRANCO_OLIVIO_GUAMAN_BASTIDAS.pdf
11	$coleccion->find(array(“campo_1”=>”valor_1”), array(“campo_2”=>”valor_2”));	Benigna	Ninguno	Este tipo de consulta se basa en aplicaciones web que utilizan PHP para enviar una matriz - en este ejemplo, una	MongoDB	Búsqueda por URL	http://oa.upm.es/43331/1/TFM_FRANCO_OLIVIO_GUAMAN_BASTIDAS.pdf

				matriz JSON- con el fin de ejecutar la consulta.			
12	$coleccion->find(array(“campo_1”=>”valor_1”), array(“campo_2”=>”valor_2”));	Maliciosa	$regex	Podemos modificar los datos dados al servidor por el método HTTP GET utilizando la URL de la aplicación web, que es la siguiente: www.myejemplo.com/getproduct.php?ref[$regex]=.	MongoDB	Búsqueda por URL	http://oa.upm.es/43331/1/TFM_FRANCO_OLIVIO_GUAMAN_BASTIDAS.pdf
13	$query = array('$where' => 'this.name === \''.$name.'\');	Benigna	$where	Con este tipo de consulta, podemos obtener datos pasando una cadena con una expresión JavaScript a través del operador de evaluación **$where**.	CouchDB	Acceso a registros	https://www.acunetix.com/blog/web-security-zone/nosql-injections/
14	"$where": "this.name === '; while(true){}'	Maliciosa	$where	El operador $where se evalúa como código Java Script, crea un bucle infinito y causa un	CouchDB	Acceso a registros	https://www.acunetix.com/blog/web-security-zone/nosql-injections/

				ataque de denegación de servicio.			
15	$map = “function() { for (var i = 0; i < this.items.length; i++) { emit(this.name, this.items[i].$param); } }”; $reduce = “function(name, sum) { return Array.sum(sum); }”; $opt = “{ out: ‘totals’ }”; $db->execute(“db.stores. mapReduce($map, $reduce, $opt);”);	Benigna	Ninguno	Los atacantes modifican estas consultas para entregar código malicioso y cambiar datos. Se utilizan con frecuencia en ataques de inyección de bases de datos con funciones de expresión Java Script.	CouchDB	Uso de Funciones	https://www.infoq.com/articles/nosql-injections-analysis/

16	db.stores.mapReduce (function() { for (var i = 0; i < this.items.length; i++){emit(this.name, this.items[i].**a); } },function (kv) { return 1; }, { out: 'x' }); db.injection.insert({success:1}); return 1;db.stores.mapReduce(function(){{emit(1,1); }},function(name, sum) { return Array. sum(sum); }, { out: 'totals' });"**	Maliciosa	Ninguno	MapReduce se utiliza para crear búsquedas o transacciones complejas en JavaScript. La función mapreduce original se cierra en la primera sección. A continuación, la información de las apps se restaura concatenando funciones adicionales, que pueden implicar actualizaciones, inserciones o eliminaciones que comprometan la confidencialidad e integridad de los datos.	CouchDB	Uso de Funciones	https://www.infoq.com/articles/nosql-injections-analysis/

Según las consultas encontradas, con frecuencia existe la probabilidad de que se origine una brecha de seguridad durante el procedimiento de inicio de sesión. La mayoría de los ataques se ocasionan en escenarios en los que los motores de datos NoSQL tienen credenciales por defecto o incluso la autenticación deshabilitada. Dado que pueden utilizarse para comparar cualquier cosa, como una cadena libre en el caso de $gt y algo que no está en el almacén de datos en el caso de $ne, los operadores de comparación como $gt, que significa mayor que, y $ne, que significa no igual a, se utilizan con frecuencia. Por defecto, la equiparación en ambas situaciones arroja un resultado positivo, devolviendo todos los registros de la colección. Del mismo modo se encuentran variables como $regex, $where y operadores de suma como $or y $and. Estas variables utilizan una expresión de Java Script para controlar los datos que se envían al servidor utilizando el método HTTP GET.

En esta investigación se realizó un estudio para utilizar la colección de datos para el aprendizaje automático hasta 2020. Para seleccionar características deterministas y obtener información especializada sobre el dominio de aplicación, se emplearon técnicas precisas atributos (basados en el formato de documento JSON) y (algoritmos de aprendizaje automatizado). A partir de los datos recogidos de múltiples fuentes, fue posible dar cuenta de palabras y bases de datos generales en las preguntas más frecuentes.

A continuación, se describe cómo se creó el conjunto de datos:

La investigación comienza con la estructura JSON para la colección de atributos, ya que el conjunto de datos se creó examinando las diferentes consultas en los almacenes de datos MongoDB, que emplea objetos JSON (Java Script Object Notation) para el filtrado de atributos.

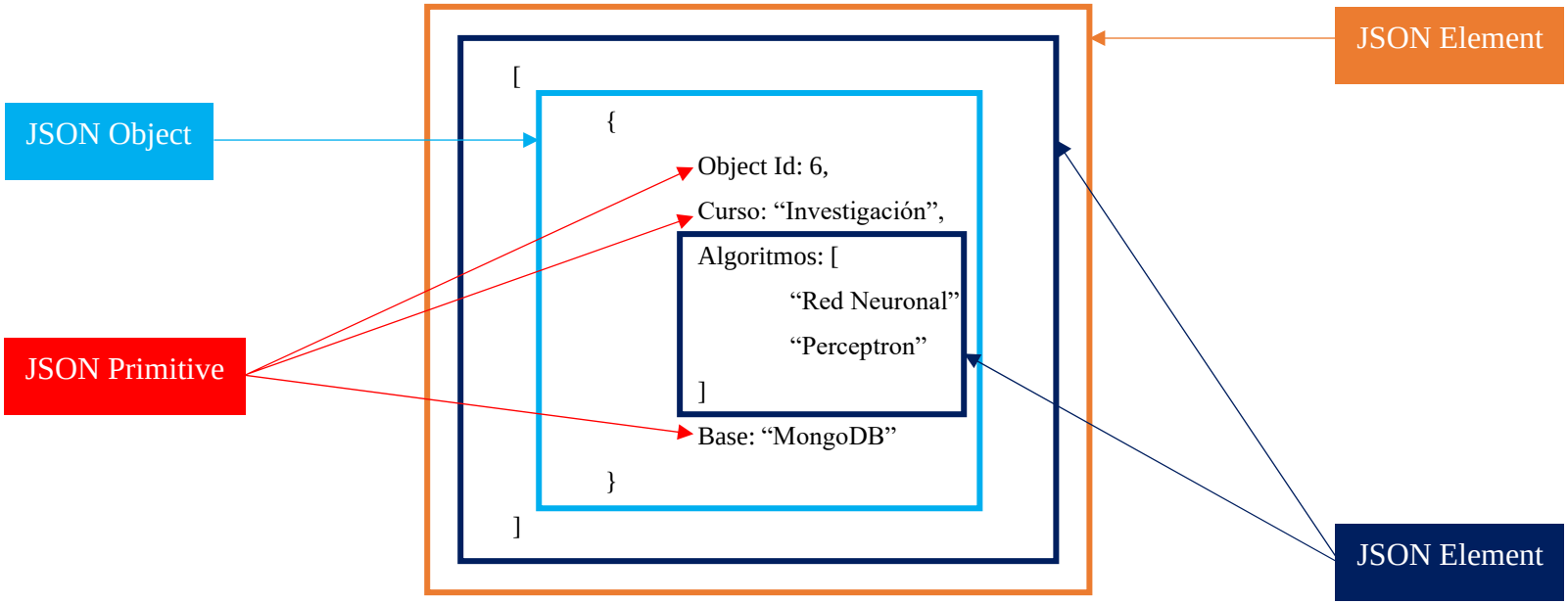
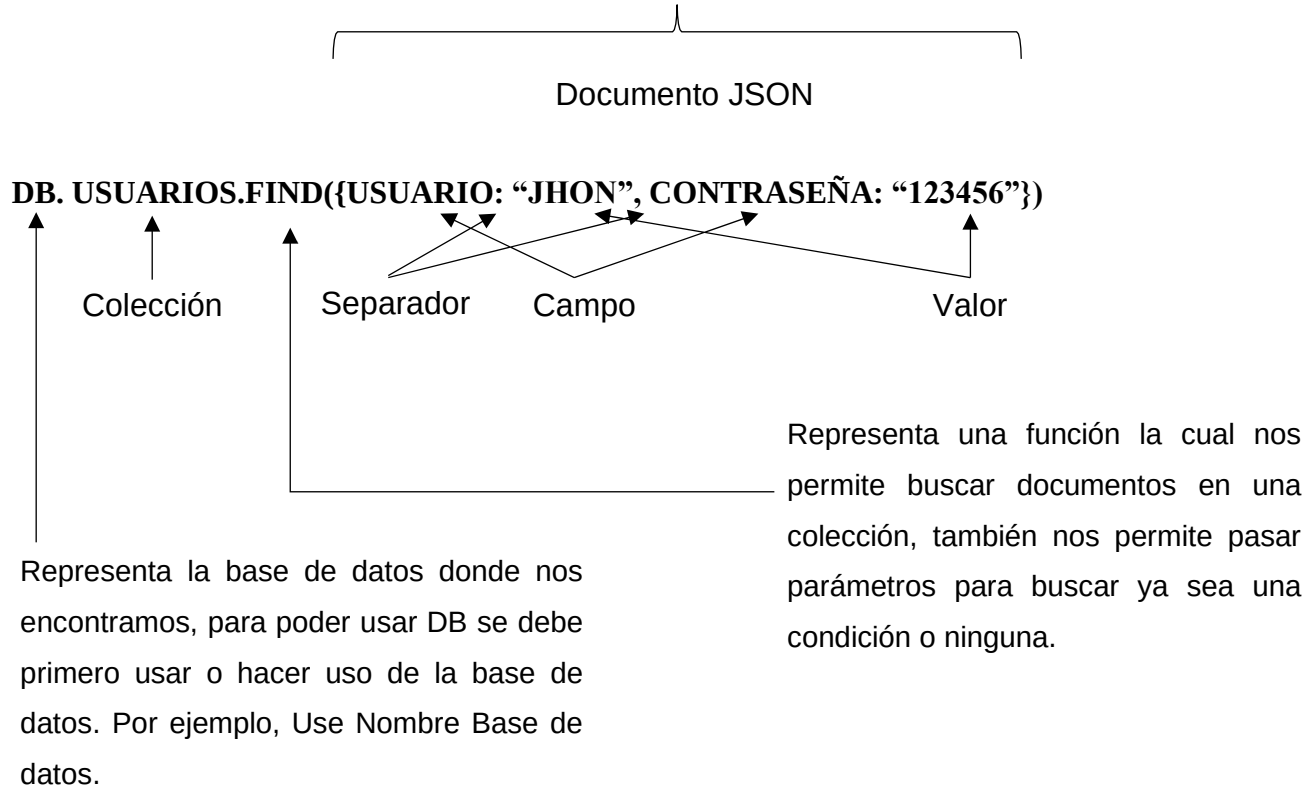


