ESCUELA POLITÉCNICA NACIONAL

FACULTAD DE CIENCIAS

ANÁLISIS Y DISEÑO DE UN MODELO PREDICTIVO PARA DETECCIÓN DE PHISHING BASADO EN URL Y CORPUS DEL CORREO ELECTRÓNICO

PROYECTO PREVIO A LA OBTENCIÓN DEL TÍTULO DE INGENIERÍA MATEMÁTICA

PROYECTO DE INVESTIGACIÓN

DOLORES FERNANDA ALBÁN TOAPANTA

dolores.alban@epn.edu.ec

Director: MSC. ROBERTO ANDRADE

roberto.andrade@epn.edu.ec

Codirector: MSC. MENTHOR URVINA

menthor.urvina@epn.edu.ec

QUITO, ABRIL 2022

DECLARACIÓN

Yo, DOLORES FERNANDA ALBÁN TOAPANTA, declaro bajo juramento que el trabajo aquí escrito es de mi autoría; que no ha sido previamente presentado para ningún grado o calificación profesional; y que he consultado las referencias bibliográficas que se incluyen en este documento.

A través de la presente declaración cedo mis derechos de propiedad intelectual, correspondientes a este trabajo, a la Escuela Politécnica Nacional, según lo establecido por la Ley de Propiedad Intelectual, por su reglamento y por la normatividad institucional vigente.

DOLORES FERNANDA ALBÁN TOAPANTA

CERTIFICACIÓN

Certificamos que el presente trabajo fue desarrollado por DOLORES FERNANDA ALBÁN TOAPANTA, bajo nuestra supervisión

Msc. Roberto Andrade	Msc. Menthor Urvina
Director del Proyecto	Codirector del Proyecto

DEDICATORIA

Dedicada a mi madre, gracias a ella soy una mejor versión de la que fui ayer. Es tan hermoso cuando en su mirada se refleja la alegría, de ver en quien te has convertido y lo que has logrado. Por esta razón este Trabajo es para ella.

ÍNDICE GENERAL

ÍNDICE DE FIGURAS

ÍNDICE DE CUADROS

RESUMEN

Uno de los delitos cibernéticos más reportados a nivel mundial es el phishing, por esta razón, actualmente se está desarrollando diversos sistemas anti-phishing (APS) para identificar este ataque en sistemas de comunicación en línea. A pesar de los esfuerzos, este ataque continúa sin cesar, teniendo como causas: la detección errónea en el ataque de día cero, el alto costo computacional y las tasas altas de falsificación. Aunque el enfoque de Machine Learning (ML) ha logrado una tasa de precisión favorable, se debe considerar que la elección y el rendimiento del vector de características es un punto clave para obtener un nivel de precisión elevado. En este trabajo, proponemos un modelo predictivo basado en ML y analizar la eficiencia de algunos esquemas anti-phishing que sirvieron para entender esta temática. El modelo propuesto consta de un módulo de selección de características que se utiliza para la construcción del vector final. Estas características se extraen de la URL, las propiedades de la página web y del corpus de correo electrónico, utilizando un sistema basado en componentes incrementales para presentar el vector resultante. El sistema utiliza modelos de clasificación, Random Forest y Naïve Bayes, que han sido entrenados en el vector de rasgos. Los experimentos se basaron en dataset compuestas por instancias de phishing y benignas. Utilizando la validación cruzada, los resultados experimentales indican una precisión del 97,5 % para las bases mencionadas en otros trabajos, mientras que para el abordaje de esta investigación a nivel local se obtuvo una precisión del 96,5 %.

Palabras clave: Anti-phishing, Ataques ciberneticos, Phishing, Middleware, Amenaza

ABSTRACT

One of the most reported cyber crimes worldwide is phishing, for this reason, various anti-phishing systems (APS) are currently being developed to identify this attack in online communication systems. Despite the best efforts, this attack continues unabated, having as causes, the erroneous detection in the zero-day attack, the high computational cost and the high rates of forgery. Although the Machine Learning (ML) approach has achieved a favorable accuracy rate, the choice and performance of the feature vector should be considered a key point to obtain a high level of accuracy. In this work, we propose a predictive model based on ML and analyze the efficiency of some anti-phishing schemes that served to understand this issue. The proposed model consists of a feature selection module that is used for the construction of the final vector. These characteristics are extracted from the URL, the web page properties and the email corpus, using a system based on incremental components to present the resulting vector. The system uses classification models, Random Forest and Naïve Bayes, which have been trained on the vector of traits. The experiments were based on datasets comprised of phishing and benign instances. Using cross-validation, the experimental results indicate a precision of 97.5 % for the bases mentioned in other works, while for the approach of this research at the local level a precision of 96.5 % was obtained.

Keywords: Anti-phishing, Cyberattacks, Phishing, Middleware, Threat

Capítulo 1
INTRODUCCIÓN

1.1. Introducción

Phishing es una clase de ataque cibernético que se usa a frecuentemente para el robo de datos personales, estos son: las credenciales de inicio de sesión y los números de tarjetas de crédito. Las técnicas de ingeniería social pretenden adquirir la identidad de los usuarios ingenuos o información confidencial sensible mediante el uso de correos electrónicos falsificados, sitios web falsos, anuncios/promociones dudosas en línea, SMS falsos de proveedores de servicios o empresas, etc. El objetivo principal es: atacar a las grandes corporaciones, instituciones financieras, compañías de pago, agencias militares y gubernamentales que generalmente sufren enormes daños financieros y de credibilidad de marca, (*e.g.*, los informes de seguridad de Estadísticas y Tendencias de 2021) indicaron que en enero de este año se registró el pico más alto en a nivel histórico en la cantidad de sitios de phishing a nivel mundial, según reporte de APWG[1] [18].

El principal sitio en sufrir este ataque sigue siendo el sector financiero, que registró el 24,9 % durante el primer trimestre, seguido por las redes sociales con el 2,.6 % y los distribuidores de servicios de sitios web con el 19,6 %. Otro sitio que presento un crecimiento acelerado es la cantidad correo electrónicos que registraron textos únicos en el campo asunto de los correos, con un total de 172793 asuntos diferentes en un solo mes.

En los últimos tiempos, los phishers han desarrollado un *ransomware* que ejecuta un código malicioso que afecta negativamente a los recursos informáticos y exige el pago de un rescate, para restaurar los recursos al estado original. La incidencia de estos correos electrónicos de phishing basados en *ransomware* según lo informado por CSO[2], mostró que el 93 % de los correos electrónicos de phishing ahora son *ransomware*. El informe observó que la mayoría de las víctimas tienden a pagar rápidamente debido a la naturaleza sensible de sus recursos.

En respuesta a la amenaza del phishing, se desarrollaron varias contramedidas llamadas APS[3]. Sin embargo, los *phishers* continúan adoptando nuevos patrones sofisticados en evolución para derrotar a los sistemas de defensa actuales.

Esto ha forzado a las víctimas a ser consientes en tomar medidas de seguridad para no caer en este delito y detectar cualquier otro tipo. Pese a esto los APS no son capaces de detectar las características de un correo con presencia de Phishing.

Y de esta manera, se produce la pérdida de información, causando una incertidumbre en los expertos en seguridad. Aunque cada día los sistemas se actualizan, pero los phishers buscan nuevas maneras de atacar y efectuar el robo a los usuarios. Motivados por esta premisa, en este proyecto se intenta determinar un esquema anti-phishing robusto para superar los desafíos actuales que enfrentan los APS existentes.

[1] **APWG:**Anti-Phishing Working Group tiene el objetivo de intentar eliminar el fraude y el robo de identidad causado por el phishing y los incidentes relacionados

[2] CSO: Chief Security Officer, informe Online, 2016

[3] APS: Anti-Phishing System

1.2. Problemática

Phishing es una amenaza informática basada en la utilización de mecanismos de ingeniería social con la finalidad de engañar a la víctima para que exponga datos personales, que más adelante se utilizaran para la suplantación de identidad en varios lugares ya sea en un sitio web, o entidades financieras. Este se considera como un problema que está presente a diario y presenta un riesgo enorme en la pérdida de información personal como contraseñas, tarjetas de crédito, cuentas bancarias, etc.

En Ecuador se han incrementado las transacciones en línea (*e.g.*, ventas, servicios bancarios, pagos servicios básicos, entre otros), que han generado como consecuencia una sociedad dependiente de la tecnología, y como resultado de dicha dependencia, una alta probabilidad de ser víctimas de fraude informático.

Dado que Ecuador cuenta con una baja legislación referente al delito informático de phishing, permite que los causantes de este tipo de delito no sean sancionados. En contraste, las victimas pueden sufrir consecuencias a nivel económico y psicológico, debido a que al ser víctima de robo de credenciales. Este contexto incrementa el temor de utilizar herramientas tecnológicas que en la actualidad son fundamentales, dado que son de gran ayuda para la situación actual que estamos viviendo.

Por esta razón, existe un notable crecimiento de los APS, cuyo objetivo es evitar el engaño que sufre el usuario al acceder a sitios web simulados con apariencia similar al de instituciones conocidas. Las victimas proporcionan sus datos en estos sitios web al observar que provienen de sitios confiables, sin sospechar que se trata de un fraude; con este fin comenzamos un estudio exhaustivo de este tema, así se identificó que alrededor del 85 % de personas han sido víctimas en este de delito al menos una vez.

Por ese motivo, es importante que los usuarios estén concienciados, pero a su vez que las empresas tomen medidas para evitar sufrir de phishing tanto en sus productos como servicios. Gracias a los métodos avanzados de ciber-inteligencia se detecta de forma rápida el ataque phishing y se responde cerrando las páginas clonadas, de esta manera se controla la exposición de información; logrando evitar robos de montos millonarios o simplemente cuidar datos personales. Por esta razón, se ha impulsado la creación de diversos métodos para identificar phishing, los cuales presentan un nivel de predicción variado dependiendo de la complejidad con la que se implementó.

Entonces, para minimizar el riesgo de ataques de phishing, las entidades financieras, empresas y personas se ven en la obligación de aumentar sus esfuerzos para protegerse. Esta es la razón principal sobre la utilización de sistemas de protección para phishing. Esta medida de protección ha tenido un crecimiento, gracias al uso tecnológico elevado que se ha dado en los últimos años, se han desarrollado técnicas de inteligencia artificial (IA) como Maquinas de vectores de Soporte, Random Forest, Redes Neuronales y el clasificador Bayesiano, que demuestran que tienen una gran capacidad para predecir y clasificar. [17] y [21]

Por lo mencionado anteriormente se pretende mejorar aquellos modelos que se basan en técnicas estadísticas como la regresión logística. Para lograr este objetivo, proponemos un modelo basado en IA para modelar la tarea de predicción de un sistema robusto contra un ataque phishing utilizando algoritmos de aprendizaje supervisado para obtener alguna mejora en los esquemas anti-phishing. El modelo predictivo consta de un módulo de selección de características que se utiliza para la construcción de un vector final para posteriormente utilizarlo en el modelado.

Las funciones para formar el conjunto de características se extraen de la URL[4] y del contenido del corpus del correo electrónico. Mediante la utilización de un sistema basado en componentes incrementales se organizan las principales características para tener con un modelo de aprendizaje automático que permita un sistema flexible y manejable. De esta manera, se pretende mostrar un enfoque con una mejor precisión de detección significativa en comparación con otros trabajos relacionados.

Cabe mencionar que la mayoría de los trabajos relacionados a este tema tienen como módulo de características las que se extraen de la URL, enfocándose principalmente en esto dejando de lado el corpus del correo donde actualmente los delincuentes cibernéticos se están enfocando, jugando con las mentes de las víctimas y sobre todo con la falta de información sobre este tema de vital importancia en la realidad que vivimos a diario.

1.3. Objetivos

Objetivo General

- Diseñar un modelo anti-phishing usando Random Forest y clasificador Bayesiano, con la finalidad de obtener un modelo predictivo mejorado, basado en el aprendizaje automático.

Objetivos Específicos

- Realizar un estudio del estado del arte sobre parámetros para detección de Phishing tanto en la URL y corpus del email; el uso de RF y el NB
- Implementar un modelo predictivo para el sistema anti-phishing, mediante la construcción de características basado en de RF y el NB.
- Realizar un análisis de desempeño mediante una validación cruzada de los resultados experimentales para indicar el rendimiento del modelo.

1.4. Justificación

En la actualidad, las grandes corporaciones y las personas realizamos grandes esfuerzos tanto a nivel monetario como a nivel investigativo para conocer los mecanismos de seguridad de la información, para ampliar nuestro panorama de conocimiento o al menos tener una pauta de lo que este delito es. Y tratar en medida ser víctima de este, Sin dejar de lado que los delincuentes cibernéticos para este caso los phishers, logran evitar estos mecanismos de defensa, utilizando estrategias como: él envió de correos infectados, SMS con publicidad falsa o que contengan links de acceso y el peligro de mediante los links de descarga entremos a paginas falsas esperando entrar a un dominio conocido. Esto motivo tanto a las empresas y a la academia a la investigación de los patrones que estos atacantes utilizan y por otro lado encontrar soluciones mediante técnicas de ML con un enfoque en la seguridad informática, sin dejar de lado a la analítica de datos.

[4]URL: Uniform Resource Locator, que en español significa Localizador Uniforme de Recursos.

1.4.1. Justificación Teórica

La trascendencia del delito informático phishing, se debe a que en la actualidad la cantidad de usuarios que utilizan internet se ha incrementado notablemente, ocasionando que accedan a los distintos sitios o reciban correos con alto contenido de phishing, originando colateralmente que la cantidad de víctimas día tras día crezca. Por tal motivo, las entidades financieras, empresas y personas en general buscan evitar la exposición al riesgo de robo de información mediante técnicas estadísticas que permiten mejorar sus sistemas anti-phishing, entre estas metodologías se puede encontrar modelos de predicción, modelos basados en el aprendizaje automático, modelos de clasificación, modelos de apilamiento, etc.

