UNIVERSIDAD ABIERTA INTERAMERICANA

Facultad de Tecnología Informática

Trabajo Final de Carrera presentado para obtener el título de
Lic. en Gestión de Tecnología informática

Técnicas de Aprendizaje Automático para la detección de spam

Alumno: Nicolas Pisani

Tutora Técnica: Dra. Claudia Pons

Profesora de Trabajo Final: Dra. Marcela Samela

Julio 2023

Resumen

El uso del correo electrónico es habitual en nuestras actividades profesionales, académicas o personales, pero no está exento de ser utilizado de forma malintencionada convirtiéndolo en uno de los principales vectores de infecciones, robo de información o publicidades no deseadas. Es necesario comprender la gravedad que estos ataques ocasionan, grandes pérdidas económicas y extorsiones se inician a través de este medio de comunicación. Los modelos de aprendizaje automático de inteligencia artificial, se encuentran en auge para su identificación y categorización. Dichas técnicas, son empleadas para combatir al correo no deseado, pero no son infalibles y pueden seguir mejorándose. Es necesario continuar investigando en el área, para poder optimizar su rendimiento. En este sentido, el presente trabajo se enfocó en la puesta a prueba de seis algoritmos de clasificación para la detección de correo no deseado, buscando medir sus niveles de exactitud. Se construyó un conjunto inicial de datos, el cual fue sometido al proceso de aprendizaje. No fue fácil predecir su ejecución, para conocer el rendimiento de las distintas técnicas siempre fue necesario la puesta a prueba de las mismas. La investigación aporta resultados que servirán de referencia. Además, contiene en un único documento la comparación de las métricas entre sí. Los resultados muestran que los niveles de exactitud de los modelos son similares, el mejor rendimiento fue obtenido por Naive Bayes.

Palabras Clave: aprendizaje automático, correo electrónico, correo no deseado, inteligencia artificial

Abstract

The use of email is common in our professional, academic or personal activities, but it is not exempt from being used maliciously, making it one of the main vectors of infections, information theft or unwanted advertising. It is necessary to understand the seriousness that these attacks cause, great economic losses and extortions are initiated through this means of communication. Artificial intelligence machine learning models are on the rise for their identification and categorization. These techniques are used to combat spam, but they are not infallible and can continue to be improved. It is necessary to continue researching in the area, in order to optimize its performance. In this sense, the present work focused on the testing of six classification algorithms for spam detection, looking to measure their accuracy levels. An initial data set was constructed, which was subjected to the learning process. It was not easy to predict its execution, to know the performance of the different techniques it was always necessary to put them to the test. The research provides results that will serve as a reference. In addition, it contains in a single document the comparison of the metrics with each other. The results show that the accuracy levels of the models are similar, the best performance was obtained by Naive Bayes.

Keywords: artificial intelligence, email, machine learning, spam

Agradecimientos

A mi novia, familia, tutora, profesora del trabajo, docentes y compañeros.

Indice

Indice de Figuras

Indice de Tablas

1. Introducción

En la actualidad, las interacciones que realizamos con otras personas se han modificado, ya no podemos decir que la comunicación analógica es la que utilizamos comúnmente. Con analógica nos referimos al habla y la escritura en papel, la cual está perdiendo terreno respecto a lo digital. Es muy común utilizar la tecnología para este fin, sobre todo en contextos como la pandemia en la que fue elaborado el presente trabajo. El correo electrónico forma parte de la comunicación provista por la tecnología de una manera importante. Además, el correo tradicional corre en desventaja, la digitalización no solo trae comodidad y rapidez sino también un beneficio ecológico. Pero este gran medio de comunicación digital, no es ajeno a los problemas, también llama la atención de delincuentes y puede ser utilizado con fines poco felices, desde masivas campañas comerciales no deseadas hasta el robo de información personal, estafas o envió de virus informáticos o en inglés malware.

Esta problemática se ha considerado como un desafío cada vez más serio y ha afectado las comunicaciones de las personas a nivel global, según indica el informe de Symantec (2017), el correo no deseado, o como se lo conoce en informática spam, ocupa gran parte del tráfico total de correo electrónico en internet. Además, indica el informe, que es el método más popular de ataque hacia una organización o usuario común. Cada día se producen nuevas campañas de spam y es imposible su clasificación manual.

Existen dos técnicas aplicadas para evaluar si un programa es benigno o maligno. Como expone Sikorski (2012), por un lado, se encuentra el análisis estático y por el otro el dinámico. Sikorski explica que las técnicas estáticas no ejecutan código, solo examinan su estructura y datos binarios mediante las conocidas firmas de virus, una especie de huella digital del archivo. En cambio, dice Sikorski que las técnicas de análisis dinámico, ejecutan el código para analizar el comportamiento durante su ejecución en ambientes controlados. Claro que no es posible analizar uno por uno cada archivo por el procesamiento y tiempo que demanda, por lo que se debe hacer un filtro previo.

Actualmente en el mercado se comenzaron a utilizar ambas técnicas en conjunto para la prevención de amenazas, incluso en lo que se refiere al correo electrónico. El informe de Symantec (2017) explica que cuando nos referimos a ataques provocados mediante correos no deseados, no solo se atañe a archivos adjuntos en los correos para infectar los ordenadores, también existen diversas amenazas que los hacen el vector de infección más utilizado

actualmente. Entre las formas de ataque se desprende del informe, el robo de información, suplantación de identidad, enlaces a sitios fraudulentos para realizar estafas, entre otros.

Como se puede suponer, no es posible seguir el ritmo de las nuevas e incalculables variaciones de ofensivas para mantener actualizadas estas bases de datos de firmas, por lo que se ha comenzado a utilizar la inteligencia artificial para mitigar este problema de la técnica estática. Explica Russel & Norving (2004) que mediante la inteligencia artificial la maquina no sigue una receta para resolver un problema, sino que encuentra su propia forma de resolverlo. Las técnicas de machine learning, que en español significa lenguaje automático, actualmente se destacan como el nuevo paradigma en el combate de estas amenazas. Aún se desconocen los límites de su funcionamiento. Es necesario poder encontrar nuevas formas de implementarlas y optimizar sus rendimientos.

El propósito de la presente investigación es poner a prueba diversos modelos de machine learning evaluando sus niveles de exactitud y eficacia en la detección de spam, aportando así evidencia en la búsqueda de mejores soluciones. Los resultados obtenidos serán de utilidad para la comparación académica en la evaluación de modelos de detección de spam. La implementación del proyecto inicia con un relevamiento de técnicas, la obtención de los datos, su procesamiento y clasificación. Luego se construyeron los modelos de machine learning para poder someter el conjunto de muestreo y finalmente se comparó la efectividad de las técnicas. Es importante medir sus niveles de exactitud, con diferentes muestreos y métodos de preparación de los datos, para entrenar los modelos de clasificación y comparar las diferencias con otros estudios. A partir de los resultados se pretende idear estrategias para optimizarlos.

1.1 Fundamentación

Gracias a la Inteligencia Artificial no simbólica, las maquinas pueden “razonar” de forma automática por inducción y aprender a partir grandes volúmenes de datos y ejemplos para poder predecir comportamiento. El motivo del presente trabajo es evaluar técnicas de machine learning y contribuir con las mediciones en pos de la búsqueda de mejores rendimientos en la clasificación de spam. Poner a prueba las técnicas con diversos muestreos en fundamental para avanzar en encontrar soluciones. La investigación realizada aporta a la actualización y puesta a prueba de modelos y técnicas de detección en constante evolución.

1.2 Planteamiento del Problema

El cambio de estrategia para la detección de correo no deseado es una necesidad. La utilización de la Inteligencia Artificial para su detección es actualmente la forma más auspiciosa, pero si bien ha mejorado su rendimiento respecto a las técnicas estáticas de firmas, aun no logra ser lo efectiva que se espera.

Entonces, nos realizamos las siguientes preguntas, ¿Cuáles son las técnicas de machine learning más eficaces para la detección de spam? ¿Tienen diferentes niveles de exactitud?

1.3 Objetivo

En la presente investigación se busca conocer la efectividad de distintas técnicas machine learning para la detección y clasificación de correo no deseado. Aplicarlas para lograr una detección temprana y automática es importante en los desafíos que se plantea la seguridad informática. Según lo expuesto el objetivo primario es el siguiente:

- Analizar técnicas de machine learning para la detección de spam

1.3.1 Objetivos Secundarios

- Construir el muestreo necesario para ser sometido a los modelos de machine learning.
- Construir modelos de clasificación.
- Determinar el porcentaje de spam detectado según las técnicas de machine learning.
- Comparar técnicas de machine learning para la detección de spam
- Identificar las técnicas de machine learning más eficaces

2. Hipótesis del Trabajo

Los riesgos que presentan los correos no deseados son muy altos, detectarlos es la mejor prevención. Para cumplir con esta tarea es necesario que estos correos no ingresen en las bandejas de entrada de las cuentas. Entonces si se puede disponer de una técnica automática que detecte correo no deseado aprendiendo y clasificando por inducción, se podrá corregir este problema.

- Diferentes técnicas de análisis con machine learning logran detectar correo no deseado con efectividad aceptable superior al 90%

3. Marco Teórico

3.1 Los Riesgos de la Tecnología

Una de las principales compañías de seguridad Eset, en un relevamiento realizado en el año 2022, indica que "Dos tercios de los encuestados señaló la infección con códigos maliciosos como la mayor preocupación en materia de ciberseguridad. A esta le sigue la preocupación por el robo de información (62%)" (Eset, 2022, p4). Este tipo de infecciones, como también explica el reporte "llegan a sus víctimas mediante descargas de cracks o programas fraudulentos, comunicaciones maliciosas vía correos electrónicos o aplicaciones de mensajería instantánea" (Eset, 2022, p10). El correo electrónico continúa siendo uno de los principales vectores para ciberataques. Además, el envío masivo incluye al spam en productos farmacéuticos, citas para adultos, pérdida de peso, entre otras publicidades no solicitadas.

3.2 Los Ataques Informáticos

Inicialmente el principal objetivo de los ataques era ver que tan lejos se podía llegar, una competencia de habilidades y destrezas en encontrar defectos en los sistemas. Sin embargo, ha dejado de ser esa la motivación. En la actualidad los ataques esconden una gran gama de intereses, siendo el redito económico el más preponderante. Incluso algunos ataques son dirigidos con el fin de robar información y/o desestabilizar la reputación de la víctima.

Es importante reconocer que cada país tipifica los delitos informáticos con sus respectivas penalidades, de acuerdo con la consideración del valor de los datos y las consecuencias de perdida de privacidad, confidencialidad e integridad. La mayor parte de las legislaciones en el campo de la informática se definen después de ocurrido el incidente, luego de una lección aprendida del mismo.

Los delitos informáticos pueden producirse durante la navegación en internet, incluyendo claro está, el uso de herramientas como el correo electrónico. En base a lo expuesto por Temperini (2014) en el 14° Simposio Argentino de Informática y Derecho, en Latinoamérica se puede catalogar en Violación de Datos Personales; Difusión de Malware; Hurto Informático; Difusión maliciosa de información; Suplantación de Identidad Digital; Grooming[1], que se refiere al acto de contactar a una persona menor de edad, con el propósito de cometer cualquier delito contra la integridad sexual de la misma; Captación o venta ilegítima de datos; Carding, se refiere a al uso no autorizado de tarjetas de crédito ; Espionaje Informático; Violación a la Intimidad.

Se considera contenido malicioso a todo lo que a través de la navegación web permita vulnerar sistemas o también con presencia de virus de computadora, conocido en inglés con el termino malware. El software espía, conocido como spyware, actúa silenciosamente enviando información a terceros sin el conocimiento del usuario, aunque no causan explícitamente perjuicios a datos y sistemas.

Específicamente en Argentina, los delitos informáticos ya forman parte del Código Penal. El ministerio de justicia y derechos humanos, regula los siguientes delitos en la ley 26.388[2], según (Portal del Estado Argentino, s.f.):

1) Delitos informáticos contra la integridad sexual
2) Delitos informáticos contra la libertad
3) Delitos informáticos contra la propiedad
4) Delitos informáticos contra la seguridad pública
5) Delitos informáticos contra la administración pública

[1] Ley 26.904, articulo 131
[2] Ley 26.388

3.3 Conceptos de Seguridad Informática

Cuando se habla de Seguridad Informática, se refiere a la práctica de preservar o defender de ataque maliciosos a todo aquel programa, computadoras, servidores, dispositivos móviles, sistemas electrónicos, redes, información y bases de datos. Para ello, es importante conocer no solo los conceptos relacionados a dicho tema, sino también cuales son los procesos, protocolos, métodos o herramientas que se pueden utilizar para hacer frente a dicha amenaza.

3.3.1 Resumen

En la siguiente sección, se describirán ciertos conceptos fundamentales relacionados a la Seguridad Informática. Se comenzará describiendo cómo pueden clasificarse las diferentes clases de piratas informáticos conocidos en informática como hackers. Luego se procederá a realizar una descripción de las variedades de programas maliciosos, en inglés malware, que pueden encontrarse y los métodos para detectarlos.

3.3.2 Hackers

Se utiliza el término en inglés Hacker, para referirse a aquella persona experta en computadoras, con habilidades y conocimientos técnicos específicos para resolver un problema. Como explica Sweigart (2013), hay dos definiciones de hacker. Un hacker es una persona que estudia un sistema, para entender sus reglas completamente y puede con su creatividad modificar su funcionamiento en nuevas direcciones. También comenta Sweigart, que el termino hacker, es usado para hacer referencia a criminales que irrumpen en un sistema de computación, violan la privacidad de las personas y causan daños. Otro autor nos ofrece una idea similar de lo que es el acto del hacker, para Erickson (2008), es encontrar de una manera hábil y contradictoria una solución a un problema.

Los hackers pueden ser clasificados en categorías. Las siguientes son las detalladas por una de las empresas más importantes dedicadas a la seguridad informática (McAfee, 2019). Para comenzar, se hace una distinción entre Hackers a los cuales se les asigna un color de sombrero para identificarlos. Los Hackers de sombrero blanco, son expertos en seguridad, que utilizan distintas técnicas para garantizar que los sistemas de información de una empresa sean seguros. Por el otro lado tenemos a los Hackers de sombrero negro, a los que comúnmente se utiliza para referirse a los Hackers con fines maliciosos. En este grupo se encuentran los que

crean virus de computadora, se infiltran en redes, etc. Por último, en esta clasificación de Hackers, se encuentran los de sombrero gris quienes no usan sus habilidades para su propio beneficio, pero cuyas practicas no llegan a ser completamente legitimas. Por ejemplo, pueden infiltrarse en una empresa para revelar una vulnerabilidad y publicarla en internet. Podría estar haciéndole un favor a la empresa, aunque también la ha vulnerado sin permiso.

McAfee continua la catalogación con los llamados Script Kiddies, es un término despectivo, utilizado para referirse a hackers de sombrero negro que usan programas descargados de internet creados por alguien más, con la finalidad de darse a conocer. También se encuentran los Hacktivistas, quienes buscan generar un cambio social. Pueden estar motivados para exponer actos criminales y deshonestos o promover posturas políticas o religiosas. No hay que dejar de lado a los Hackers patrocinados por el estado. Los mismos disponen de tiempo y fondos ilimitados para enfocarse en personas, corporaciones y gobiernos. Las corporaciones también contratan sus Hackers, son llamados Hackers espías, para infiltrarse en la competencia y robar sus secretos comerciales. Podrían hacerlo desde fuera u obtener empleo como topo. Su único propósito es conseguir los objetivos requeridos por sus clientes para recibir dinero. Similares a estos últimos, son los Alteradores. Se trata de una persona que se encuentra dentro de una organización, que utiliza su acceso a los sistemas para fugar información. Por último, McAfee menciona a los Ciberterroristas. Los cuales suelen verse motivados por creencias religiosas o políticas para afectar infraestructuras con el objetivo de instigar miedo, violencia y caos.

Por otro lado, han proliferado las comunidades en donde los hackers comparten conocimiento y además distribuyen sus herramientas de forma gratuita o paga. Incluso suelen encontrarse publicaciones en las que se ofrece dinero a cambio del descubrimiento de nuevas vulnerabilidades en determinado sistema o programa informático.

Todas las categorías, anteriormente detalladas, aplican al utilizar al correo electrónico como medio para lograr fines delictivos. El uso del spam es una de las técnicas utilizadas.

3.3.3 Malware

El término malware proviene de la abreviación del inglés de malicious software, es decir programa malicioso. Puede ser utilizado para comprometer funcionalidades del sistema, robo de información, saltear controles de acceso, o cualquier otra forma de causar daño al sistema

sobre el cual se está ejecutando. Dividiremos los malwares en distintas categorías dependiendo de su propósito. Es necesario tener presente que todas las clases aquí expuestas de archivos maliciosos, pueden ser enviadas en un correo electrónico. Como indica Elisan (2015) son descargados como archivos adjuntos o mediante un hipervínculo en el cuerpo del correo.

Para realizar una clasificación del malware, existen muchos autores y criterios. Tomamos como referencia las variedades expuestas por Sikorski (2012). Comenzamos con el malware Backdoor, el cuál provee una "puerta trasera" al sistema para los atacantes. No causa ningún daño, pero facilita el acceso al sistema, de modo que puedan realizar autenticaciones y ejecuciones de comandos. Luego continúa con los Botnets, que son similares a los anteriormente descriptos, pero todas las computadoras infectadas con el mismo botnet reciben las mismas instrucciones desde un servidor. Sikorski menciona los Downloaders, código que existe solo para descargar otro malware. Comúnmente son instalados luego que el atacante ya tiene acceso al sistema, para así poder descargar códigos malignos adicionales. Además, el autor incluye a Information-stealing malware. Dicha categoría incluye malware que recolecta información de la computadora de la víctima y generalmente la envía al atacante. Continúa con los Launchers, usados para activar o ejecutar otros programas maliciosos. El autor también menciona a los Rootkit, utilizados para encubrir la existencia de otro código. Suma a la lista a los Scareware, que están designados para asustar a los usuarios infectados para que compre alguna cosa, generalmente simular ser soluciones de antivirus. Sikorski finaliza con las dos siguientes clasificaciones. El Spam-sending malware, infecta la computadora de un usuario con la finalidad de usarla para enviar spam. Este malware genera ganancias a los atacantes permitiéndoles vender servicios de spam. Por último, se encuentran los Worms o Virus, que son códigos maliciosos que pueden copiarse a sí mismos e infectar computadoras adicionales.

3.3.4 Métodos de Detección del Malware

Como indica Sikorski (2012) todas las técnicas de detección de malware pueden ser divididas en dos grandes categorías: estática o dinámica. El análisis estático involucra examinar el malware sin ejecutarlo. El análisis dinámico comprende su ejecución. Básicamente, el autor indica que el análisis estático consiste en examinar el archivo ejecutable, mediante firmas que permiten reconocerlo por la información que contienen. Su detección es rápida, pero puede ser inefectiva ante malwares sofisticados. El dinámico, en cambio, comprende la ejecución del malware y observar su comportamiento en el sistema. Para realizar esta tarea, se debe armar un

entorno seguro previamente, para evitar los riesgos o daños. Su detección conlleva más tiempo y recursos.

Cuando realizamos un análisis estático, el primer paso es verificar el malware a través de un programa antivirus, que ya lo ha identificado. Sikorski (2012) explica que dependen de una base de datos de piezas identificables de código sospechoso (firmas) y de análisis de coincidencia de patrones (heurística) para reconocer estos archivos. Además, el autor comenta que el método más común usado para identificar malware es el hashing (huella digital del malware). Continúa explicando que el software malicioso se corre a través de un programa de hashing que produce un único hash (código alfanumérico) que permite detectar el malware. Suele ser utilizado comúnmente para este propósito, indica Sikorski, el algoritmo MD5 o también el SHA-1.

Escapa a esta investigación el uso como soporte de un antivirus, ya que es un complemento para la detección del spam. El aprendizaje que se realizó en el presente trabajo, se basó en una comparación estática de patrones y datos en los mensajes, que permitió determinar una decisión sobre el correo utilizando machine learning. El spam, como se indicó anteriormente, es un medio para la transmisión de malware, pero no es su identidad, su identificación se puede realizar tanto si contiene estos archivos o no. El propósito de mencionar los métodos de análisis y el malware, se debe a que lo entendemos necesario para comprender los peligros de esta clase de correos.

3.4 Spam

3.4.1 Resumen

En la presente sección se abordan conceptos sobre el correo electrónico y el spam. Entenderlos es necesario para comprender el problema que acarrea y su necesaria detección.

3.4.2 Correo Electrónico

El avance de la tecnología en los últimos 50 años ha traído consigo grandes beneficios a la población en general. Refiriéndonos puntualmente al área de las comunicaciones, la misma ha permitido el intercambio de información desde cualquier punto del planeta, uniendo personas, empresas y gobiernos. Una de las herramientas utilizadas con este propósito es el

correo electrónico. Desde los comienzos de Arpanet[3] en adelante, para poder realizar el envío de los mensajes fueron necesarios protocolos de red específicos. Los protocolos son un conjunto de reglas y normas que fijan como deben comunicarse dos o más componentes de una red. Uno de los utilizados para la transmisión del correo electrónico es el SMTP, de sus siglas en inglés simple mail transfer protocol. Sin entrar en minuciosos detalles técnicos que exceden el presente trabajo, intentaremos dar algunos conceptos al respecto.

Según Klensin (2008) en la RFC[4] 5321 titulada SMTP, el objetivo del protocolo es transferir correo electrónico de forma confiable y eficiente. Cuando un cliente SMTP tiene un mensaje que transmitir, indica Klensin, éste establece un canal de transmisión de dos vías a un servidor SMTP. Además, en el documento se menciona que esta comunicación se puede realizar de dos formas. De una manera directa, entre el remitente original y el destinatario final. De una manera indirecta, puede ocurrir en una serie de saltos a través de sistemas intermediarios.

De lo anteriormente expuesto emergen dos importantes puntos a tener en cuenta, el primero es que el protocolo garantiza que la trasferencia sea confiable y eficiente. Pero solo se refiere a la transferencia como el proceso que permite el envío y recepción, no se ocupa de lo que se envía y tampoco de quien lo envía. El segundo, es que pueden existir intermediarios en la transferencia. Es en estos puntos donde la seguridad entra en juego. El correo electrónico puede ser utilizado tanto por entidades legítimas, como por delincuentes para masivas campañas comerciales, información fraudulenta e incluso envío de virus de computadora.