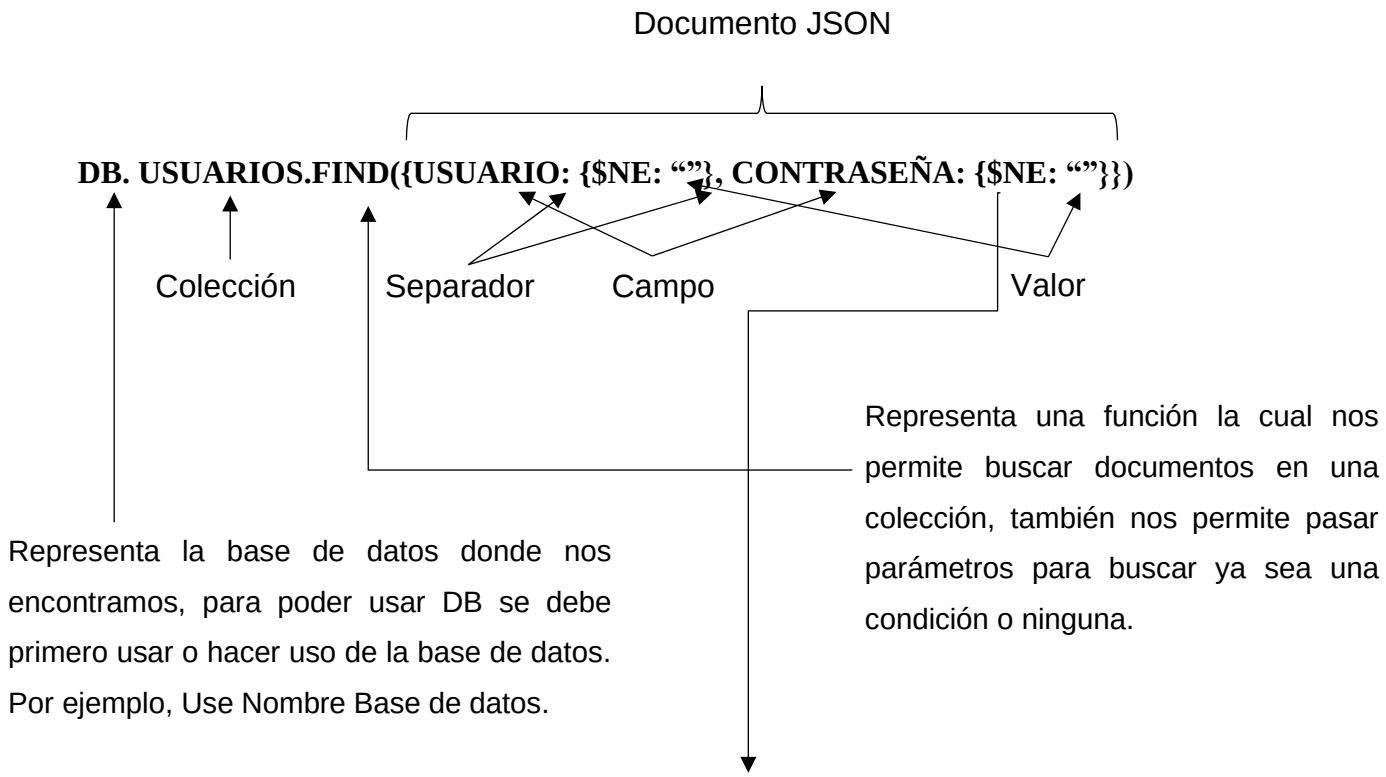
Fig. 20. Estructura de documento JSON.

La base de datos NoSQL MongoDB consulta diferentes datos según criterios especificados por el usuario utilizando la función FIND (). A continuación, ofrecemos dos búsquedas distintas para ilustrar y demostrar cómo un operador del mismo almacén de datos puede transformar de una consulta segura en una consulta arriesgada.

Consulta Benigna:



Consulta Maliciosa:



El conjunto de datos puede construirse utilizando los documentos JSON una vez que se conoce el formato JSON y se dispone de una consulta benigna y malintencionada de un operador lógico del mismo motor de datos. Para ello, se realizó un análisis exhaustivo en el que se dividió la estructura en varias columnas, a saber, el campo, el separador, el valor, el operador, el campo y el inicio y el final de la estructura JSON. En el cuadro 12 se especifica la construcción y las características que se eligieron.

Tabla XI.

Documentación de la Base de Datos.

Denominación del atributo	Descripción del atributo ¿Qué representa? ¿Cómo atribuye a identificar la etiqueta?	Probables valores	Descripción del probable valor	Tipo variable
Column1	Devuelve el comienzo de la estructura compatible con JSON. - Explica un carácter específico que aparece en el documento JSON. - Característica para determinar la malicia o benignidad de la consulta.	[0 - 255]	123: Averigua el código ASCII del primer carácter de inicio de JSON.	Continua
Column2	Corresponde al campo estructural que compone JSON.	[0 - 11]	Determinar el valor posible del operador en una consulta de JSON.	Discreta

	- Define un campo específico dentro de JSON. - Contribuye a determinar si la consulta es maliciosa o benigna.			
Column3	Devuelve el separador de la estructura en formato JSON. - Simboliza un campo específico en el documento JSON. - Característica para determinar la malicia o benignidad de la consulta.	[0 - 255]	58: Indica el carácter separador del código ASCII de JSON.	Continua
Column4	Devuelve el comienzo de la estructura compatible con JSON. - Explica un carácter específico que aparece en el documento JSON. - Característica para determinar la malicia o benignidad de la consulta.	[0 - 255]	123: Averigua el código ASCII del primer carácter de inicio de JSON.	Continua
Column5	Devuelve el carácter que pertenece al operador registrado en JSON.	[0 - 255]	36: Averigua el código ASCII del carácter dólar, que es un	Continua

	- Explica una característica del operador del documento JSON. - Característica para determinar la malicia o benignidad de la consulta.		componente del operador de JSON.	
Column6	Corresponde al nombre del operador de la estructura que contiene JSON. - Simboliza un operador específico en JSON. - Característica para determinar la malicia o benignidad de la consulta.	[0-6]	Determinar el valor posible del operador en una consulta de JSON.	Discreta
Column7	Devuelve el separador de la estructura en formato JSON. - Simboliza un campo específico en el documento JSON. - Característica para determinar la malicia o benignidad de la consulta.	[0 - 255]	58: Indica el carácter separador del código ASCII de JSON.	Continua
Column8	Corresponde a un valor de la consulta dentro de la estructura JSON. - Explica un valor específico encontrado en la consulta.	[0-4]	Simboliza un valor que puede incluirse en una consulta de JSON.	Discreta