En Europa, los modelos de predicción se encuentran entre los más usados por ser de buen poder predictivo y de fácil implementación. La finalidad de este proyecto de investigación es brindar un aporte de mayor influencia en el diseño de esquemas anti-phishing que otras categorías como similitud visual y enfoque basado en motores de búsqueda. Por tal motivo, aunque otros modelos de *Machine Learning* (ML) como *K-Nearest-Neighbor* (KNN) se han utilizado en problemas de phishing, tanto Random Forests (RF) como el Naive Bayes (NB) se han encontrado adecuados debido a su atributo de clasificación binaria y simplicidad [6], [12] y [20].

1.4.2. Justificación Metodológica

Los investigadores han desarrollado una gran variedad de técnicas tanto estadísticas como de IA que son utilizadas para la construcción de modelos de clasificación, siendo una de ellas el método de ensamble como lo señalan [25] y [26]. Sin embargo, en respuesta a la amenaza que representa el phishing, se desarrollaron varias contramedidas llamadas APS. Aunque existan estos sistemas, se debe tener en cuenta que los phishers continúan adoptando nuevos patrones sofisticados en evolución para derrotar las técnicas de defensa actuales.

Específicamente, la mayoría de los APS existentes tienen problemas de posibilidad de un ataque de día cero, sobrecarga computacional superflua, altas tasas de falsos positivos y negativos [9] y [28]. Si bien algunos métodos existentes innovaron una o más características para lograr resultados prometedores, otros extrajeron un subconjunto del corpus de características existente para lograr los mismos resultados [2]. No obstante, varios APS que utilizan ML, lograron una tasa de precisión prometedora con un pico de 97,62 % [19], [36] y [37], la elección y el rendimiento del vector de características en estos algoritmos es la fuente tanto para una mejora como el caso contrario en los APS.

Llevándonos a la necesidad de examinar la posibilidad de utilizar el corpus de funciones de phishing existentes para crear un esquema anti-phishing sólido con una eficiencia significativa. Nuestro objetivo es proponer un sistema de phishing eficiente, es decir, un vector de características con una precisión de detección significativa en más de un algoritmo ML. Específicamente, se elige RF y NB para la evaluación de nuestro vector de características, porque la mayoría de los APS existentes utilizan estos algoritmos con mayor frecuencia que otros para comparar su enfoque en la mayoría de la literatura disponible.

A continuación, se describe brevemente el contenido de este trabajo.

En el capítulo 2 presenta la revisión de literatura acerca de esta temática, siendo estos el Análisis de características para detección, Módulo de selección de funciones y Los filtros usados para la detección. En esta sección se trató de explicar cómo se fue desarrollando el

entendimiento de este tipo de ataques la forma de clasificar estos rasgos y sobre todo se trató de explicar de una manera general y sencilla a comparación de otros estudios solo enfocados en esta explicación. En la siguiente parte se estudiará Los modelos de clasificación enfocándonos en los modelos que se utilizara para la modelación de esta investigación siendo estos: RF y NB.

La descripción del conjunto de datos y El análisis y exploración de estos es la base del Capítulo 3, siguiendo con la construcción del vector de características y el modelo predictivo. Para el vector se escogió las características descritas a detalle en este capítulo, además, al hacer este proceso se tiene una pauta para trabajos posteriores y con esto obtener alguna mejora ya sea en tiempo de ejecución o precisión. Adicionalmente, mostramos la evaluación del modelo tanto para los datos descargables/experimentales y datos locales.

En la capítulo 4, se presentan los resultados para el método propuesto para el conjunto de datos descargables y el conjunto de prueba para detección de correo locales, para el ultimo dataset se opta por utilizar los correos de 3 cuentas, una cuenta de Gmail, Outlook personal y una de institución educativa nacional, para probar el conjunto de características y ver el nivel de predictibilidad que se obtiene con estos datos; también se realizara una comparación de estos resultados y una comparación con los que se tiene de trabajos anteriores, para posteriormente presentar una discusión sobre el tema que se desarrolló en este trabajo.

Las conclusiones y recomendaciones se exponen en el capítulo 5.

Nota: En la parte final del documento se exponen los Apéndices correspondientes a este trabajo entre ellos se encuentra el código empleado para obtener los resultados expuestos en la sección de resultados.

Capítulo 2
MARCO TEÓRICO

En este capítulo se describen las nociones teóricas necesarias para comprender la metodología para la clasificación de URL utilizando métodos estadísticos para descubrir las propiedades léxicas y basadas en *host* de las URL de sitios web maliciosos. Con el propósito de entender como clasificar la presencia de un ataque malicioso a gran escala. Para la construcción del modelo de predicción. De forma general estas fueron tomadas de [14], [30] y [38], y serán continuamente utilizadas en este trabajo. Para alcanzar una buena comprensión de la metodología, es importante revisar el concepto de aprendizaje estadístico o *statistical learning*. Acorde a [22], el aprendizaje estadístico se relaciona con un conjunto grande de herramientas para la comprensión de datos, que pueden incluir varias situaciones.

Una vez que los datos han sido entendidos, se puede tener una idea de que camino tomar ó que técnica es la que se debe emplear específicamente para el problema que se pretende desarrollar, es decir, para predecir la probabilidad de ser víctima de robo de información o suplantación de identidad y simultáneamente obtener conclusiones a manera de deducción en los resultados de un modelo estadístico, se ha dividido este capítulo en cuatro secciones principales: metodología de investigación, análisis de características para detección de phishing, problemas de clasificación y su modelamiento, evaluación y selección del modelo y evaluación de su desempeño. Las referencias principales son [8], [14], [30], [34] y [38]. Se ha incluido las demostraciones y gráficos de algunos resultados, por su relevancia en el desarrollo del trabajo.

2.1. Metodología de Investigación

La problemática que se tiene para esta investigación, toma en cuenta que el Phishing es un conflicto social y en la actualidad se busca una solución eficiente. En base a esto se puede determinar qué proceso tiene como principal objetivo realizar una investigación eficiente, para tomar una acción eficaz.

Según [11], la investigación tiene una similitud con los métodos de investigación mixtos, ya que utiliza un conjunto de datos que contiene variables de tipo cuantitativo o cualitativo, pero cabe recalcar que esta problemática se centra en una situación específica.
Para este trabajo de investigación en particular se utilizamos la metodología CRISP-DM[1] que es un modelo de proceso independiente para la minería de datos. Consta de seis fases iterativas desde la comprensión del problema hasta el desarrollo. Se encuentra disponible en el Cuadro 5.1 encontrada en el Apéndice A, en esta, se describe las tareas y el resultado de estas fases.

A continuación, se va a ir desarrollando estas etapas a medida que se va avanzando la investigación, es importante mencionar que para el análisis de características de todo lo referente a la detección de phishing se realizó una revisión de literatura teniendo como principales fuentes [2], [9], [13], [24] y [30], para posteriormente realizar una comparación de

[1] CRISP-DM:Cross Industry Standard Process for Data Mining

resultados de los métodos que se propusieron en cada una de estas investigaciones esto se visualizara en la sección de resultados.

2.2. Análisis de características para detección de phishing

La definición de ataque phishing es un caso típico de clasificación binaria, ya que una comunicación en línea (*i.e.*, correo electrónico, sitio web o chat electrónico) se clasifica como: Phish o benigna.

Tratándolo más formalmente, sea w una solicitud que necesita clasificación, es decir

$$w \xrightarrow{x} \{\text{Phish, benigna}\}. \tag{2.1}$$

Entonces x es el sistema anti-phishing que contiene características, $f_i \in w$ dado que

$$w = \sum_{i}^{n} f_i, \quad n \in \mathbb{N},$$

i.e. conjunto no vacío.

Por lo tanto, una solicitud contiene al menos una característica (*e.g.*, enlaces, etiquetas HTML, scripts, certificado SSL, etc.), sobre la cual se puede consultar o clasificar la predicción de su estado. Dado que estas características pueden variar de sencillas a complejas, el modelo propuesto utiliza la evaluación de frecuencia de características para la composición de vectores de características representada por $x = \{x_1, x_2, ..., x_n\}$ que asignan la etiqueta y a cada $f_i \in w$, de modo que la etiqueta y es una clase binaria representada como:

$$y = \begin{cases} 1, & \text{si es phishing,} \\ 0, & \text{si es benigna,} \end{cases} \tag{2.2}$$

Representado (2.2) como

$$x_i : f(w) \rightarrow y$$

La ecuación (2.1) describe el problema de clasificación donde, dado un dato de entrenamiento D, que contiene $(w_1, w_2, ..., w_n)$ y cada w_i contiene un conjunto de características $(f_1, f_2, ..., f_m)$. Además, los datos de entrenamiento son un conjunto de clases. $C = (C_1, C_2)$ que representa sitios legítimos y de suplantación de identidad que:

$$C_1 = \{w - i, f_i \mid w_i \in D, y = benigna, i = 1, ..., m\},$$
$$C_2 = \{w - i, f_i \mid w_i \in d, y = phishing, i = m + 1, ..., p\}.$$

Por tanto, cada caso $w_i \in D$ se le puede dar una clase $c_i \in C$ y se representa como un par $(w_i, (c_i))$ dónde c_i es una clase de C asociada con el caso w_i en los datos de entrenamiento. Sea H el conjunto de clasificadores para $D \rightarrow C$, donde cada caso $c_i \in C$ se le da una clase y el objetivo es encontrar un clasificador $h_i \in H$ que maximiza la probabilidad de que $h(c_i) = c$ para cada caso de prueba. En el sistema propuesto, se eligen dos clasificadores de aprendizaje automático más comunes para la clasificación de phishing, a saber, RF y NB, para investigar el rendimiento del conjunto de características/vector y maximizar la precisión del enfoque propuesto. Estos sistemas se explicarán posteriormente en las siguientes secciones.

2.2.1. Módulo de selección de funciones

Un proceso de extracción de características implica la identificación de ciertas características en un conjunto particular de datos, (*e.g.*, spam o phishing o benignas, etc.) Tales características generalmente se marcan como huellas digitales, ocurren con poca o ninguna probabilidad de ocurrencia fuera de lo conocido. En la mayoría de los casos, estos rasgos suelen excluirse mutuamente de las otras. Se han extraído numerosas características, pero el problema de las características más representativas sigue siendo una preocupación. En este enfoque, utilizaremos la evaluación de características basada en el análisis de frecuencia de varias características recopiladas de conjuntos de datos de investigación y literatura existente. Esto se define como un módulo de selección de características (FSM) que consta de:

- Las características de la URL
- Las propiedades del documento web
- Las características del email

Estos tres componentes se consideran un filtro en FSM y cada factor se organiza en el enfoque para tener un sistema basado en componentes. Con base en este enfoque, los tres filtros se construyen como un filtro unitario y uno compuesto para lograr gradualmente un planteamiento de detección eficiente. El primero consta de las heurísticas definidas en cada componente (unidad de cálculo) y su puntuación de predicción que se (dispara) conjuntamente al siguiente filtro, el filtro compuesto representa la suma de todos los filtros de unidad del sistema, es decir $S_0 \subseteq S_1 \subseteq S_2 ... \subseteq S_C$ dónde S_C se construyó a partir de los subconjuntos de otras unidades. Por lo tanto, esté es el que unió computacionalmente el sistema al algoritmo de clasificación. La fase de extracción de características toma datos mixtos como entrada desde donde se extraen las características de la URL y las propiedades del documento web.

2.2.2. Las características de la URL (filtro F1)

Las características de la URL representan las características asociadas con las direcciones web donde se puede recuperar una página en particular de Internet. Los phishers suelen manipular las URL legítimas de diferentes formas para engañar a los usuarios desprevenidos. Las características de la URL se extraen ya sea una URL absoluta o una URL relativa mediante el análisis de la estructura de enlaces en el DOM (Dominio). Para la extracción de identidad de URL, FSM considera **href** y **src** atributos de los enlaces de anclaje, en particular las etiquetas $< a >$, $< area >$, $< link >$, $< img >$ y $< script >$ del árbol de DOM de una página web donde normalmente se encuentran las direcciones web. Basándose en el estudio preliminar, se construyó el módulo de selección de características mediante una serie de consultas sobre ciertos rasgos de la URL seleccionadas de las investigaciones existentes *e.g.*, [1], [16], [36] y [42]

Por ejemplo, la consulta sobre el uso del símbolo @ en una ruta de URL no arroja ninguna coincidencia para las URL legítimas a pesar de que el número de apariciones es menor en comparación con otras características. Se elige incluir el símbolo debido a su aparición desconocida en URL benignas. Por lo tanto, esta función proporciona un plano de vector de soporte marcado de la distancia marginal entre las URL de phishing y las URL benignas.

Basado en la metodología de evaluación de características de frecuencia, el algoritmo 1 para el 'Análisis de frecuencia de evaluación de características en URL' se encuentra disponible en el Apéndice 1-B.

Dada una lista de características inicial $F_{URL(n)}$, el algoritmo solo selecciona las características que se encuentren en la fuente de datos. Luego, la frecuencia de ocurrencia de la característica seleccionada se determina usando la ecuación (2.3). Si este valor supera el límite de exclusión, la característica se incluye en la nueva lista de características, *S*. Este proceso se repite hasta que se agota la lista de funciones inicial. A continuación, se clasifica la nueva lista de funciones *S*, y se determina el rendimiento de cada función. Luego, se selecciona una nueva característica final de dimensión, *m*, que consiste en características de mejor desempeño. Las características de URL seleccionadas en el enfoque utilizan una información de frecuencia (FI) cuyos valores se encuentran entre 1 y 0. Este valor describe el peso estadístico de cada característica en toda la base de datos. Es decir,

$$FI = F_{url} / \sum DB \tag{2.3}$$

$$0 \ll FI \ll 1$$

donde 0 significa que no se encontró ninguna ocurrencia y 1 significa que se encontraron en todas las ocurrencias.