En el envío de correo electrónico intervienen principalmente tres conceptos: El SMTP, la sintaxis que especifica el formato de los mensajes y el MIME, de sus siglas en inglés Multipurpose Internet Mail Extensions, descritos en las RFC 2821, RFC 2822 y RFC 2045 respectivamente.

El actual estándar SMTP fue explicado en el año 2001 en RFC 2821 por Klensin. En él se especifica la estructura de una dirección de correo electrónico, que se compone de dos partes. La primera es el identificador del usuario o del buzón del usuario. Se utiliza para identificar el buzón en el servidor de correo, es la parte anterior al símbolo @. La segunda parte es el identificador del dominio, es decir el nombre de la organización más un sufijo de Internet estándar, al que corresponde el correo electrónico. Se utiliza para encauzar el correo electrónico

[3] Fue una red descentralizada de computadoras construida en 1969 por el Departamento de Defensa de los Estados Unidos, se convirtió en el inicio de Internet.

[4] Conjunto de documentos que sirven de referencia para las normas y estándares de internet y sus protocolos, por la organización IETF (Internet Engineering Tasking Force)

hacia su destino, mediante una consulta al DNS, de las siglas en inglés domain name system. Es la parte posterior al @. Una dirección de ejemplo de correo electrónico quedaría determinada de la forma usuario@dominio.

La estructura de un correo electrónico básicamente está dividida en tres partes, según Resnick en la RFC 2822 (2001). En dicho documento se menciona primero al cuerpo del correo, el cual es el contenido en sí, es la información que da el motivo del envío del mensaje. Luego le sigue el encabezado, donde se enuncia los datos del remitente, los servidores por donde ha pasado el correo y los programas utilizados para su envío. Por último, la envoltura del mensaje, en ella se encuentra la información del destinatario, que será utilizada por los servidores para hacer entregar el correo al destino.

El primer paso para enviar un mensaje es comunicarse con el servidor de correo, siguiendo los pasos indicados por el protocolo SMTP, en la RFC 2821 según Klensin (2001). En dicha RFC se especifica que el protocolo contempla un proceso de identificación a través de los comandos HELO o EHLO. Luego, se continúa explicando que, queda reflejada la IP del host origen en el servidor. Se especifica la dirección de correo del remitente y la dirección destino. Una vez establecida la comunicación, que es equivalente a crear la envoltura del mensaje, resta crear los encabezados y el cuerpo del mensaje. Esto es realizado por el agente usuario, programa de correo, que se esté utilizando. Éste se encarga de colocar las cabeceras correspondientes establecidas por la RFC 2822 como detalla Resnick (2001), haciendo coincidir, si no se especifica otra cosa, los campos Form y To con los detallados en MAIL FROM y en RCPT TO respectivamente. Continuando con el proceso, se da el formato correspondiente al texto del mensaje y al contenido del mismo, siguiendo el estándar MIME descrito en la RFC 2045 según Freed & Borenstein (1996).

Una vez que el mensaje es entregado al servidor SMTP, éste inserta una cabecera Message-ID y otra Received al principio del mismo, y se lo entrega a otro servidor de correo, utilizando para esta comunicación el protocolo SMTP. El segundo servidor SMTP comprueba la dirección de correo que se ha pasado en el RCPT TO para redireccionar ese mensaje a su destino y añade una cabecera Received encima de la que colocó el anterior. Estos pasos se repetirán tantas veces como sea necesario hasta que el correo llegue a su destino.

Figura 1

Cabecera de un correo electrónico

```
Return-path: <user@example.com>
Received: from mac.com ([10.13.11.252])
  by ms031.mac.com (Sun Java System Messaging Server 6.2-8.04 (built Feb 28
  2007)) with ESMTP id <0JMI007ZN7PETGC0@ms031.mac.com> for user@example.com; Thu,
  09 Aug 2007 04:24:50 -0700 (PDT)
Received: from mail.dsis.net (mail.dsis.net [70.183.59.5])
  by mac.com (Xserve/smtpin22/MantshX 4.0) with ESMTP id l79BOnNS000101
  for <user@example.com>; Thu, 09 Aug 2007 04:24:49 -0700 (PDT)
Received: from [192.168.2.77] (70.183.59.6) by mail.dsis.net with ESMTP
  (EIMS X 3.3.2) for <user@example.com>; Thu, 09 Aug 2007 04:24:49 -0700
Date: Thu, 09 Aug 2007 04:24:57 -0700
From: Frank Sender <sender@example.com>
Subject: Test
To: Joe User <user@example.com>
Message-id: <61086DBD-252B-46D2-A54C-263FE5E02B41@example.com>
MIME-version: 1.0 (Apple Message framework v752.2)
X-Mailer: Apple Mail (2.752.2)
Content-type: text/plain; charset=US-ASCII; format=flowed
Content-transfer-encoding: 7bit
```

Nota. Tomado de https://tecnologiafacil.org/como-rastrear-un-correo-electronico-hasta-su-ip-de-origen

En la Figura 1 se pueden observar algunos de los campos que se especifican en la RFC 2822 Resnick (2001). En el campo de Return-Path aparece la dirección del remitente que se especificó en la transacción SMTP con el comando MAIL FROM que no tiene que coincidir obligatoriamente con el contenido de la cabecera From. Los campos que verdaderamente detallan información sobre la procedencia del mensaje son los rotulados como Received. En el correo presentado en la Figura 1 se observa que hay tres cabeceras con este nombre, lo cual indica que el mensaje ha pasado por tres servidores de correo.

3.4.3 Definición del Spam

Su definición depende de cada uno de los autores y diccionarios que podamos consultar, sin embargo, todas tienen similitudes entre sí. Optamos por recurrir a la Real Academia Española como esclarecimiento del concepto, la cual define al correo basura como "correo electrónico de distribución masiva y contenido normalmente publicitario o malicioso, que se recibe sin haberlo solicitado" (Real Academia Española, s.f., definición 1 informática).

Como fue comentado en secciones anteriores, el protocolo SMTP se creó de forma insegura, para ser usado en ARPANET sin pensar en ningún uso comercial. En la explosión de masividad de Internet en 1994, como indica el documento de la Internet Society[5] (2014), se comenzó a utilizar al SMTP para distribuir mensajes con cualquier tipo de información comercial a miles de buzones y adoptando una nueva práctica de negocio. La facilidad y bajo

[5] Organización no gubernamental y sin fines de lucro, dedicada al desarrollo mundial de internet.

costo del envío de miles de mensajes en periodos cortos de tiempo, ha provocado una gran expansión de esta práctica. Según el informe de una de las empresas líderes en seguridad, Symantec (2017) indica que, el spam representa la mitad del tráfico del correo. Incluso este mismo informe remarca que el correo es el método más popular para los atacantes para esparcir códigos maliciosos. Incluso, suelen ser elaborados con el objetivo de realizar ataques dirigidos a victimas específicas.

Actualmente las estafas realizadas mediante el spam, continúan siendo básicamente las mismas que mencionan Costales y Flynt (2005) que las agrupa en cuatro categorías. La primera es la estafa o el fraude relacionado con engaños, mentiras seductoras, donde la víctima con su consentimiento envía dinero. La segunda enmarcada en robos bancarios o información de tarjetas de crédito. En este tipo de estafa, el correo electrónico aparenta proceder del banco o de la compañía de la tarjeta de crédito, por lo general alertando que la cuenta se encuentra cerrada o la tarjeta bloqueada a menos que cierta información personal sea validada. Un hipervínculo, o también conocido en inglés con el término link, es incluido dentro del mensaje del correo, para que la visita del destinatario al sitio web aparente ser legítima. Pero el correo y el sitio web son una fachada para el engaño, en realidad es mantenido por alguien esperando robar la información personal de la víctima. La tercera categoría, mencionada por los autores, es el robo de contraseñas. El spam envía información sobre nuevas membresías gratis en alguna organización, club o servicio, que al igual que en el caso anterior suele ser una fachada. El spam incluye en el mensaje un link hacia un sitio de registro de alta. Con la premisa de que los usuarios suelen usar la misma clave para cada cuenta que utilizan, roban dicha contraseña ingresada en el registro y luego la someten a prueba en todas las posibles cuentas del usuario. Por último, Costales y Flynt mencionan a los virus, los cuales mediante defectos en el HTML[6] usado en el mensaje del correo o como archivo adjunto explotando vulnerabilidades del sistema operativo, pueden infectar un ordenador y robar información leyendo cada tecla del teclado pulsada. También puede convertir el mismo en un generador de spam sin el consentimiento del usuario.

[6] Lenguaje de marcado para la elaboración de páginas web

3.4.4 Características del Spam

Para una persona puede ser relativamente fácil, aunque a veces no, reconocer un spam. Pero para una computadora es mucho más difícil. A continuación, se presentan los conceptos básicos de los comportamientos del spam.

Los autores Costales y Flynt (2005) abordan una serie de características sobre el spam, para evaluar parámetros que permitan su identificación. Según dichos autores, la primera es la falsificación de la dirección del remitente. Explican que cuando un usuario contesta un mensaje de correo, utilizando el protocolo SMTP, la respuesta generalmente se envía a la dirección listada en el campo From del encabezado. Pero cuando el mensaje es rebotado, habitualmente es enviada a la dirección de la envoltura del mensaje, que es utilizada por el servidor de correo. Los autores señalan que sitios que envían spam conocen esta propiedad y la usan a su beneficio. Saben que muchos de los correos enviados serán rebotados. Para evitar recibir los mensajes rebotados o ser identificados por sistemas de detección de spam, estos sitios advierten los autores, utilizan un falso remitente en la envoltura del mensaje, mintiendo en su identidad. Costales y Flynt continúan explicando que para evitar que el spam sea rechazado por utilizar direcciones que no existen, usualmente son usadas direcciones de remitentes reales, pero que nos les pertenecen. Este procedimiento no es ilegal, el protocolo SMTP no contiene ningún estándar que impida realizarlo. A continuación, en las figuras 2 y 3 se muestran ejemplos del análisis de una cabecera del mensaje falsificado.

Figura 2

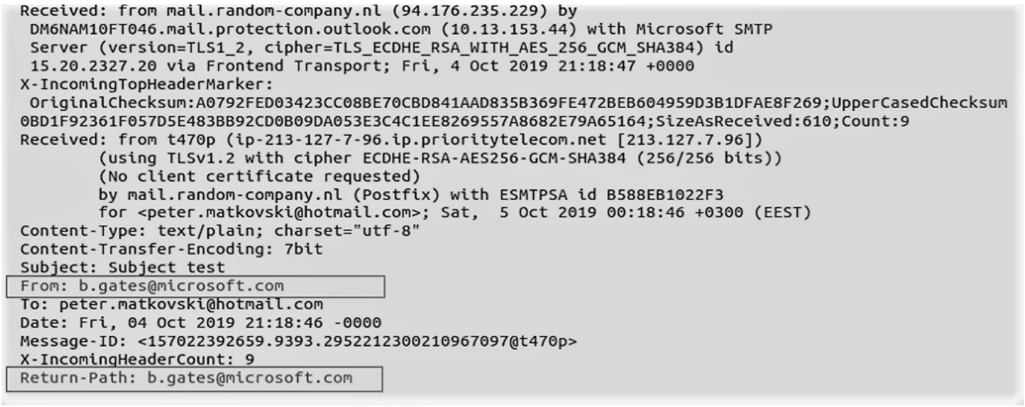
Cabecera campo From falsificado

```
Received: from mail.random-company.nl (94.176.235.229) by
 BN7NAM10FT042.mail.protection.outlook.com (10.13.156.218) with Microsoft SMTP
 Server (version=TLS1_2, cipher=TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384) id
 15.20.2327.20 via Frontend Transport; Fri, 4 Oct 2019 22:06:43 +0000
X-IncomingTopHeaderMarker:
 OriginalChecksum:6729BA403590CBA43ABB1A57C5394EB2166D97875EE6A189409FDB73E6D144E5;UpperCasedChecksum:
2C87EB7CB54295766072777C618C3BE2C3BCCB732AE85F844EADFD2402C52547;SizeAsReceived:421;Count:4
Received: from [127.0.1.1] (ip-213-127-7-96.ip.prioritytelecom.net [213.127.7.96])
        (using TLSv1.2 with cipher ECDHE-RSA-AES256-GCM-SHA384 (256/256 bits))
        (No client certificate requested)
        by mail.random-company.nl (Postfix) with ESMTPSA id EAB35B1022F3
        for <peter.matkovski@hotmail.com>; Sat,  5 Oct 2019 01:06:41 +0300 (EEST)
To: peter.matkovski@hotmail.com
From: b.gates@microsoft.com
Subject: test email
X-IncomingHeaderCount: 4
Message-ID:
 <cbb0f6c4-cd1b-46e9-8925-d9e6c2244b42@BN7NAM10FT042.eop-nam10.prod.protection.outlook.com>
Return-Path: test@random-company.nl
Date: Fri, 4 Oct 2019 22:06:43 +0000
```

Nota. La imagen muestra como el campo From fue adulterado para hacerse pasar por b.gates@microsoft.com. El campo Return-Path, dirección de rebote, no fue modificado. Tomado de https://cobracr.com/analisis-forense-de-encabezados-falsificados-email-spoofing-en-phishing/

Figura 3

Cabecera campo From y Return-Path falsificados



Nota. La imagen muestra como el campo From y el campo Return-Path fueron adulterados para hacerse pasar por b.gates@microsoft.com. Tomado de https://cobracr.com/analisis-forense-de-encabezados-falsificados-email-spoofing-en-phishing/

La segunda característica que los autores Costales y Flynt (2005) mencionan es disfrazar el título del mensaje, en inglés subject, del encabezado. Remarcan que es tentador examinar el título para identificar un spam, pero nunca se puede confiar completamente de este, ya que no todos los correos spam son creados de forma igual. Algunos títulos pueden ser bien redactados y aparentar ser benignos, otros no y dar pistas. Por este motivo, indican los autores, que se puede capturar spam, aunque se corre el riesgo de marcar correo verdadero como no deseado. Dos ejemplos de spam presentados por los autores son: " Subject: Learn how thousands are making a fortune with eBay..." o "Subject: Visit PlayboyPlus for New Playmate Pics" (Costales y Flynt, 2005). Igualmente, los autores indican que algunos de los atacantes que envían spam suelen utilizar trucos como codificar en base64[7] el título, haciéndolo difícil para su detección. Pero esto no es determinante para reconocer un spam, ya que es un método utilizado por otros idiomas para insertar títulos legales en los correos.

La tercera de las características mencionadas, es el camuflaje dentro del HTML en el cuerpo del mensaje. Costales y Flynt (2005) explican como a través del uso de los comentarios dentro de dicho lenguaje de marcado, pueden ocultarse palabras claves que podrían ser identificadas como pertenecientes al spam. Por ejemplo, indican que, un comentario en HTML se escribe de la forma: <!-- esto es un comentario -->, para ocultar la palabra Viagra se puede

[7] Sistema de numeración posicional que usa 64 como base. Es la mayor potencia que puede ser representada usando únicamente los caracteres imprimibles de ASCII

recurrir a: V<!--LIMA-->ia<!--CEBOLLA-->gr<!--LIMON-->a. También indican los autores que se pueden utilizar los caracteres especiales del HTML para camuflar un texto, por ejemplo &#097 representa el carácter de la letra a. Una manera de utilizar este método, para ocultar una referencia a una dirección de correo en el cuerpo del mensaje, sería el siguiente:

<a href="mailto:nacho@ejem-plo.com">

Luego de decodificarlo, se revela la dirección oculta tras el código:

<a href="mailto:nacho@ejemplo.com">

Antes de continuar, cabe aclarar que el termino URL, de sus siglas en inglés Uniform Resource Locator, se refiere a una dirección de internet. El término IP, corresponde a las Siglas del inglés Internet Protocol, es una etiqueta numérica, por ejemplo "192.0.10.1" que identifica, de manera lógica y jerárquica, a una interfaz en la red. Costales y Flynt (2005) indican que se puede codificar una URL, dentro del cuerpo del mensaje. Explican que ciertos correos maliciosos pueden contener links a sitios web referenciando a una URL para ser accedida por la víctima. Para intentar ocultar las direcciones del sitio web, se utilizan los valores hexadecimales[8] % + número. Por ejemplo, la letra “a” se codifica en %61. Incluso, como explican más adelante los autores, las URL pueden ser escritas con su dirección IP, intentando ocultar las mismas con su equivalente en número hexadecimal, previamente convertidas a decimal. Un ejemplo, la IP 209.15.13.134, se convierte al decimal en 3507424646, luego se convierte en hexadecimal en D10F0D86. De esta forma la referencia <a href="http://209.15.13.134"> pasa a ser la siguiente, agregando el prefijo 0x para dificultar aún más su detección, <a href="http://0xD10F0D86">. Técnicas como redirecciones, también son utilizadas para engañar en las URL, manipulando en el link de la referencia un sitio seguro o conocido y además incluir una redirección al sitio del atacante. Ejemplo de esto último, es un link referenciando a yahoo.com, pero redirigiendo el acceso del link al sitio del atacante http://realsite.com, de la siguiente manera:

<a href="http://rd.yahoo.com/bypass/winkie/food/thumb*big/dairy/noisy/-gyroscope/middle/fred/*http://real.site">.

[8] Es el sistema de numeración posicional que tiene como base el 16.

Continuando lo expuesto por Costales y Flynt (2005) también señalan la utilización de JavaScript[9] para envolver las URL, mantenerlas secretas y que luego sean decodificadas por los navegadores. En el siguiente ejemplo, se puede observar un script utilizado

```
<script language="JScript.Encode">#@~^hQAAAA=3D=3D~@#@&[Km!:+ YcADbYn'E@>
!(o"bHA~?"Z'r4OYa)Jz+!+ O, FF+R8*f&^kxV 4YhVr~qq9:C{!P_2&!C:'TPwI)\AAr"92"> '!,j/I}
SdqHMxEWE@*@!&qwI)\A@*BbI@#@&AyIAAA==^#~@</script>
```

Otra característica es intentar engañar los detectores de firmas. En computación, los detectores de firmas trabajan calculando un valor para un bloque de texto, un mensaje o archivos. Este método suele ser utilizado comúnmente por los antivirus. El MD5 es un algoritmo de reducción criptográfico de 128 bits ampliamente usado. Uno de sus usos es el de comprobar que algún archivo no haya sido modificado. Por ejemplo, si se utiliza el generador de hash MD5 para el siguiente texto: no responder correos de bancos solicitando información personal. Su resultado es ce84035d17c88b6f3a1b44314dcc7adf. El resultado es único, la modificación de un solo carácter en el texto modifica completamente todo el hash generado. Dicha firma, es guardada en una base de datos, para luego comparar con un nuevo ejemplo y buscar coincidencias para detectar que el mensaje ya ha sido visto antes. Como explican Costales y Flynt (2005), programas de detección de spam utilizan técnicas similares para reconocimiento. Pero aclaran que todas estas formas de detección comparten la acción de examinar texto del mensaje. Los sitios que envían spam, conocen la existencia de analizadores de texto, por lo que agregan texto aleatorio en cada mensaje enviado. Los autores explican que, para detectar spam, se debe evitar el texto aleatorio y hacer foco en partes del mensaje que no cambian, por ejemplo, referencias web, direcciones de mail o números de teléfono cuando sea posible.

3.4.5 Métodos Simples de Detección de Spam

En los comienzos sobre la protección del spam, fueron creadas simples contramedidas mediante la observación de las características básicas del spam y los métodos de envío. Las mismas han sido muy efectivas y se utilizaron algunas de estas ideas para realizar la investigación del presente trabajo. Según Tan (2016) se clasifican las técnicas simples de protección de cuatro maneras. Se puede evitar que los remitentes de spam obtengan las

[9] Lenguaje de programación. Se define como orientado a objetos, basado en prototipos, imperativo, débilmente tipado y dinámico.

direcciones de correo electrónico cambiando las direcciones almacenadas por mínimos caracteres, Tan lo llama protección de dirección. Otra es el filtro de palabras clave, la cual es una forma de juzgar las clases de correos testeando si existen o no ciertas palabras predefinidas. Por ejemplo, palabras sexuales o violentas. Se puede elaborar una serie de filtros conocidos como lista negra y lista blanca. Ambos métodos están basados en reconocimiento de la identidad del remitente. Lista negra, continúa Tan, es una manera de filtrar el spam rechazando correos de una dirección especifica. Explica que algunas veces, cierta información contenida en la cabecera del correo suele ser adulterada por el creador del spam. En cambio, señala que lista blanca permite solo recibir correo de direcciones declaras. Por último, Tan menciona que se encuentra la lista gris y desafío-respuesta. El autor indica que se utiliza cuando los correos que no son reconocidos se rechazan temporalmente, hasta que sean considerados legítimos. El método de desafío-respuesta se refiere a direcciones de correo que no están en lista blanca que son agregadas luego de su correspondiente verificación.

3.4.6 Enfoques Inteligentes de Detección de Spam

Según Tan (2016) los enfoques de detección inteligente son las tecnologías más efectivas y ampliamente utilizadas en el campo. Por un lado, indica el autor, son altamente automatizados y no necesitan de mucha intervención humana. Por otro lado, se caracterizan por su alta precisión, robustez y fuerte tolerancia al ruido y puede adaptarse a los cambios dinámicos del contenido de los correos electrónicos. Tan remarca que la detección de spam es un problema de clasificación, el cual puede ser resuelto por métodos de machine learning. Estos métodos, extraen información del conjunto de datos de entrenamiento y construyen clasificadores basados en los principios de aprendizaje para catalogar nuevas muestras de correos electrónicos.

Tan menciona que excepto de la necesaria participación del humano en la generación del proceso de entrenamiento, los procesos de aprendizaje y clasificación son completamente automáticos. Mientras tanto, el modelo de aprendizaje se puede adaptar a los cambios dinámicos del contenido de los correos electrónicos ajustando las especificaciones de entrenamiento y actualizando los clasificadores. Como indica Tan, muchos de los métodos clásicos de machine learning han sido aplicados satisfactoriamente en la detección de spam, incluyendo Naive Bayes, Support Vector Machine, K-Nearest Neighbor, Artificial Neural Network y Boosting. Su rendimiento depende del armado del modelo, la extracción y clasificación de los datos.