	- Característica para determinar si una consulta es dañina o benigna.			
Column9	Es equivalente al cierre de JSON. - Indica un carácter único que aparece en JSON. - Característica para determinar la malicia o benignidad de la consulta.	[0 - 255]	125: Averigua el código ASCII del carácter de cierre de JSON.	Continua
Column10	Corresponde el separador de la estructura que forma parte de JSON. - Representa un carácter en especial dentro del documento JSON. - Atribuye a identificar si la consulta es benigna o maliciosa.	[0 - 255]	44: Indica el carácter separador del código ASCII de JSON.	Continua
Column11	Coincide con el campo de la estructura que es un componente de JSON. - Simboliza un campo específico de JSON.	[0 - 11]	Determinar el valor posible del operador en una consulta de JSON.	Discreta

	- Característica para determinar la malicia o benignidad de la consulta.			
Column12	Devuelve el separador de la estructura en formato JSON. - Simboliza un campo específico en el documento JSON. - Característica para determinar la malicia o benignidad de la consulta.	[0 - 255]	58: Indica el carácter separador del código ASCII de JSON.	Continua
Column13	Devuelve el comienzo de la estructura compatible con JSON. - Explica un carácter específico que aparece en el documento JSON. - Característica para determinar la malicia o benignidad de la consulta.	[0 - 255]	123: Averigua el código ASCII del primer carácter de inicio de JSON.	Continua
Column14	Devuelve el carácter que pertenece al operador registrado en JSON. - Explica una característica del operador del documento JSON.	[0 - 255]	36: Averigua el código ASCII del carácter dólar, que es un componente del operador de JSON.	Continua

	- Característica para determinar la malicia o benignidad de la consulta.			
Column15	Corresponde al nombre del operador de la estructura que contiene JSON. - Simboliza un operador específico en JSON. - Característica para determinar la malicia o benignidad de la consulta.	[0 - 6]	Determina el valor posible del operador en una consulta de JSON.	Discreta
Column16	Devuelve el separador de la estructura en formato JSON. - Simboliza un campo específico en el documento JSON. - Característica para determinar la malicia o benignidad de la consulta.	[0 – 255]	58: Indica el carácter separador del código ASCII de JSON.	Continua
Column17	Equivale al valor de una consulta dentro de JSON. - Representa un valor específico de la consulta.	[0 - 4]	Simboliza un valor que puede incluirse en una consulta de JSON.	Discreta

	- Característica para determinar la maliciosidad o benignidad de la consulta.			
Column18	Es equivalente al cierre de JSON. - Indica un carácter único que aparece en JSON. - Característica para determinar la malicia o benignidad de la consulta.	[0 - 255]	125: Averigua el código ASCII del carácter de cierre de JSON.	Continua
Column19	Es equivalente al cierre de JSON. - Indica un carácter único que aparece en JSON. - Característica para determinar la malicia o benignidad de la consulta.	[0 - 255]	125: Averigua el código ASCII del carácter de cierre de JSON.	Continua

A continuación, se presenta el grafico de la construcción del Dataset.

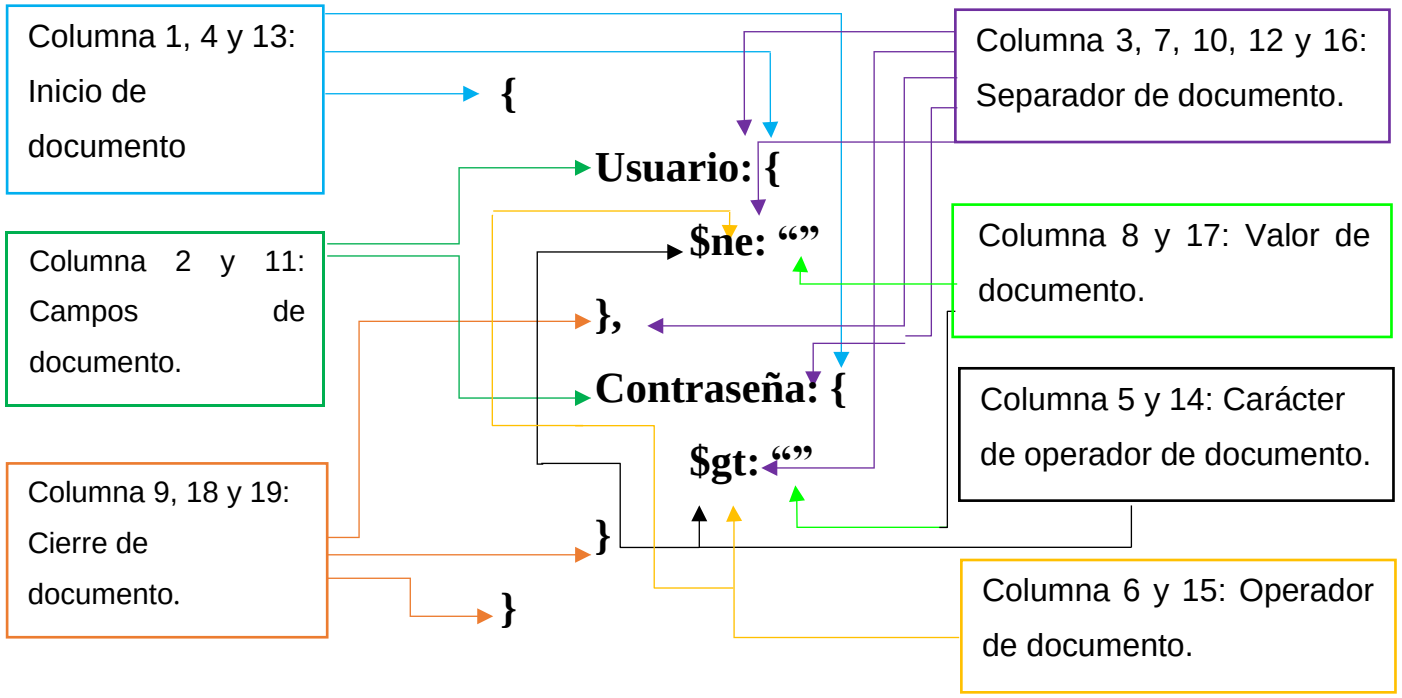


Fig. 21. Representación Gráfica de construcción de Dataset.

Tabla XII.

Explicación de cada etiqueta.

Designación de la etiqueta	**Clase (probables valores)**	**Descripción de la clase**
Query	Benigna	extrae una conclusión de la consulta basándose en los atributos suministrados.
	Maliciosa	extrae una conclusión de la consulta basándose en los atributos suministrados.

Para elegir los superiores algoritmos de clasificación de aprendizaje automático, se realizó un estudio hasta 2020. Modificando sus hiperparámetros, se eligieron seis métodos que proporcionan mejores modelos entrenados. Para asegurarse de que el modelo no está sobre ajustado, hay que tener en cuenta la coherencia de los indicadores de rendimiento en los procesos de entrenamiento y validación de los conjuntos de datos.

Implementación del Decision Tree (Árbol de decisión).

El objetivo general de este clasificador es catalogar las consultas como benignas o maliciosas. El conjunto de datos y el nodo inicial, también conocido como nodo padre, son los objetos principales del algoritmo.

Cada nodo ejecuta una consulta basada en los posibles valores del atributo; el árbol dirige la ruta en función del resultado.

A continuación, se presenta una figura que describe el funcionamiento de un árbol.

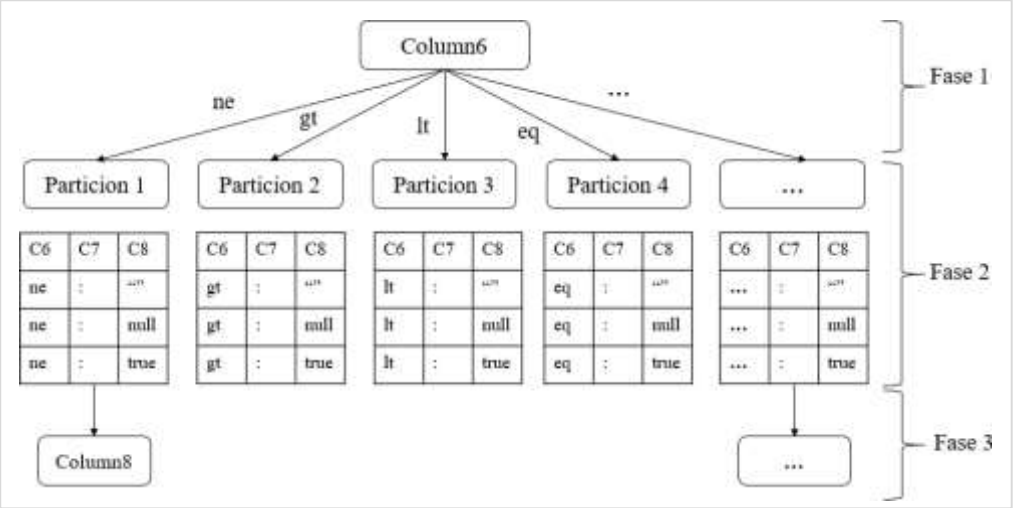


Fig. 22. Funcionamiento del algoritmo árbol de decisión.