Después de una observación detallada del FI, FSM construye el valor para cada característica de URL existente y, al final, las siguientes características de URL fueron seleccionados en nuestro enfoque ya que su FI excedió el límite de exclusión definido dentro del sistema. Una ilustración simple del límite de exclusión (es decir, valor umbral,θ) se da como:

$$0,10 \ll FI_{phishing} \ll 1, \quad 0 \ll FI_{benigna} \ll 0,20$$

1. **URL con símbolo @:** esto implica el uso del símbolo @ en la ruta URL de un sitio web. Este símbolo se utiliza para redirigir el tráfico al sitio de phishing cuyo nombre de dominio sigue inmediatamente al símbolo @.

   Por ejemplo, $https://secure-document-viewer-login.du.r.appspot.com/-eid=jdoe1@emailhost.c$ dirigirá a un usuario a emailhost en lugar de jdoelemail. El símbolo @ generalmente viene con un nombre de dominio más corto a diferencia de otros símbolos como (-) ó ($\cdot$).

   Por lo tanto, si la URL contiene el símbolo @, la suplantación de identidad es legítima.

2. **Uso de la dirección IP como URL:** esto implica el uso de la dirección IP para representar el nombre de dominio de un sitio web. Por lo general, esta práctica es muy común para ocultar la información original de un nombre de dominio. Por lo tanto, dicha dirección IP generalmente denota suplantación de identidad o dominio sospechoso.

   Si la URL contiene la ruta del dominio como dirección IP, entonces existe suplantación de identidad.

3. **URL con código de carácter hexadecimal:** los phishers suelen ocultar las URL de phishing mediante el uso de códigos hexadecimales para representar los números en la dirección IP. Cada código hexadecimal suele comenzar con un símbolo (%).

   Por ejemplo, $https://manga-now.com/es/hakam\%20new/piled.php-email$ que fue informado en septiembre de 2020 por PhishTank donde se evidencia que se utilizó el código de caracteres hexadecimal.

Si la URL contiene caracteres hexadecimales, entonces es phishing; de lo contrario, es legítimo.

4. **Longitud de la URL:** esto implica obtener una longitud de la URL de más de 45 caracteres. Por ejemplo, $HTTP//womenincoachingsuccess.com$ que se encuentra en la base de datos de PhishTank es una URL legítima. Una observación meticulosa de la base de datos de PhishTank indicó que cualquier longitud de más de 45 es probable que sea phishing.

   Si la longitud de la URL es superior a 45, entonces es suplantación de identidad. De lo contrario, legítima

5. **URL con múltiples (//)**: Esto implica el uso de más de un (//) en la ruta del nombre de dominio de una URL. Una consulta de búsqueda en la base de datos de 1 millón de PhishTank de URL legítimas en formato .csv excel devuelve 0 para esta función.

   Por ejemplo, $http://alahmadymotors.org//wp-content/images//home.html$, se observa 3 repeticiones de este símbolo. Por lo tanto, si la URL contiene varios (//), la suplantación de identidad (phishing). De lo contrario, es legítimo.

6. **Presencia de Http en las URL:** es la presencia de $Http$ en la URL. Indica que la página web no cifra los datos, razón por lo que intercambia información con el navegador por medio del estándar HTTP. Google Chrome clasifica estos sitios como inseguros.

7. **Presencia de Https en las URL:** es la presencia de $Https$ en la URL. Este símbolo de seguridad no asegura que la página web esté protegida de las amenazas. Estos lugares cifran la información difundida entre el sitio y los usuarios.

   Efectivamente, cifrar los datos que se difunden es positivo, ya que la información que se intercambia entre el navegador y el sitio no es asequible a terceros. Esto nos brinda la confianza de introducir las contraseñas y los datos de la tarjeta de crédito sin el riesgo de intromisión de por medio.

   Por ejemplo $https://mantania.gr/mbres/box$ se pensaría que se dirige a una página oficial y que nuestros datos están protegidos. Pero el problema es que los datos personales aún pueden ser robados desde el mismo sitio. Cabe mencionar que una página con presencia de phishing puede obtener un certificado de seguridad sin trabas y cifrar todo el tráfico que ocasiona.

   Por lo tanto, una conexión segura no equivale a un sitio seguro

Cabe señalar que se omitieron ciertas características de la URL, como el uso de (-) y el número de puntos. Esto se debe a que se observa que dichas características suelen estar relacionadas con la longitud de la URL. Es decir, el número de puntos suele alargar la longitud de la URL.

Por ejemplo, $http://sonic-ally.com/owaOutlook/xchan_owa/$ reportado en septiembre de 2021 por PhishTank había alargado la longitud de la URL con más de tres puntos. Además, no se incluye el guion porque este carácter tiene aproximadamente un 11 % ocurrencias en la base de datos de 1 millón. Además, el símbolo del guion a menudo se asocia con URL alargadas como se observa en nuestra consulta ejecutada en el corpus de datos

de phishing. Por lo tanto, nuestra elección de la longitud de la URL a nuestro leal saber y entender es suficiente para acomodar estas características omitidas.

En el Cuadro 2.1 se ilustra la tasa de frecuencia de las características de URL seleccionadas y descritas anteriormente.

S / N	Función de URL	Frecuencia / Proporción
1	Símbolo @ en URL	0,23
2	URL basada en IP	0,5
3	URL con código hexadecimal	0,45
4	Longitud de URL larga	0,68
5	Varias barras en la ruta de la URL	0,82
6	Http presente en la URL	0,78
7	Https presente en la URL	0,53

CUADRO 2.1. La frecuencia del conjunto de características de URL. Fuente [30]

2.2.3. Las propiedades del documento web (filtro F2)

Las propiedades del documento web de una página web se extraen de la etiqueta de documento. Este proceso de extracción se basa en el concepto de método Término-Frecuencia de un Documento de Frecuencia Inversa (TF-IDF). El método se utiliza para extraer un conjunto de palabras clave del documento d, que se recopila de varias partes de una página web. El TF-IDF refleja la estadística numérica de cuán relevante es una característica para un documento en un corpus de datos. El valor de TF-IDF aumenta proporcionalmente al número de veces que aparece una característica en el documento, pero se compensa con la frecuencia de la característica en el corpus. Por tanto, un término particular definido como t tiene un peso TF-IDF alto si el término tiene una frecuencia alta en un documento D dado y una frecuencia baja si el término es relativamente poco común.

Dado un documento dy, su conjunto de identidad de términos t, el FSM usa la medición de la tasa de frecuencia para determinar la inclusión de una característica en la clase discriminatoria, el Algoritmo de Análisis de frecuencia de evaluación de características se encuentra utilizable en el Apéndice 2-B.

El algoritmo 2 presenta el flujo de la metodología del sistema, y las notaciones para este algoritmo se encuentran el cuadro 5.2, disponible en el Apéndice 1-B.

Teniendo una base una lectura extensa sobre este tema, se seleccionan estas características debido a su detección y la tasa de evaluación arrojó resultados prometedores en Positivo verdadero con falsos positivos insignificantes. Además, dado que las representaciones de nomenclatura de diversas características se pueden presentar de manera diferente en diversas formas.

Existe la necesidad de asegurar una inter-correlación característica a característica. Sobre la base de este requisito, se introduce una función de evaluación de selección de rasgos basada en correlación. Esta evaluación heurística se representa como:

$$M = \frac{k \cdot a}{\sqrt{k + k(k-1) \cdot b}}. \tag{2.4}$$

Donde M que tan bueno es el conjunto principal del subconjunto de características que contiene k características, a es la correlación media y b es la inter-correlación promedio característica-característica. El numerador de la ecuación (2.4) indica qué tan predictivo es un conjunto de características y el denominador proporciona cuánta redundancia hay entre estos rasgos. Esto hace que el enfoque tenga un costo computacional relativamente bajo y evite el sobreajuste en el método de selección. Por lo tanto, esto es importante para identificar qué tan importante es una característica en particular.

Con base en esta evaluación numérica, el FSM considera las siguientes características $(F_{V2} = F_1, F_2, ..., F_5)$ donde el valor 0 indica que no es phishing, $y > 0$ indica un estado sospechoso o que es phishing. Aunque varias características satisfacen la condición mayor que cero, se procede a elegir las que tienen más del 20 % de ocurrencia (valor de umbral).

Se presenta las propiedades de los documentos web que se tomó en cuenta para este estudio.

1. **Comprobación del nombre de dominio:** en la mayoría de los casos, los nombres de dominio del sitio web (D_n) tienen una fuerte relación con su contenido (C) que describe la naturaleza de los productos o servicios ofrecidos por la página web. Las palabras clave de este nombre de dominio suelen formar parte de la URL del dominio base y deben formar la etiqueta de la mayoría de los enlaces. Por lo tanto, si el conjunto de palabras clave de una página no está relacionado con su contenido (al menos el 75 %), entonces es un ataque.

$$F2_1 = \begin{cases} 0, & \text{si} \quad C \subseteq 0.75 * D_n, \\ > 0, & \text{si} \quad C \not\subset 0.75 * D_n. \end{cases}$$

2. **El nombre de dominio en la ruta de la URL:** algunas URL de phishing agregan el nombre de dominio de un sitio web legítimo dentro del segmento de ruta de una URL en un intento de estafar a los usuarios para que crean que están tratando con un sitio web auténtico. Esto implica que esta función puede detectar igualmente el uso de prefijo o sufijo por parte de los phishers al remodelar un nombre de dominio sospechoso, ya que la autenticidad será baja debido a un conjunto de identidades de palabras clave inapropiadas. Por lo tanto, si el nombre de dominio en la ruta de una URL contiene un prefijo o sufijo (D_{ps}) que no está indicado en su contenido, entonces es phishing.

$$F2_2 = \begin{cases} 0, & \text{si} \quad D_{ps} = D_{url}, \\ > 0, & \text{si} \quad D_{ps} \neq D_{url}. \end{cases}$$

3. **Acortamiento anormal de URL:** los atacantes utilizan acortadores de URL para ocultar las URL largas cuando solicitan a usuarios que inicien sesión en sus cuentas a través de un enlace, estó sucede especialmente en sitios de redes sociales. Si un patrón de marca de tiempo de acortamiento de enlace y el número de codificadores no es similar al servicio de acortamiento de URL (USS) genuino, como $Bitly$, $goo.gl$, $Owl.ly$, $Deck.ly$, $Su.pr$, etc., entonces es probable que sea suplantación de identidad.

$$F2_3 = \begin{cases} 0, & \text{si} \quad L_{t,enc(n)} = USS_{t,enc(n)}, \\ > 0, & \text{si} \quad L_{t,enc(n)} \neq USS_{t,enc(n)}. \end{cases}$$

4. **Código malicioso descargable:** la mayoría de los sitios o correos electrónicos de phishing contienen instrucciones para descargar ciertos archivos que se utilizan para perpetrar ataques basados en software delictivo. Si una página web contiene un enlace de descarga activo que contiene una extensión específica como *.aaa*, *.abc*, *.exx*, $.help_restore$, extensión de 6 a 7 caracteres aleatorios, entonces es suplantación de identidad y es sospechosa.

$$F2_4 = \begin{cases} 0, & \text{si} \quad Dg(link) = \text{ext-malisiosa}, \\ > 0, & \text{si} \quad Dg(link) \neq \text{ext-malisiosa}. \end{cases}$$

4. **Dominio de lista negra (DLN):** dado que la lista negra de URL sospechosas ha producido resultados prometedores en la detección de phishing, el dominio de lista negra se utiliza como una función en nuestro enfoque para administrar una lista de sitios de phishing detectados localmente para evitar el cálculo superfluo en un dominio malicioso ya conocido. Esta función ofrece las ventajas de proporcionar recursos para actualizar nuestro corpus de funciones y reducir los gastos generales. Si el nombre de dominio utilizado en el campo de acción de un formulario de inicio de sesión o URI (D_C) se encuentra en el dominio de la lista negra, es probable que sea phishing. De lo contrario, es legítimo.

$$F2_5 = \begin{cases} 0, & \text{si} \quad D_C \not\subset DLN, \\ > 0, & \text{si} \quad D_C \subseteq DLN. \end{cases}$$

El Cuadro 2.2, se presenta el análisis de frecuencia de las características seleccionadas para las propiedades de los documentos web

S / N	Característica web	Frecuencia / Proporción
1	Verificación del nombre de dominio	0,65
2	Nombre de dominio en la ruta de la URL	0,31
3	Acortamiento anormal de URL	0,24
4	Código malicioso descargable	0,73
5	Dominio de lista negra	0,77

CUADRO 2.2. Tasa de frecuencia de las funciones web seleccionadas. Fuente [30]

Dada una identidad (es decir, maliciosa o legítima) y un conjunto de características, la labor de determinar la veracidad de una solicitud se efectúa con un algoritmo de clasificación. Este aprende automáticamente cómo hacer predicciones precisas de instancias desconocidas basadas en observaciones pasadas o entrenadas. La precisión (*i.e.*, verdaderos positivos, falsos positivos, tasa de recuperación, verdaderos negativos, etc.) y los recursos-requisitos (*i.e.*, tiempo de entrenamiento, tiempo de respuesta, sobrecarga de memoria, etc.) con los que se realizan las predicciones determinan la eficiencia de dichos algoritmos de clasificación.

2.2.4. Las propiedades del corpus del email (filtro F3)

Hay varias medidas técnicas que las organizaciones pueden implementar para prevenir, o al menos reducir la cantidad de spam. Algunas de las técnicas más comunes incluyen el filtrado basado en contenido para el mensaje, el uso de listas para permitir que el email sea

procesado y denegar donde se pueden filtrar los correos electrónicos que se originan en una dirección IP específica, una dirección de correo o un nombre de dominio y el filtrado colaborativo.
A veces, hay casos en los que el filtrado de spam no tiene éxito en evitar que el *malware* de productos básicos u otros correos electrónicos no solicitados lleguen. En estos casos, es extremadamente crítico contar con controles de seguridad adicionales, como controles preventivos en los puntos finales, como el software de protección del punto final que puede detectar y prevenir la ejecución de software malicioso. Al igual que con cualquier contramedida técnica, generalmente existen métodos que también se divulgan y discuten públicamente y que pueden usarse para eludir un control de seguridad o evadir la detección. Estudiar la efectividad de los filtros de spam existentes contra los correos electrónicos de phishing para determinar qué técnicas existen que podrían ser utilizadas por los phishers para evitar ser detectados y marcados como spam.