Como se mencionó anteriormente, los detectores de spam basados en firmas pueden ser muy efectivos si el spam ya es conocido o ya ha sido descubierto por alguna herramienta antivirus. Sin embargo, no resultan útiles para la detección de aquellos spam capaces de cambiar sus remitentes y estructuras constantemente. La necesidad por encontrar nuevos métodos de detección está dada por el alto grado de propagación que poseen los spams. Esto ha llevado a que se comenzaran a explorar nuevas alternativas capaces de brindar una solución a este problema. Métodos de detección y clasificación utilizando técnicas de machine learning han arrojado muy buenos resultados.

3.5 La Inteligencia Artificial

Hasta hace unos pocos años, la Inteligencia Artificial o IA parecía un asunto futurista que difícilmente podría cambiar el día a día en el corto plazo. Sin embargo, hoy ya es una realidad que comienza a revolucionar varios aspectos de nuestra sociedad. Russel & Norving (2004) responden a la pregunta de que es la IA concibiendo por un lado a sistemas que piensan y actúan como humanos. Y por el otro lado a sistemas que piensan y actúan racionalmente. Los autores destacan que el enfoque racional, basado en las leyes del pensamiento lógico de la mente, son los que construyen la IA.

Alan Turing planteó realizar una prueba para dar una definición de inteligencia, se la conoce como la Prueba de Turing. “El computador supera la prueba si un evaluador humano no es capaz de distinguir si las respuestas, a una serie de preguntas planteadas, son de una persona o no” (Russell & Norvig, 2004, p. 3)

Un concepto central en la inteligencia artificial es el de agente racional. Básicamente “un agente es cualquier cosa capaz de percibir su medioambiente con la ayuda de sensores y actuar en ese medio utilizando actuadores” (Russel & Norving, 2004, p.37). El termino actuadores se utiliza para indicar el elemento que reacciona a un estímulo realizando una acción. Además, la palabra percepción se usa para indicar que el agente puede recibir entradas o estímulos a estos sensores. Por lo tanto, “un agente tomara una decisión en un momento dado dependiendo de la secuencia completa de percepciones hasta ese instante” (Russel & Norving, 2004, p.38).

Vale aclarar que, según Russel & Norving, para entender el funcionamiento de la inteligencia artificial, se debe comprender que la secuencia completa de percepciones depende de un aprendizaje previo que debe realizar el agente. Luego podrá tomar una acción, obteniendo

un resultado mejor, en función de una nueva percepción. Dicho de otro modo, se puede precisar que "en cada posible secuencia de percepciones, un agente racional deberá emprender aquella acción que supuestamente maximice su medida de rendimiento, basándose en las evidencias aportadas por la secuencia de percepciones y en el conocimiento que el agente mantiene almacenado." (Russel & Norving, 2004, p.41).

Se puede decir que hablamos de Inteligencia Artificial cuando la maquina no sigue un algoritmo o receta para resolver un problema, sino que encuentra su propia forma de resolverlo.

3.6 Machine Learning

3.6.1 Machine Learning

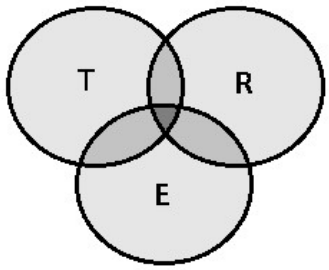
Tanto en la IA como en machine learning las neuronas artificiales son un concepto importante. Warren McCulloch y Walter Pitts, los autores reconocidos en el primer trabajo de IA, propusieron un modelo construido con neuronas artificiales, donde cada una puede estar activada o desactivada. La activación se puede definir como una "respuesta a la estimulación producida por una cantidad suficiente de neuronas vecinas" (Russell & Norvig, 2004, p. 19).

Este concepto de neurona artificial, necesita de algún procedimiento lógico para funcionar. Mitchell (1997) remarca la importancia del aprendizaje inductivo en el machine learning. Explica que los algoritmos de aprendizaje inductivo pueden, en el mejor de los casos, garantizar que la hipótesis se ajuste al objetivo buscado en base a los datos de entrenamiento. A falta de más información, se supone que la mejor hipótesis con respecto a las posibilidades desconocidas, es la hipótesis que mejor se ajusta a los datos de entrenamiento observados. En otras palabras, Mitchell dice que cualquier hipótesis, que se aproxima bien a la función objetivo sobre un conjunto suficientemente grande de ejemplos de entrenamiento, también se aproximará bien a la función objetivo sobre otros ejemplos no observados.

De una manera un poco más simple, Sarkar et al. (2018) expresa que el machine learning es un área que consiste en algoritmos de aprendizaje que mejoran su rendimiento al ejecutar alguna tarea, usando la experiencia, a través del tiempo. Los componentes se representan en la figura 4.

Figura 4

Componentes del algoritmo de aprendizaje



Nota. T: tarea; R: rendimiento; E: experiencia.

En la figura anterior, Sarkar et al. (2018) define a una tarea T en un problema del mundo real a ser resuelto. La experiencia E la obtienen los algoritmos de machine learning, o modelos, a partir de lo que aprenden del conjunto de datos. Este proceso de ir ganando experiencia, explica Sarkar, es iterativo y es conocido como entrenamiento del modelo. En cuanto al rendimiento R, el autor lo define como una medida cuantitativa o métrica, para determinar que tan bien el algoritmo o modelo está ejecutando la tarea T, con experiencia E. Suele referirse a este resultado como precisión.

El proceso de aprendizaje del modelo consta de dos etapas:

- Entrenamiento (training): su objetivo es aprender a partir de un conjunto de datos conocidos. Para ello se deben utilizar conceptos matemáticos, principalmente algebra lineal y estadística, junto a algoritmos de aprendizaje.
- Evaluación (testing): a partir de lo aprendido en la etapa anterior, se predicen o infieren resultados mediante nuevos datos.

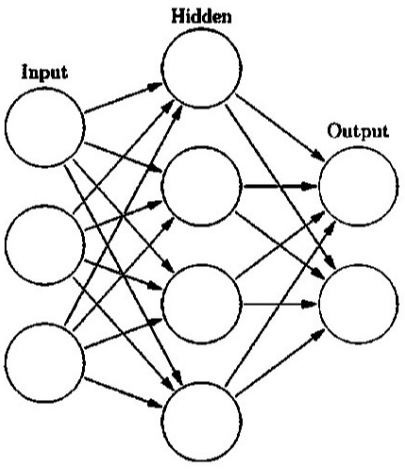
Como indica Mitchell (1997) aunque la tarea de aprendizaje es determinar una hipótesis idéntica al concepto de un objetivo sobre todo el conjunto de instancias, la única información disponible sobre este objetivo es su valor sobre los ejemplos de entrenamiento. Por lo tanto, indica el autor, que los algoritmos de aprendizaje inductivo pueden, en el mejor de los casos, garantizar que la hipótesis de salida se ajuste al objetivo sobre los datos de entrenamiento. A falta de más información, continúa el autor, se asume que la mejor hipótesis con respecto a instancias que no fueron contempladas es la hipótesis que mejor se ajusta a los datos de entrenamiento observados. Esta es la suposición fundamental de la inducción.

Un concepto importante es el Deep Learning o aprendizaje profundo. Como explica Sarkar et al. (2018), es un sub campo del machine learning que recientemente se ha vuelto prominente. Sarkar indica que los algoritmos basados en Deep Learning implican el uso de conceptos del aprendizaje de representaciones, donde varias representaciones de los datos se aprenden en diferentes capas. Simplificando la definición, se intenta construir una inteligencia de maquina representando los datos como una jerarquía de capas de conceptos, donde cada capa de conceptos es construida desde otra capa más simple. En cualquier técnica básica de machine learning supervisada, explica Sarkar, que básicamente lo que intentamos es aprender a mapear entre nuestra muestra de datos y nuestra salida (output) y entonces intentar predecir la salida de muestras de datos nuevas.

En este sentido, una de las arquitecturas que utilizan Deep Learning son las Redes Neuronales Artificiales o RNA. El mismo autor nos ofrece una clara definición, indicando que es un modelo computacional y una arquitectura que simula las neuronas biológicas y la forma en que funciona nuestro cerebro. Sarkar explica que las Redes Neuronales Artificiales usualmente cuentan con capas interconectadas de nodos. Estas interconexiones son análogas a nuestra red de neuronas de nuestros cerebros. La arquitectura tiene una capa de entrada (input layer), una capa de salida (output layer) y al menos una capa oculta (hidden layer) entre ellas unidas con interconectores, como se muestra en la figura 5.

Figura 5

Red Neuronal Artificial típica



Nota. Tomado de Practical Machine Learning with Python: A Problem-Solver's Guide to Building Real-World Intelligent Systems. (p. 31) Por Sarkar, D., Bali, R. y Sharma, T., 2018, Apress.

Sarkar et al. (2018) explica que cualquier RNA básica siempre tendrá múltiples capas de nodos, patrones específicos de conexión y enlaces entre las capas, pesos de las conexiones y funciones de activación para los nodos que convierten entradas en salidas. El proceso de aprendizaje de la red usualmente implica una función de costo y el objetivo es optimizar la función de costo, es decir minimizar el costo. Estos pesos continúan siendo actualizados en el proceso de aprendizaje.

Como si fuera poco, además es necesario mencionar el algoritmo de Backpropagation o retro propagación. El mismo es una técnica muy popular para entrenar RNA, tiene dos etapas principales, la propagación y la actualización de pesos. Sarkar et al. (2018) divide en estas dos etapas una serie de pasos. Para la propagación, el primer paso es que los datos de entrada de los vectores de muestra sean propagados adelante a través de la red neuronal para generar los valores de salida en la capa de salida. Luego compara el vector de salida generado, con el vector de salida actual o deseado para ese dato del vector de entrada. Computa la diferencia del error en las unidades de salida. Finalmente, retro propaga o propaga en dirección contraria los valores de error para generar los deltas en cada nodo. En cuanto a la actualización del peso, consta de dos pasos. El primero es computar los gradientes de los pesos mediante la multiplicación del delta (error) de salida y la activación de entrada. Luego, usa la tasa de aprendizaje para determinar el porcentaje del gradiente para ser sustraído desde el peso original y actualiza el peso de todos los nodos.

Sarkar aclara que estas dos etapas, son repetidas múltiples veces con múltiples iteraciones o épocas hasta que obtenemos resultados satisfactorios. Usualmente, indica Sarkar, la retro propagación se utiliza en algoritmos o funciones de optimización.

3.6.2 Clasificación de métodos de aprendizaje

Los métodos de aprendizaje pueden ser clasificados en cuatro, basados en la supervisión humana en el proceso de aprendizaje. Dichos métodos que exponemos a continuación están basados en las definiciones de Sarkar et al. (2018). El primero, según dicho autor, es el aprendizaje supervisado, el cual realiza su entrenamiento con los datos de entrada (conjunto de entrenamiento) y sus respectivos datos de salida (conocido como etiqueta o label). Explica que el concepto consiste en que el algoritmo pueda aprender las relaciones que existen entre las variables independientes *X1, X2, ..., Xn* y la variable dependiente *Y*, y de este modo utilizar el aprendizaje para realizar predicciones sobre nuevos datos o datos desconocidos. El segundo

método que menciona Sarkar es el aprendizaje no supervisado, el cual requiere que el modelo sea entrenado con una serie de observaciones tanto de las variables independientes, como de la dependiente. Sin embargo, señala que en muchos casos la información de la variable dependiente es desconocida, por lo que es imposible entrenar un modelo. Son en estas situaciones en las que un modelo no supervisado es extremadamente útil. Sarkar explica que estos algoritmos intentan aprender cuales son las relaciones, patrones o estructuras internas inherentes a los datos sin ningún tipo de ayuda o supervisión, como en el caso del aprendizaje no supervisado. El aprendizaje no supervisado se concentra más en intentar extraer conocimiento o información útil de los datos, más que en predecir salidas. El tercero que menciona el autor es el aprendizaje semi-supervisado, siendo una combinación de los métodos supervisados y no supervisados. Indica Sarkar que estos métodos normalmente utilizan para su entrenamiento una gran cantidad de datos que no poseen etiqueta, y una pequeña que sí. Existen múltiples técnicas disponibles en la forma de métodos generativos, graficas basados en métodos, métodos basados en heurística, etc. Por último, el cuarto método mencionado por Sarkar, es el aprendizaje por esfuerzos. Donde se tiene un agente que se lo quiere entrenar en un ambiente determinado y durante un periodo de tiempo, de modo de ir mejorando su performance basándose en las acciones que éste ejecuta sobre dicho ambiente. Normalmente, comenta Sarkar, el agente comienza con un conjunto de estrategias o reglas para interactuar con el ambiente. Al observar dicho ambiente, toma acciones particulares basándose en las reglas o políticas y observando el estado actual del ambiente. De acuerdo a la acción que tomó, el agente obtiene una recompensa o una penalización. Este es un proceso iterativo en el que el algoritmo irá aprendiendo y modificando su estrategia de ser necesario, de modo de obtener la recompensa deseada afirma Sarkar.

3.6.3 Tareas de métodos de aprendizaje

Es posible categorizar los métodos de aprendizaje de acuerdo al tipo de salida que se desea obtener de los algoritmos de machine learning. Haciendo referencia a las definiciones de Sarkar et al. (2018), destacamos las siguientes dos tareas llamadas Clasificación y Regresión.

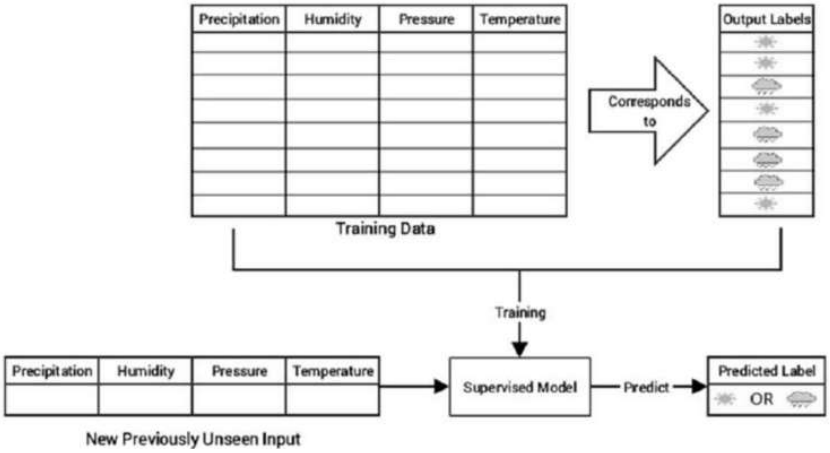
La tarea basada en Clasificación, indica Sarkar, se encuentran dentro del aprendizaje supervisado, en donde el objetivo principal es predecir una categoría o etiqueta para cada uno de los datos de entrada basándose en lo que el modelo ha aprendido en la etapa de entrenamiento. El autor explica que estas etiquetas de salida también son conocidas como clases

o etiquetas de clase, las cuales por naturaleza son categóricas, por lo que no poseen orden y son discretas. Por lo tanto, Sarkar indica que cada etiqueta de salida pertenecerá a una clase o categoría discreta especifica. Enuncia que existe una amplia variedad de algoritmos que pertenecen a esta familia, pero los más importantes son los modelos Lineales, como Logistic Regression (regresión logística), Naive Bayes y Support Vector Machines. También se encuentran los modelos no paramétricos, como los K vecinos más cercanos (K-nearest neighbors). Luego los modelos basados en Arboles como arboles de decisión (Decision Trees). Ademas los modelos de Ensamble como Random Forest y Gradient Boosted Machines (boosting).

Sarkar et al. (2018) explica que la tarea de clasificación es binaria cuando lo que se tiene son dos categorías para diferenciar. Un ejemplo de clasificación binaria podría ser el problema de clasificación de emails. En ese problema lo que se desea es distinguir entre dos categorías: “Es Spam” o “No es Spam”. En el ejemplo, propuesto por los autores, de la figura 6, se distingue dos categorías: “soleado” o “no soleado” es decir lluvioso en este ejemplo.

Figura 6

Aprendizaje supervisado, clasificación binaria



Nota. Clasificación para la predicción del tiempo, soleado o lluvioso. Tomado de Practical Machine Learning with Python: A Problem-Solver’s Guide to Building Real-World Intelligent Systems. (p. 36) Por Sarkar, D., Bali, R. y Sharma, T., 2018, Apress.

La multi-clases, según Sarkar et al. (2018), se considera una extensión al problema de clasificación binario. En este caso, explican, se tienen más de dos categorías o clases que el dato puede pertenecer. Por ejemplo, un problema de clasificación de multi-clases propuesto por los autores, es determinar la categoría de dígitos del cero al nueve que han sido escritos a mano.

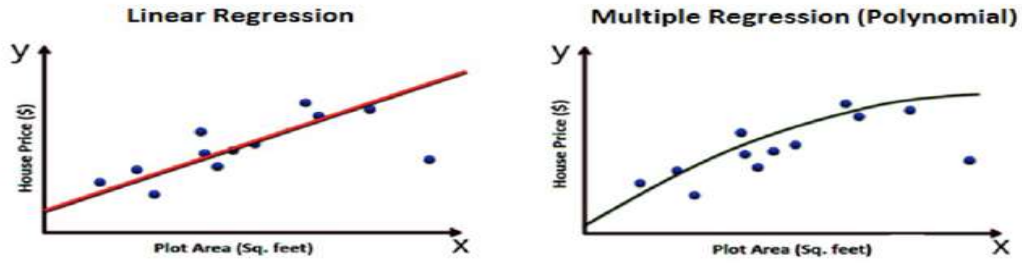
Con lo cual, lo que se tiene es un problema de clasificación de diez clases. La clasificación multi-etiqueta, es mencionada por los autores cuando los problemas de clasificación normalmente involucran datos que pueden pertenecer a más de una categoría o poseer más de una etiqueta. Los autores dan el ejemplo en el caso de la predicción de la categoría de artículos de noticias que pueden tener múltiples etiquetas, como política, ciencia, religión, etc.

Continuando con la categorización de las tareas, se encuentra la Regresión. Sarkar et al. (2018) señala que los modelos basados en regresiones son entrenados a partir de un conjunto de observaciones, cada una conformada por un conjunto de variables independientes y un valor continuo como dependiente. Así el modelo hará uso de estos valores para aprender las relaciones que existen entre las variables independientes y su correspondiente variable dependiente. A partir de este conocimiento, Sarkar explica que el modelo sabrá como predecir nuevos valores continuos cuando le sean suministradas nuevas observaciones no vistas anteriormente.

Sarkar et al. (2018) propone uno de los más comunes ejemplos del mundo real sobre regresión, la predicción de precios de casas. Plantean los autores, que se puede crear un modelo de regresión simple para predecir los precios de la vivienda en función de los datos relacionados con las áreas de terreno en metros cuadrados. En la figura 7, se muestran dos posibles modelos de regresión, basados en diferentes métodos para predecir los precios de las casas basados en los metros cuadrados del terreno.

Figura 7

Aprendizaje supervisado, modelos de regresión



Nota. Tomado de Practical Machine Learning with Python: A Problem-Solver's Guide to Building Real-World Intelligent Systems. (p. 37) Por Sarkar, D., Bali, R. y Sharma, T., 2018, Apress.

La idea de este ejemplo, comentan los autores, es que se intenta determinar si hay alguna relación o asociación entre la característica del área de los datos y la variable de salida, que es

el precio de la casa y es lo que se quiere predecir. Señalan que cuando se aprenda esta relación, se podrá predecir el precio de las casas en un futuro dada cualquier área de tierra. Como se puede apreciar en la figura 7, se presentan dos tipos de modelos con el propósito de mostrar que puede haber multiples maneras de construir el modelo. El principal objetivo es minimizar los errores durante el entrenamiento y validación del modelo parar en un futuro realizar buenas predicciones.

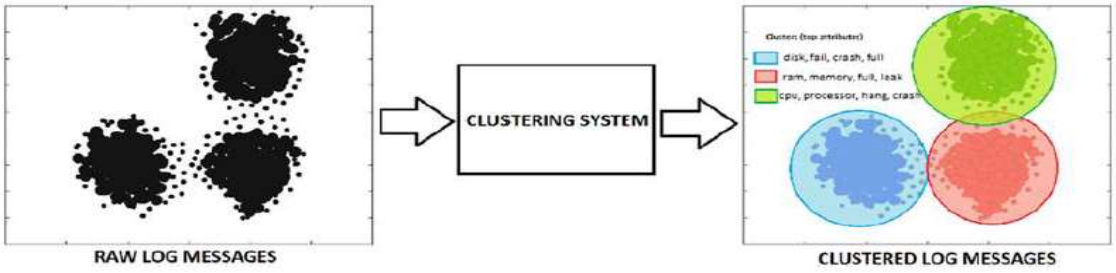
Sarkar et al. (2018) explican que el modelo de regresión lineal intenta generar relaciones de datos con una característica o variable X descriptiva y una variable Y de respuesta donde el objetivo es predecir Y. En cambio, los autores explican que la regresión múltiple es conocida como una regresión multivariable. Señalan que este método intenta generar relaciones donde se tiene una respuesta o variable Y de salida, pero múltiples variables en forma de vector X en lugar de una única variable. En la vida real, se suelen utilizar más características, como la cantidad de habitaciones, número de baños, cantidad de pisos, etc. Entonces, explican los autores, que basados en todos esos atributos, el modelo intenta aprender una relación, entre cada característica del vector en lugar de una única variable característica. Con esto la idea es predecir Y basados en las diferentes características presentes en X.

Finalizando con los métodos abarcados en el presente trabajo, mencionaremos el Clustering, el cual pertenece a los modelos de algoritmos no supervisados, según continúa explicando Sarkar et al. (2018). Dicho proceso consiste en agrupar datos similares que no hayan sido previamente etiquetados o clasificados. Las salidas de este tipo de modelos son grupos de datos segregados que son similares entre sí, pero diferentes a los miembros de los demás grupos. La mayor diferencia que existe con los modelos supervisados es que aquí no se tienen los datos previamente clasificados para entrenar al modelo, por lo que se utiliza todo el conjunto de datos como entrada.

En la figura 8 se observa el ejemplo de un sistema de clustering, agrupando mensajes de registro o logs según grupos afines.

Figura 8

Aprendizaje no supervisado, clustering



Nota. Mensajes de registro o logs, Celeste: disco, Naranja: memoria ram, Verde: CPU, procesador. Tomado de Practical Machine Learning with Python: A Problem-Solver's Guide to Building Real-World Intelligent Systems. (p. 37) Por Sarkar, D., Bali, R. y Sharma, T., 2018, Apress.