Para facilitar la comprensión, en la Figura 22 se observa un ejemplo por fases de cómo construir un árbol de decisión utilizando la colección de datos. La columna 6 se elige como nodo raíz para comenzar el desarrollo del árbol en la fase 1. Para este nodo se selecciona el mejor atributo de la colección de datos y se aplican las condiciones "ne" y "ne". Para este nodo se selecciona el mejor atributo de la colección de datos y, a continuación, se comparan las condiciones "ne", "gt", "lt" y "eq". La creación de divisiones según los criterios especificados y las filas que puede ocupar el atributo es la fase dos. A estas divisiones se les aplican todas las ramas posibles de la propiedad Columna6, que se realizan para el conjunto de datos completo. La tercera fase consiste en seleccionar y añadir a las posibles ramas las mejores características de cada división; este proceso se repite para Columna8.

A continuación, se presenta una figura que describe como utilizar la librería RepTree.

```
public static double[] getQuery(int numAttributes) {
      double[] column;
      column = new double[numAttributes];
      column[1]=123;
      column[2]=0;
      column[3]=58;
      column[3]=123;
      column[…]=…;
      return column;
   }

Classifier repTree = new weka.classifiers.trees.REPTree();
Instances trainingData = GenerateTestVessels.getData();
repTree.buildClassifier(trainingData);
System.out.println(repTree);
```

Fig. 23. Código al utilizar la librería RepTree.

Como se ilustra en la Figura 23, los valores se devuelven en una matriz al final del procesamiento de cada columna. Esto crea una matriz doble [] para cada partición, que comprende todos los atributos, en este caso, las columnas. El objetivo de este procedimiento es categorizar cada muestra. El árbol se entrena simplemente instanciando la clase, invocando la función buildClassifier(), proporcionando los datos de entrenamiento y entrenando el algoritmo utilizando el paquete weka.classifiers.trees y RepTree.

Implementación del algoritmo Random Forest

El núcleo de esta técnica es la combinación de varios ejemplares de árboles de decisión para producir respuestas sustentadas en patrones. Cada árbol de decisión se prepara utilizando una colección de datos separada e independiente, ya que cada modelo se inicia con un subconjunto de datos distinto.

A continuación, se presenta una figura que describe el funcionamiento de Random Forest.

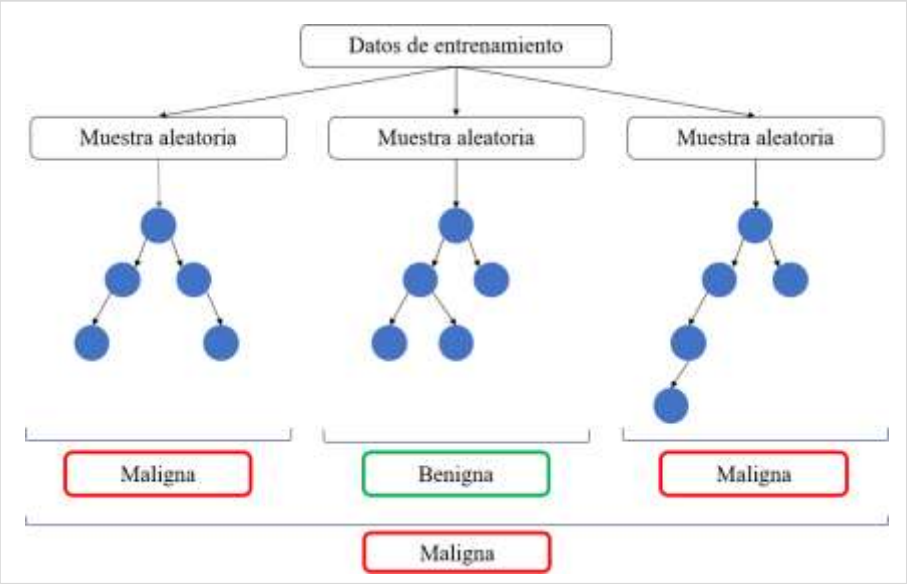


Fig. 24. Funcionamiento de Random Forest.

En la figura 24 se analiza la subdivisión del conjunto de datos, la creación de varios modelos de árboles de decisión y su integración. Por consiguiente, para entrenar a cada árbol de decisión a abordar el mismo problema -a saber, determinar si una consulta es benigna o maliciosa- se utilizan muestras de datos diferentes. Combinando los resultados de cada clasificador, el ensacado reduce la varianza de la predicción y aumenta la precisión ajustando algunos errores.

A continuación, se muestra el código para utilizar Random Forest:

```
/**Librerías*/
import weka.classifiers.Classifier;
import weka.classifiers.Evaluation;
import weka.classifiers.trees.RandomForest;

public class RandomForestDemo {

   /** definir el nombre del conjunto de datos*/
   public static final String TRAINING_DATA_SET_FILENAME="Dataset_Training.arff";

   public static Instances getDataSet(String fileName) throws IOException {
   /** se configura el archive para cargar los datos*/
      int classIdx = 1;
      /** el arffloader para cargar el archivo*/
            ArffLoader loader = new ArffLoader();
      /** cargar los datos de entrenamiento*/
      loader.setSource(RandomForestDemo.class.getResourceAsStream("/" + fileName));
   }

   /** Se procesa la entrada para devolver la estadística*/
   public static void process() throws Exception {
      Instances trainingDataSet = getDataSet(TRAINING_DATA_SET_FILENAME);
            RandomForest forest=new RandomForest();
            forest.setNumTrees(10); /** cantidad de arboles de decisión*/
      forest.buildClassifier(trainingDataSet);
      /** Se entrena el algoritmo con los datos de entrenamiento*/
      Evaluation eval = new Evaluation(trainingDataSet);

       /** Imprime el resumen del algoritmo*/
      System.out.println("** Evaluación de los árboles de decisión con el conjunto de datos**");
      System.out.println(eval.toSummaryString());
      System.out.print(" la expresión para los datos según el algoritmo es:");
      System.out.println(forest);
```

Fig. 25. Código al utilizar la librería Random Forest.

En la Figura 25 se observa el proceso algorítmico para entrenar la colección de datos y describe cómo cargar el archivo y sus datos de entrenamiento. Una vez dividido la colección de datos, se decide la cantidad de árboles de decisión que deben construirse.

Después, la colección de datos se maneja para entrenar el algoritmo, de modo que pueda producir la evaluación del árbol de decisión y comunicar datos que el algoritmo ha determinado.

Implementación de Red Neuronal

Al igual que el cerebro puede adquirir intelecto mediante procesos de aprendizaje, las redes neuronales también pueden retener los conocimientos adquiridos. Mantienen las capas de entrada, encubierta y de salida como sus tres niveles.

$$y = \sum_{i}^{n} x_i w_i$$

Donde:

y: Indica las etiquetas de la colección de datos, que pueden ser benignas o maliciosas.

n: Representa el número de ejemplares presentes en la colección de datos.

x: Es el parámetro dependiente que comprende los atributos de la colección de datos.

w: Es el peso de la vinculación entre las capas de las neuronas de entrada y de encubierta.

A continuación, se muestra el siguiente algoritmo en código fuente:

```
package weka.classifiers.functions;
/**Librerías*/
import weka.classifiers.Classifier;
import weka.classifiers.evaluation.NumericPrediction;
import weka.classifiers.timeseries.WekaForecaster;
import java.util.List;

public Instances getWineData() throws Exception {
      /**Se carga el conjunto de datos*/
      ConverterUtils.DataSource ds = new ConverterUtils.DataSource("Dataset_Training.arff ");
      return ds.getDataSet();
}
public Classifier configureRNN() {
      /**Se configura la red neuronal para la extracción de los datos*/
      RnnForecaster cls = new RnnForecaster();
      /**Se utiliza el paquete wekaDeeplearning4j y se configura modelos RNN con capas LSTM*/
      LSTM lstm = new weka.dl4j.layers.LSTM();
      /**Función de activación*/
      lstm.setActivationFn(new ActivationTanH());
      lstm.setNOut(10);
      /**Capas de salida*/
      RnnOutputLayer out = new RnnOutputLayer();
      out.setLossFn(new LossMSE());
      cls.setLayers(lstm, out);
      cls.setNumEpochs(2000);
      return cls;
}
```

Fig. 26. Código al utilizar la librería RnnForecaster.