Es por esta razón que para este filtro se analizara en contenido de los correos tanto en los que presentan phishing como los emails reales mediante la técnica de Text Mining con el propósito de detectar patrones adicionales a los que se conocen por defecto que son:

- El remitente no corresponde con el servicio que envía el correo
- La gramática con errores
- URLs falsas disfrazadas en hipervínculos
- Archivos adjuntos
- Correo de un servicio que no utilizado o no contratado

2.2.5. Text Mining

Para esta subsección se tomará como referencia [4], [5] y [23], que nos permite entender sobre la analítica de texto que, en los últimos años, hemos visto una explosión en la cantidad de datos que provienen de varias fuentes y están disponibles en muchos formatos. Esto ha sido posible, principalmente, por el avance masivo en el Internet. Varios dispositivos, como los teléfonos inteligentes, permiten la comunicación y el uso de aplicaciones en cualquier momento y en cualquier lugar; muchas actividades como ir de compras, interactuar con instituciones gubernamentales, o brindando apoyo a los clientes, se están moviendo hacia entornos digitales; muchos documentos están siendo digitalizados; la gente se encuentra e interactúan en plataformas virtuales.

La transformación de la Web, ha sido el punto de partida para expresar ideas, recomendaciones o actitudes, es obvio que los textos escritos en un lenguaje natural son una forma natural de comunicación humana. Los documentos textuales están, por lo tanto, fuertemente relacionados con los seres humanos. Actividades y se han convertido en una fuente que vale la pena analizar, los resultados de dicho análisis pueden aportar información útil en prácticamente todos los dominios. Como la capacidad de las personas para analizar textos sigue siendo esencialmente la misma, la disponibilidad de más datos requiere nuevos métodos computacionales para encontrar algo útil en grandes colecciones de documentos. Así, una disciplina conocida como la minería de texto se ha vuelto muy popular y atractiva. donde se puede definir como un proceso de conocimiento intensivo en el que un usuario interactúa

con una colección de documentos mediante el uso de herramientas analíticas para identificar y explorar interesantes patrones. Las aplicaciones se pueden encontrar en marketing, inteligencia competitiva, banca, salud, manufactura, seguridad, ciencias naturales y muchos otros dominios.

Las computadoras solo pueden analizar el aspecto sintáctico de los textos, lo que significa que son capaces de reconocer cómo se ordenan las palabras en los documentos. Porque los textos en un lenguaje natural se escriben utilizando una gramática, unos patrones sintácticos en un texto puede identificarse más o menos fácilmente. La semántica se refiere al significado de una palabra o grupo de palabras en un contexto. Sin una comprensión perfecta de un lenguaje no es posible entender completamente el significado. Afortunadamente, es posible resolver muchos problemas prácticos incluso sin la comprensión completa de un texto porque la sintaxis y la semántica suelen estar estrechamente relacionadas. Si dos textos usan las mismas palabras y estructuras sintácticas, es probable que sean semánticamente similares y pueden, por ejemplo, pertenecer a la misma clase de documentos.

Los documentos generalmente se pueden analizar de dos maneras diferentes. La estadística o el enfoque de aprendizaje automático utiliza la representación matemática de los textos. Los métodos lingüísticos, utilizando técnicas de procesamiento del lenguaje natural, representan textos utilizando modelos de lenguaje donde se contienen el significado y las diferentes relaciones. La minería de texto utiliza ambos enfoques para encontrar conocimiento, generalmente en un gran número de textos.

Relación de la minería de textos con la minería de datos

La minería de texto abarca una amplia variedad de tareas que pueden aportar información sobre diferentes aspectos de los textos. Las tareas típicas de minería de texto incluyen:

1. **Categorización de documentos:** asignación de un documento a una o más categorías predefinidas (*e.g.*, asignación de un artículo de periódico a una o más categorías, etiquetado de correos electrónicos como spam o ham);

2. **Agrupamiento**: agrupar documentos según su similitud, *e.g.*, para identificar documentos que comparten un tema común;

3. **Resumen:** encontrar las partes más importantes en uno o más documentos y crear un texto significativamente más corto que el original;

4. **Recuperación de información:** recuperación de documentos que coinciden con una consulta que representa la información necesaria de una gran colección de documentos;

5. **Extraer el significado de los documentos** o sus partes, identificar temas ocultos, analizando sentimientos, opiniones o emociones;

6. **Extracción de información:** extracción de información estructurada como entidades, eventos o relaciones de textos no estructurados;

7. **Minería de asociaciones:** encontrar asociaciones entre conceptos o términos en textos;

8. **Análisis de tendencias:** observar cómo cambian los conceptos contenidos en los documentos a tiempo;

9. **Traducción automática:** convertir un texto escrito en un idioma a otro idioma.

Algunas de las tareas de minería de texto son muy similares a las tareas de minería de datos. La minería de datos es el proceso automático o semiautomático de encontrar conocimiento implícito, previamente desconocido y potencialmente útil en colecciones de datos electrónicos. El conocimiento tiene una forma de patrones estructurales en los datos que pueden usado para hacer predicciones o dar respuestas en el futuro.

La minería de datos incluye muchos métodos, herramientas, algoritmos o modelos diferentes. Todos ellos requieren que los datos estén en forma estructurada, Esto significa que los datos se pueden representar en forma tabular como en una base de datos relacional. Los datos toman la forma de un conjunto de ejemplos (o instancias, puntos de datos, observaciones) descritos por valores específicos de sus características (o atributos, variables, campos).

Las características pueden ser de varios tipos.

- **categórico:** (nominal) el dominio es un conjunto discreto de valores donde el orden no tiene sentido;
- **binario:** un tipo especial de atributo categórico con solo dos posibles valores;
- **ordinal:** el dominio es un conjunto discreto de valores que se pueden ordenar;
- **numérico:** el valor de una característica es un número, ya sea entero o continuo.

Una unidad de texto puede ser una oración, unas pocas oraciones combinadas en un párrafo, o textos mucho más largos, como páginas web, correos electrónicos, artículos o libros. A veces, un texto puede ser solo unas pocas palabras que no son una oración válida, lo cual es bastante típico, (*e.g.*, para publicaciones cortas en redes sociales). Para poder aplicar los métodos de minería de datos a los textos, estos deben ser convertido en una representación estructurada. Una clásica representación estructurada de textos como vectores en un modelo de espacio vectorial se conoce como bolsa de palabras y el proceso de inferir esta representación que se basa en incrustar palabras en un espacio vectorial continuo.

Hay algunos problemas relacionados con la representación de la bolsa de palabras, que no son tan típicos para las tareas de minería de datos en general.

- Independientemente de la complejidad de las funciones, el espacio de entrada para los problemas de minería de texto es grande. No es raro que los diccionarios de varios, los lenguajes naturales contienen cientos de miles de palabras. Al considerar no solo las palabras como características sino también, (*e.g.*, las combinaciones, posiciones mutuas, o relaciones gramaticales entre palabras, la complejidad aumenta aún más). Por supuesto, no todas las palabras de un idioma aparecerán en la mayoría de las colecciones de documentos. Por otra parte, incluso relativamente las colecciones pequeñas pueden contener decenas de miles de palabras únicas.
- Los datos contienen mucho ruido que a menudo viene dado por la naturaleza de los datos. Los documentos de texto generalmente son creados por personas que hacen muchos errores, incluidos errores tipográficos, errores gramaticales y etiquetado incorrecto (*e.g.*, asignar una estrella en lugar de cinco estrellas a una reseña positiva).

- Los vectores que representan documentos son muy escasos, si comparamos el número de palabras únicas en un documento (decenas de palabras para mensajes cortos) el tamaño del diccionario (decenas de miles), la diferencia es enorme. Por lo tanto, solo una pequeña fracción de los elementos vectoriales son valores distintos de cero.

- En muchas tareas, solo una pequeña fracción del contenido suele ser relevante (*e.g.*, para una tarea de clasificación de documentos), solo unos pocos cientos de palabras del diccionario son necesarios para asignar correctamente los documentos a sus clases con una precisión relativamente alta.

- En muchas situaciones, la distribución de los datos en términos de clases o temas no está equilibrado. Esto significa que algunas clases tienen muchas más instancias, que los demás, lo que generalmente es un problema para el aprendizaje.

2.2.6. El proceso de minería de textos

Encontrar conocimiento útil en una colección de documentos de texto implica muchos pasos diferentes. Para organizarlos en un orden significativo, podríamos mirar el proceso general. Consta de los siguientes pasos:

a) **Definición del problema:** este paso es en realidad independiente de cualquier acción que posteriormente podrá tomarse. Aquí, el dominio del problema debe ser entendido y las preguntas a responder, definidas.

b) **Recopilación de datos necesarios:** es necesario identificar las fuentes de los textos que contienen la información deseada y recopilar los documentos. Los textos pueden provenir de una empresa (base de datos interna o archivo) o de fuentes externas, de la web, por ejemplo. En este caso, los rastreadores web deben implementarse con frecuencia para capturar directamente el contenido de páginas web. Después de la recuperación, los textos se almacenan para que estén listos para un análisis más extenso.

c) **Características definitorias:** Es necesario definir características que caractericen bien los textos y que sean adecuadas para la tarea dada. Las características comúnmente se basan sobre el contenido de los documentos. Un enfoque muy simple, bolsa de palabras, con ponderación de atributos binarios, toma cada palabra como una característica booleana. Su valor indica si la palabra está o no en un documento. Otros métodos es usar esquemas de ponderación más complicados o características que son derivados de las palabras (palabras modificadas, combinaciones de palabras, etc.).

d) **Analizar los datos:** En el proceso de encontrar patrones en los datos. De acuerdo con el tipo de tarea a resolver (*e.g.*, clasificación), se selecciona el modelo o algoritmo y se definen sus propiedades y parámetros. Seguidamente, el modelo se aplicará a los datos para encontrar una solución al problema. Para solucionar un problema concreto, suelen estar disponibles más modelos. La elección no se da explícitamente de antemano. Los modelos tienen diferentes características que influyen en el proceso de minería de datos y su resultado, algunos modelos tienen mayor complejidad computacional que los demás. Según la utilización del modelo, se puede preferir la creación a la aplicación rápida o viceversa. La idoneidad de un modelo a menudo depende en gran medida de los datos. Por lo tanto, seleccionar un modelo adecuado, encontrar la estructura correcta y ajustar los parámetros a menudo requiere mucho esfuerzo experimental.

e) **Interpretación de los resultados:** Aquí se obtienen algunos resultados del análisis. Necesitamos mirarlos cuidadosamente y relacionarlos con el problema que queríamos resolver. Esta fase puede incluir verificación y validación, que son pasos para aumentar la confiabilidad de los resultados.

Proseguimos, en la siguiente sección vamos a hablar de los modelos de clasificación, y de esta manera buscar una manera para modelar estos algoritmos y posteriormente usarlos en el tema de investigación que abordamos en este trabajo.

2.3. Problemas de clasificación y su modelamiento

En el aprendizaje estadístico se tiene principalmente problemas supervisados en este caso son dos, siendo estos: regresión y clasificación. Para este trabajo de investigación se trabajará con los problemas de clasificación, donde en este se tiene un conjunto de entrenamiento $(x_1, y_1), ..., (x_n, y_n)$, frecuentemente se divide la base en dos conjuntos de datos que son: entrenamiento/training y prueba/test, siguiendo normalmente el criterio de Pareto que es de 80 % y 20 %, donde la base de entrenamiento se usa para construir el clasificador, y el otro conjunto se usa como indica su nombre para evaluar el modelo. Recordemos que existen varias técnicas de modelamiento, que visualizará en el trascurso del documento.

2.3.1. Métodos basados en árboles

En esta parte de la teoría, describimos métodos basados en árboles para regresión y clasificación. Estos implican segmentar el espacio predictor en diversas regiones. Con la finalidad de hacer una predicción para una observación dada, normalmente usamos el valor de la respuesta modal para los datos de entrenamiento para la región a la que corresponde. Dado que el conjunto de las reglas de división utilizadas para segmentar el espacio del predictor se puede resumir en un árbol, estos tipos de enfoques se conocen como métodos de árboles de decisión.

Los métodos basados en árboles realmente son sencillos y útiles para la explicación. Sin embargo, por lo general, no son competitivos con los mejores enfoques de aprendizaje supervisado, en términos de predicción exactitud. Por lo tanto, en este capítulo también presentamos embolsado, bosques aleatorios, boosting y árboles de regresión aditivos bayesianos. Cada uno de estos enfoques implica la producción de múltiples árboles que luego se combinan para producir un solo predicción de consenso. Veremos que combinando una gran cantidad de árboles a menudo puede resultar en mejoras dramáticas en la precisión de la predicción, en las expensas de alguna pérdida en la interpretación.

Los fundamentos de los árboles de decisión se logran aplicar para los problemas de regresión y de clasificación. Para comenzar con este estudio lo haremos con los problemas de regresión para posteriormente ir a los de clasificación.

Entonces, para mejor comprensión de los árboles de regresión se ejemplifica su ejecución utilizando un ejemplo simple en este caso la predicción de los salarios de los jugadores béisbol.