Los modelos de clustering pueden ser de diferentes tipos, de acuerdo con sus metodologías o principios que dividimos en tres. El primero es el clustering basado en partición, que definirá una noción de similaridad. Esta similaridad es una característica que se deriva de los atributos de entrada luego de aplicadas funciones matemáticas. Luego, una vez encontradas estas similitudes, se agrupan los datos que son similares en un solo grupo y separados de aquellos que son diferentes. Los modelos de clustering basados en partición, en general son desarrollados utilizando técnicas recursivas para clasificarlos. Por ejemplo, se comienza con una porción arbitraria de los datos y, basados en alguna medida de similitud, se continúa reasignando datos hasta que se llegue a un punto estable según algún criterio. Ejemplos de estas técnicas son: K-means, K-medoids, CLARANS, etc. El segundo es el clustering Jerárquico, que se diferencian del clustering basado en partición por la manera en la que son desarrollados y por cómo trabajan. Dentro del paradigma del clustering jerárquico, se comienza con, o bien todos los datos en un solo grupo (divisive clustering), o todos los datos en diferentes grupos (aglomerative). Según el punto de entrada elegido, se continúa dividiendo el gran grupo en grupos más pequeños o clusters basados en algún criterio de similitud, o bien se puede continuar juntando diferentes grupos o clusters en grupos más grandes basados en el mismo criterio. El proceso concluye cuando se llega a una condición preestablecida. Para finalizar las metodologías de clustering mencionamos al basado en densidad. Como se había mencionado, los dos métodos anteriores son fuertemente dependientes de la noción de distancia. Lo que conduce a estos algoritmos a encontrar clusters de datos de forma esférica. Esto puede ser un problema si los datos se encuentran ubicados arbitrariamente. Esta limitación podría ser resuelta

si en lugar de tener en cuenta el concepto de distancia, se utilizara el concepto de densidad de los datos para desarrollar el modelo. Entonces, la metodología para encontrar puntos ya no consiste en encontrar puntos próximos a uno en particular, sino a determinar ´áreas que contengan puntos. Este tipo de modelos no resultan simples de interpretar como los de métricas de distancia, pero ayudan con clusters que no son necesariamente de forma esférica. Ejemplos de estos modelos son DBSCAN y OPTICS.

3.7 Modelos de Clasificación

Respecto al machine learning, el problema de la detección de spam, puede ser considerado un problema de clasificación. En el cual lo que se intenta identificar es si una muestra es un correo malicioso. Podemos decir que la clasificación es binaria: la muestra es o no es un spam.

3.7.1 Resumen

En la presente sección, se proporciona un marco teórico a algunos de los métodos y técnicas que fueron utilizados en la investigación.

3.7.2 Naive Bayes

Es un algoritmo de clasificación de machine learning que utiliza el modelo de red de Bayes o método Bayesiano. Para comprender acerca del mismo, cabe mencionar lo siguiente:

> El aprendizaje bayesiano simplemente calcula la probabilidad de cada hipótesis dados los datos, y realiza predicciones sobre estas bases. Es decir, se realizan todas las predicciones utilizando todas las hipótesis, ponderadas por sus probabilidades, y no utilizando únicamente la mejor hipótesis. De esta manera, el aprendizaje se reduce a inferencia probabilística. (Russel & Norving, 2004, p.812)

En este sentido, los autores indican que el modelo Naive Bayes, Bayes simple o ingenuo se define porque “asume que los atributos son condicionalmente independientes entre sí, dada la clase” (Russel & Norving, 2004, p.818).

Para definir al teorema Bayes de la figura 9 de forma precisa, tenemos que introducir ciertas notaciones. Mitchell (1997) indica que escribiremos P(h) para denotar la probabilidad inicial que tiene la hipótesis h, previamente de que hayamos observado los datos de entrenamiento. Además, aclara que suele llamarse a P(h) la probabilidad previa de h y puede reflejar cualquier conocimiento previo que tengamos sobre la posibilidad de que h sea una hipótesis correcta. El autor indica que, si no tenemos tal conocimiento previo, entonces podríamos simplemente asignar la misma probabilidad previa a cada hipótesis candidata. Prosigue que, de manera similar, escribiremos P(D) para denotar la probabilidad previa de que se observen los datos de entrenamiento D. A continuación, escribiremos P(D|h) para denotar la probabilidad de observar los datos D dado un mundo en el que se cumple la hipótesis h. De forma más general, escribimos P(x|y) para denotar la probabilidad de x dada y. En los problemas de machine learning, indica Mitchell, estamos interesados en la probabilidad P (h|D) de que h se cumpla dados los datos de entrenamiento observados D. Además, el autor explica que se denomina P(h|D), probabilidad posterior de h, porque refleja nuestra confianza en que h se mantiene después de haber visto los datos de entrenamiento D. Mitchell remarca observar que la probabilidad posterior P(h|D) refleja la influencia de los datos de entrenamiento D, en contraste con la probabilidad anterior P(h), que es independiente de D. Según Mitchell el teorema de Bayes es la piedra angular de los métodos de aprendizaje bayesianos porque proporciona una forma de calcular la probabilidad posterior P(h|D), a partir de la probabilidad previa P(h), junto con P(D) y P(D|h).

Figura 9

Teorema de Bayes

$$P(h|D) = \frac{P(D|h)P(h)}{P(D)}$$

Nota. Tomado de Machine Learning. (p. 156). Por Mitchell, T. 1997. McGraw-Hill

Como se puede deducir intuitivamente de la figura 9, P(h|D) aumenta con P(h) y con P(D|h) de acuerdo al teorema de Bayes. Es también razonable ver que P(h|D) decrece a medida que P(D) incrementa.

El clasificador Bayes ingenuo, como indica Mitchell (1997) se aplica a tareas de aprendizaje en las que cada instancia x se describe mediante una conjunción de valores de atributo y donde la función de destino f(x) puede tomar cualquier valor de algún conjunto finito V. Además el autor indica que un conjunto de ejemplos de entrenamiento de la función de

destino u objetivo (target) es proporcionada y es presentada una nueva instancia, descrita por la tupla de valores de atributo (al, a2.. .an). Luego se le pide predecir el valor objetivo (target) o clasificación, para esta nueva instancia.

El enfoque Bayesiano, según Mitchell (1997) para clasificar la nueva instancia es el de asignar el valor objetivo más probable vMAP, dados los valores de atributos (a1, a2…an) que describe la instancia, según la siguiente definición $v_{MAP} = \underset{v_j \in V}{\text{argmax}} P(v_j|a_1, a_2 \ldots a_n)$

El clasificador Bayes ingenuo está basado, como indica Mitchell, en la suposición simplificada de que los valores de atributos son condicionalmente independientes dado el valor objetivo. El autor quiere decir que, la suposición es que dado el valor objetivo de la instancia, la probabilidad de observar la conjunción a1, a2…an es justo el producto de las probabilidades de cada atributo individual $P(a_1, a_2 \ldots a_n|v_j) = \prod_i P(a_i|v_j)$. Finalmente, Mitchell explica que, sustituyéndolo en la anterior ecuación, luego de reescribirla, tenemos el enfoque utilizado por el clasificador de Bayes ingenuo, como se observa en la figura 10

Figura 10

Clasificador de Naive Bayes

$$v_{NB} = \underset{v_j \in V}{\text{argmax}} P(v_j) \prod_i P(a_i|v_j)$$

Nota. Tomado de Machine Learning. (p. 176). Por Mitchell, T. 1997. McGraw-Hill

Una diferencia interesante, remarca Mitchell (1997), entre el método de aprendizaje bayesiano ingenuo y otros métodos de aprendizaje es que no hay una búsqueda explícita a través del ritmo de las posibles hipótesis. En cambio, indica el autor que, la hipótesis se forma sin buscar, simplemente contando la frecuencia de varias combinaciones de datos dentro de los ejemplos de entrenamiento. A esto sumamos que “El aprendizaje de Bayes simple funciona sorprendentemente bien en una gran variedad de aplicaciones” (Russel & Norving, 2004, p.818). El método al asumir independencia generalmente simplifica el problema.

3.7.3 Logistic Regression

Para abordar la técnica Logistic Regression de machine learning, debemos entender la regresión. Según Stamp (2018), la regresión a menudo se contrasta con la clasificación. El autor indica que, en un problema de clasificación, se clasifican muestras en dos o más categorías. Sin

embargo, aclara que en regresión no se retrocede. En cambio, la regresión puede verse como el proporcionar una puntuación (o probabilidad) en lugar de una clasificación directa. Stamp define en forma general que el análisis de regresión puede ser visto como un método para razonamiento sobre la relación entre dos o más cosas.

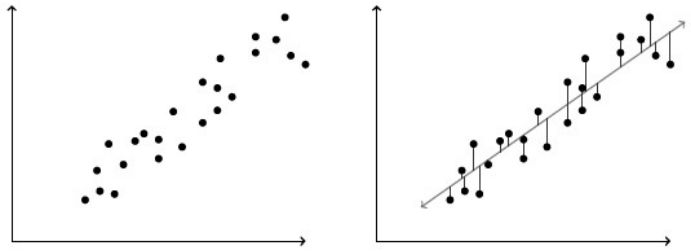
A continuación, se trata el tema de regresión lineal (linear regression), luego el de regresión logística (logistic regression).

En linear regression, Stamp (2018) indica que, se determina un modelo lineal que mejor se ajuste a los datos dados, en el sentido de que el modelo minimiza la suma del error cuadrático de los términos. El problema, continua el autor, puede ser resuelto eficientemente usando el algoritmo lineal de mínimos cuadrados. Stamp, propone el ejemplo del precio de las casas, para entender gráficamente la regresión lineal.

Como se puede observar en la Figura 11, la regresión lineal o línea de regresión ha reducido datos en dos dimensiones a una sola dimensión.

Figura 11

Línea de Regresión



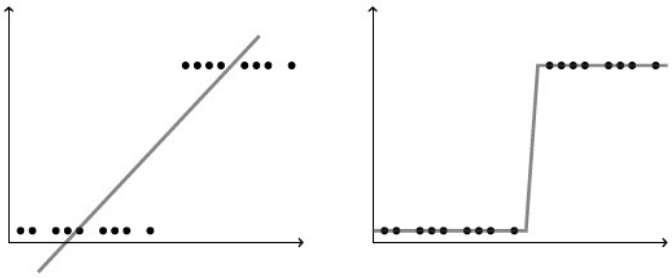
Nota. Tomado de Introduction to Machine Learning with Applications in Information Security (p. 206) por Stamp, M. 2018. CRC Press

En cuanto al problema de clasificación binaria, solo hay dos resultados posibles. Podemos utilizar el ejemplo de clasificar muestras de correo en spam o no spam. Stamp, plantea un caso similar clasificando software como malware o benigno. Según Stamp (2018), linear regression es un método simple e intuitivo, pero no ideal para problemas de clasificación binaria, ya que solo hay dos resultados, sin casos en el medio. En la Figura 12 se observa que la línea de regresión no es muy útil. El problema, indica Stamp, es que los datos se agrupan alrededor de spam (1) y no spam (0), mientras que la línea es uniforme en el espacio entre ellos. Se puede obtener una solución, indica el autor, si se cambia la linear regression por alguna otra

función. El autor continúa comentando que se busca una función que no desperdicie mucho esfuerzo en el área entre los dos grupos, ya que ahí nada interesante ocurre. Una posible solución es una función lineal por partes, representada en el gráfico de la derecha de la Figura 12.

Figura 12

Ejemplos de Regresión



Nota. Tomado de Introduction to Machine Learning with Applications in Information Security (p. 206) por Stamp, M. 2018. CRC Press

En matemáticas, la solución anteriormente mencionada, es definida como función logística (logistic function), para todo t, 0<F(t)<1

$$F(t) = \frac{1}{1 + e^{-t}}$$

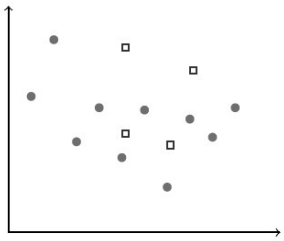
En logistic regression, señala Stamp (2018), usamos la función logística F(t) para modelar los datos, donde t es una función lineal en los datos. Se puede interpretar, indica Stamp, a la función logística como la probabilidad de que se pertenezca a la clase o grupo asociado al problema de clasificación binario. Es importante remarcar el concepto de Stamp en la comparación de logistic regression con Naive Bayes. Según Stamp (2018), en Naive Bayes se hace una suposición de independencia, lo que hace que el problema resultante sea relativamente más fácil de resolver. En contraste, enuncia Stamp, para logistic regression directamente modelamos los parámetros relevantes sin suposiciones simplificadoras. Concluye Stamp diciendo que logistic regression provee una solución aproximada a un problema exacto, mientras Naive Bayes otorga una solución exacta a un problema aproximado.

3.7.4 K-Nearest Neighbors

El algoritmo de K-Nearest Neighbors, según Stamp (2018), es quizás uno de los más sencillos de implementar. El autor indica que, dado un conjunto de datos etiquetados de entrenamiento y una función de distancia, K-NN clasifica un punto X basándose en los k elementos de dicho conjunto que se encuentren más cerca de X. Propone que consideremos los datos de entrenamiento de la Figura 13, el conjunto consiste de diez elementos de circulo sólido y cuatro elementos de cuadrado hueco.

Figura 13

K-nearest neighbors datos de entrenamiento

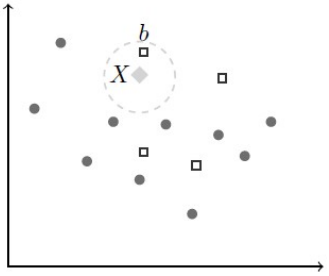


Nota. Tomado de Introduction to Machine Learning with Applications in Information Security (p. 178) por Stamp, M. 2018. CRC Press.

Stamp propone suponer que queremos clasificar el diamante que está etiquetado con la X en la Figura 14 usando 1-NN (K-NN con k=1), que es también conocido con el término del algoritmo del vecino más cercano. Como el punto más cercano a X es el cuadrado hueco etiquetado b, clasificaremos a X como un cuadrado hueco.

Figura 14

K-nearest neighbors 1-NN

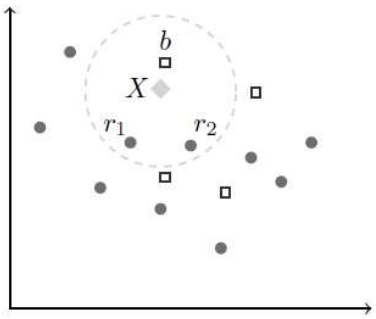


Nota. Tomado de Introduction to Machine Learning with Applications in Information Security (p. 178) por Stamp, M. 2018. CRC Press.

Se puede presentar el caso, como explica Stamp, de pretender clasificar X usando 3-NN representado en la Figura 15. Podemos observar que los tres puntos cercanos a X consisten de un cuadrado hueco y dos círculos sólidos, etiquetados b, r1 y r2 respectivamente. Como la mayoría de los tres puntos cercanos en el conjunto de entrenamiento son círculos sólidos, usando 3-NN, clasificamos X del mismo tipo que los círculos sólidos.

Figura 15

K-nearest neighbors 3-NN



Nota. Tomado de Introduction to Machine Learning with Applications in Information Security (p. 178) por Stamp, M. 2018. CRC Press.

Stamp (2018) señala que el número de vecinos más cercanos (k) y la medida de distancia son componentes clave para el algoritmo de K-Nearest Neighbors. Un valor pequeño de k resultará en una baja precisión, sobre todo en conjuntos de datos con mucho ruido, dado que cada instancia del conjunto de entrenamiento tiene un alto peso durante el proceso de decisión. Un valor grande de k, continúa el autor, disminuye la performance del algoritmo. Además, si el valor es muy grande, el modelo puede hacer un sobreajuste, dificultando la separación entre clases, lo cual también resultaría en menos precisión. Una buena regla es utilizar un k menor a la raíz cuadrada de n, siendo n el número total de patrones de entrenamiento.

Su facilidad y sencillez de implementar son un aspecto positivo, pero no resultan aptos para todos los casos. Si el conjunto de datos se encuentra distribuido desigualmente, el algoritmo de K-Nearest Neighbors no tendrá un buen rendimiento afirma Stamp.

3.7.5 Support Vector Machines

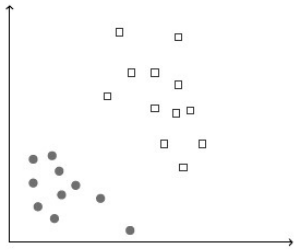
El algoritmo Support Vector Machines (SVM) de machine learning, es utilizado según Stamp (2018), generalmente para resolver problemas de clasificación. El objetivo principal de

un SVM, explica el autor, es encontrar un hiperplano que separe las clases de la mejor manera posible. Donde un hiperplano, explica Stamp, es definido como un subespacio de una dimensión menos que el espacio en el cual se está trabajando. Por ejemplo, si los datos con los cuales se trabaja viven en un espacio bidimensional, un hiperplano es simplemente una línea. Si el hiperplano existe, se dice que los datos son linealmente separables.

En la siguiente Figura 16, el autor propone el ejemplo de muestras de malware (círculos solidos) y muestras benignas (cuadrados vacios). Los datos son claramente linealmente separables.

Figura 16

SVM datos de entrenamiento

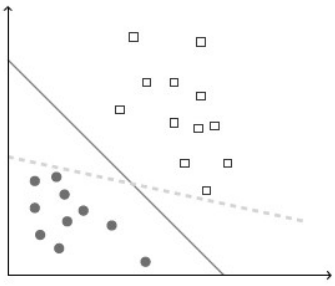


Nota. Tomado de Introduction to Machine Learning with Applications in Information Security (p. 98) por Stamp, M. 2018. CRC Press

En la Figura 17, se puede apreciar dos distintos hiperplanos separados. Cualquiera de estos hiperplanos que se seleccione puede ser usado para clasificación, es decir se clasifica en base al lado en que se encuentre del hiperplano.

Figura 17

SVM hiperplanos

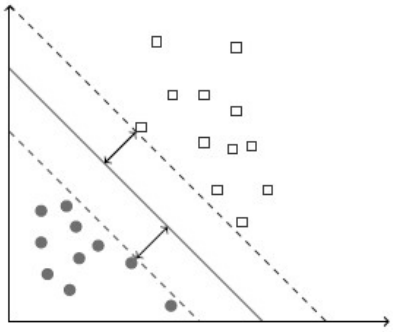


Nota. Dos hiperplanos en línea sólida y punteada. Tomado de Introduction to Machine Learning with Applications in Information Security (p. 98) por Stamp, M. 2018. CRC Press

A la hora de elegir un hiperplano para el SVM, señala Stamp, lo que se busca es uno que maximice el margen. El margen se define, continúa el autor, como la distancia mínima entre el hiperplano y cualquier elemento del training set. En el ejemplo, el hiperplano optimo es dado por el de la línea sólida. Si se observa la Figura 18, las flechas son las que representan el margen. Por consiguiente, la línea solida es la óptima separación.

Figura 18

SVM margen



Nota. Tomado de Introduction to Machine Learning with Applications in Information Security (p. 99) por Stamp, M. 2018. CRC Press

De acuerdo a lo comentado por Stamp (2018), no se tiene garantías de que los datos puedan ser separados linealmente, puede ocurrir que no exista un hiperplano separador. Continúa explicando que SVM emplea dos técnicas para manejar datos de entrenamiento que no son separables linealmente. Un margen suave permite algunos errores de clasificación cuando se determina un hiperplano de separación. Cuantos más errores de clasificación se pueda soportar, en general, mayor es el margen. En SVM, Stamp indica que existe un parámetro definido por el usuario, que especifica la suavidad del margen. En la práctica, el valor de éste parámetro puede determinarse por ensayo y error.

Otra técnica empleada, continúa el autor, en el entrenamiento de SVM es mapear los datos de entrada a un espacio de características donde el problema de construir un hiperplano de separación es más manejable. En la práctica, generalmente implica transformar los datos de entrada a un espacio de características de mayor dimensión.

Un concepto importante, retomando la Figura 18, es que los puntos de datos de entrenamiento que se encuentran en cualquiera de las líneas punteadas, son conocidos como vectores de soporte (support vectors). La función Kernel, como explica Stamp (2018),

utilizando los vectores de soporte (X) y los productos internos vectoriales, nos permite mejorar la separabilidad al trabajar en un espacio dimensional superior. El autor presenta las ecuaciones de cuatro funciones de Kernel. Se definen a continuación.

Lineal (Linear Kernel):

$$K(X_i, X_j) = X_i \cdot X_j$$

Aprendizaje de máquina polinómico (Polynomial learning machine):

$$K(X_i, X_j) = (X_i \cdot X_j + 1)^p$$

Función de base radial Gaussiana (Gaussian radial basis function o RBF):

$$K(X_i, X_j) = e^{-(X_i - X_j) \cdot (X_i - X_j)/(2\sigma^2)}$$

Perceptrón de dos capas (Two.layer perceptron):

$$K(X_i, X_j) = \tanh\big(\beta_0(X_i \cdot X_j) + \beta_1\big)$$

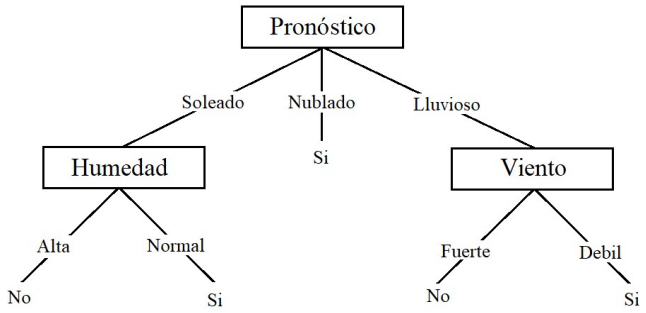
3.7.6 Decision Tree

El algoritmo Decision Tree (árbol de decisión) es fácil de construir, permite la creación de complejos procesos de decisión y sus resultados son muy intuitivos de interpretar. Según Mitchell (1997), clasifica las instancias ordenándolas hacia abajo en el árbol desde la raíz hasta algún nodo hoja. Comenta que cada nodo en el árbol declara una prueba de algún atributo de la instancia, cada rama descendiente de ese nodo corresponde a uno de los posibles valores para ese atributo. Una instancia se clasifica, indica Mitchell, comenzando en el nodo raíz del árbol, evaluando el atributo específico para ese nodo, luego bajando por la correspondiente rama del árbol al valor del atributo. El autor menciona que éste proceso es entonces repetido por el sub-árbol en el nuevo nodo.