La Figura 26 muestra cómo se utilizó la técnica de red neuronal y la herramienta WEKA para entrenar la colección de datos. El método getDataSet() se maneja para obtener la colección de datos. Las capas LSTM se configuran en la conformación final de la red neuronal, que emplea WEKA.DL4J. La información puede conservarse añadiendo bucles a la red, permitiendo a la LSTM recapitular estados anteriores y manejar ese conocimiento para mejorar los pronósticos.

Implementación de Multilayer Perceptron (Perceptrón Multicapa)

El MP usa la estrategia de backpropagation, que busca repartir el error de la red entre las neuronas de las capas encubiertas, para modificar los valores del umbral y los pesos sinápticos.

A continuación, se presenta una figura de describe el funcionamiento del Perceptron Multicapa:

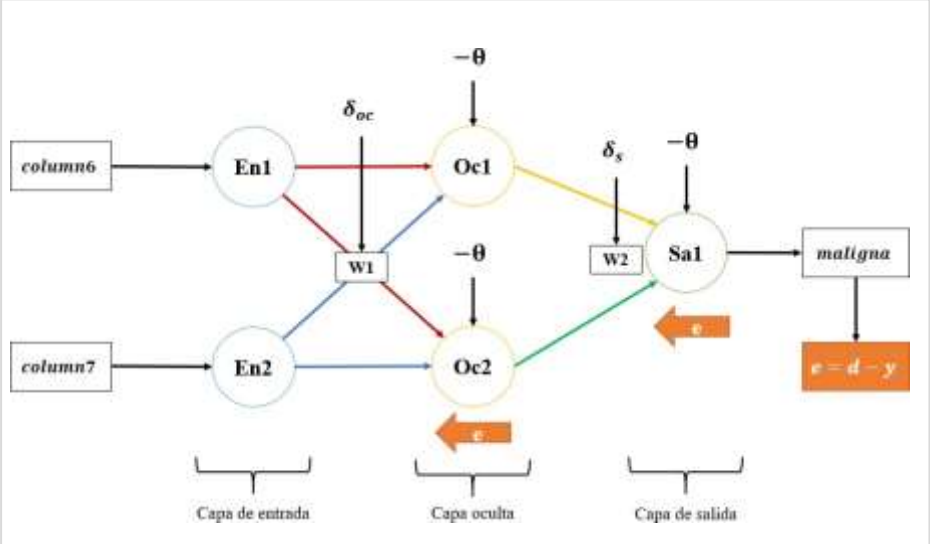


Fig. 27. Funcionamiento de perceptrón multicapa.

En la Figura 27, y denota las etiquetas benigna o maliciosa respectivamente, y x indica las propiedades del conjunto de datos. Mediante la siguiente fórmula, el algoritmo de retro propagación determina en primer lugar el error de red:

$$e = \frac{\partial E}{\partial y} \rightarrow (d - y)$$

Donde:

d: es el valor deseado

y: es la partida maliciosa o benigna

El paso 2 es ajustar la matriz de los pesos w2

$$w2 = w2 + (\lambda * \delta_s + S_{ocu})$$

$$\delta_s = \frac{\partial y}{\partial Y} * e$$

Donde:

w2: Denota el peso de la conexión entre las neuronas.

λ: Representa el factor de enseñanza, que entalla los pesos en función de la información de acceso.

δ_s: Es la derivada de la función tangente hiperbólica, la cual se multiplica por el error.

S_{ocu}: Son las salidas en la capa oculta

Finalmente, deje ajustarse el umbral en las neuronas de la capa encubierta:

$$\theta_{ocu} = \theta_{ocu} + (\lambda * \delta_{ocu})$$

Donde:

θ_{ocu}: es el valor del umbral

δ_{ocu}: Es la derivada de las capas enmascaradas de la función hiperbólica.

λ: factor de aprendizaje

A continuación, se muestra cómo demostrar este método utilizando su propia biblioteca de algoritmos en forma de código fuente.

```
package perceptron;

/**Librerias*/
import weka.core.Instances;
import weka.classifiers.*;
import weka.classifiers.functions.*;
import java.io.BufferedReader;
import java.io.FileReader;

public static void main(String args[])
{
  try{
  /**Carga del conjunto de datos*/
  Instances data = new Instances(new BufferedReader(new FileReader(" Dataset_Training.arff ")));
  data.setClassIndex(data.numAttributes()-1);
  /**Uso de la librería de perceptron multicapa*/
  weka.classifiers.Classifier model1= new weka.classifiers.functions.MultilayerPerceptron();
  model1.buildClassifier(data); /**Se genera el clasificador*/
  /**Evaluación del modelo*/
  Evaluation evaluation1=new Evaluation(data);
  evaluation1.evaluateModel(model1,data);
  System.out.println("----------------");
  /**Muestra en resumen las estadísticas de rendimiento*/
  System.out.println(evaluation1.toSummaryString());
  System.out.println(model1.toString());

  } catch (Exception ex) {
     ex.printStackTrace();
  }
}
```

Fig. 28. Código al utilizar la librería Multilayer Perceptron.

Implementación del algoritmo Support Vector Machine

Con esta metodología, se genera un modelo que puede discernir si una consulta reciente es maliciosa o benigna. Para lograrlo, explora un hiperplano que establezca un distanciamiento óptimo entre las clases. Como resultado, las consultas originadas de vectores etiquetados se ubicarán en un lado específico del hiperplano.

$$d(\bar{x}) = \sum_{i=1}^{m} \alpha_i y_i K(\bar{x}_i, \bar{x}) + b$$

Donde:

α_i: variable que define el algoritmo

$\bar{x}_i$: los atributos de la colección de datos

$K(\bar{x}_i, \bar{x})$: es la función kernel que utiliza un dominio dimensional para mapear estratégicamente las muestras.

La función del kernel gaussiano o base radial

Utiliza un método alternativo en el que se amplía la dimensionalidad.

$$K(\bar{x}_i, \bar{x}) = e^{(-\frac{\|x_1 - x_2\|^2}{2\sigma^2})}$$

Donde:

x: Representa los atributos del conjunto de preparación.

σ: Indica el ancho del kernel.

Tipo de SVM

$$C - SVC$$

Donde:

C: Representa la variable reguladora, útil para equilibrar la complejidad del modelo con el error de entrenamiento.

A continuación, se muestra cómo demostrar este método utilizando su propia biblioteca de algoritmos en forma de código fuente.

A continuación, se presenta una figura de describe el código de la librería LibSVM:

```
/**Librerías*/
import weka.classifiers.Evaluation;
import weka.classifiers.functions.LibSVM;
import weka.core.Instances;
import weka.core.converters.ConverterUtils.DataSource;

public class Classifier {
  public static void main(String[] args) throws Exception{
    /**Carga del conjunto de datos*/
    DataSource source = new DataSource("Dataset_Training.arff ");
    Instances dataset = source.getDataSet();
    dataset.setClassIndex(dataset.numAttributes()-1);

    /**Uso de la librería de LibSVM*/
    LibSVM libsvm = new LibSVM();
    String[] options = new String[4];
    /**-S tipo de SVM | -K función kernel | -G gama de la function kernel |
       -C establece el parámetro C del tipo de SVM
    */
    options[0] = "-S"; options[1] = "-G";
    options[2] ="-K"; options[3] = « -C";

    libsvm.setOptions(options);
    libsvm.buildClassifier(dataset); /**Se genera el clasificador*/

    /**Evaluación del modelo*/
    Evaluation eval3 = new Evaluation(dataset);
    eval3.evaluateModel(libsvm, dataset);
    /**Muestra en resumen las estadísticas de rendimiento*/
    System.out.println(eval3.toSummaryString());
  }
}
```

Fig. 29. Código al utilizar la librería LibSVM.

Implementación del algoritmo K-NN

Está ampliamente reconocido que K-NN es un método de predicción fundamentado en instancias. Encuentra instancias de consultas de clasificación y crea un espacio n-dimensional a partir de ahí.

$$D_{m,n} = \begin{bmatrix} d_1 = & c_1 & c_2 & \cdots & q_1 \\ d_2 = & c_1 & c_2 & \cdots & q_2 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ d_m = & c_m & c_m & \cdots & q_n \end{bmatrix}$$

Donde cada instancia de consulta está representada por d_m, cada atributo del conjunto de datos está representado por c_m y la etiqueta a presagiar (es decir, si es o no una consulta malintencionada) está indicada q_n.

Para entender el funcionamiento del algoritmo con la colección de datos, se plantea el siguiente ejemplo:

Tabla XIII.

Muestra de datos utilizada para ilustrar el funcionamiento del algoritmo K-NN.

columna1	columna2	columna3	columna4	columna5	columna6	Query
[123]	[0]	[58]	[123]	[36]	[0]	[1]
[123]	[1]	[58]	[123]	[36]	[0]	[1]
[123]	[4]	[58]	[123]	[36]	[0]	[1]
[123]	[2]	[58]	[123]	[36]	[0]	[0]
[123]	[3]	[58]	[123]	[36]	[0]	[0]
[123]	[4]	[58]	[123]	[36]	[0]	[0]

La Tabla 14 presenta una porción reducida del conjunto de datos empleado para categorizar las consultas como peligrosas o seguras.