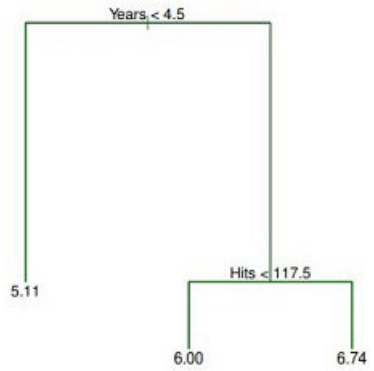


FIGURA 2.1. Para los datos de *Hitters*, un árbol de regresión para predecir el registro salario de un jugador de béisbol, basado en la cantidad de años que ha jugado en las ligas mayores y la cantidad de *hits* que hizo en el año anterior, en un dado un nodo interno, la etiqueta (de la forma $X_j < t_k$) indica la rama izquierda que emana de esa escisión, y la rama de la derecha corresponde a $X_j \geq t_k$. (*e.g.*, la partición a nivel superior del árbol) da como resultado dos ramas grandes. La rama izquierda corresponde a $Years < 4,5$, y la rama derecha corresponde a $Years >= 4,5$. El árbol tiene dos nodos internos y tres nodos terminales, o hojas. El número en cada hoja es la media de la respuesta de las observaciones, que caen ahí. Fuente [14]

2.3.2. Árboles de regresión

Para el ejemplo práctico de predicción de los salarios de los jugadores de béisbol mediante árboles de regresión usamos el conjunto de datos de bateadores para predecir el salario de un jugador de béisbol en función de Años (la cantidad de años que ha jugado en las ligas mayores) y Aciertos (el número de aciertos que hizo en el año anterior). primero quitamos observaciones a las que les faltan valores $Salary$, y $log-transformSalary$ para que su distribución tiene una forma más típica de campana.

La figura 2.1 muestra un árbol de regresión ajustado a estos datos. Y esto consiste en una serie de reglas de partición, empezando en la parte superior del árbol. La división superior asigna observaciones que tienen en $Years < 4,5$ a la rama izquierda. El salario previsto para estos jugadores viene dado por el valor medio de respuesta para los jugadores en el conjunto de datos. Para tales jugadores, el salario promedio logarítmico es 5,107, entonces hacemos una predicción de $e^{5,107}$ miles de dólares, es decir \$165174.

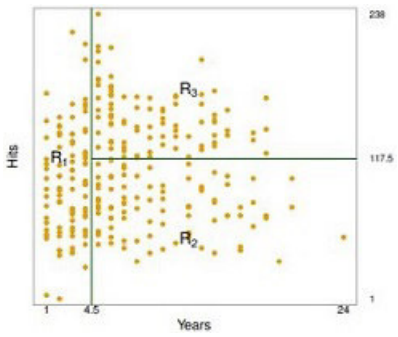


FIGURA 2.2. La partición de tres regiones para el conjunto de datos *Hitters* del árbol de regresión ilustrado en la Figura 2.1. Fuente [14]

Los jugadores con $Years >= 4,5$ se asignan a la rama derecha, y entonces ese grupo se

subdivide en $Hits$. En general, el árbol se estratifica o segmenta a los jugadores en tres regiones del espacio predictor: jugadores que han jugado durante cuatro años o menos, jugadores que han jugado durante cinco o más años y que hicieron menos de 118 *hits* el año pasado, y los jugadores que tienen jugó durante cinco años o más y que hizo al menos 118 *hits* el año pasado. Estas regiones se pueden describir como

$$R_1 = \{X \mid Years < 4,5\},$$
$$R_2 = \{X \mid Years >= 4,5, Hits < 117,5\},$$

y

$$R_3 = \{X \mid Years >= 4,5, Hits >= 117,5\}.$$

La Figura 2.2 ilustra las regiones en función de $Years$ e $Hits$. Los salarios previstos para estos tres grupos son $\$1.000 \times e^{5,107} = \$165.174, \$1.000 \times e^{5,999} = \$402834$, y $\$1000 \times e^{6,740} = \$845346$ respectivamente.

De acuerdo con la analogía del árbol, las regiones R_1, R_2 y R_3 se conocen como nodos terminales u hojas del árbol. Como es el caso de la figura 2.1, la decisión los árboles se dibujan típicamente al revés, en el sentido de que las hojas están en la parte inferior del árbol. Los puntos a lo largo del árbol donde el espacio predictor se divide se denominan nodos internos. En la Figura 2.1, los dos nodos internos se indican con el texto $Years < 4,5$ y $Hits < 117,5$. Asociamos a las secciones de los árboles que unen los nodos.

Podríamos interpretar el árbol de regresión que se muestra en la Figura 2.1 de la siguiente manera: Los años son el factor más importante para determinar el salario, y los jugadores con menos experiencia ganan salarios más bajos que los jugadores más experimentados. Dado que un jugador tiene menos experiencia, el número de golpes que hizo en el año anterior parece jugar poco papel en su salario. Pero entre los jugadores que han estado en las ligas mayores durante cinco años o más, el número de hits hecho en el año anterior afecta el salario, y los jugadores que hicieron más Los éxitos del año pasado tienden a tener salarios más altos.

El árbol de regresión que se muestra en la primera imagen de esta sección es probablemente una simplificación excesiva de la verdadera relación entre $Hits$, $Years$ y $Salary$. No obstante, presenta ventajas a comparación de otros modelos de regresión: es más fácil entender, dado que, posee una buena representación gráfica.

Predicción a través de la estratificación del espacio de características

Ahora discutimos el proceso de construcción de un árbol de regresión. Se podría decir que, hay dos pasos.

1. Dividimos el espacio predictor, el conjunto de valores posibles para $X_1, X_2, .., X_p$ en J regiones distintas y no superpuestas, $R_1, R_2, ..., R_J$.

2. Para cada una de las contemplaciones que se encuentra en la región R_j, se realiza la misma predicción, sencillamente es el promedio de los valores que se tiene como resultados para las contemplaciones del entrenamiento en R_j.

Por ejemplo, en el Paso 1 se tenemos dos bloques, R_1 y R_2, y que la media de los resultados de las observaciones de entrenamiento es 10 en el primer bloque, mientras que la

media para el segundo bloque es 20. Entonces para una observación dada $X = x$, si $x \in R_1$ predeciremos un valor de 10, y si $x \in R_2$ pronosticaremos un valor de 20.

Ahora elaboramos el Paso 1 anterior. ¿Cómo construimos las regiones $R_1, ..., R_J$?. En teoría, las regiones podrían tener cualquier forma. No obstante, se elige dividir el espacio del predictor en rectángulos de alta dimensión, por simplicidad y facilidad de interpretación del modelo predictivo que dé como resultado. El objetivo es encontrar cajas $R_1, ..., R_J$ que minimicen la suma residual de cuadrados (RSS), dada por

$$\sum_{j=1}^{J} \sum_{i \in R_j} (y_i - \hat{y}_{R_j})^2,$$

donde $\hat{y}_{R_j}$ es la media de los resultados en el conjunto de entrenamiento en la j-ésima caja. El costo computacional es demasiado alto para cada partición del espacio de características en los bloques J. Por consiguiente, se toma el enfoque de observar desde arriba en dirección a abajo lo vamos a conocer como división binaria recursiva. Tiene este nombre, dado que el proceso empieza desde arriba del árbol, para posteriormente dividir de forma sucesiva el espacio predictor; cada separación forma dos ramas nuevas. Este procedimiento es altamente costoso dado que se toma la división óptima para la edificación del árbol, pudiendo analizar que en un futuro se tendrá un buen árbol.

Ejecutando la división binaria recursiva, se selecciona primero el predictor X_j y el punto de incisión s al dividir el espacio predictor en las regiones $\{X \mid X_j < s\}$ y $\{X \mid X_j \geq s\}$ conduce a la reducción más alta para RSS. (La notación $\{X \mid X_j < s\}$ significa *la región del espacio predictor en el que X_j toma un valor menor que s*.) Es decir, se considera todos predictores $X_1, ..., X_p$, y los valores factibles del punto de corte s para cada uno de los predictores, y escoger el predictor y el punto de inserción, de tal manera que el árbol que se obtuvo de resultado tiene el RSS más pequeño. Con mayor detalle, para cualquier j y s, tenemos que definir el par de semiplanos

$$R_1(j,s) = \{X \mid X_j < s\} \quad \text{y} \quad R_2(j,s) = \{X \mid X_j \geq s\},$$

y buscamos el valor de j y s que minimice la ecuación

$$\sum_{i:x_i \in R_1(j,s)} (y_i - \hat{y}_{R_1})^2 + \sum_{i:x_i \in R_2(j,s)} (y_i - \hat{y}_{R_2})^2, \tag{2.5}$$

donde $\hat{y}_{R_1}$ es la respuesta media para las observaciones de entrenamiento en $R_1(j,s)$, y $\hat{y}_{R_2}$ es la respuesta promedio de las observaciones de entrenamiento en $R_2(j,s)$. Encontrar los valores de j y s que minimizan (2.5) se puede hacer con bastante rapidez, especialmente cuando el número de características p no es demasiado grande.

Seguidamente, reiteramos el desarrollo, en busca del predictor óptimo y al igual que el punto de corte para fraccionar los datos con el fin de minimizar el RSS para cada una de las regiones. a pesar de, tener como propósito dividir el espacio predictor en su totalidad, solo se dividió una de las dos regiones antes mencionadas. Ahora tenemos tres regiones, replicamos el proceso anterior es decir dividir una de estas tres regiones, para minimizar el RSS. El desarrollo sigue hasta alcanzar el criterio.

Una vez creadas las regiones $R_1, ..., R_J$, predecimos la respuesta para una observación de prueba, utilizando el promedio de las observaciones en la región a donde estas observación de prueba pertenecen. En la Figura 2.3 se muestra un ejemplo de cinco regiones de este

enfoque.

Poda de árboles

El desarrollo explicado en la parte anterior dio predicciones esperadas sobre el conjunto de entrenamiento, pero hay que pensar que el sobreajuste de los registros, lleve a un mal rendimiento del conjunto de prueba, y la razón es que se puede tener un árbol complejo. Un árbol con menos regiones $R_1, ..., R_J$) daría como resultado una menor varianza y un mejor entendimiento, pero puede dar un sesgo un poco más alto. La solución a esto es armar un árbol que no sobrepase el umbral de disminución en el RSS. Dando como resultado árboles pequeños, aunque hay que pensar que, aunque la primera división no sea buena la siguiente será buena, y esto nos conducirá a una gran reducción en RSS posteriormente.

Así, el camino a seguir es construir árboles grandes T_0, para luego hacer un proceso de podado y así, obtener un ala más pequeño. Todos nos preguntamos el mejor método de podar, pero hay que dejarse llevar por la intuición dado que nuestra meta es elegir un subárbol que tenga la tasa de error de prueba más pequeña. Una vez que se tiene un subárbol, se puede estimar el error de prueba usando la validación cruzada o el manejo del conjunto de validación. No obstante, calcular el error de CV para cada subárbol es muy pesado, dado que se tiene varios subárboles pequeños. Para remediar esta situación es tomar una muestra del conjunto original de subárboles.

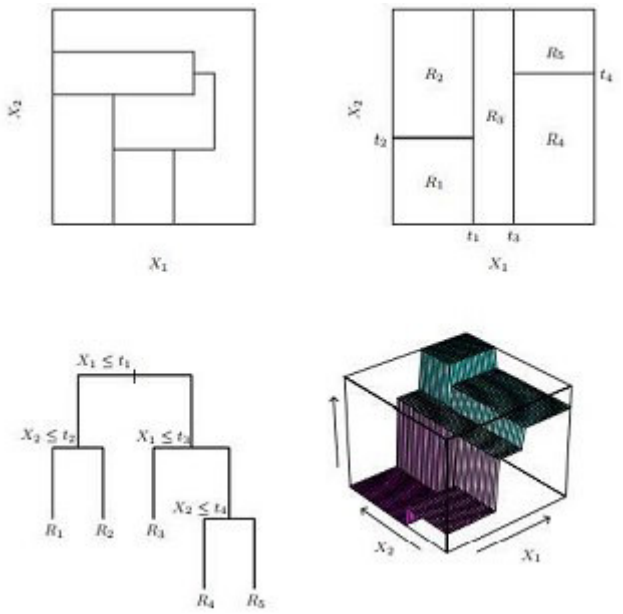


FIGURA 2.3. Arriba a la izquierda: una división del espacio de características bidimensionales que podría no ser el resultado de la división binaria recursiva. Arriba a la derecha: la salida de división binaria recursiva en un ejemplo bidimensional. Abajo a la izquierda: un árbol correspondiente a la partición en el panel superior derecho. Abajo a la derecha: Un gráfico en perspectiva de la superficie de predicción correspondiente a ese árbol. Fuente [14]

Reducción de la complejidad de costos-también conocida como poda del eslabón más débil-nos da una manera de hacer precisamente esto, considerar una cadena de árboles indexados por un parámetro de uniformidad no negativo α.

Para cada valor α le corresponde un subárbol $T \subset T_0$ de manera que

$$\sum_{m=1}^{|T|} \sum_{i:x_i \in R_m} (y_i - \hat{y}_{R_m})^2 + \alpha \mid T \mid, \tag{2.6}$$

Donde, $\mid T \mid$ denota el número de nodos terminales del árbol T, y R_m es subconjunto del espacio predictor, que corresponde al nodo terminal m-ésimo, y $\hat{y}_{R_m}$ es la respuesta predicha asociado con R_m, siendo este, la media de las observaciones de entrenamiento en R_m. El factor de ajuste α examina la estabilidad entre la complejidad del subárbol y el ajuste a los datos de entrenamiento. Cuando $\alpha = 0$, entonces el subárbol T será igual a T_0, dado que la ecuación (2.6) solo mide el error de entrenamiento.

A pesar de que, la medida α aumenta, hay que pensar que el costo por tener un árbol con diversos nodos terminales es alto, por lo que la ecuación (2.6) tenderá a bajar para un subárbol más pequeño. Resulta que a medida que aumentamos α desde cero en (2.6), las ramas se podan del árbol de forma anidada y predecible, obteniendo así la totalidad.

La cadena de subárboles en función de α es fácil. Así, seleccionamos el valor de α utilizando la validación cruzada. Con el conjunto completo de datos se obtiene el subárbol respectivo a α, el proceso descrito se lo encuentra en el Algoritmo 3y este a su vez lo podemos encontrar en el Apéndice 3-B para entenderlo de mejor manera.