Tomamos el ejemplo propuesto por Mitchell, para graficar la estructura en la Figura 19. En el mismo se plantea la decisión de jugar al tenis.

Figura 19

Decision Tree estructura de ejemplo



Nota. Adaptado de Machine Learning. (p. 53). Por Mitchell, T. 1997. McGraw-Hill

El algoritmo básico, lleva el nombre de ID3. El mismo, como indica Mitchell (1997), aprende arboles de decisión al construirlos de arriba hacia abajo, comenzando con la pregunta ¿Cuál atributo debe ser probado en la raíz del árbol? Para responder esa pregunta, el autor explica que se evalúa cada instancia del atributo usando una prueba estadística para determinar que tan bien clasifica los ejemplos de entrenamiento solo. El mejor atributo se selecciona y se utiliza como prueba en el nodo raíz del árbol. Un descendiente del nodo raíz es creado por cada posible valor de ese atributo, los ejemplos de entrenamiento son ordenados al correspondiente nodo descendiente. Luego se repite todo el proceso usando los ejemplos de entrenamiento asociados con cada nodo descendiente, para seleccionar el mejor atributo para probar en ese punto en el árbol. Finalmente, el autor indica que este proceso forma una búsqueda aceptable de un árbol, en la cual el algoritmo nunca retrocede para reconsiderar elecciones anteriores.

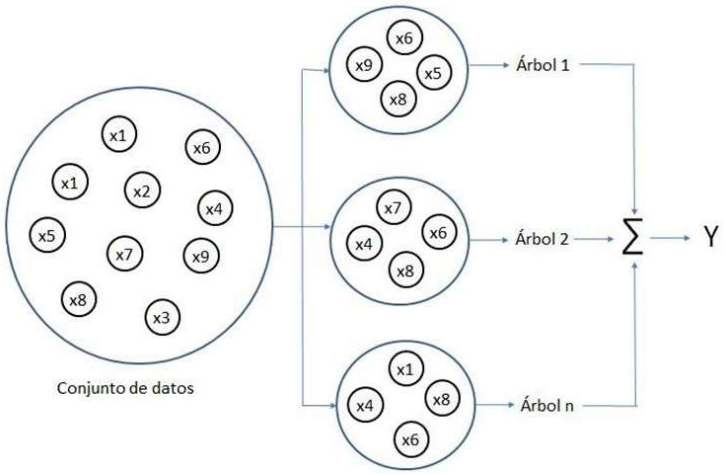
3.7.7 Random Forest

El algoritmo Random Forest es otra técnica de aprendizaje supervisado, pero que genera múltiples árboles de decisión sobre un conjunto de datos de entrenamiento. Es una generalización de un árbol de decisión (decisión tree). Según Stamp (2018), una forma de generalizar es entrenar múltiples árboles. Explica que se puede seleccionar diferentes subconjuntos de datos de entrenamiento y construir un árbol por cada subconjunto, luego se utiliza una mayoría de los votos de los arboles resultantes para determinar la clasificación. Señala el autor, que el procedimiento es conocido como embolsar (bagging) las observaciones,

dicha estrategia tiene menos probabilidades de sobreajuste (overfit) ya que tiende a generalizar mejor los datos de entrenamiento. En la Figura 20 se puede observar los subconjuntos

Figura 20

Random Forest subconjuntos y combinación



Dividimos la serie de datos en varios subconjuntos compuestos aleatoriamente de muestras, de ahí el término random (aleatorio) de random forest. Las salidas de todos los árboles se combinan en un resultado final Y (conocida como ensamblado) que se obtiene mediante alguna regla, generalmente el promedio, cuando las salidas de los árboles del ensamblado son numéricas y conteo de votos, cuando las salidas de los árboles del ensamblado son categóricas.

3.7.8 Redes Neuronales

Las Redes Neuronales o redes neuronales artificiales, como ya se explicó en sección anterior en este trabajo, son una clase de técnicas de machine learning que intentan modelar la interconexión neuronal del cerebro. Haciendo referencia a esto último "la neurona se dispara cuando una combinación lineal de sus entradas excede un determinado umbral" (Russel & Norving, 2004, p.838)

Para comprender su funcionamiento, los autores realizan la siguiente explicación:

> Las redes neuronales están compuestas de nodos o unidades … conectadas a través de conexiones dirigidas. Una conexión de la unidad *j* a la unidad *i* sirve para propagar la activación *aj* de *j* a *i.* Además cada conexión tiene un peso numérico Wji asociado, que

determina la fuerza y el signo de la conexión. Cada unidad *i* primero calcula una suma ponderada de sus entradas. (Russel & Norving, 2004, p.839)

A continuación, la ecuación de la suma ponderada:

$$in_i = \sum_{j=0}^{n} W_{j,i} a_j$$

Luego, los autores señalan que, se aplica una función de activación g a esta suma para producir la salida:

$$a_i = g(in_i) = g\left(\sum_{j=0}^{n} W_{j,i} a_j\right)$$

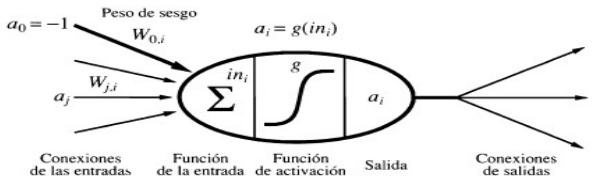
Es importante detenernos en comprender un poco más sobre la función de activación, sin entrar en conceptos muy específicos, exponemos la siguiente explicación:

La función de activación *g* se diseña con dos objetivos. Primero, queremos que la unidad esté activa (cercana a +1) cuando se proporcionen las entradas correctas, e inactiva (cercana a 0) cuando se den las entradas erróneas. Segundo, la activación tiene que ser no lineal, en otro caso la red neuronal en su totalidad se colapsaría con una sencilla función lineal. (Russel & Norving, 2004, p.839)

Un modelo matemático para una neurona se puede observar en la figura 21.

Figura 21

Redes Neuronales modelo de neurona



Nota. Tomado de Inteligencia Artificial, un enfoque moderno (p. 839) por Russell, S. & Norvig, P. 2004. Pearson

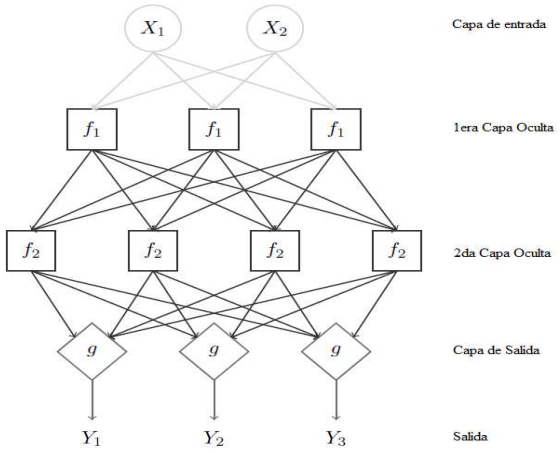
Como señala Stamp (2018), hay muchos tipos diferentes de redes neuronales, pero nos centraremos en el perceptron[10] multicapa (MLP). El perceptron multicapa es una técnica de aprendizaje supervisado que incluye una capa de entrada, una o más capas ocultas y

[10] es una neurona artificial, por tanto, una unidad de red neuronal.

una capa de salida. Cada capa está conectada completamente a la capa anterior. La siguiente Figura 22 representa un MLP con dos capas ocultas:

Figura 22

Redes Neuronales MLP



Nota. Adaptado de Introduction to Machine Learning with Applications in Information Security (p. 180) por Stamp, M. 2018. CRC Press.

Las Redes Neuronales son muy efectivas para problemas de alta complejidad, cuentan con poderosas opciones de puesta a punto para evitar problemas en el ajuste del modelo. Si bien estas redes pueden resultar muy robustas y poderosas, también pueden ser complejas y difíciles de implementar.

3.8 Estado del Arte

Para la elaboración de este trabajo, se hallaron las siguientes investigaciones recientes relacionadas con machine learning y spam.

Chaudhari et al (2022) publicaron un documento donde implementan y comparan el rendimiento de detección de spam utilizando machine learning. Evalúan tres técnicas Naive Bayes, Decision Tree y Multilayer Perceptron. Sobre las dos técnicas en común con nuestro trabajo, observamos un valor de resultado muy similar. Los resultados presentados aquí, refuerza las evidencias actuales sobre estos clasificadores.

Varun Kumar et al (2002) publicaron un artículo donde solo se evalúa Naive Bayes y Decision Tree en la detección de Spam. En el mismo encontramos que realizaron una muy breve

descripción de los pasos en la implementación, a diferencia del anterior documento que no lo poseía. Recurrieron, como en nuestra investigación, a Google Colab para desarrollar el proyecto. Los resultados de exactitud son similares en Naive Bayes, pero Decision Tree es inferior en un 4,9% respecto a nuestra métrica.

Airan & Lal Bhari (2022) publicaron un documento donde evalúan cuatro técnicas de machine learning para la clasificación de spam. Cuentan con un desigual dataset de correo spam y ham al igual que la presente investigación. Sin embargo, los resultados de las técnicas Decision Tree, Support Vector y Random Forest son muy diferentes. Los mismos arrojan un bajo rendimiento. Lamentablemente no se conoce el conjunto inicial de datos.

Abayomi-Alli et al (2022) publicaron un artículo de investigación sobre un método de aprendizaje profundo para la clasificación de spam. La propuesta utiliza redes neuronales recurrentes (RNN), específicamente el modelo Bi-LSTM (Long-Short Term Memory bidireccional). Realizan el entrenamiento con dos datasets distintos, uno es propio del proyecto y el otro es tomado de UCI, al igual que nuestra investigación. Comparan los resultados del método elegido con otras publicaciones. Pero no utilizan el dataset de UCI para comparar los resultados con las técnicas de Machine Learning utilizadas en nuestra investigación.

Reddy & Kakulapati (2021) publicaron una investigación sobre la clasificación de spam utilizando el algoritmo de Random Forest. En la misma se obtiene el resultado de exactitud muy similar al registrado por nuestra investigación.

En los trabajos relevados no se encontraron investigaciones recientes sobre las técnicas Logistic Regression, tampoco se pudo consultar los detalles de implementación y en general no se conoce el conjunto inicial de datos. Tampoco se pudo tener evidencia de cómo se trabajaron los datasets para que pudieran utilizarse en el entrenamiento de los métodos.

En general Naive Bayes obtiene los mejores rendimientos en las investigaciones realizadas. Las técnicas de machine learning y redes neuronales son la estrategia para abordar el problema del spam en la actualidad. Investigaciones recientes del Sinhgad Institute of Technology Lonavala en la India[11], crearon una nueva técnica para la detección automática de correos electrónicos no deseados. Este modelo se basa en la selección de características multi-objetivo bajo una técnica de aprendizaje profunda nueva y prometedora. La diferencia con otros métodos desarrollados antes, es que el modelo se entrena en un conjunto de datos de imagen y

[11] https://link.springer.com/article/10.1007/s41315-021-00217-9

texto. Cuando se usan datasets con texto, dos técnicas de extracción de características son usadas, como Term Variance (TV) y Term Frecuency-Inverse Document Frecuency (TF-IDF). Esta última utilizada también en el presente trabajo. En el manejo de datasets de imágenes son usadas como extracción Fisher Discriminate Analysis (FDA), Walsh-Hadamard Transform (WHT) y color correlogram (correlograma de color). Para reducir la complejidad del entrenamiento, se realiza la selección de características multi-objetivo mediante el algoritmo meta-heuristico hibrido Grey-Sail Fish Optimization (G-SFO). La eficiencia del modelo sugerido se encuentra en evaluación.

4. Metodología

4.1 Modalidad de la investigación

En el presente capítulo se explican los pasos para realizar la investigación, detallando el cronograma, las variables y los datos iniciales necesarios. Se enfoca en el diseño de las técnicas utilizadas para el análisis del conjunto de datos, también se describe la estructura y formato original de los mismos. La información resultante es la que conforma el muestreo que se utiliza en los algoritmos.

La metodología orienta el proceso de investigación, con ella definimos una serie de pasos que se deben cumplir para documentar el trabajo. Mediante la observación, la hipótesis y la experimentación se realiza la demostración de un hecho o suceso. En este trabajo se ponen a prueba técnicas de machine learning sobre un mismo conjunto de datos, para así comparar la efectividad de precisión de dichas técnicas. Por lo expuesto se realizó una investigación con enfoque metodológico cuantitativo. El objetivo es analizar técnicas de machine learning para la detección de spam.

La investigación se divide en dos etapas, la primera es obtener el conjunto de datos (dataset) con ejemplos de mails y spams y su posterior acondicionamiento para ser utilizado por el modelo de machine learning en su entrenamiento. La segunda, es la construcción, puesta a prueba y comparación de los modelos de Support Vector, K-Nearest Neighbors, Naive Bayes, Decision Tree, Logistic Regression y Random Forest.

A continuación, se enumeran los procesos realizados en la investigación, divididos en dos etapas:

1) Procesamiento de los datos:
    - Obtener el dataset
    - Procesamiento y exploración del Dataset
    - Construir gráfica para identificar palabras frecuentes en spam y ham
    - Remover palabras vacías y signos de puntuación
    - Convertir texto en vectores
2) Construcción del modelo de clasificación de spam
    - Separar los datos en los sets de entrenamiento y prueba del modelo
    - Utilizar clasificador Sklearn para construir el modelo

- Entrenar el modelo
- Obtener valor de exactitud

Se aclara que Sklearn es una librería que permite acceder y utilizar los algoritmos de Machine Learning en Python.

4.2 Cronograma

Se presenta a continuación Gantt del presente trabajo, se detallan las actividades y la duración de periodos que representan una quincena

Figura 23

Cronograma

4.3 Variables de Investigación

Determinar la importancia de las variables no sólo ayuda a lograr una mejor interpretación de los datos, sino también permite seleccionar aquellas características que son realmente importantes para el modelo de predicción y comprobar la hipótesis.

En este sentido, se utilizan las variables de:

- **Técnicas de machine learning**: Support Vector, K-Nearest-Neighbor, Naive Bayes, Decision Tree, Logistic Regression, Random Forest.
- **Datos**: texto del mensaje de correo, etiqueta que define si es o no spam
- **Exactitud**: valor de detección de spam, a partir del entrenamiento, para cada técnica de clasificación.

El proceso de distinguir las características que son las más útiles o relevantes para resolver un determinado problema, es la selección de las variables o atributos. Si las características son elegidas de manera que puedan existir valores similares para resultados diferentes entonces no importa que tan bueno sea el algoritmo de machine learning, siempre cometerá errores. Para evitar este problema se analizaron los datos a fin de tener un conjunto significativo que permita un correcto aprendizaje.

4.4 Población del trabajo y conjunto inicial de datos

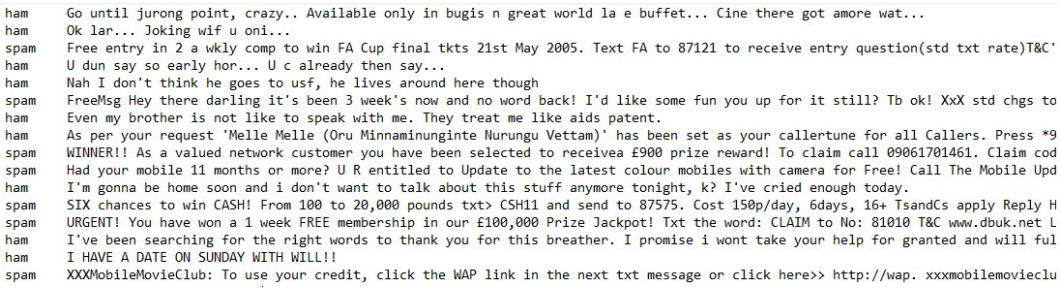
La construcción del modelo de clasificación de machine learning está presentada por la obtención de los datos. Como se mencionó anteriormente, se obtuvieron datos de tipo texto y número de acuerdo a las características elegidas. El conjunto inicial de datos (dataset) fue obtenido del sitio web UCI Machine Learning Repository[12], el cual contiene una colección de base de datos útiles para utilizar en sistemas de aprendizaje.

De acuerdo al dataset mencionado, contamos con una población del trabajo de 5574, es decir el número de mensajes de correo a utilizar por el modelo. Estos datos corresponden a atributos reales, no ficticios, en idioma inglés. El motivo de dicha elección se justifica en realizar un aprendizaje válido, sobre una base a datos reales, para la presente investigación.

[12] Del sitio web: https://archive.ics.uci.edu/ml/datasets/SMS+Spam+Collection

Figura 24

Conjunto inicial de datos

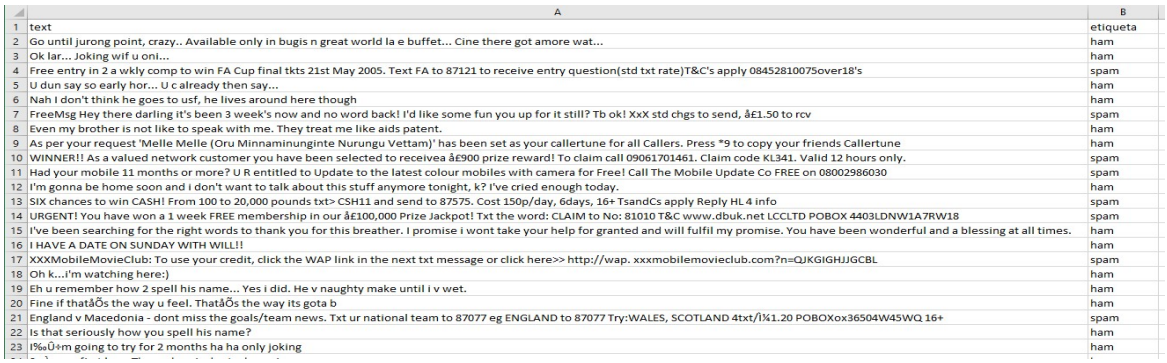


```
ham     Go until jurong point, crazy.. Available only in bugis n great world la e buffet... Cine there got amore wat...
ham     Ok lar... Joking wif u oni...
spam    Free entry in 2 a wkly comp to win FA Cup final tkts 21st May 2005. Text FA to 87121 to receive entry question(std txt rate)T&C'
ham     U dun say so early hor... U c already then say...
ham     Nah I don't think he goes to usf, he lives around here though
spam    FreeMsg Hey there darling it's been 3 week's now and no word back! I'd like some fun you up for it still? Tb ok! XxX std chgs to
ham     Even my brother is not like to speak with me. They treat me like aids patent.
ham     As per your request 'Melle Melle (Oru Minnaminunginte Nurungu Vettam)' has been set as your callertune for all Callers. Press *9
spam    WINNER!! As a valued network customer you have been selected to receivea £900 prize reward! To claim call 09061701461. Claim cod
spam    Had your mobile 11 months or more? U R entitled to Update to the latest colour mobiles with camera for Free! Call The Mobile Upd
ham     I'm gonna be home soon and i don't want to talk about this stuff anymore tonight, k? I've cried enough today.
spam    SIX chances to win CASH! From 100 to 20,000 pounds txt> CSH11 and send to 87575. Cost 150p/day, 6days, 16+ TsandCs apply Reply H
spam    URGENT! You have won a 1 week FREE membership in our £100,000 Prize Jackpot! Txt the word: CLAIM to No: 81010 T&C www.dbuk.net L
ham     I've been searching for the right words to thank you for this breather. I promise i wont take your help for granted and will ful
ham     I HAVE A DATE ON SUNDAY WITH WILL!!
spam    XXXMobileMovieClub: To use your credit, click the WAP link in the next txt message or click here>> http://wap. xxxmobilemovieclu
```

Antes de iniciar el análisis de los datos que conforman el dataset, es necesario estudiar la propia estructura del archivo. El mismo fue transformado a un archivo de extensión .csv o documento Excel de Microsoft Office, delimitado por punto y coma, de 5574 filas y 2 columnas con cada uno de sus atributos extraídos y utilizados como variables. En las columnas, se observan: el texto del mensaje del correo y la etiqueta que clasifica si el correspondiente mensaje es spam o no lo es.

Figura 25

Conjunto inicial de datos .csv



	A	B
1	text	etiqueta
2	Go until jurong point, crazy.. Available only in bugis n great world la e buffet... Cine there got amore wat...	ham
3	Ok lar... Joking wif u oni...	ham
4	Free entry in 2 a wkly comp to win FA Cup final tkts 21st May 2005. Text FA to 87121 to receive entry question(std txt rate)T&C's apply 08452810075over18's	spam
5	U dun say so early hor... U c already then say...	ham
6	Nah I don't think he goes to usf, he lives around here though	ham
7	FreeMsg Hey there darling it's been 3 week's now and no word back! I'd like some fun you up for it still? Tb ok! XxX std chgs to send, å£1.50 to rcv	spam
8	Even my brother is not like to speak with me. They treat me like aids patent.	ham
9	As per your request 'Melle Melle (Oru Minnaminunginte Nurungu Vettam)' has been set as your callertune for all Callers. Press *9 to copy your friends Callertune	ham
10	WINNER!! As a valued network customer you have been selected to receivea å£900 prize reward! To claim call 09061701461. Claim code KL341. Valid 12 hours only.	spam
11	Had your mobile 11 months or more? U R entitled to Update to the latest colour mobiles with camera for Free! Call The Mobile Update Co FREE on 08002986030	spam
12	I'm gonna be home soon and i don't want to talk about this stuff anymore tonight, k? I've cried enough today.	ham
13	SIX chances to win CASH! From 100 to 20,000 pounds txt> CSH11 and send to 87575. Cost 150p/day, 6days, 16+ TsandCs apply Reply HL 4 info	spam
14	URGENT! You have won a 1 week FREE membership in our å£100,000 Prize Jackpot! Txt the word: CLAIM to No: 81010 T&C www.dbuk.net LCCLTD POBOX 4403LDNW1A7RW18	spam
15	I've been searching for the right words to thank you for this breather. I promise i wont take your help for granted and will fulfil my promise. You have been wonderful and a blessing at all times.	ham
16	I HAVE A DATE ON SUNDAY WITH WILL!!	ham
17	XXXMobileMovieClub: To use your credit, click the WAP link in the next txt message or click here>> http://wap. xxxmobilemovieclub.com?n=QJKGIGHJJGCBL	spam
18	Oh k...i'm watching here:)	ham
19	Eh u remember how 2 spell his name... Yes i did. He v naughty make until i v wet.	ham
20	Fine if thatåÕs the way u feel. thatåÕs the way its gota b	ham
21	England v Macedonia - dont miss the goals/team news. Txt ur national team to 87077 eg ENGLAND to 87077 Try:WALES, SCOTLAND 4txt/ì¼1.20 POBOXox36504W45WQ 16+	spam
22	Is that seriously how you spell his name?	ham
23	I‰Û÷m going to try for 2 months ha ha only joking	ham

Como se puede observar en la figura 25, la columna text contiene los mensajes de correo o lo que comúnmente se encuentra en el cuerpo del correo. Por otra parte, la columna etiqueta, representa la clasificación de los mensajes. Se aclara que la denominación ham, es utilizada usualmente en el lenguaje informático para identificar correo que no es spam. Es importante destacar en la Figura 25, la línea 17, la cual contiene un enlace a una página web maliciosa here>> http://wap. xxxmobilemovieclub.com?n=QJKGIGHJJGCBL. Este es un claro ejemplo de lo mencionado en el marco teórico del presente trabajo.