Un punto negro en la Figura 30 indica un nuevo dato que hay que clasificar. El algoritmo encuentra las K ocurrencias más cercanas a este punto para identificar su etiqueta tras valorar la distancia entre este punto y todos los puntos del conjunto de datos.

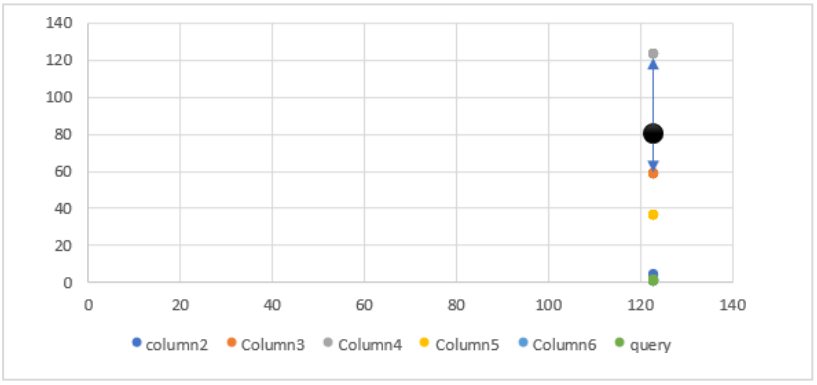


Fig. 30. Simulación del funcionamiento de K-NN.

A continuación, se muestra cómo demostrar este método utilizando su propia biblioteca de algoritmos en forma de código fuente.

```
import weka.classifiers.Classifier;
import weka.classifiers.lazy.IBk;
import weka.core.Instance;
import weka.core.Instances;

public class KNN {
        public static void main(String[] args) throws Exception {
                /**Carga del conjunto de datos*/
                BufferedReader datafile = readDataFile("Dataset_Training.arff ");
                 /**Se cargan las instancias del dataset*/
                Instances data = new Instances(datafile);
                data.setClassIndex(data.numAttributes() - 1);

                 /**Instancia de prueba*/
                Instance first = data.instance(0);
                Instance second = data.instance(1);
                data.delete(0);
                data.delete(1);
                 /**Uso de la libreria IBk*/
                Classifier ibk = new IBk();
                ibk.buildClassifier(data); /**Se genera el clasificador*/
                 /**Calcula las probabilidades a la clase para la isntancia de
                prueba*/
                double class1 = ibk.classifyInstance(first);
                double class2 = ibk.classifyInstance(second);

                System.out.println("first: " + class1 + "\nsecond: " + class2);
        }
}
```

Fig. 31. Código al utilizar la librería IBk.

Para mejorar los modelos entrenados, se eligieron seis algoritmos y se ajustaron sus hiperparámetros en las secciones anteriores. Para impedir el sobreajuste de los modelos, se tomaron decisiones basadas en la coherencia de los indicadores de rendimiento tanto en el entrenamiento como en la validación de los conjuntos de datos. A continuación, figura una lista completa de clasificadores y los parámetros ideales para cada uno.:

Tabla XIIIV.

Configuración de parámetros empleada en los seis algoritmos.

Algoritmos de Clasificación	**Parámetros**	**Valor**
Decision Tree	No tiene parámetros	Null
Random Forest	Sucesos fuera de la muestra %	10
	Cantidad de iteraciones	200
	Cantidad de arboles	200
Multilayer Perceptron / Neuronal Network	Tasa de aprendizaje	0.05
	Épocas máximas	2000
	Cantidad de capas ocultas	4
Support Vector Machine	Tipo de SVM	C-SVC, C=1
	Función del kernel	Función base radial: $e^{(-\frac{\|x1-x2\|^2}{2\sigma^2})}$
	Pesos de clase	{1,1}
K-NN	Cantidad de vecinos más cercanos	5

IV. CONCLUSIONES Y RECOMENDACIONES

4.1. Conclusiones

Tras analizar resultados de la investigación y teniendo en cuenta los objetivos particulares que se habían fijado, elegimos los mejores algoritmos de aprendizaje automático para la categorización. Para completar este proceso, hubo que revisar a fondo las dieciocho técnicas que se habían encontrado en estudios anteriores sobre la identificación de amenazas de inyección en motores de datos SQL y NoSQL entre 2015 y 2020.

Los accesos al registro, las solicitudes al servidor y el uso de funciones JavaScript se encontraban entre las diversas consultas benignas y peligrosas encontradas durante un examen exhaustivo de las amenazas de inyección de motores de datos NoSQL. Se desvelo que los asaltos más frecuentes se producían durante las situaciones de inicio de sesión, cuando las consultas hostiles se convertían en inocuas utilizando los operadores $ne y $gt, poniendo en peligro la seguridad del almacenamiento de datos NoSQL.

Se encontró un patrón en la producción de la colección de datos, que consta de 19 atributos conectados a la estructura JSON, tras un examen detallado de las consultas y de la estructura JSON del motor de datos MongoDB. Se recogieron 509 muestras en total, la mayoría procedentes del caso de inicio de sesión. Para su inclusión en la colección, estas instancias se convirtieron en varios tipos de valores, como caracteres ASCII y valores numéricos.

Los algoritmos Neural Network, Multilayer Perceptron, Random Forest, Árbol de decision, K-NN (K-Nearest Neighbors) y SVM (Support Vector Machine) resultaron ser los más apropiados para llegar a los objetivos designados en función de la métrica de precisión. En términos de precisión, los algoritmos Perceptrón multicapa y Red neuronal obtuvieron los mejores resultados.

Al final, estos métodos se pusieron en práctica para identificar ataques de inyección NoSQL. Utilizando consultas NoSQL benignas y maliciosas, se creó un modelo de clasificación y se utilizaron los seis métodos mencionados en pruebas iterativas. Los algoritmos de Perceptrón Multicapa y Red Neuronal produjeron superiores resultados, con una precisión de clasificación del 98,7% para los 509 datos utilizados en el ejemplo de inicio de sesión. Cuando estos algoritmos se compararon con otros analizados, también mostraron un tiempo de ejecución eficiente y un bajo consumo de recursos informáticos.

4.2. Recomendaciones

SVM, redes neuronales, perceptrones multicapa, árboles de decisión, K-NN y Random Forest son sólo algunas de las técnicas de clasificación que se utilizaron en este estudio. Pero investigar otras técnicas que no se tuvieron en cuenta en este estudio podría producir superiores resultados en términos de identificación de asaltos de inyección en motores de datos NoSQL, lo que podría mejorar la funcionalidad del modelo.

La investigación, centrada principalmente en MongoDB, la base de datos más popular, ha descrito una serie de consultas benignas y maliciosas. Se han estudiado muchos escenarios para las consultas, y el trabajo futuro pretende definir otras bases de datos NoSQL indefensas a asaltos de inyección, como Cassandra, CouchDB, Redis y Hbase.

Al construir el conjunto de datos, se eligieron las propiedades y etiquetas adecuadas utilizando un patrón basado en la estructura JSON. Para mejorar este procedimiento, podría ser útil crear un patrón que pueda gestionar diversas consultas de todos los motores de datos NoSQL susceptibles de sufrir asaltos de inyección, teniendo en cuenta elementos como documentos, columnas y valores clave, entre otros.

La identificación de asaltos de inyección contra la base de datos NoSQL MongoDB ha sido el enfoque principal de esta investigación. En consecuencia, se aconseja utilizar este método para descubrir asaltos de inyección en diferentes tipos de bases de datos NoSQL.

REFERENCIAS

[1] R. Neha, NoSQL Security, India: Elsevier Inc, 2018.

[2] C. Belmer, «NullSweep,» 6 Agosto 2019. [En línea]. Available: https://nullsweep.com/a-nosql-injection-primer-with-mongo/.

[3] M. Curphey y D. Groves, «OWASP,» 2017. [En línea]. Available: https://wiki.owasp.org/images/5/5e/OWASP-Top-10-2017-es.pdf.

[4] C. Mulling, «Database Trends and Applications,» 10 Octubre 2010. [En línea]. Available: https://www.dbta.com/Columns/DBA-Corner/Defining-Database-Performance-70236.aspx.

[5] C. Chapman, «The Daily Swig,» 5 Marzo 2020. [En línea]. Available: https://portswigger.net/daily-swig/high-severity-regex-bugs-discovered-in-parse-server.

[6] R. Islam, S. Islam, Z. Ahmed, A. lqbal y R. Shahriyar, «Automatic Detection of NoSQL Injection Using Supervised Learning,» *2019 IEEE 43rd Annual Computer Software and Applications Conference (COMPSAC),* pp. 760-769, 2019.