Las figuras 2.4 y 2.5 muestran los resultados de ajustar y podar un árbol de regresión en los datos de *Hitters*, usando nueve de las características. Primero, al azar dividió el conjunto de datos por la mitad, lo que produjo 132 observaciones en el conjunto de entrenamiento y 131 observaciones en el conjunto de prueba. Luego construimos un gran árbol de regresión en los datos de entrenamiento y varió α en (2.6) para crear subárboles con diferente número de nodos terminales. Finalmente, realizamos una validación cruzada de seis veces para estimar el error medio cuadrado (MSE) con validación cruzada de los árboles como una función de α. (Elegimos realizar una validación cruzada de seis veces porque 132 es un múltiplo exacto de seis.)

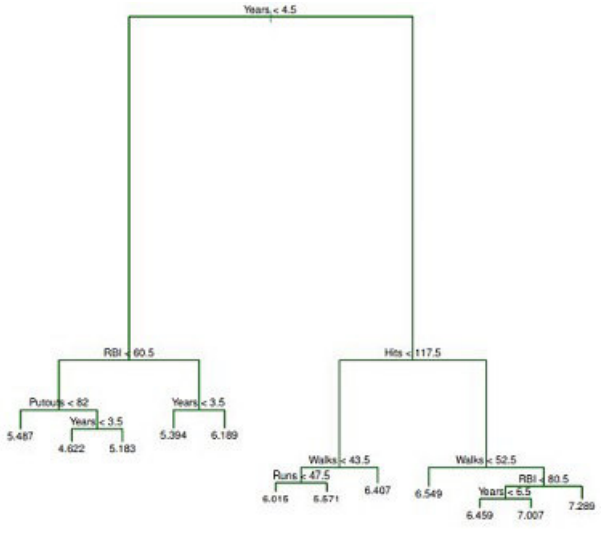


FIGURA 2.4. Análisis para el árbol de regresión en el Dataset de *Hitters*. El árbol sin podar que resulta de la división codiciosa de arriba hacia abajo en los datos de entrenamiento. Fuente [14]

En la figura 2.4 se visualiza un árbol de regresión no podado. La curva verde en la Figura 2.5 muestra el error de la validación cruzada (CV) en función del número de hojas,[2] mientras que la curva naranja indica el error de prueba. Adicionalmente se visualiza las barras de error estándar alrededor de los errores estimados. Se referencia, la curva de error de entrenamiento en color negro. El error del CV es una aproximación adecuada del error de prueba: el error toma su mínimo para un árbol de tres nodos, mientras que el error de prueba también desciende en el árbol de tres nodos (aunque toma su valor más bajo en el árbol de diez nodos). El podado del árbol que tiene tres nodos terminales se observa en la Figura 2.1.

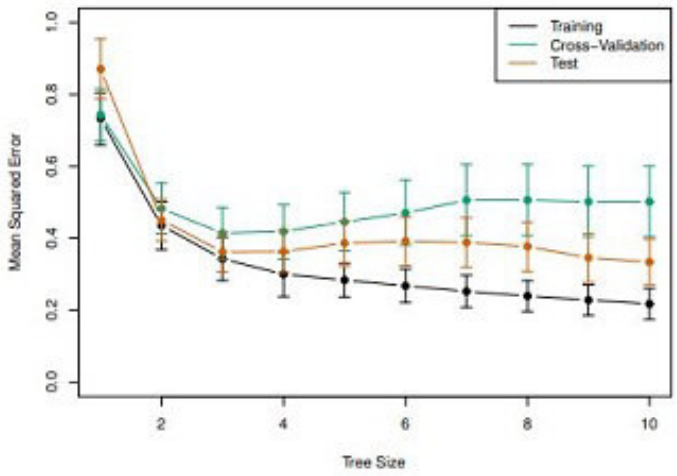


FIGURA 2.5. Análisis de árbol de regresión para los datos de *Hitters*. El entrenamiento, la validación cruzada y el MSE de prueba se muestran como una función del número de nodos terminales en el árbol podado. Se muestran las bandas de error estándar. El mínimo se produce un error de validación cruzada en un árbol de tamaño tres. Fuente [14]

2.3.3. Árboles de clasificación

Un *árbol de clasificación* es parecido a un árbol de regresión, la diferencia es usado para predecir una respuesta cualitativa en lugar de cuantitativa. Notemos que, para el árbol de regresión, la respuesta que predice es la media de las observaciones que corresponden al mismo nodo terminal. En contraste, para un árbol de clasificación, predecimos que cada observación pertenece a la clase de entrenamiento que ocurre más comúnmente observaciones en la región a la que pertenece. Al analizar el resultado obtenido del árbol de clasificación, los resultados que tienen importancia son la predicción del nodo terminal y las proporciones entre las observaciones del conjunto de entrenamiento que caen en esta región.

La labor de crecimiento de un árbol de clasificación o regresión es relativamente semejante. De la misma manera que, en la regresión, se utiliza división recursiva binaria para el crecimiento del árbol de clasificación. En esta, estructuración de clasificación, el criterio RSS no se utiliza para las divisiones binarias. Se tiene como opción la tasa de error de clasificación para este caso. Ya se piensa designar un dato en el bloque con mayor frecuencia en la región, la tasa de error de clasificación es sencillamente la división de las observaciones

[2] Aunque el error de CV se calcula como una función de α, es conveniente mostrar el resultado en función de $\mid T \mid$, el número de hojas; esto se basa en la relación entre α y $\mid T \mid$ en el árbol original crecido a todos los datos de entrenamiento.

en esa región que no corresponden a la clase normal:

$$E = 1 - \max_k(\hat{p}_{mk}).$$

Donde $\hat{p}_{mk}$ es la proporción de observaciones en el conjunto de entrenamiento en la m-ésima región de la k-ésima clase. Sin embargo, resulta que el error de clasificación no es lo suficientemente sensible para el cultivo de árboles, y en la práctica estas dos medidas son preferibles. El índice de Gini está definido por

$$G = \sum_{k=1}^{K} \hat{p}_{mk}(1 - hat\,p_{mk}), \tag{2.7}$$

La varianza total por medio de las clases K, es fácil observar que el índice de Gini admite un pequeño valor de todos los $\hat{p}_{mk}$ está entre 1 o 0. Debido a lo cual el índice de Gini es conocido como medida de pureza en el nodo: un valor pequeño de Gini nos muestra que las observaciones son de una sola clase.

Una alternativa al índice de Gini es la entropía, dada por

$$D = -\sum_{k=1}^{K} \hat{p}_{mk} \log hat\,p_{mk}.$$

Como $0 \leq \hat{p}_{mk} \leq 1$, se sigue que $0 \leq \hat{p}_{mk} \log hat\,p_{mk}$. Uno puede demostrar que la entropía tomará un valor cercano a cero si los $\hat{p}_{mk}$ están todos cerca de cero o cerca de uno. Por lo tanto, el índice de Gini, la entropía tendrá un valor pequeño, si el nodo m-ésimo es puro. De hecho, resulta que el índice de Gini y la entropía son bastante similares numéricamente.

Al construir un árbol de clasificación, ya sea el índice de Gini o la entropía, estos se utilizan normalmente para evaluar cada división especialmente, dado que estas dos medidas son más susceptibles a la pureza del nodo que la tasa de error de clasificación. Cualquiera de estos tres enfoques podría usarse al podar el árbol, pero la tasa de error de clasificación es preferible si la precisión de predicción del último árbol podado es la meta.

La figura 2.6 permite visualizar un ejemplo del conjunto de datos de $Heart$. Se presenta el resultado HD binario para 303 pacientes que presentaron dolor torácico. Un valor de resultado de S indica la presencia de enfermedad cardíaca según una prueba angiográfica, mientras que No significa que no hay enfermedad cardíaca. Hay 13 predictores que incluyen edad, sexo, col (una medida de colesterol) y otros factores cardíacos, y mediciones de la función pulmonar. La validación cruzada da como resultado un árbol con seis nodos terminales. Se pensó, que las variables predictoras toman valores continuos, pero hay que notar, que los árboles de decisión se pueden edificar con presencia de variables predictoras cualitativas. *e.g.*, en los datos cardíacos, algunos de los predictores, como $Sexo$, $Thal$ (prueba de esfuerzo con talio), y $ChestPain$, son cualitativos. Por lo tanto, una división en una de estas variables equivale a asignar algunos de los valores cualitativos a una rama y asignando el resto a la otra rama. En la Figura 2.6, algunos de los nodos internos corresponden a la división de variables cualitativas. por ejemplo, el nodo interno superior corresponde a dividir $Thal$.

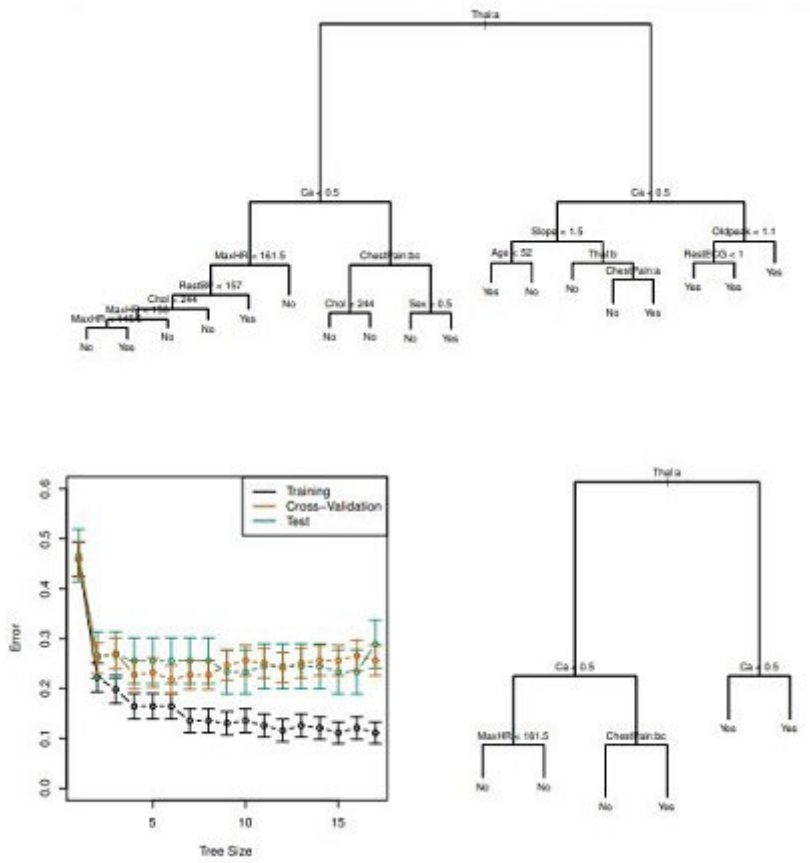


FIGURA 2.6. Datos del corazón. Arriba: El árbol sin podar. Abajo a la izquierda: error de validación cruzada, entrenamiento y error de prueba, para diferentes tamaños del árbol podado. Fondo Derecha: El árbol que tuvo el proceso de podado concerniente al error mínimo de validación cruzada. Fuente [14]

El texto $Thal$: a indica que la rama izquierda que sale de ese nodo consiste en observaciones con el primer valor de la variable $Thal$ (normal), y el nodo de la derecha consta de las observaciones restantes (defectos fijos o reversibles). El texto *ChestPain*: bc dos divisiones hacia abajo del árbol a la izquierda indica que la rama izquierda que sale de ese nodo consta de observaciones con el segundo y tercer valor de la variable *ChestPain*, donde los posibles valores son: angina típica, angina atípica, dolor no anginoso y asintomático. La figura 2.6 tiene una característica sorprendente: algunas de las divisiones producen dos nodos terminales que tienen el mismo valor predicho. Por ejemplo, considere la división $RestECG < 1$ cerca de la parte inferior derecha del árbol sin podar. Independientemente del valor de *RestECG*, se pronostica un valor de respuesta de *S* para esas observaciones. ¿Por qué, entonces, se realiza la división? La división se realiza porque conduce a una mayor pureza de los nodos. Es decir, las 9 observaciones correspondientes a la hoja de la derecha tienen un valor de respuesta *S*, mientras que $7/11$ de los correspondientes a la hoja izquierda tienen un valor de respuesta de *Si*. Suponga que tenemos una observación de prueba que pertenece a la región dada por esa hoja de la derecha. Tenemos la certeza que el valor de respuesta es *S*. Por el contrario, si una prueba de observación pertenece a la región dada por la hoja de la izquierda, entonces su valor de respuesta es probablemente *S*, pero estamos mucho menos seguros. Aunque el split $RestECG < 1$ no reduce el error de clasificación, mejora el Índice de Gini y la entropía, que son más sensibles a la pureza del nodo.

2.3.4. Árboles versus modelos lineales

Los árboles de regresión y clasificación tienen un sabor muy diferente al de los enfoques clásicos para la regresión y la clasificación. En particular, la regresión lineal asume un modelo de la forma

$$f(x) = \beta_0 + \sum_{j=1}^{P} X_j \beta_j, \tag{2.8}$$

mientras que los árboles de regresión asumen un modelo de la forma

$$f(x) = \sum_{m=1}^{M} c_m \cdot 1_{(X \in R_m)}, \tag{2.9}$$

donde $R_1, ..., R_M$ representan una partición del espacio de características, como en la figura 2.3. ¿Qué modelo es mejor? Depende del problema en cuestión. Si la relación entre las características y la respuesta está bien aproximada por un modelo lineal como en 2.8, entonces un enfoque como la regresión lineal probablemente funcione bien y superará a un método como un árbol de regresión que no explota esta estructura lineal. Si, en cambio, existe una relación altamente no lineal y compleja entre las características y la respuesta como como indica el modelo (2.9), como conclusión se tiene que los árboles de decisión superan el enfoque clásico. Un ejemplo ilustrativo se muestra en la Figura 2.7. Los rendimientos del enfoque clásico en los árboles pueden estimar el error de la prueba, utilizando tanto la validación cruzada como el conjunto de validación.

Por supuesto, pueden entrar en juego otras consideraciones más allá del simple error de prueba, jugar en la selección de un método de aprendizaje estadístico; por ejemplo, en ciertos escenarios, la predicción usando un árbol puede ser preferible en aras de la interpretabilidad y la visualización.