Muchas veces se trabaja con datos con magnitudes que resultan muy diferentes entre sí. Esto puede ser un problema para muchos algoritmos de machine learning que son sensibles al estandarizado de atributos, los cuales podrían tener un comportamiento erróneo si sus valores no fueran similares. Dependiendo del problema que se quiere abordar, se pueden utilizar diferentes técnicas, tanto de estandarización, como de normalización de los datos, para que estos resulten útiles al utilizar los algoritmos de machine learning. Para la presente investigación se recurrió a la remoción de palabras vacías, signos de puntuación y tokenización.

Además, la presencia de datos faltantes, incorrectos o nulos en un dataset suele tener un impacto negativo en la performance del modelo que se entrene. Por este motivo la primera tarea que suele llevarse a cabo en todo previo procesamiento de datos es la de búsqueda e identificación de los mismos y su correspondiente corrección.

A continuación, exponemos algunos ejemplos de la depuración de los datos.

En la siguiente Figura 26, se puede observar los caracteres *<#>* en las filas 436, 437 y la ~ en la fila 440 que no revisten de importancia para el análisis, por lo cual se eliminaron del texto de los mensajes.

Figura 26

Depuración de los datos

434	Congrats! Nokia 3650 video camera phone is your Call 09066382422 Calls cost 150ppm Ave call 3mins vary from mobiles 16+ Close 300	spam
435	Booked ticket for pongal?	ham
436	You available now? I'm like right around hillsborough & <#> th	ham
437	The message sent is askin for <#> dollars. Shoul i pay <#> or <#> ?	ham
438	Ask g or iouri, I've told the story like ten times already	ham
439	How long does applebees fucking take	ham
440	Hi hope u get this txt~journey hasnt been gd,now about 50 mins late I think.	ham

Estos caracteres suelen utilizarse en el formato HTML para representar símbolos como se indica en la Figura 27.

Figura 27

Símbolos HTML

Entidad	Carácter	Descripción	Traducción
&lt;	<	less than	signo de menor que
&gt;	>	more than	signo de mayor que
&amp;	&	ampersand	ampersand
&quot;	"	quotation mark	comillas
	(espacio en blanco)	non-breaking space	espacio en blanco
&apos;	'	apostrophe	apóstrofo

Como se indicó anteriormente no revisten de importancia para el entrenamiento, recurriendo a su eliminación. además, al resultado se lo modificó para que el mensaje tuviera sentido de ser, es decir tenga coherencia:

Figura 28

Depuración de los datos cont.

434	Congrats! Nokia 3650 video camera phone is your Call 09066382422 Calls cost 150ppm Ave call 3mins vary from mobiles 16+ Close 300	spam
435	Booked ticket for pongal?	ham
436	You available now? I'm like right around hillsborough &amp th	ham
437	The message sent is askin for dollars. Shoul i pay 10 or 13 ?	ham
438	Ask g or iouri, I've told the story like ten times already	ham
439	How long does applebees fucking take	ham
440	Hi hope u get this txt journey hasnt been gd,now about 50 mins late I think.	ham

La existencia de valores nulos o mensajes vacíos en el dateset no puede permitirse. Por este motivo no hubo otra alternativa más que eliminar las filas correspondientes.

Luego de realizar el análisis del conjunto inicial de datos, disponemos de un total de 5568 mensajes depurados. En cuanto a la clasificación de los mismos, se dividen en 747 catalogados como spam y 4821 como ham (no spam). Para el aprendizaje que requiere el modelo se utilizó el 80% del total de los mensajes, dejando un 20% restante para la prueba o evaluación.

4.5 Técnicas de recolección de datos a emplear

La recolección de los datos, como se ha explicado, necesita de una serie de filtros para la correcta utilización de los mismos en los algoritmos de aprendizaje. En la presente investigación de recurrió al método de aprendizaje supervisado. En el cual, se utilizan tanto los datos de entrenamiento y los datos de prueba. El primer conjunto de entrenamiento corresponde a lo que formará parte de la experiencia de aprendizaje, conteniendo la información significativa para el algoritmo. El segundo conjunto, el de prueba, permite validar la experiencia adquirida y predecir la información nueva. Esto puede percibirse como el proceso de transformar una entrada particular en una salida deseada, para ello tiene que haber pasado por el aprendizaje de la obtención de la salida.

Una de las técnicas utilizadas en el proceso de recolección de datos fue la división de las palabras contenidas en los mensajes clasificados como spam y como ham. De esta manera, se puede trabajar el universo de cada grupo de palabras de forma que el algoritmo pueda realizar

el aprendizaje correctamente. Se utilizaron dos contenedores definidos con el nombre palabras_ham y palabras_spam. Una vez cargados los contenedores con las palabras, se recurrió a una de las técnicas más frecuentemente recurridas en implementaciones de machine learnig. La técnica es Natural Language Processing (NLP), la cual permite interpretar el lenguaje humano mediante un amplio rango de tareas de análisis, extracción, procesamiento y visualización. El principal objetivo es tokenizar el texto. Las herramientas o clasificadores de machine learning normalmente trabajan con números. Al trabajar con texto, sea el caso de los mensajes de correo analizados en el presente trabajo, es necesario convertirlos a datos numéricos para que pueda ser interpretado por el algoritmo. Esta técnica es llamada proceso de tokenización y consiste en segmentar el texto en frases o palabras. La mencionada operación permite dividir una frase en tokens o grupo de tokens, estos últimos llamados ngram, permitiéndonos identificar su frecuencia de aparición, como así también reconocer si se trata de sustantivos o verbos. A su vez, se eliminan las palabras vacías que no aportan significado, como artículos, pronombres, preposiciones, etc.

Para ejecutar la técnica anteriormente mencionada, utilizamos la librería Natural Language Toolkit (NLTK) escrita en el lenguaje de programación Phyton. En la Figura 29 se puede comparar el texto previo y posterior a la tokenización y limpieza de palabras vacías.

Figura 29

Librería NLTK

	text
0	Go until jurong point, crazy.. Available only ...
1	Ok lar... Joking wif u oni...
2	Free entry in 2 a wkly comp to win FA Cup fina...
3	U dun say so early hor... U c already then say...
4	Nah I don't think he goes to usf, he lives aro...

	text
0	Go jurong point crazy Available bugis n great ...
1	Ok lar Joking wif u oni
2	Free entry 2 wkly comp win FA Cup final tkts 2...
3	U dun say early hor U c already say
4	Nah dont think goes usf lives around though

Se recurrió a la librería de Phyton wordcloud para armar una nube de palabras y poder identificar las que más peso o aparición tienen en los mensajes. La nube de palabras es un gráfico que resalta las palabras con el tamaño de su fuente y colores. Resulta de utilidad para comprender el universo de los datos, con los cuales el algoritmo realizará el aprendizaje.

Como se mencionó previamente, es necesario trabajar con datos numéricos en machine learning, por lo que se recurrió a transformar los datos de la columna etiquetas. Con el método

llamado replace de Phyton, realizamos dicha tarea reemplazando la palabra ham y spam por 0 y 1 respectivamente.

Figura 30

Método replace

	text	etiqueta
0	Go until jurong point, crazy.. Available only ...	0
1	Ok lar... Joking wif u oni...	0
2	Free entry in 2 a wkly comp to win FA Cup fina...	1
3	U dun say so early hor... U c already then say...	0
4	Nah I don't think he goes to usf, he lives aro...	0
5	FreeMsg Hey there darling it's been 3 week's n...	1
6	Even my brother is not like to speak with me. ...	0
7	As per your request 'Melle Melle (Oru Minnamin...	0
8	WINNER!! As a valued network customer you have...	1
9	Had your mobile 11 months or more? U R entitle...	1

Nota: los valores numéricos de la izquierda representan el número de fila, no pertenecen al conjunto inicial de datos.

4.6 Técnicas de análisis de datos relevados

Para efectuar el análisis de los datos que se obtuvieron en la investigación, se necesitaron previamente el uso de librerías de Phyton en la construcción de los modelos y representación de las mediciones. La primera que mencionaremos es Pandas, la cual nos permite trabajar con estructuras de datos en Python, analizar los datos y trabajar con diferentes tipos de datos provenientes de bases de datos, estadísticas, etc. La librería Numpy admite realizar funciones matemáticas y estadísticas de manera eficiente. Esencialmente, introduce los objetos en arreglos multidimensionales, los cuales permiten operaciones por bloques de manera equivalente a operaciones escalares. Sklearn es la librería que permite acceder y utilizar los algoritmos de Machine Learning en Python, a partir de este repositorio se hace referencia a las técnicas utilizadas en la presente investigación. Por otro lado, se utilizó Matploitlib, que facilita generar gráficos, histogramas, espectros de potencia, gráficos de barras. Por último, Seaborn que está basada en matplotlib, suministra una interfaz de alto nivel para gráficos y estadísticas atractivas.

Para medir el rendimiento de cada técnica de machine learning, se recurrió a la métrica accuracy_score. Podemos definirla como el ratio entre las predicciones correctas y las

predicciones totales. Vale aclarar que la suma de verdaderos positivos y verdaderos negativos, son el conjunto de las predicciones correctas. La librería SKlearn o Scikit-Learn implementa la métrica sklearn.metrics.accuracy_score[13] que puede utilizarse en clasificación binomial y multi-clase para devolver el porcentaje de predicciones correctas. Toma los valores en el rango de 0 y 1. El 0 representa el peor clasificador posible, de exactitud 0, es decir que ninguna muestra seria bien clasificada. En cambio, el valor 1 significa un clasificador ideal, de exactitud plena, todas las muestras serian bien clasificada. La exactitud de un modelo de clasificación está definida por la siguiente ecuación de acuerdo a lo expuesto por Sarkar et al. (2018):

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

El significado de las variables es explicado junto al concepto de matriz de confusión mas adelante. Cabe destacar que la exactitud de la clasificación por sí sola puede ser engañosa si tiene un número desigual de observaciones en cada clase o si tiene más de dos clases en su conjunto de datos. En la investigación se utilizó el mismo conjunto de datos para realizar el aprendizaje y evaluación de cada modelo. Respecto a la función, la misma acepta los siguientes parámetros: (y_true, y_pred, *, normalize=True, sample_weight=None). Para el presente trabajo solo se utilizaron los dos primeros, el parámetro y_true recibe del conjunto de prueba las etiquetas correctas. En cuanto a y_pred toma las etiquetas predichas por el algoritmo clasificador.

Para poder obtener un mayor detalle sobre el rendimiento de los algoritmos, se utilizó la técnica de matriz de confusión, utilizando la librería confusion_matrix de sklearn.metrics. Dicha matriz es una de las herramientas más conocidas de evaluar modelos de clasificación. Si bien la matriz en sí misma no es una métrica, su representación puede ser manipulada para obtener mediciones. La matriz de confusión ofrece soluciones tanto para modelos de clasificación binaria como así también de multi-clases. Dicha herramienta permite gráficamente representar los resultados del algoritmo.

Como indica Sarkar et al. (2018) la matriz de confusión presenta la estructura de la Figura 31. En la misma se observan las etiquetas de los posibles escenarios del modelo de predicción en contraposición con los datos reales. Es una técnica para resumir el rendimiento del algoritmo de clasificación.

[13] Web oficial https://scikit-learn.org/stable/modules/generated/sklearn.metrics.accuracy_score.html

Figura 31

Matriz de confusión

	Etiqueta Predicha: NO	Etiqueta Predicha: SI
Etiqueta Real: NO	VN	FP
Etiqueta Real: SI	FN	VP

La matriz comprende las siguientes definiciones, adaptándolas a la presente investigación:

- VP (Verdadero Positivo): El modelo predice un mensaje como spam y realmente es spam según el conjunto de datos.
- FP (Falso Positivo): El modelo predice un mensaje como spam, pero no lo es según el conjunto de datos.
- FN (Falso Negativo): El modelo predice que el mensaje no es spam, pero si lo es según el conjunto de datos.
- VN (Verdadero Negativo): El modelo predice que el mensaje no es spam y realmente no lo es según el conjunto de datos.

Continúa Sarkar et al. (2018) detallando métricas que se derivan de la matriz de confusión. La precesión se define como el número de predicciones hechas que son realmente correctas sobre el total de predicciones positivas.

$$\text{Precisión} = \frac{VP}{VP+FP}$$

Una medida del modelo que identifica el porcentaje de datos relevantes lleva el nombre de Recall. Se define como el número de instancias de la clase positiva que fueron correctamente predichas.

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

En muchos casos lo que se requiere es una optimización balanceada entre la precisión y la sensibilidad, utilizando la formula F1 Score.

$$\text{F1 Score} \frac{2 \text{ X Precisión X Recall}}{\text{Precisión X Recall}}$$

Sklearn provee además un reporte de clasificación con las principales métricas mencionadas, se ejecuta mediante sklearn.metrics.classification_report. El mismo es utilizado en la sección de resultados. Para una mejor comprensión del rendimiento de cada técnica de clasificación, se prefirió recurrir al análisis de la matriz de confusión, ya que permite una clara interpretación del aprendizaje y detección de spam.

4.7 Características del informe final

Como se mencionó en el marco teórico de la presente investigación, existen diversos modelos que pueden ser utilizados para problemas de clasificación, se seleccionaron los siguientes porque demuestran buenos rendimientos en clasificación binaria. Lo cual se ajusta a nuestra investigación de detección binaria de spam o no spam.

- Support Vector
- K-Nearest-Neighbor
- Naive Bayes
- Decision Tree
- Logistic Regression
- Random Forest

Todos ellos fueron ejecutados tomando como datos de entrada el mismo dataset pre procesado. Los resultados luego fueron evaluados utilizando la librería scikit-learn y la matriz de confusión en Phyton.

El informe sigue los pasos diseñados en la modalidad de la investigación. La implementación, no se efectuó por cada modelo en forma individual, fueron puestos a prueba en la misma ejecución.

5. Informe Final

5.1 Propuesta

En el presente capitulo se exhibe la ejecución del algoritmo en lenguaje de programación Phyton y los modelos de machine learning evaluados. Además, se detalla cómo se realizó la implementación y finalizando con los resultados de las predicciones de spam. La investigación se divide en dos etapas, en el procesamiento de los datos y en la construcción del modelo de clasificación de spam. Se ha utilizado el entorno de Google Colab[14] para desarrollar el proyecto.

La propuesta es desarrollar un modelo de machine learning que permita verificar la hipótesis.

5.2 Implementación

a) Procesamiento de los datos

a.1) Obtener el dataset

Para obtener el dataset o conjunto inicial de datos, necesitamos importar las dependencias necesarias para poder trabajar el modelo en Phyton

Figura 32

Importar librerías

```
#importar librerias
%matplotlib inline
import string
import matplotlib.pyplot as plt
import csv
import sklearn
import pickle
from wordcloud import WordCloud
import pandas as pd
import numpy as np
import nltk
from nltk.corpus import stopwords
from sklearn.feature_extraction.text import CountVectorizer, TfidfTransformer
from sklearn.tree import DecisionTreeClassifier
from sklearn.model_selection import GridSearchCV,train_test_split,StratifiedKFold,cross_val_score,learning_curve
```

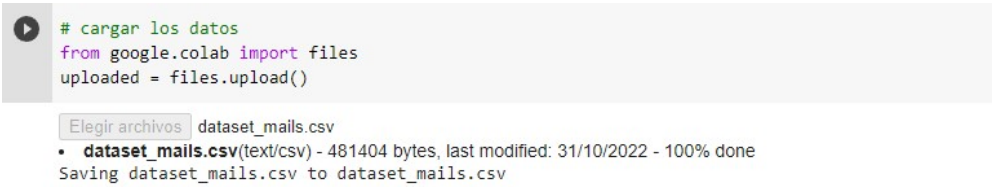
[14] https://colab.research.google.com/

Resaltaremos de la Figura 32, la librería Scikit-learn, también llamada Sklearn, utilizada en machine learning. Provee una selección de herramientas de aprendizaje y modelado estadístico, incluyendo clasificación, regresión, clustering, entre otras. La librería Pandas es usada por científicos de datos para limpieza de datos y análisis. Numpy se puede utilizar para realizar una amplia variedad de operaciones matemáticas en vectores y matrices. NLTK es un conjunto de herramientas creado para trabajar con NLP (Natural Language Processing) en Python. Nos proporciona varias bibliotecas de procesamiento de texto con muchos conjuntos de datos de prueba. Se puede realizar una variedad de tareas utilizando NLTK, como tokenización, ya mencionada en secciones anteriores.

A continuación, en la Figura 33, se muestra la carga del archivo dataset_mails.csv al proyecto.

Figura 33

Carga de dataset



Luego se vierte el conjunto de datos inicial, contenido en el archivo CSV, dentro del objeto data. Para interpretar el archivo se utilizan parámetros como el (;) punto y coma, que declara que la estructura del mismo se separa mediante dicho carácter. Se puede corroborar la correcta lectura de los datos mostrando las primeras 5 filas y sus dos columnas text y etiqueta.

Figura 34

Contenido del dataset

	text	etiqueta
0	Go until jurong point, crazy.. Available only ...	ham
1	Ok lar... Joking wif u oni...	ham
2	Free entry in 2 a wkly comp to win FA Cup fina...	spam
3	U dun say so early hor... U c already then say...	ham
4	Nah I don't think he goes to usf, he lives aro...	ham

a.2) Procesamiento y exploración del Dataset

Se imprime en pantalla la cantidad de filas (5568), es decir mensajes de correo, como así también la cantidad de columnas (2) obtenidas luego de la carga.

Figura 35

Filas y columnas del dataset

```
# imprimir formato, la cantidad de filas (mails) y columnas (variables)
data.shape
```

```
(5568, 2)
```

También se imprimen los nombres de las columnas, esto es necesario luego para poder identificar correctamente los datos que contienen.

Figura 36

Nombre de filas y columnas

```
#obtener los nombres de las columna del dataset
data.columns
```

```
Index(['text', 'etiqueta'], dtype='object')
```

Se verifica la cantidad de mails o mensajes clasificados como spam o ham en el dataset. El resultado es que la muestra contiene 747 mails que son spam y 4821 mails que no son spam.

Figura 37

Cantidad de ham y spam

```
#mostrar total de mails clasificados por spam y ham(se refiere a mail no spam)
data['etiqueta'].value_counts()
```

```
ham     4821
spam     747
Name: etiqueta, dtype: int64
```

Se descarga el paquete PUNKT, dicho tokenizador divide un texto en una lista de oraciones usando un algoritmo no supervisado para construir un modelo para abreviaturas y palabras. Debe entrenarse en una gran colección de texto sin formato en un idioma específico antes de que pueda usarse.

Figura 38

Descarga de paquete punkt

```
nltk.download("punkt")
import warnings
warnings.filterwarnings('ignore')
```

```
[nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data]   Unzipping tokenizers/punkt.zip.
```

a.3) Construir gráfica para identificar palabras frecuentes en spam y ham

Para realizar la gráfica de nube de palabras, se crean y se inicializan dos contenedores para las palabras de cada categoría.

Figura 39

Contenedores de palabras

```
#se crean dos contenedores para las palabras a tokenizar
palabras_ham = ''
palabras_spam = ''
```

Se efectúa el proceso de tokenización en la Figura 40. Según sea la etiqueta del conjunto de datos, se divide el mensaje en palabras y se agrega a su correspondiente contenedor.

Figura 40

Tokenization

```
#Tokenization (lista de tokens), que será usado por el analizador
# se llena de los mensajes catalogados como spam
for val in data[data['etiqueta'] == 'spam'].text:
    text = val.lower()
    tokens = nltk.word_tokenize(text)
    for palabras in tokens:
        palabras_spam = palabras_spam + palabras + ' '

# se llena de los mensajes catalogados como ham
for val in data[data['etiqueta'] == 'ham'].text:
    text = text.lower()
    tokens = nltk.word_tokenize(text)
    for palabras in tokens:
        palabras_ham = palabras_ham + palabras + ' '
```

Se construye el gráfico de nube de palabras, que permite observar cuales son las más frecuentes en los mensajes de spam y ham. Se especifica el ancho y altura del mismo.

Figura 41

Wordcloud

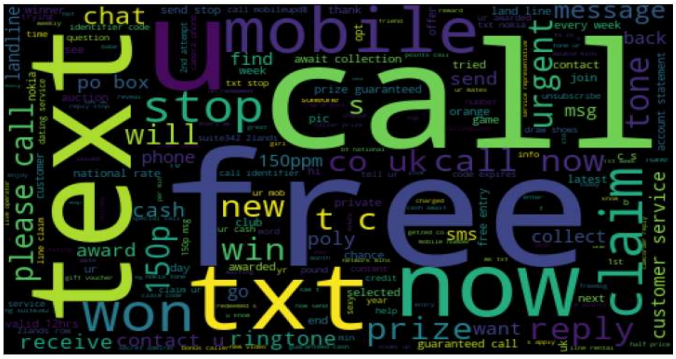
```
#Se crea con la herramienta worldcloud una nube de las palabras
spam_wordcloud = WordCloud(width=500, height=300).generate(palabras_spam)
ham_wordcloud = WordCloud(width=500, height=300).generate(palabras_ham)
```

Resultado de las palabras analizadas en los mensajes de spam

Figura 42

Wordcloud spam

```
#mostrar Spam Wordcloud
plt.figure( figsize=(10,8), facecolor='w')
plt.imshow(spam_wordcloud)
plt.axis("off")
plt.tight_layout(pad=0)
plt.show()
```



Resultado de las palabras analizadas en los mensajes de ham

Figura 43

Wordcloud ham

```
#Mostrar Ham wordcloud
plt.figure( figsize=(10,8), facecolor='g')
plt.imshow(ham_wordcloud)
plt.axis("off")
plt.tight_layout(pad=0)
plt.show()
```

Para la correcta interpretación de los datos, al igual que en la tokenización, es necesario reemplazar a números los datos de la columna etiqueta. Para realizarlo se reemplazó la palabra ham y spam por un 0 y 1 respectivamente.