[7] A. Cuzzocrea y H. Shahriar, «Data masking techniques for NoSQL database security: A systematic review,» *IEEE Xplore,* pp. 2-7, 2017.

[8] A. Ron, E. Bronshtein y A. Shulman-Peleg, «No SQL, No Injection? Examining NoSQL Security,» *ResearchGate,* 2015.

[9] R. Islam, S. Islam, Z. Ahmed, A. lqbal y R. Shahriyar, «Automatic Detection of NoSQL Injection Using Supervised Learning,» *2019 IEEE 43rd Annual Computer Software and Applications Conference (COMPSAC),* pp. 760-769, 2019.

[10] F. Guamán, «Vulnerabilidades de Bases de Datos NoSQL,» UPM, Madrid, 2014.

[11] Chema, A.;, «ElevenPaths,» 28 octubre 2014. [En línea]. Available: https://empresas.blogthinkbig.com/como-funcionan-las-mongodb-injection/.

[12] L. Joyanes, «Big Data, Análisis de grandes volúmenes de datos en organizaciones,» *Alfaomega,* 2016.

[13] M. Altarade, «The Definitive Guide to NoSQL Databases,» 2020. [En línea]. Available: https://www.toptal.com/database/the-definitive-guide-to-nosql-databases.

[14] A. Bondi, «Characteristics of Scalability and Their Impact on,» *DL.ACM.ORG,* 2000.

[15] C. Caballero y R. Montoya, «Almacenamiento de la información e introducción a SGBD,» *Paraninfo,* pp. 74-75, 2016.

[16] W. Tutte, «Graph Theory,» *Combridge Mathematical Library,* p. 30, 2001.

[17] C. Requena, «Internet Archive,» 12 Marzo 2018. [En línea]. Available: https://web.archive.org/web/20110217092700/http://www.nosql.es/blog/.

[18] Bezos, B.;, «Amazon Web Service,» 06 junio 2020. [En línea]. Available: https://docs.aws.amazon.com/es_es/dynamodb/?id=docs_gateway.

[19] J. Palma y R. Marín, Inteligencia Artificial, Madrid: McGrawHill, 2008, pp. 3-8.

[20] A. Norman, Aprendizaje Automático En Acción, San Francisco: Kindle Edition, 2017.

[21] M. Kang y N. Jameson, «Machine Learning,» Maryland, Sons Ltd, 2018, pp. 85-88.

[22] D. Powers, «Evaluation: From Precision, Recall and F-Factor,» *ResearchGate,* pp. 2-17, 2007.

[23] P. Nicolas, «Scala for Machine Learning,» Reino Unido, Packt Publishing Ltd., 2015, pp. 69-70.

[24] Liitle, J.; Bangert, S.; Moler, C.; , «MathWorks,» 14 junio 2020. [En línea]. Available: https://www.mathworks.com/help/phased/examples/detector-performance-analysis-using-roc-curves.html.

[25] Page, L.; Brin, S.;, «Developers,» 11 junio 2020. [En línea]. Available: https://developers.google.com/machine-learning/crash-course/classification/roc-and-auc?hl=es-419.

[26] H. Lan y Y. Pan, «A Crowdsourcing Quality Prediction Model Based on,» *IEEE/ACIS,* pp. 315-319, 2019.

[27] M. Segal, «Machine Learning Benchmarks and Random Forest Regression,» *eScholarship,* pp. 2-15, 2004.

[28] David Cournapeau, «scikit-learn.org,» 14 octubre 2020. [En línea]. Available: https://scikit-learn.org/stable/modules/neighbors.html.

[29] A. Akbulut, A. Sengur, Y. Guo y F. Smarandache, «NS-k-NN: Neutrosophic Set-Based k-Nearest Neighbors Classifier,» *symmetry,* p. 179, 2 agosto 2017.

[30] E. Caicedo y J. López, Una aproximación práctica a la redes neuronales artificiales, Santiago de Cali, Colombia: Programa Editorial Universidad del Valle, 2017.

[31] X. Wang, X. Lin y X. Dang, Supervised learning in spiking neural networks: A review of algorithms, People's Republic of China, 2020.

[32] A. Palmer, J. Montaño y R. Jiménez, Perceptron Multicapa, España, 2001.

[33] Q. Ye, B. Lin y Y.-J. Li, SENTIMENT CLASSIFICATION FOR CHINESE REVIEWS: A COMPARISON BETWEEN SVM AND SEMANTIC APPROACHES,

China: Proceedings of the Fourth International Conference on Machine Learning and Cybernetics, 2005.

[34] M. Awad y R. Khanna, Support Vector Machines for Classification, India, 2015.

[35] E. Carmona, Máquinas de Vectores Soporte (SVM), Madrid, España, 2016.

[36] J. Burón, «Aplicación del Análisis en Componentes Independientes a la Clasificación de Texturas,» *Bibing,* pp. 48-59, 2015.

[37] R. Fernández, R. Seijo, P. Suárez y R. Martínez, «Statistical Model for Prediction of Diabetic Foot Disease in Type 2 Diabetic Patients,» *Medisur,* p. 10, 2016.

[38] K. Zhang, «A Machine Learning based Approach to Identify,» *IEEE/ACM International Conference on Automated Software Engineering (ASE),* pp. 1286-1287, 2019.

[39] J. Chaparro, B. Giraldo y S. Rodon, «Evaluación del clasificador Naïve Bayes como herramienta de diagnóstico en Unidades de Cuidado Intensivo,» *Tecnología ¦ Journal Technology,* pp. 88-91, 2013.

[40] A. Olivieri, «Regresión lineal en analítica química,» *SlideShare,* pp. 28-36, 2016.

[41] L.-:. Pinto, «Análisis de la aplicación de algoritmos de K-means y Continuous Max-Flow a la segmentación de imágenes en color,» *IDUS,* pp. 37-54, 2015.

[42] D. Pascual, «Algoritmos de Agrupamiento basados en densidad y Validación de clusters,» *CERPAMID,* pp. 130-141, 2010.

[43] J. Troyano, V. Díaz y F. Enriquez, «researchgate,» 10 Junio 2002. [En línea]. Available: https://www.researchgate.net/publication/267565005_Aplicacion_de_Modelos_de_Markov_y_Maquinas_SVM_al_Reconocimiento_de_Entidades.

[44] I. Mauny, «APISECURITY,» 24 Enero 2019. [En línea]. Available: https://apisecurity.io/issue-15-fortnite-hack-tls-mitm-attacks-sql-injections-for-nosql/.

[45] J. Macqueen, «SOME METHODS FOR CLASSIFICATION AND ANALYSIS OF MULTIVARIATE OBSERVATIONS,» *projecteuclid,* pp. 281-297, 2008.

[46] I. Pérez y B. León, «Logica Difusa,» *Casa del Libro,* pp. 59-65, 2007.

[47] A. Ron, E. Bronshtein y A. Shulman-Peleg, «No SQL, No Injection? Examining NoSQL Security,» *ResearchGate,* 2015.

[48] F. Romero, «Metologia de la investigación,» 2020. [En línea]. Available: http://metodos-investigacion-unicordoba.blogspot.com/p/blog-page_8.html.

[49] A. Olusola y A. Temidayo, «Application of Data Masking in Achieving Information Privacy,» *IOSR Journal of Engineering,* pp. 13-20, 2014.

[50] A. Hernández, «La recolección y análisis de datos cuantitativos,» *UNY,* pp. 3-16, 2013.

[51] R. Islam, S. Islam, Z. Ahmed, A. Lqbal y R. & Shahriyar, «Automatic Detection of NoSQL Injection Using Supervised Learning,» *IEEE 43rd Annual Computer Software and Applications Conference (COMPSAC),* pp. 760-769, 2019.

[52] M. Hasan, Z. Balbahaith y M. Tarique, «Detection of SQL Injection Attacks: A Machine Learning Approach,» *International Conference on Electrical and Computing Technologies and Applications (ICECTA),* pp. 1-6, 2019.

[53] K. Kamtuo y C. Sooleck, «Machine Learning for SQL Injection Prevention on Server-Side Scripting,» *International Computer Science and Engineering Conference (ICSEC),* pp. 1-6, 2016.

[54] K. Zhang, «A Machine Learning based Approach to Identify SQL Injection Vulnerabilities,» *34th IEEE/ACM International Conference on Automated Software Engineering (ASE),* pp. 1286-1288, 2019.

[55] A. Uwagbole, J. Buchana y L. Fan, «Applied Machine Learning Predictive Analytics to SQL Injection Attack Detection and Prevention,» *IFIP/IEEE Symposium on Integrated Network and Service Management (IM),* pp. 1087-1090, 2017.