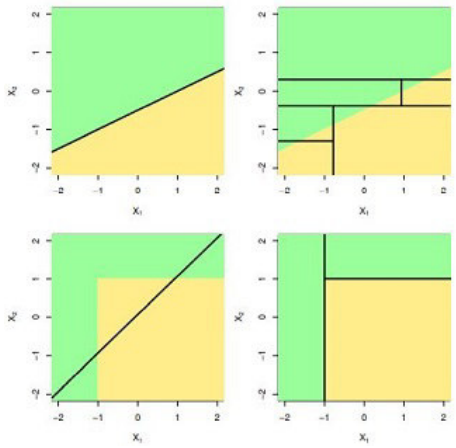


FIGURA 2.7. Fila superior: un ejemplo de clasificación bidimensional en el que el verdadero límite de decisión es lineal y está indicado por las regiones sombreadas. un clásico el enfoque que supone un límite lineal (izquierda) superará a un árbol de decisión que realiza divisiones paralelas a los ejes (derecha). Fila inferior: aquí el verdadero límite de decisión no es lineal. Aquí un modelo lineal es incapaz de capturar el verdadero límite de decisión (izquierda), mientras que un árbol de decisión es exitoso (derecha). Fuente [14]

2.4. Bagging, Boosting y Random Forests

Un método de conjunto es un enfoque que combina muchos modelos simples de (bloques de construcción) para obtener un modelo único y potencialmente muy poderoso. Estos modelos simples de bloques de construcción a veces se conocen como modelos débiles, ya que pueden conducir a predicciones mediocres por sí mismos. Ahora analizaremos el Bagging, Boosting y Random Forests. Estos son métodos de conjunto para los cuales el simple el bloque de construcción es una regresión o un árbol de clasificación.

2.4.1. Bagging

El Bootstrap, es una idea extremadamente poderosa. Es usado en varias ocasiones en las que calcular la desviación estándar de una cantidad relativamente grande es complicada. Vemos aquí que el Bootstrap se puede utilizar en un contexto completamente diferente, con el fin de mejorar los métodos de aprendizaje estadístico, como los árboles de decisión.

Los árboles de decisión que se discutió en la sección anterior sufren de una gran variación. Si dividimos la base de entrenamiento en dos partes aleatoriamente, y ajustamos un árbol en estas dos divisiones, los resultados que tendremos serán muy diferentes. Contrariamente, el desarrollo con baja varianza produce resultados semejantes, si se aplica repetidamente al conjunto de datos; la regresión lineal tendrá una varianza pequeña, si la razón de n a p es relativamente grande. *Bootstrap aggregation, o bagging*, es un proceso para reducir la varianza para un método de aprendizaje estadístico; Lo presentamos aquí porque es particularmente útil y de uso frecuente en el contexto de los árboles de decisión.

Dado un conjunto de n observaciones independientes $Z_1,...,Z_n$, con varianza σ^2, cada una de ellas, la varianza de la media $\bar{Z}$ está dada por σ^2/n. Es decir que, promediar un conjunto de observaciones baja la varianza. En consecuencia, para reducir la varianza y aumentar la precisión del conjunto de prueba hay que tomar varios conjuntos de entrenamiento de la población, realizar un modelo de predicción para cada conjunto de entrenamiento, y obtener el promedio de las predicciones resultantes. Es decir, podríamos calcular $\hat{f}^1(x), \hat{f}^2(x), ..., \hat{f}^B(x)$ usando B conjuntos de entrenamiento separados, y promediarlos para obtener un único modelo de aprendizaje estadístico de baja varianza, dado por

$$\hat{f}_{avg}(x) = \frac{1}{B}\sum_{b=1}^{B} \hat{f}^b(x).$$

Indudablemente, esto no es viable porque de forma general no tenemos paso a múltiples conjuntos de entrenamiento. En su lugar, podemos arrancar, tomando repetidas muestras del conjunto de datos de entrenamiento. Con este método producimos B diferentes conjuntos de datos de entrenamiento con Bootstrapped. Luego entrenamos nuestro método en el conjunto de entrenamiento con Bootstrapped bth para obtener $\hat{f}^{*b}(x)$, y finalmente promediar todas las predicciones, para obtener

$$\hat{f}_{bag}(x) = \frac{1}{B}\sum_{b=1}^{B} \hat{f}^{*b}(x).$$

Esto se llama Bagging. Si bien el Bagging puede mejorar las predicciones para muchos métodos de regresión, es particularmente útil para los árboles de decisión. Para aplicar Bagging a los árboles de regresión, simplemente construimos árboles de regresión B usando entrenamiento Bootstrapped de B conjuntos y promediar las predicciones resultantes. Estos árboles

crecen profundos, y no se podan. Por consiguiente, cada árbol en forma particular posee una varianza alta, a su vez un sesgo bajo. Promediar estos B árboles reduce la varianza. Bagging demuestra dar una buena precisión al mezclar cientos o incluso miles de árboles en un solo proceso. Se ha descrito el proceso de Bagging para la regresión, es decir, cuando se quiere predecir un resultado cuantitativo Y. Pensemos para un problema de clasificación donde Y es cualitativo. ¿Qué deberíamos hacer para extender el Bagging? el proceso más sencillo es el que se muestra a continuación. En cada prueba dada, podemos notar la clase predicha por cada B árbol, y tomar el mayor: la predicción es la clase que tiene la frecuencia más alta entre las B predicciones. La Figura 2.8 indica los resultados de los árboles Bagging en los datos de $Heart$. La tasa del error de prueba se muestra como una función de B, el número de árboles construidos utilizando conjuntos de datos de entrenamiento de Bootstrapped. Vemos que la tasa de error Bagging de prueba es ligeramente menor en este caso que la tasa de error de prueba obtenida de un solo árbol. El número de B árboles no es un factor decisivo en el Bagging; usamos un valor B muy grande que no conducirá a un sobreajuste. En la práctica nosotros usamos un valor de B lo suficientemente grande como para que el error se haya establecido. Utilizando $B = 100$ es suficiente para lograr un buen rendimiento en este ejemplo.

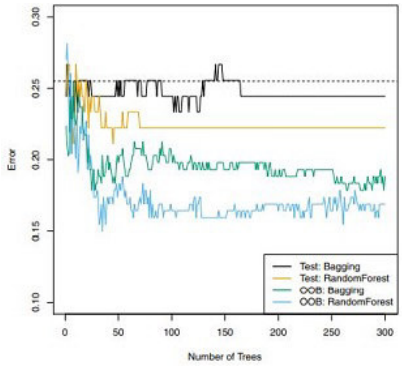


FIGURA 2.8. Resultados de Bagging y Random Forests para los datos de $Heart$. La prueba error (negro y naranja) se muestra como una función de B, el número de conjuntos Bootstrapped de entrenamiento utilizados. Se aplicaron en Random Forests con $m = \sqrt{p}$. la línea discontinua indica el error de prueba resultante de un solo árbol de clasificación, el verde y las trazas azules muestran el error OOB. Fuente [14]

Out-of-Bag Estimación de error fuera de bolsa

Resulta que hay una forma muy sencilla de estimar la prueba error de un modelo Bagged, sin pensar en la validación cruzada o el enfoque del conjunto de validación. Tomemos en cuenta que el Bagging busca que los árboles se ajusten muchas veces a los subconjuntos de observaciones con Bootstrapped, uno puede mostrar que, en media, cada árbol Bagged utiliza alrededor de $2/3$ de las observaciones. El resto de las observaciones se usa para ajustar un árbol que tiene como nombre observación fuera de la bolsa (OOB). Predecir la respuesta para la $i-sima$ observación utilizando cada uno de los árboles en que la observación fue OOB. Esto produce $B/3$ de las predicciones para la i-ésima observación. Con la intensión de encontrar una predicción para la i-ésimo observación, hay que promediar las respuestas predichas cuando la regresión es el objetivo, pero para la clasificación se toma la mayoría de votos. Esto lleva a una sola predicción OOB para la i-ésima observación. Una predicción OOB se puede obtener de esta manera para cada una de las n observaciones, de las cuales

el MSE del OOB general para la regresión o error de clasificación para la clasificación se calcula.

El error OOB que da como respuesta es un error válido de la estimación prueba para el modelo en Bagged, ya que la respuesta para cada observación se predice usando solo los árboles que no encajaban. La Figura 2.8 muestra el error OOB en los datos de *Heart*. Podemos demostrar que con B lo suficientemente grande, el error OOB es virtualmente equivalente para dejar uno fuera de error de validación cruzada. El enfoque OOB para estimar el error de prueba es particularmente conveniente cuando se realiza Bagging en grandes conjuntos de datos para los cuales la validación cruzada sería computacionalmente onerosa.

Medidas de importancia variable

Como hemos podido notar, el Bagging nos brinda como resultado una precisión mejorada sobre la predicción con el uso de solo un árbol. Tristemente, es complicado entender el modelo que dio como resultado. Hay que recordar que una ventaja de estos árboles es que fácil de entender, como el que se visualiza en la Figura 2.1. No obstante, cuando se agrupa un numero grande de árboles, visualmente no es posible representarlo con el uso de un solo árbol, y peor notar que variables o variables son las que más aportan para el proceso. Por esta razón, el Bagging mejora la predicción, aunque la interpretabilidad da mucho que desear.

Hay que notar que el conjunto de árboles Bagged es complicado de entender a comparación de un solo árbol, podemos obtener un resumen cada predictor usando el RSS, pero solo para regresión o el índice de Gini para árboles de clasificación. Para los árboles de regresión, la cantidad total del RSS reduce debido a fracciones sobre un predictor dado, mediando sobre todos los B árboles. Un valor grande indica un predictor importante. Al igual, en el contexto de la clasificación de árboles Bagging, se suma la cantidad total que el índice de Gini (2.7) reduce por fraccionar en un predictor dado, obteniendo el promedio sobre todos los B árboles.

En la Figura 2.9 se visualiza la representación gráfica de la importancia de las variables en el conjunto de datos de *Heart*. Vemos la disminución media en el índice de Gini para cada variable, en relación con la mayor. Las variables con mayor disminución promedio con respecto al índice de Gini son: *Thal*, *Ca* y *ChestPain*.

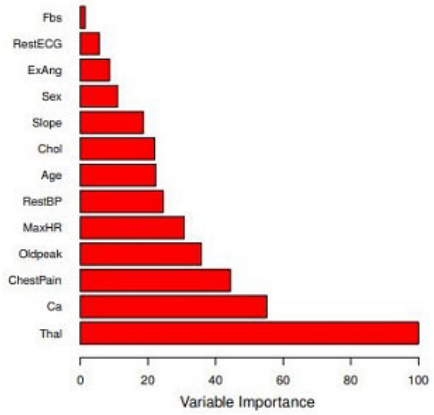


FIGURA 2.9. Un gráfico de la importancia de la variable para los datos de Herat. La importancia de la variable se calcula utilizando la disminución media del índice de Gini y se expresa en términos máximos relativos. Fuente [14]

2.4.2. Boosting

Ahora discutimos sobre el Boosting, otro enfoque más para mejorar las predicciones resultantes de un árbol de decisión. Al igual que Bagging, Boosting es en general un enfoque que se puede aplicar a muchos métodos de aprendizaje estadístico para regresión o clasificación. Aquí restringimos al contexto de los árboles de decisión. Recuerde que el Bagging implica la creación de múltiples copias del conjunto de datos de entrenamiento original usando el programa de bootstrap, ajustando un árbol de decisión separado para cada uno, y luego combinar todos los árboles para crear un solo modelo predictivo. En particular, cada árbol se basa en un conjunto de datos de arranque, independiente de los otros árboles. Boosting funciona de manera similar, excepto que los árboles son crecen secuencialmente: cada árbol se cultiva usando información de previa de árboles crecidos. El boosting no implica un muestreo de bootstrap; en cambio cada uno de los árboles se ajustan a una versión modificada del conjunto de datos original.

Considere primero el escenario de regresión. Al igual que bagging, boosting implica combinar una gran cantidad de árboles de decisión, $\hat{f}^1, ..., \hat{f}^B$. ¿Cuál es la idea detrás de este procedimiento? A diferencia de ajustar un solo árbol de decisión grande a los datos, lo que equivale a ajustar los datos de forma estricta y potencialmente sobreadaptado, el enfoque de refuerzo aprende lentamente. Dada la corriente modelo, ajustamos un árbol de decisión a los residuos del modelo. es decir, nosotros ajustamos un árbol utilizando los residuos actuales, en lugar del resultado Y , como respuesta. Luego agregamos este nuevo árbol de decisión a la función ajustada para para actualizar los residuos. Cada uno de estos árboles puede ser bastante pequeño, con solo algunos nodos terminales. Para encajar arbolitos a los residuales, poco a poco mejoramos $\hat{f}$ en las zonas donde no funciona bien. El parámetro de contracción λ ralentiza el proceso aún más, permitiendo que más árboles de diferentes formas ataquen los residuos. En general, los enfoques de aprendizaje estadístico que aprenden lentamente tienden a desempeñarse bien. Tenga en cuenta que al Boosting, a diferencia del Bagging, que la construcción de cada árbol depende en gran medida de los árboles que ya han crecido. Acabamos de describir el proceso de árboles boosting de regresión. Los árboles boosting de clasificación procede de una manera similar pero un poco más compleja.

2.4.3. Random Forests

El Bagging o Bootstrap es una técnica para reducir la varianza de una función de predicción estimada. Bagging parece funcionar especialmente bien para procedimientos de varianza grandes y sesgos pequeños, como árboles. Para la regresión, simplemente ajustamos el mismo árbol de regresión muchas veces a las versiones Bootstrap sampled de los datos de entrenamiento, y promediamos el resultado.

Como dice [7] Random Forests es una modificación de Bagged que construye un conjunto grande de árboles descorrelacionados para después promediarlos. En muchos problemas, el rendimiento de los bosques aleatorios es muy similar al Bagging, y son sencillos de entrenar y ajustar. Como consecuencia, al azar los bosques son populares y se implementan en una variedad de paquetes.