Figura 44

Columna etiqueta

```
#Modificar o reemplaza la palabra ham y spam por 0 y 1 respectivamente
data = data.replace(['ham','spam'],[0, 1])
data.head(10)
```

	text	etiqueta
0	Go until jurong point, crazy.. Available only ...	0
1	Ok lar... Joking wif u oni...	0
2	Free entry in 2 a wkly comp to win FA Cup fina...	1
3	U dun say so early hor... U c already then say...	0
4	Nah I don't think he goes to usf, he lives aro...	0
5	FreeMsg Hey there darling it's been 3 week's n...	1
6	Even my brother is not like to speak with me. ...	0
7	As per your request 'Melle Melle (Oru Minnamin...	0
8	WINNER!! As a valued network customer you have...	1
9	Had your mobile 11 months or more? U R entitle...	1

a.4) Remover palabras vacías y signos de puntuación

Las palabras vacías y los signos de puntuación no contribuyen al modelo de clasificación, por lo cual deben ser removidas de los mensajes. Usando la librería NLTK realizamos dicho proceso. Primero debemos descargar el paquete de palabras vacías stopwords.

Figura 45

Stopwords

```
#descargar el paquete de palabras vacias, es el nombre que reciben las palabras sin significado como artículos,
#pronombres, preposiciones, etc. que son filtradas antes o después del procesamiento de datos en lenguaje natural.
nltk.download('stopwords')
```

```
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Unzipping corpora/stopwords.zip.
True
```

Se efectúa la limpieza de los mensajes sobre las palabras vacías y los signos de puntuación. Para realizar dicho proceso se recurre a los métodos string.punctuation y stopwords.words('english'). Se debe definir el idioma para el correcto funcionamiento del método.

Figura 46

Texto procesado

```
#funcion para limpiar el texto y devolver tokens
#Puntuacion son [!"#$%&'()*+,-./:;<=>?@[\]^_`{|}~]
#Palabras vacias en el procesamiento del lenguaje natural, palabras sin importancia o significado.
import string
def texto_procesado(text):

    text = text.translate(str.maketrans('', '', string.punctuation))
    text = [word for word in text.split() if word.lower() not in stopwords.words('english')]

    return " ".join(text)
```

Se imprime el resultado del proceso anterior, en la Figura 47, mostrando las primeras 5 filas de mensajes. Si se compara con la Figura 34, se advierte que los signos de puntuación y preposiciones fueron removidos. Se aclara que los puntos suspensivos, a la derecha del texto, corresponden a que el mensaje no cabe en su totalidad dentro del ancho de la columna. Esto ocurre solo en la salida en pantalla, el mensaje se encuentra completo en el dataset.

Figura 47

Resultado del texto procesado

```
data = data.astype(str)
data['text'] = data['text'].apply(texto_procesado)
data.head()
```

	text	etiqueta
0	Go jurong point crazy Available bugis n great ...	0
1	Ok lar Joking wif u oni	0
2	Free entry 2 wkly comp win FA Cup final tkts 2...	1
3	U dun say early hor U c already say	0
4	Nah dont think goes usf lives around though	0

El conjunto de datos ya se encuentra preparado, dividimos los datos y creamos el data frame.

Figura 48

Dataframes con los datos procesados

```
#crear dataframe de los datos procesados
text = pd.DataFrame(data['text'])
etiqueta = pd.DataFrame(data['etiqueta'])
```

a.5) Convertir texto en vectores

Necesitamos convertir palabras en vectores. Los vectores son espacios de almacenamiento contiguos que contienen elementos del mismo tipo. La conversión se puede realizar mediante la técnica Count Vectorizer o usando TF-IDF Vectorizer. La segunda opción es mejor porque no solo se concentra en la frecuencia de las palabras, sino que también proporciona la importancia de las palabras. Nos permite remover las palabras que son menos importantes para el análisis. Por lo tanto, hace que la construcción del modelo sea menos compleja al reducir las dimensiones de entrada. En la implementación se incluyen ambos métodos. Los mismos se pueden consultar en la documentación oficial de la librería Scikit-Learn[15]

Se utiliza a continuación Count Vectorizer para convertir palabras en vectores. Primero contamos la cantidad de palabras que contiene nuestro dataset. Como se puede observar en la Figura 49, el valor es de 11.412 palabras.

Figura 49

Count Vectorizer

```
#Contar cuantas veces una palabra aparece en el dataset dentro de la columna texto

from collections import Counter

total_counts = Counter()
for i in range(len(text)):
    for palabra in text.values[i][0].split(" "):
        total_counts[palabra] += 1

print("Total de palabras en el dataset: ", len(total_counts))
```

```
Total de palabras en el dataset:  11412
```

Se imprimen las primeras 15 palabras ordenadas en forma decreciente según la frecuencia. Recurrimos al método sorted para dicho ordenamiento.

Figura 50

Palabras ordenadas

```
# Ordenar en forma decreciente (palabras con la mas alta frecuencia aparecen primero) se muestran las primeras 15
vocab = sorted(total_counts, key=total_counts.get, reverse=True)
print(vocab[:15])
```

```
['u', '2', 'call', 'U', 'get', 'Im', 'ur', '4', 'know', 'go', 'like', 'dont', 'come', 'got', 'time']
```

[15] https://scikit-learn.org/stable/

Se procede a realizar el mapeo de palabras a un índice. El mapeo es el proceso de especificar cómo un conjunto de información se relaciona con otro, en este caso el objetivo es relacionar las palabras con el vector.

Figura 51

Mapeo de palabras a índice

```
#Se mapean las palabras a un indice
vocab_size = len(vocab)
word2idx = {}
#print vocab_size
for i, palabra in enumerate(vocab):
    word2idx[palabra] = i
```

A continuación, se construye el vector utilizando el dataframe text, previamente cargado con los mensajes de la columna text del conjunto de datos. Al pie de la Figura 52 se muestra en pantalla el resultado, revelando que la cantidad de palabras no se ha modificado (11.412). Las palabras solo han sido medidas por su frecuencia en los 5568 mensajes.

Figura 52

Palabras del dataset

```
[22] # Se vuelca texto al vector
     def texto_a_vector(text):
         vector_palabra = np.zeros(vocab_size)
         for palabra in text.split(" "):
             if word2idx.get(palabra) is None:
                 continue
             else:
                 vector_palabra[word2idx.get(palabra)] += 1
         return np.array(vector_palabra)
     vector = np.zeros((len(text), len(vocab)), dtype=np.int_)
     for i, (_, text_) in enumerate(text.iterrows()):
         vector[i] = texto_a_vector(text_[0])
```

```
vector.shape
```

```
(5568, 11412)
```

En la Figura 53 se puede observar como la cantidad de palabras se ha reducido, al utilizar TF-IDF Vectorizer. El método al tokenizar los mensajes, transforma el texto en vectores de características que se pueden usar como entrada para el clasificador mejorando el rendimiento. Se diferencia respecto a CountVectorizer radica en que éste pondera aquellas palabras que aparecen mucho en un mensaje, pero no en muchos mensajes. Esto permite que una palabra como por ejemplo saludos, que aparece en prácticamente todos los mensajes de correo, no sea

tenida en cuenta en el caso de TFidVectorizer. Este último método conjetura que la mayor parte de los mensajes tendrán la palabra saludos y que no tiene ninguna relevancia en la clasificación de los mensajes. Por este motivo, el número de palabras disminuyó a 9.449. El método puede fallar si no se utiliza stop words, debido a que puede ponderar artículos o preposiciones lo cual fue descartado en pasos previos.

Figura 53

Palabras del dataset luego de TFidVectorizer

```
#convertir los datos de texto en el vectors
from sklearn.feature_extraction.text import TfidfVectorizer

vectorizer = TfidfVectorizer()
vectors = vectorizer.fit_transform(data['text'])
vectors.shape
```

```
(5568, 9449)
```

b) Construcción del modelo de clasificación de spam

b.1) Separar los datos en los sets de entrenamiento y prueba del modelo

Se utiliza el método train-test split para poder entrenar a nuestro detector de spam de correo electrónico. El objetivo es que reconozca y clasifique los correos electrónicos no deseados. El método es una técnica de división que se utiliza para evaluar el rendimiento de un algoritmo de aprendizaje automático. Podemos usarlo para clasificación o regresión de cualquier algoritmo de aprendizaje supervisado.

El procedimiento implica tomar un conjunto de datos y dividirlo en dos conjuntos de datos separados. El primer conjunto de datos se utiliza para ajustar el modelo y se denomina conjunto de datos de entrenamiento (train). El segundo conjunto de datos es el de prueba (test), donde proporcionamos los elementos para evaluar el modelo. Finalmente, se realizan las predicciones, comparándolas con el resultado real del conjunto de prueba.

En la Figura 54 se puede observar que datos_del_modelo contiene los mensajes previamente trasformados. Para indicarle al modelo las respuestas correctas, se asigna la clasificación correspondiente de cada mensaje (spam, ham) contenida en data[‘etiqueta’]. Entonces la función X_train, X_test, y_train, y_test = train_test_split(datos_del_modelo, data['etiqueta'], test_size=0.20, random_state=100) divide datos_del_modelo y data[‘etiqueta’] en mensajes de entrenamiento asignados a X_train, a su vez asignando las etiquetas de

entrenamiento a y_train. Luego en X_test e y_test los mensajes y etiquetas destinados a la prueba del modelo respectivamente. El parámetro test_size=0.20 establece que el 20% de los datos de datos_del_modelo y data['etiqueta'] serán reservados al conjunto de prueba, dejando así un 80% de los datos para el aprendizaje del modelo. En cuanto a random_state=100, establecemos un valor fijo para que en cada ejecución se obtenga la misma división del conjunto de datos, de esta manera no obtener variaciones en el resultado final del entrenamiento.

Figura 54

Train-test split

```
[31] #Dividir el dataset en set de entrenamiento y prueba
     #80% entrenamiento (training) y 20% prueba (test)
     X_train, X_test, y_train, y_test = train_test_split(datos_del_modelo, data['etiqueta'], test_size=0.20, random_state=100)
```

b.2) Utilizar clasificador Sklearn para construir el modelo

Para construir el modelo se utilizan los clasificadores que se encuentran en la librería sklearn. La misma contiene los algoritmos que se importarán para luego poder comparar sus rendimientos. En la Figura 55 se encuentran los siguientes: LogisticRegression, SVC (Support Vector), MultinomialNB (Naive Bayes), DecisionTreeClassifier (Decision Tree), KNeighborsClassifier (K-Nearest Neighbors) y RandomForestClassifier (Random Forest). La librería accuracy_score se utiliza para devolver el valor de exactitud del modelo, toma valor 1 cuando no hay error y 0 en el peor de los casos.

Figura 55

Sklearn

```
[32] #importar los paquetes sklearn para construir los clasificadores
     from sklearn.linear_model import LogisticRegression
     from sklearn.svm import SVC
     from sklearn.naive_bayes import MultinomialNB
     from sklearn.tree import DecisionTreeClassifier
     from sklearn.neighbors import KNeighborsClassifier
     from sklearn.ensemble import RandomForestClassifier
     from sklearn.metrics import accuracy_score
```

b.2) Entrenar el modelo

Se inicializan los clasificadores, a continuación, se comenta sus detalles.

En el clasificador Support Vector (SVC) fue seleccionada la función de kernel sigmoid[16], la cual está definida con la ecuación del perceptrón de dos capas, como fue explicado en el marco teórico. Para el mencionado kernel escogido, el valor de gamma[17] 0.8 fue el que mejores resultados permitió registrar. El parámetro permite definir el ancho y pendiente de la función. Para el clasificador K-Nearest Neighbors se configuró la cantidad de vecinos en 65, presentando sus mejores resultados para el dataset utilizado. En Naive Bayes (MultinomialNB) se definió el valor 0.2 para el parámetro alpha. El cual permite mediante la técnica de suavizado de Laplace abordar el problema de la probabilidad cero en el algoritmo. Este valor fue el que mejores resultados proporcionó. En el caso de Decision Tree, la cantidad mínima para un nodo interno del árbol pueda separarse (min_samples_split) es de 7. Las pruebas arrojaron mejores resultados junto con el valor elegido de 42 en random_state. En el clasificador Logistic Regression se configuró el parámetro solver con liblinear[18], ya que es recomendado para datasets pequeños y penalidad l1 dando los mejores resultados. Por último, en Random Forest se configuró mediante n_estimators[19], el número de 65 árboles por bosque. El código se muestra en la Figura 56.

Figura 56

Clasificadores

```
[75] #inicializar los modelos clasificadores
     svc = SVC(kernel='sigmoid', gamma=0.8)
     knc = KNeighborsClassifier(n_neighbors=65)
     mnb = MultinomialNB(alpha=0.2)
     dtc = DecisionTreeClassifier(min_samples_split=7, random_state=42)
     lrc = LogisticRegression(solver='liblinear', penalty='l1')
     rfc = RandomForestClassifier(n_estimators=65, random_state=100)
```

Se realiza un ciclo for para que, en una misma ejecución, se pueda entrenar y obtener el valor de exactitud de cada algoritmo clasificador.

[16] https://scikit-learn.org/stable/modules/svm.html#kernel-functions
[17] https://www.researchgate.net/publication/344458945_The_effect_of_gamma_value_on_support_vector_machine_performance_with_different_kernels
[18] https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html
[19] https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html

Figura 57

Entrenar los clasificadores

```
[34] #crear un diccionario de variables y modelos
     clfs = {'Support Vector' : svc,'K-NN' : knc, 'Naive Bayes': mnb, 'DecisionTree': dtc, 'LogisticRegression': lrc, 'RandomForest': rfc}
```

```
[35] #adaptar los datos a los modelos
     def train(clf, datos_del_modelo, targets):
         clf.fit(datos_del_modelo, targets)

     def predict(clf, datos_del_modelo):
         return (clf.predict(datos_del_modelo))

     puntaje_vector_palabras_predichas = []
     for k,v in clfs.items():
         train(v, X_train, y_train)
         pred = predict(v, X_test)
         puntaje_vector_palabras_predichas.append((k, [accuracy_score(y_test , pred)]))
```

5.3 Resultados Obtenidos

Los resultados de exactitud de todas las técnicas de machine learning puestas a prueba, se muestran en la Figura 58.

Figura 58

Resultados de exactitud

```
[('Support Vector', [0.9766606822262118]),
 ('K-NN', [0.9425493716337523]),
 ('Naive Bayes', [0.9820466786355476]),
 ('DecisionTree', [0.9676840215439856]),
 ('LogisticRegression', [0.9587073608617595]),
 ('RandomForest', [0.9757630161579892])]
```

A continuación, exponemos los reportes de clasificación y matriz de confusión, separados por cada técnica en particular. Se aclara que, para interpretar los resultados de las métricas, se debe tener en cuenta lo explicado en Metodología y lo siguiente:

- Reporte de clasificación: el valor 0 corresponde a la etiqueta ham (no spam) y el valor 1 a spam.
- Matriz de confusión: los valores 0 y 1 pertenecen a la clasificación binaria de no y si respectivamente. El eje vertical corresponde a las etiquetas reales del dataset y el eje horizontal las etiquetas predichas.

Tabla 1

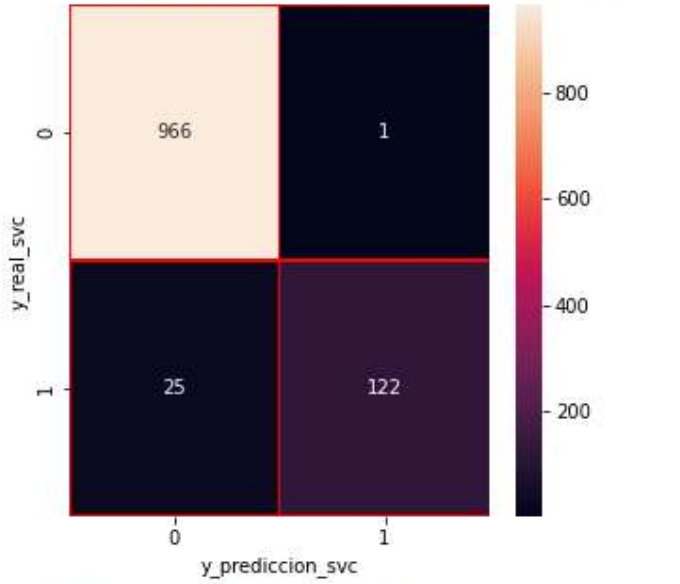
Support Vector

Reporte de clasificación:

```
Support Vector
              precision    recall  f1-score   support

           0       0.97      1.00      0.99       967
           1       0.99      0.83      0.90       147

    accuracy                           0.98      1114
   macro avg       0.98      0.91      0.95      1114
weighted avg       0.98      0.98      0.98      1114


Exactitud:  0.9766606822262118
```

Matriz de confusión:

Support Vector
Cantidad de mensajes en set de prueba: (1114,)



Exactitud: 0.9766606822262118

Tabla 2

K-Nearest-Neighbor

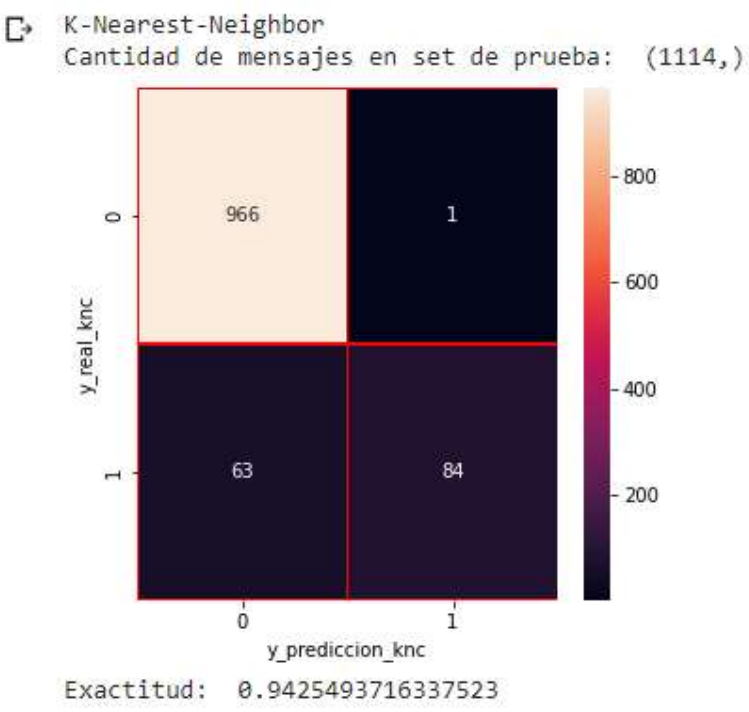
Reporte de clasificación:

K-Nearest-Neighbor

	precision	recall	f1-score	support
0	0.94	1.00	0.97	967
1	0.99	0.57	0.72	147
accuracy			0.94	1114
macro avg	0.96	0.79	0.85	1114
weighted avg	0.95	0.94	0.94	1114

Exactitud: 0.9425493716337523

Matriz de confusión:



Exactitud: 0.9425493716337523

Tabla 3

Naive Bayes

Reporte de clasificación:

```
Naive Bayes
              precision    recall  f1-score   support

           0       0.98      0.99      0.99       967
           1       0.96      0.90      0.93       147

    accuracy                           0.98      1114
   macro avg       0.97      0.95      0.96      1114
weighted avg       0.98      0.98      0.98      1114


Exactitud:  0.9820466786355476
```

Matriz de confusión:

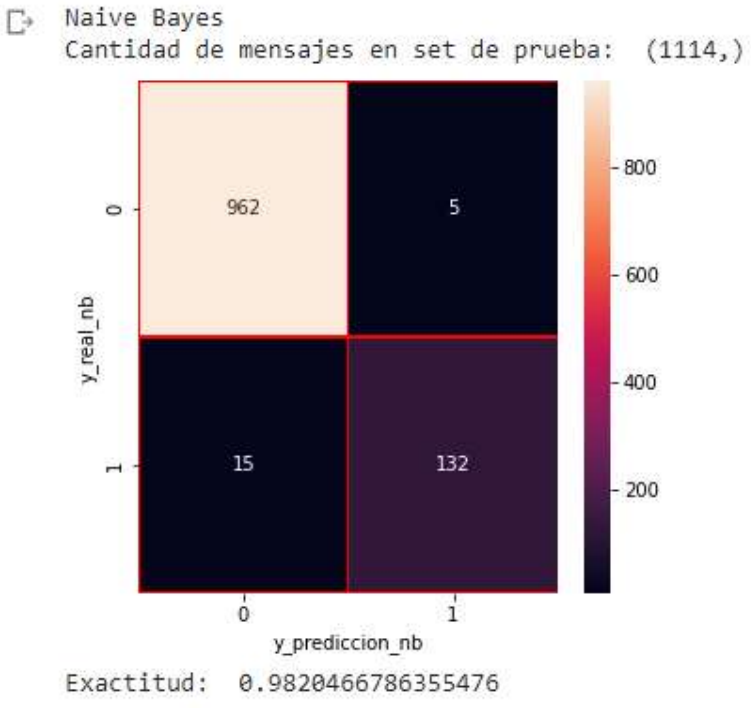


Tabla 4

Decision Tree

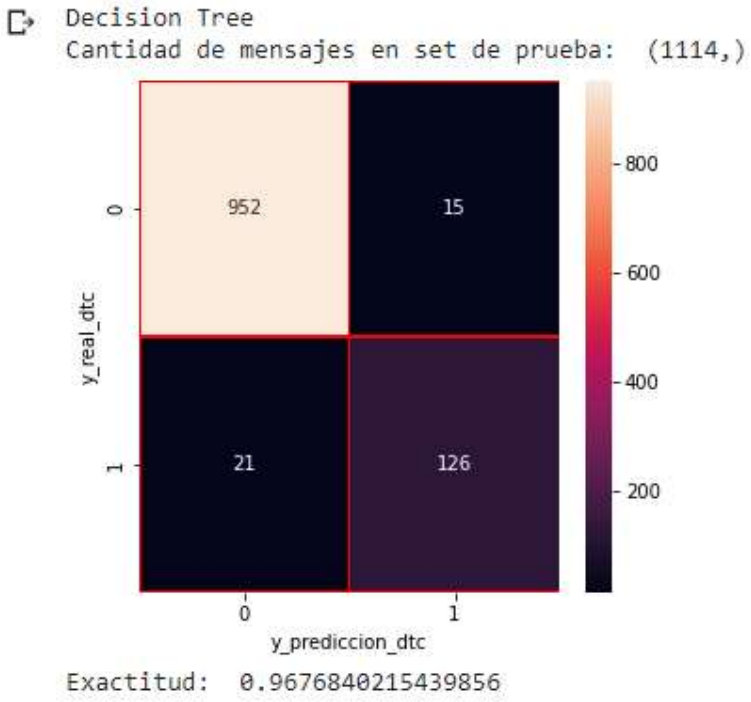
Reporte de clasificación:

Decision Tree

	precision	recall	f1-score	support
0	0.98	0.98	0.98	967
1	0.89	0.86	0.88	147
accuracy			0.97	1114
macro avg	0.94	0.92	0.93	1114
weighted avg	0.97	0.97	0.97	1114

Exactitud: 0.9676840215439856

Matriz de confusión:



Exactitud: 0.9676840215439856

Tabla 5

Logistic Regression

Reporte de clasificación:

```
Logistic Regression
              precision    recall  f1-score   support

           0       0.96      0.99      0.98       967
           1       0.95      0.72      0.82       147

    accuracy                           0.96      1114
   macro avg       0.96      0.86      0.90      1114
weighted avg       0.96      0.96      0.96      1114


Exactitud:  0.9587073608617595
```

Matriz de confusión:

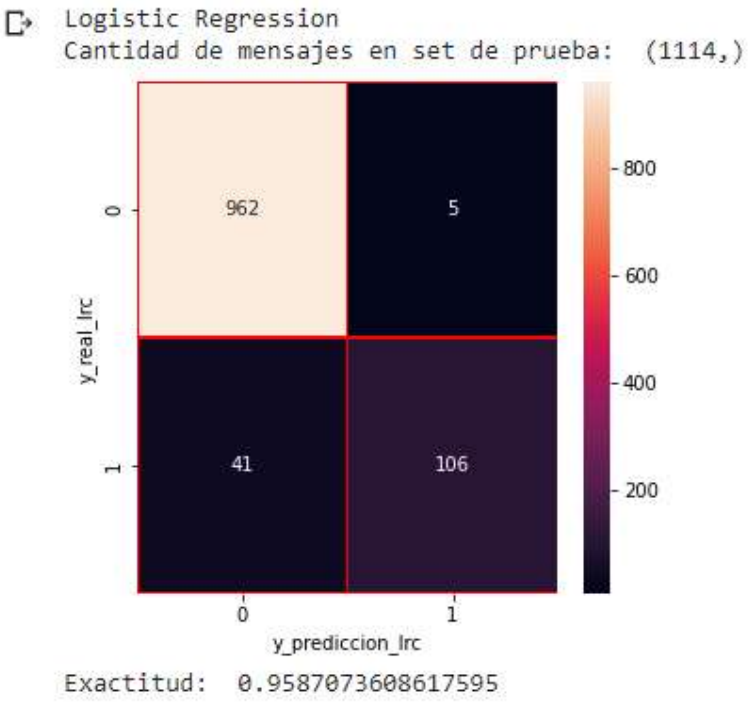


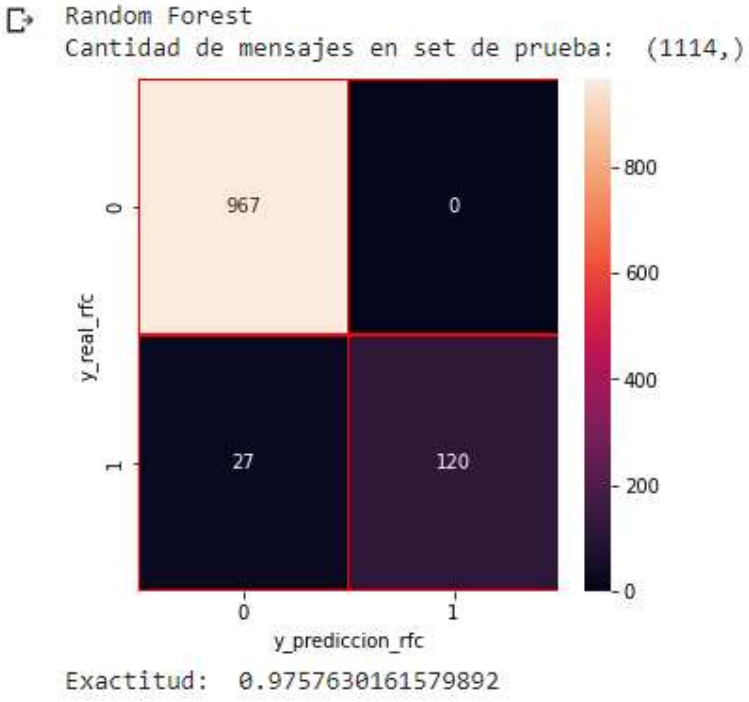
Tabla 6

Random Forest

Reporte de clasificación:

```
Random Forest
              precision    recall  f1-score   support

           0       0.97      1.00      0.99       967
           1       1.00      0.82      0.90       147

    accuracy                           0.98      1114
   macro avg       0.99      0.91      0.94      1114
weighted avg       0.98      0.98      0.97      1114


Exactitud:  0.9757630161579892
```

Matriz de confusión:



A continuación, en la Tabla 7, se detallan los porcentajes de spam detectados según las técnicas de machine learning. Para realizarla, se calculó el porcentaje tomando en cuenta la cantidad de spam en el set de prueba (147) y los verdaderos positivos de la matriz de confusión de cada clasificador.

Tabla 7

Porcentajes de spam detectados

Clasificador	Spam detectado
Support Vector	82,99%
K-NN	57,14%
Naive Bayes	89,80%
Decision Tree	85,71%
Logistic Regression	72,11%
Random Forest	81,63%

Se advierte que el mayor porcentaje lo registra Naive Bayes, el resto de las técnicas se encuentran cercanas al mismo, salvo Logistic Regression y K-Nearest-Neighbor que quedan relegadas.

Utilizando los resultados del informe, sobre el nivel de exactitud de cada técnica de la Figura 58, se elabora la siguiente comparativa para expresar la distancia o diferencia entre cada una. La misma se puede consultar en la Tabla 8.

Tabla 8

Comparativa de exactitudes

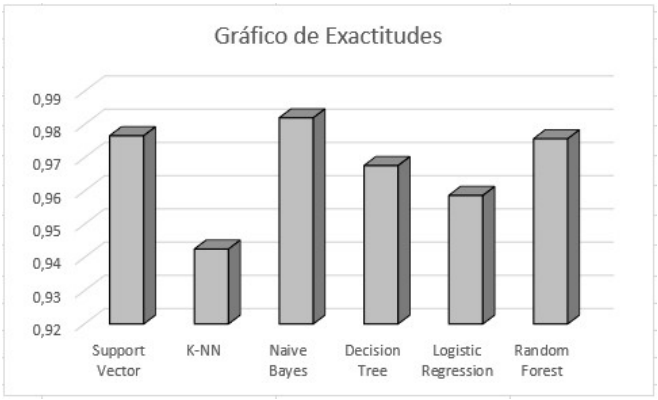
	Support Vector	K-NN	Naive Bayes	Decision Tree	Logistic Regression	Random Forest
Support Vector	0,976660682226211	-0,034111310592459	0,005385996409336	-0,008976660682226	-0,017953321364452	-0,000897666068222
K-NN	0,034111310592459	0,942549371633752	0,039497307001795	0,025134649910233	0,016157989228007	0,033213644524237
Naive Bayes	-0,005385996409336	-0,039497307001795	0,982046678635547	-0,014362657091562	-0,023339317773788	-0,006283662477558
Decision Tree	0,008976660682226	-0,025134649910233	0,014362657091562	0,967684021543985	-0,008976660682226	0,008078994614004
Logistic Regression	0,017953321364452	-0,016157989228007	0,023339317773788	0,008976660682226	0,958707360861759	0,017055655296230
Random Forest	0,000897666068222	-0,033213644524237	0,006283662477558	-0,008078994614004	-0,017055655296230	0,975763016157989

La matriz simétrica contiene en su diagonal los valores de exactitud correspondientes. La diferencia de dichos valores respecto a cada técnica completa la matriz. Como se puede observar, el algoritmo de clasificación con el mejor rendimiento respecto del peor, tiene una diferencia en módulo de 0,039497307001795, correspondiente a la comparación entre K-Nearest-Neighbor y Naive Bayes.

A continuación, en la Figura 59, se representan con un gráfico de barras los valores de exactitud de cada algoritmo.

Figura 59

Gráfico de barras de los clasificadores



6. Conclusiones

6.1 Conclusiones

En base a los resultados, señalamos que dentro de las técnicas de clasificación de machine learning utilizadas para la detección de spam, Naive Bayes fue la que obtuvo el mejor rendimiento. Por contraparte, K-Nearest-Neighbor se ubicó en último lugar. Aunque, el margen de exactitud no es significativo entre ambas. Si analizamos detenidamente K-NN cuenta con 63 falsos negativos en la predicción de etiquetas. Siendo éste el número más alto comparado con el resto de las técnicas. El mayor número de verdaderos positivos lo registra Naive Bayes con 132. Es decir, que es el modelo que mejor predice un mensaje como spam. Aunque Random Fores, K-NN y Support Vector son levemente mejores en predecir verdaderos negativos respecto a Naive Bayes. Dicho caso es cuando el modelo predice correctamente que el mensaje no es spam. Decision Tree cuenta la mayor cantidad de falsos positivos, con un total de 15. Se destaca en este aspecto la técnica de Random Forest, la cual no registra ninguno.

Estamos en condiciones de afirmar que la hipótesis de investigación se cumple. No se evidencian diferencias significativas en el nivel de exactitud de las técnicas. Los valores de efectividad superan el 90%. Sin embargo, destacamos que existen diferencias en los porcentajes de verdaderos positivos. Se entiende que la exactitud de los modelos clasificadores, también contempla las predicciones correctas de mensajes ham, es decir verdaderos negativos. En este

aspecto, todos lograron una métrica aceptable. Aducimos el dispar nivel de porcentaje de detección de spam, a la cantidad de etiquetas spam en el conjunto inicial de datos. Recordemos que, durante la implementación, se dividió el dataset en un conjunto de entrenamiento y de prueba. Como bien detalla el reporte de clasificación correspondiente a cada técnica, el resultado de dicha segmentación destinó dentro del dataset de prueba a 967 mensajes con la etiqueta ham y 147 con la etiqueta spam.

Teniendo en cuenta lo analizado anteriormente, podemos concluir que no hay una técnica que se destaque completamente del resto en cuanto al nivel de exactitud. Aunque, claro está, en grandes volúmenes de mensajes estas pequeñas diferencias saldrían a la luz.

6.2 Futuras líneas de investigación

Algunas propuestas que podrían ser tenidas en cuenta, en una investigación futura, son las de utilizar un dataset con un mayor número de mensajes, que contenga un porcentaje similar de etiquetas spam y ham en la muestra. Sería interesante verificar con dicho dataset, los valores que se obtienen en el nivel de porcentaje de detección de spam. También se podría aumentar el detalle del análisis extendiendo las características para entrenar el modelo. Es decir, incluir columnas como dirección IP o dirección de correo del remitente. Por último, entendemos necesario poner a prueba en la implementación una red neuronal y combinar técnicas de clasificación para evaluar si las mismas mejoran los niveles de exactitud.

7. Referencias

Abayomi-Alli, O., Misra, S., Abayomi-Alli, A. (2022). A deep learning method for automatic SMS spam classification: Performance of learning algorithms on indigenous dataset. Ostfold University College, Halden, Noruega.

Airan, N. & Lal Bhari, P. (2002). Email Spam Classification Using Machine Learning. Poornima Institute Of Engineering And Technology, Jaipur, Rajasthan.

Chaudhari, D., Kolambe, D., Patil, R. y Puranik, S. (2022). Email Spam Detection Using Machine Learning and Python. SSBT's College of Engineering and Technology, Bambhori, Jalgaon, India.

Costales, B. & Flynt, M. (2005). Sendmail Filters: a guide for fighting spam. Pearson Education, Inc. ISBN 0-321-21333-5.

Elisan, C.C. (2015). Advanced Malware Analysis. Mc Graw Hill.

Erickson, J. (2008). Hacking: the art of exploitation. No Strach Press

Freed, N. & Borenstein, N. (1996). RFC 2045. Multipurpose Internet Mail Extensions. (MIME) Part One: Format of Internet Message Bodies. https://www.rfc-editor.org/rfc/rfc2045.txt

Eset (2022). Security Report Latinoamérica 2022. https://www.welivesecurity.com/wp-content/uploads/2022/07/ESET-security-report-LATAM-2022.pdf

Internet Society (2014). https://www.internetsociety.org/wp-content/uploads/2017/08/History20of20Spam.pdf

Klensin, J. (2001). RFC 2821. Simple Mail Transfer Protocol. https://www.rfc-editor.org/rfc/rfc2821.txt

Klensin, J. (2008). RFC 5321. Simple Mail Transfer Protocol. https://www.rfc-editor.org/rfc/rfc5321

McAfee (2019). 9 tipos de hackers y sus motivaciones. https://www.mcafee.com/blogs/es-es/family-safety/9-tipos-de-hackers-y-sus-motivaciones/

Mitchell, T. (1997). Machine Learning. McGraw-Hill. ISBN 0070428077

Portal del Estado Argentino. (s.f.). Delitos Informáticos. https://www.argentina.gob.ar/justicia/derechofacil/leysimple/delitos-informaticos#titulo-1

Real Academia Española. Correo. En Diccionario de la lengua española. Recuperado en 16 de diciembre de 2022, de https://dle.rae.es/correo?m=form#G4FTxy7

Reddy, K. & Kakulapati, V. (2021). Classification of Spam Messages using Random Forest Algorithm. Sreenidhi Institute of Science & Technology, Hyderabad, Telangana, India.

Resnick, P. (2001). RFC 2822. Internet Message Format. https://www.rfc-editor.org/rfc/rfc2822.txt

Russell, S. & Norvig, P. (2004). Inteligencia Artificial, un enfoque moderno. Pearson

Sarkar, D., Bali, R. y Sharma, T. (2018). Practical Machine Learning with Python: A Problem-Solver's Guide to Building Real-World Intelligent Systems. Apress.

Sikorski, M & Honig, A. (2012). Practical Malware Analysis. No Starch Press.

Stamp, M. (2018). Introduction to Machine Learning with Applications in Information Security. CRC Press.

Sweigart, A. (2013). Hacking Secret Ciphers with Python. ISBN 978-1482614374

Symantec. (2017). https://docs.broadcom.com/doc/istr-email-threats-2017-en

Tan, Y. (2016). Anti-Spam Techniques Based on Artificial Immune System. CRC Press.

Temperini, M. G. I. (2014). Delitos Informáticos en Latinoamérica: Un estudio de derecho comparado. 2da. Parte. 14º Simposio Argentino de Informática y Derecho. http://sedici.unlp.edu.ar/handle/10915/42145

Varun Kumar, K., Ramamoorthy, M. (2022). Naive Bayes Classifier Algorithm for Spam Detection of Email to Improve Accuracy and in Comparison with Decision Tree Algorithm. Saveetha University, Chennai, TamilNadu, India.

9. Anexos

Anexo A: Código en lenguaje Phyton de evaluación del proyecto

```
### Support Vector ###
#Evaluar el modelo prueba (test) del dataset
#ham = 0
#spam = 1
from sklearn.metrics import classification_report,confusion_matrix, accuracy_score
pred = svc.predict(X_test)
print('Support Vector')
print(classification_report(y_test ,pred ))
print()
print('Exactitud: ', accuracy_score(y_test,pred))


from sklearn.metrics import confusion_matrix
import seaborn as sns
### Support Vector ###
y_prediccion_svc = svc.predict(X_test)
y_real_svc = y_test
cm = confusion_matrix(y_real_svc, y_prediccion_svc)
f, ax = plt.subplots(figsize =(5,5))
sns.heatmap(cm,annot = True,linewidths=0.5,linecolor="red",fmt = ".0f",ax=ax)
plt.xlabel("y_prediccion_svc")
plt.ylabel("y_real_svc")
print('Support Vector')
print('Cantidad de mensajes en set de prueba: ',y_test.shape)
plt.show()
print('Exactitud: ', accuracy_score(y_test,y_prediccion_svc))


### K-Nearest-Neighbor ###
#Evaluar el modelo prueba (test) del dataset
#ham = 0
#spam = 1
from sklearn.metrics import classification_report,confusion_matrix, accuracy_score
pred = knc.predict(X_test)
print('K-Nearest-Neighbor')
print(classification_report(y_test ,pred ))
print()
print('Exactitud: ', accuracy_score(y_test,pred))


from sklearn.metrics import confusion_matrix
import seaborn as sns
### K-Nearest-Neighbor ###
```

```
y_prediccion_knc = knc.predict(X_test)
y_real_knc = y_test
cm = confusion_matrix(y_real_knc, y_prediccion_knc)
f, ax = plt.subplots(figsize =(5,5))
sns.heatmap(cm,annot = True,linewidths=0.5,linecolor="red",fmt = ".0f",ax=ax)
plt.xlabel("y_prediccion_knc")
plt.ylabel("y_real_knc")
print('K-Nearest-Neighbor')
print('Cantidad de mensajes en set de prueba: ',y_test.shape)
plt.show()
print('Exactitud: ', accuracy_score(y_test,y_prediccion_knc))


### Naive Bayes ###
#Evaluar el modelo prueba (test) del dataset
#ham = 0
#spam = 1
from sklearn.metrics import classification_report,confusion_matrix, accuracy_score
pred = mnb.predict(X_test)
print('Naive Bayes')
print(classification_report(y_test ,pred ))
print()
print('Exactitud: ', accuracy_score(y_test,pred))


from sklearn.metrics import confusion_matrix
import seaborn as sns
### Naive Bayes ###
y_prediccion_nb = mnb.predict(X_test)
y_real_nb = y_test
cm = confusion_matrix(y_real_nb, y_prediccion_nb)
f, ax = plt.subplots(figsize =(5,5))
sns.heatmap(cm,annot = True,linewidths=0.5,linecolor="red",fmt = ".0f",ax=ax)
plt.xlabel("y_prediccion_nb")
plt.ylabel("y_real_nb")
print('Naive Bayes')
print('Cantidad de mensajes en set de prueba: ',y_test.shape)
plt.show()
print('Exactitud: ', accuracy_score(y_test,y_prediccion_nb))


### Decision Tree ###
#Evaluar el modelo prueba (test) del dataset
#ham = 0
#spam = 1
from sklearn.metrics import classification_report,confusion_matrix, accuracy_score
pred = dtc.predict(X_test)
print('Decision Tree')
print(classification_report(y_test ,pred ))
print()
```

```
print('Exactitud: ', accuracy_score(y_test,pred))


from sklearn.metrics import confusion_matrix
import seaborn as sns
### Decision Tree ###
y_prediccion_dtc = dtc.predict(X_test)
y_real_dtc = y_test
cm = confusion_matrix(y_real_dtc, y_prediccion_dtc)
f, ax = plt.subplots(figsize =(5,5))
sns.heatmap(cm,annot = True,linewidths=0.5,linecolor="red",fmt = ".0f",ax=ax)
plt.xlabel("y_prediccion_dtc")
plt.ylabel("y_real_dtc")
print('Decision Tree')
print('Cantidad de mensajes en set de prueba: ',y_test.shape)
plt.show()
print('Exactitud: ', accuracy_score(y_test,y_prediccion_dtc))


### Logistic Regression ###
#Evaluar el modelo prueba (test) del dataset
#ham = 0
#spam = 1
from sklearn.metrics import classification_report,confusion_matrix, accuracy_score
pred = lrc.predict(X_test)
print('Logistic Regression')
print(classification_report(y_test ,pred ))
print()
print('Exactitud: ', accuracy_score(y_test,pred))


from sklearn.metrics import confusion_matrix
import seaborn as sns
### Logistic Regression ###
y_prediccion_lrc = lrc.predict(X_test)
y_real_lrc = y_test
cm = confusion_matrix(y_real_lrc, y_prediccion_lrc)
f, ax = plt.subplots(figsize =(5,5))
sns.heatmap(cm,annot = True,linewidths=0.5,linecolor="red",fmt = ".0f",ax=ax)
plt.xlabel("y_prediccion_lrc")
plt.ylabel("y_real_lrc")
print('Logistic Regression')
print('Cantidad de mensajes en set de prueba: ',y_test.shape)
plt.show()
print('Exactitud: ', accuracy_score(y_test,y_prediccion_lrc))


### Random Forest ###
#Evaluar el modelo prueba (test) del dataset
#ham = 0
```

```
#spam = 1
from sklearn.metrics import classification_report,confusion_matrix, accuracy_score
pred = rfc.predict(X_test)
print('Random Forest')
print(classification_report(y_test ,pred ))
print()
print('Exactitud: ', accuracy_score(y_test,pred))


from sklearn.metrics import confusion_matrix
import seaborn as sns
### Random Forest ###
y_prediccion_rfc = rfc.predict(X_test)
y_real_rfc = y_test
cm = confusion_matrix(y_real_rfc, y_prediccion_rfc)
f, ax = plt.subplots(figsize =(5,5))
sns.heatmap(cm,annot = True,linewidths=0.5,linecolor="red",fmt = ".0f",ax=ax)
plt.xlabel("y_prediccion_rfc")
plt.ylabel("y_real_rfc")
print('Random Forest')
print('Cantidad de mensajes en set de prueba: ',y_test.shape)
plt.show()
print('Exactitud: ', accuracy_score(y_test,y_prediccion_rfc))
```