[56] A. Akbulut, A. Sengur, Y. Guo y F. Smarandache, «NS-k-NN: Neutrosophic Set-Based k-Nearest Neighbors Classifier,» *symmetry,* p. 179, 2 agosto 2017.

[57] P. J. Roque Martin.

ANEXOS

Anexo 1. Resolución de aprobación del proyecto de investigación.

FACULTAD DE INGENIERÍA, ARQUITECTURA Y URBANISMO RESOLUCIÓN N°1314-2020/FIAU-USS

Pimentel, 17 de julio de 2020

VISTO:

El Acta de reunión N°1606-2020, de fecha 16 de junio de 2020 del Comité de investigación de la Escuela profesional de INGENIERIA DE SISTEMAS, para la ejecución de la Tesis: **"DETECCIÓN AUTOMÁTICA DE ATAQUES DE INYECCIÓN MEDIANTE APRENDIZAJE AUTOMÁTICO EN BASES DE DATOS NOSQL"**, presentado por el(los) tesista(s) **PAICO CHILENO DANIEL y VALDERA CONTRERAS JHON HARRY,** del Programa de estudios **INGENIERIA DE SISTEMAS**, y;

CONSIDERANDO:

Que, de conformidad con la Ley Universitaria N° 30220 en su artículo 48º que a letra dice: "La investigación constituye una función esencial y obligatoria de la universidad, que la fomenta y realiza, respondiendo a través de la producción de conocimiento y desarrollo de tecnologías a las necesidades de la sociedad, con especial énfasis en la realidad nacional. Los docentes, estudiantes y graduados participan en la actividad investigadora en su propia institución o en redes de investigación nacional o internacional, creadas por las instituciones universitarias públicas o privadas.";

Que, de conformidad con el Reglamento de grados y títulos en su artículo 21° señala: "Los temas de trabajo de investigación, trabajo académico y *tesis* son *aprobados por el Comité de Investigación* y derivados a la facultad o Escuela de Posgrado, según corresponda, para la emisión de la resolución respectiva. El *periodo de vigencia de los mismos será de dos años*, a partir de su aprobación. En caso un tema perdiera vigencia, el Comité de Investigación evaluará la ampliación de la misma.

Que, de conformidad con el Reglamento de grados y títulos en su artículo 24º señala: La tesis es un estudio que debe denotar rigurosidad metodológica, originalidad, relevancia social, utilidad teórica y/o práctica en el ámbito de la escuela profesional. Para el grado de doctor se requiere una tesis de máxima rigurosidad académica y de carácter original. Es individual para la obtención de un grado; *es individual o en pares para obtener un título profesional*. Asimismo, en su artículo 25° señala: "El tema debe responder a alguna de las líneas de investigación institucionales de la USS S.A.C.".

Que, en el Acta de reunión N°1606-2020 de fecha 16 de junio de 2020, del Comité de investigación de la Escuela profesional de INGENIERIA DE SISTEMAS, se indica entre los acuerdos la aprobación del Proyecto de tesis denominado "DETECCIÓN AUTOMÁTICA DE ATAQUES DE INYECCIÓN MEDIANTE APRENDIZAJE AUTOMÁTICO EN BASES DE DATOS NOSQL" de la línea de investigación de INFRAESTRUCTURA, TECNOLOGÍA Y MEDIO AMBIENTE, a cargo de PAICO CHILENO DANIEL y VALDERA CONTRERAS JHON HARRY en condición de estudiante, del Programa de estudios INGENIERIA DE SISTEMAS.

Estando a lo expuesto, y en uso de las atribuciones conferidas y de conformidad con las normas y reglamentos vigentes;

SE RESUELVE:

ARTÍCULO 1°: APROBAR, el Proyecto de Tesis denominado **"DETECCIÓN AUTOMÁTICA DE ATAQUES DE INYECCIÓN MEDIANTE APRENDIZAJE AUTOMÁTICO EN BASES DE DATOS NOSQL",** perteneciente a la línea de investigación de **INFRAESTRUCTURA, TECNOLOGÍA Y MEDIO AMBIENTE**, a cargo de **PAICO CHILENO DANIEL y VALDERA CONTRERAS JHON HARRY**, del Programa de estudios **INGENIERIA DE SISTEMAS.**

ARTÍCULO 2°: ESTABLECER, que la inscripción del Título de Proyecto de tesis se realice a partir de emitida la presente resolución y tendrá una vigencia de dos (02) años.

ARTÍCULO 3°: DEJAR SIN EFECTO, toda Resolución emitida por la Facultad que se oponga a la presente Resolución.

REGÍSTRESE, COMUNÍQUESE Y ARCHÍVESE

Dr. Mario Fernando Ramos Hascal
Decano - Facultad de Ingeniería, Arquitectura y Urbanismo
UNIVERSIDAD SEÑOR DE SIPÁN SAC.

MBA. María Noelia Sialer Rivera
Secretaria Académica - Facultad de Ingeniería, Arquitectura y Urbanismo
UNIVERSIDAD SEÑOR DE SIPÁN SAC.

Anexo 2. Formato para informe de consumo de CPU.

Consumo de CPU	
Ítem	**Valor**
Uso	
Velocidad	
Procesos	
Subprocesos	
Tiempo	

Anexo 3. Formato de informe de consumo de memoria.

Consumo de Memoria	
Ítem	**Valor**
Uso	
Disponibilidad	
Confirmada	
En caché	
Tiempo	

Anexo 4. Formato para informe de promedio de tiempo de respuesta.

Promedio de Tiempo de respuesta	
Ítem	**Valor**
Velocidad	
Tiempo	
CPU	
Memoria	
Disco	

Anexo 5. Formato para el registro de matriz de confusión.

		Resultados Clasificación	
		Consulta benigna	Consulta maliciosa
Resultados Reales	Consulta benigna		
	Consulta maliciosa		

Ítem	**Valor**
Verdadero positivo (VP)	
Falso Positivo (FP)	
Verdadero Negativo (VN)	
Falso Negativo (FN)	

Ítem	**Valor**
Exactitud	
Precisión	
Exhaustividad	
F-Score	
AUC	

Verdadero Positivo (VP)		**Falso Positivo (FP)**	
Ítem	**Valor**	**Ítem**	**Valor**
Realidad		Realidad	
Predicción del modelo AA		Predicción del modelo AA	
Numero de resultados		Numero de resultados	
Falso Negativo (FN)		**Verdadero Negativo (FP)**	
Ítem	**Valor**	**Ítem**	**Valor**
Realidad		Realidad	
Predicción del modelo AA		Predicción del modelo AA	
Numero de resultados		Numero de resultados	

Anexo 6. Ranking de modelos aprendizaje automático con respecto a la exactitud.

	Modelos de Aprendizaje Automático	Exactitud	Autor
1	Dicotomizador Iterativo (ID3)	91.66%	[36]
2	Detector Automático de Interacción Chi-cuadrado (CHAID)	73.20%	[37]
3	Árbol De Decisión	93.40%	[38]
4	Naive Bayes	78.00%	[39]
5	Regresión Ordinaria Por Mínimos Cuadrados	90.00%	[40]
6	K-NN	91.61%	[41]
7	K Vecinos más próximos (KNN)	92.00%	[9]
8	Algoritmos Basados en la Densidad	74.00%	[42]
9	Redes Neuronales	95.30%	[38]
10	Análisis de elementos principales (ACP)	83.30%	[37]
11	Análisis de elementos independientes (ACI)	84.00%	[37]
12	Regresión Logística	91.50%	[38]
13	Adaboost	91.78%	[9]
14	Markov	73.87%	[43]
15	Super Vector Machine (SVM)	95.40%	[38]
16	Random Forest	93.60%	[38]
17	Xgboost	89.51%	[6]
18	Perceptrón Multicapa	95.30%	[38]

Reporte de similitud

NOMBRE DEL TRABAJO

DETECCIÓN AUTOMÁTICA DE ATAQUES DE INYECCIÓN MEDIANTE APRENDIZAJ E AUTOMÁTICO EN BASES DE DATOS N OSQL

AUTOR

Jhon Harry / Daniel Valdera Contreras / Paico Chileno

RECUENTO DE PALABRAS

15009 Words

RECUENTO DE CARACTERES

83078 Characters

RECUENTO DE PÁGINAS

121 Pages

TAMAÑO DEL ARCHIVO

1.6MB

FECHA DE ENTREGA

Apr 30, 2024 8:50 AM GMT-5

FECHA DEL INFORME

Apr 30, 2024 8:52 AM GMT-5

18% de similitud general

El total combinado de todas las coincidencias, incluidas las fuentes superpuestas, para cada base de datos.

- 18% Base de datos de Internet
- 1% Base de datos de publicaciones
- Base de datos de Crossref
- Base de datos de contenido publicado de Crossref
- 3% Base de datos de trabajos entregados

Excluir del Reporte de Similitud

- Material bibliográfico
- Material citado
- Coincidencia baja (menos de 8 palabras)