Definición de Random Forests

La idea esencial del Bagging es promediar varios modelos ruidosos, pero algo insesgados y, de esta manera, reducir la varianza. Los árboles son los principales candidatos para el

bagging, ya que logran capturar interacciones complicadas. De la fuente [38] se obtiene el Algoritmo 4 Random Forests para regresión o clasificación está en la disponible en el Apéndice 4-B.

Si se profundiza en los árboles lo suficiente, se tiene un sesgo relativamente bajo. Puesto que, los árboles son notablemente ruidosos, el promedio tiene un beneficio grande. Además, dado que cada árbol generado en el Bagging se distribuye de manera idéntica (id), la posibilidad del promedio de B árboles es lo mismo que la expectativa de cualquiera de ellos. Esto significa que el sesgo de los árboles en conjunto es el mismo que el de los árboles individuales, y la única mejora es por medio de la disminución de la varianza. Esto contrasta con el impulso, donde los árboles se generan de forma adaptativa para eliminar el sesgo y, por lo tanto, no son *id*

Un promedio de B variables aleatorias (iid), cada una con varianza σ^2, tiene variance $\frac{1}{B}\sigma^2$. Si las variables son simplemente *id* distribuidas de manera idéntica, pero no necesariamente independiente con correlación positiva por pares ρ, la varianza del promedio es

$$\rho\sigma^2 + \frac{1-\rho}{B}\sigma^2 \tag{2.10}$$

Como B aumenta, el segundo término desaparece, pero el primero permanece y, por lo tanto, el tamaño de la correlación de pares de árboles en conjuntos limita los beneficios de promediar. La idea en los bosques aleatorios 4 es mejorar la reducción de la varianza del bagging reduciendo la correlación entre los árboles, sin aumentar demasiado la varianza. Esto se logra en el proceso de crecimiento de los árboles mediante la selección aleatoria de las variables de entrada.

Específicamente, al hacer crecer un árbol en un conjunto de datos de arranque:

Antes de la división, selec m $\leq$ p variables de entrada al azar para la división.

Normalmente valores para m son $\sqrt{p}$ o incluso tan bajo como 1.

Después B tales árboles $T(x;\Theta_b)_1^B$ se generá, el predictor del bosque aleatorio (regresión)

$$\hat{f}_{rf}^B = \frac{1}{B}\sum_{b=1}^{B} T(x;\Theta_b). \tag{2.11}$$

Como en la Sección anterior, Θ_b caracteriza el b_{th} árbol aleatorio en términos de variables divididas, puntos de corte en cada nodo y valores de nodo terminal.

Intuitivamente, reduciendo m reducirá la correlación entre cualquier par de árboles en el conjunto y, por lo tanto, mediante (2.10) reducirá la varianza del promedio.

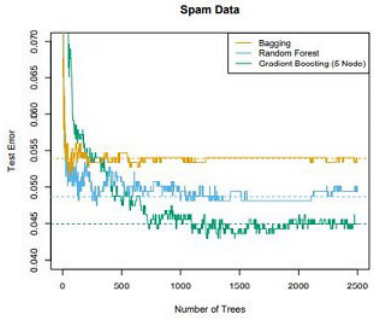


FIGURA 2.10. Bagging, Random Forest, y el gradiente boosting, aplicado a los datos de *spam*. Para impulsar,5-nodos utilizaron en los árboles, y el número de árboles fue elegido por validación cruzada de (2500 árboles). Cada paso en la figura corresponde a un cambio en una sola clasificación errónea (en un conjunto de prueba). Fuente [38]

No todos los estimadores pueden mejorarse modificando los datos de esta manera. Parece que los estimadores altamente no lineales, como los árboles, son los que más se benefician. Para árboles bootstrap, ρ es típicamente pequeño (0,05 o menos es típico; vea la Figura 2.10), mientras que σ^2 no es mucho mayor que la varianza del árbol original. Por otro lado, el bagging no cambia estimaciones de forma lineal, como la media muestral (de ahí su varianza); la correlación por pares entre las medias bootstrap es de aproximadamente el 50 %. Los bosques aleatorios son populares. Leo Breiman[3] la colaboradora Adele Cutler mantiene un sitio web[4] de Random Forest donde el software está disponible gratuitamente, con más de 3000 descargas reportadas en 2002.
Los autores hacen grandes afirmaciones sobre el éxito de los bosques aleatorios: más precisos, más interpretables y similares. En nuestra experiencia, Random Forests funciona notablemente bien, con muy poco ajuste requerido. Un clasificador de bosque aleatorio logra un error de clasificación errónea del 4,88 % en spam para los datos de prueba, que se comparan bien con todos los demás métodos, y no es significativamente peor que el aumento de gradiente al 4,5 %. Bagging alcanza un 5,4 %, que es significativamente peor que cualquiera de los dos (utilizando la prueba de McNemar), por lo que en este ejemplo parece que la aleatorización adicional ayuda.

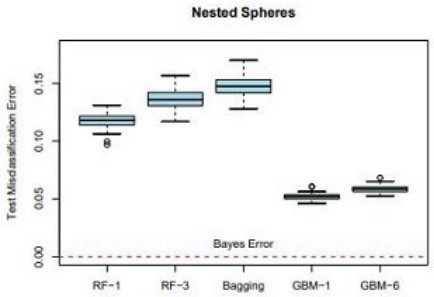


FIGURA 2.11. Los resultados de 50 simulaciones del modelo de esferas anidadas en $\mathbb{R}^{10}$. El límite de decisión de Bayes es la superficie de una esfera (aditivo). $RF-3$ se refiere a un bosque aleatorio con $m=3$, y $GBM-6$, un modelo potenciado por gradiente con orden de interacción seis; de manera similar para $RF-1$ y $GBM-1$. Fuente [38]

[3]Lamentablemente, Leo Breiman murió en julio de 2005.
[4]http://www.math.usu.edu/adele/forests/

La figura 2.11 muestra la progresión del error de prueba en 2500 árboles para los tres métodos. En este caso, hay alguna evidencia de que el aumento de gradiente ha comenzado a sobreajustarse, aunque la validación cruzada de 10 veces eligió los 2500 árboles.

En está figura adicionalmente muestra los resultados de una simulación[5] El impulso supera fácilmente a los bosques aleatorios aquí. Note que entre más pequeño m es mejor en este ejemplo, aunque parte de la razón podría ser que el límite de decisión real es aditivo.

- Los bosques aleatorios se estabilizan en alrededor de 200 árboles, mientras que en 1000 árboles el impulso continúa mejorando. El aumento se ralentiza por la contracción, así como por el hecho de que los árboles son mucho más pequeños.

2.4.4. Detalles de Random Forests

Hemos pasado por alto la distinción entre bosques aleatorios para clasificación versus regresión. Cuando se usa para la clasificación, un bosque aleatorio obtiene un voto de cada clase de árbol y luego clasifica usando el voto de la mayoría. Cuando se usa para la regresión, las predicciones de cada árbol en un punto objetivo x simplemente se promedian. Además, de los diversos estudios se hacen las siguientes recomendaciones:

- Para la clasificación, el valor preestablecido para m es $\lfloor\sqrt{p}\rfloor$ el tamaño del nodo mínimo es uno.

- Para la regresión, el valor predeterminado para m es $\lfloor p/3\rfloor$ y el mínimo tamaño es de cinco nodos.

En la práctica, los mejores valores para estos parámetros dependerán del problema y deben tratarse como parámetros de ajuste. En la Figura 2.5. $m = 6$ funciona mucho mejor que el valor predeterminado $\lfloor 8/3\rfloor = 2$.

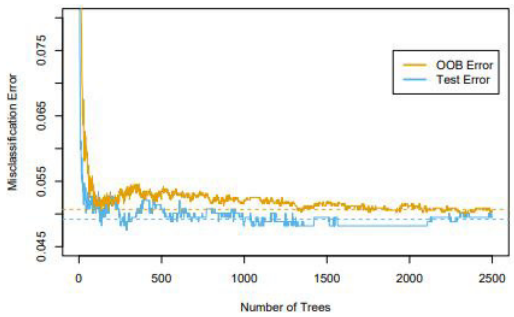


FIGURA 2.12. Error OOB calculado en una data de entrenamiento de correos no deseados, comparado con el error de prueba calculado en el equipo de test [38].

$(RFm = 6)$; una prueba de *Wilcoxon* sobre las diferencias medias en errores absolutos tiene un $p - value$ de $0,007$. Para un m grande los bosques aleatorios no funciono.

[5]Detalles: los bosques aleatorios se ajustaron usando el paquete tress Python, con 500 árboles. Los modelos de aumento de gradiente se ajustaron usando el paquete revisar, con el parámetro de contracción establecido en 0,05 y 2000 árboles.

2.4.5. Importancia de la variable

Se pueden construir parcelas de importancia variable para bosques aleatorios exactamente de la misma manera que para los modelos con aumento de gradiente. En cada división de cada árbol, la mejora en el criterio de división es la medida de importancia atribuida a la variable de división, y se acumula sobre todos los árboles del bosque por separado para cada variable.

El impulso ignora algunas variables por completo, mientras que el bosque aleatorio no. La selección de variable dividida candidata aumenta la posibilidad de que una sola variable se incluya en un bosque aleatorio, mientras que no se produce tal selección con el refuerzo. Los bosques aleatorios también usan oob muestras para construir una importancia variable medida, aparentemente para medir la fuerza de predicción de cada variable.

Cuando el *b*th árbol ha crecido, el OOB de las muestras se pasan por el árbol y se registra la precisión de la predicción. Entonces los valores para la *j*th variable se permuta aleatoriamente en el oob muestras, y la precisión se calcula de nuevo. La disminución en la precisión como resultado de esta permutación se promedia sobre todos los árboles y se usa como una medida de la importancia de la variable *j* en el bosque aleatorio.

La aleatorización anula efectivamente el efecto de una variable, muy parecido a establecer un coeficiente en cero en un modelo lineal. Esto no mide el efecto sobre la predicción si esta variable no estuviera disponible, porque si el modelo se reajustara sin la variable, se podrían usar otras variables como sustitutos.

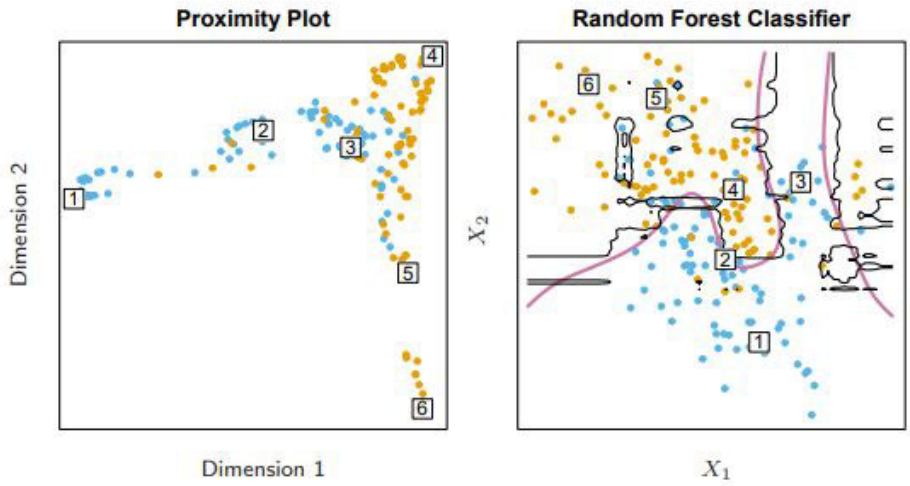


FIGURA 2.13. (Izquierda): Gráfico de proximidad para el clasificador Random Forests. (Derecha): Límite de decisión y datos de entrenamiento para bosque aleatorio sobre los datos. Se han identificado seis puntos en cada parcela. Fuente [38]

Las zonas de proximidad para Random Forests a menudo se ven muy similares, independientemente de los datos, lo que arroja dudas sobre su utilidad. Suelen tener forma de estrella, un brazo por clase, que es más pronunciada cuanto mejor es la clasificación. Dado

que el conjunto de datos es bidimensional, podemos mapear puntos desde la gráfica de proximidad a las coordenadas originales y obtener una mejor comprensión de lo que representan. Parece que los puntos en las regiones puras se asignan por clases a las extremidades de la estrella, mientras que los puntos más cercanos a los límites de decisión se asignan más cerca del centro. Esto no es sorprendente cuando consideramos la construcción de las matrices de proximidad. Los puntos vecinos en regiones puras a menudo terminarán compartiendo un depósito, ya que cuando un nodo terminal es puro, ya no lo es dividido por un algoritmo aleatorio de crecimiento de árboles forestales. Por otro lado, los pares de puntos que están cerca, pero pertenecen a clases diferentes a veces compartirán un nodo terminal, pero no siempre.

2.4.6. Bosques aleatorios y sobreajuste

Al tener un número grande de variables, pero la división de variables importantes es pequeña, es probable que los bosques aleatorios tengan un desempeño deficiente con m pequeños. En cada división, la probabilidad de que se seleccionen las variables relevantes puede ser pequeña. La figura 2.14 evidencia los resultados de una simulación que avala esta aseveración. Los detalles se encuentran la descripción de la figura. En la parte superior de cada par vemos la probabilidad hipergeométrica de que una variable relevante sea seleccionada en cualquier división por un árbol forestal aleatorio (en esta simulación, las variables relevantes son todas iguales en dimensión).

A medida que esta probabilidad se vuelve pequeña, aumenta la brecha entre el aumento y Random Forests. Cuando aumenta el número de variables relevantes, el rendimiento de los bosques aleatorios es sorprendentemente robusto a un aumento en el número de variables de ruido. Por ejemplo, con 6 variables relevantes y 100 variables de ruido, la probabilidad de que una variable relevante sea seleccionado en cualquier división es 0.46, suponiendo que $m = \sqrt{(6+100)} \approx 10$. De acuerdo con la Figura 2.14, esto no perjudica el desempeño de los bosques aleatorios en comparación con el impulso. Esta solidez se debe en gran parte a la relativa insensibilidad del costo de clasificación errónea al sesgo y la varianza de las estimaciones de probabilidad en cada árbol.

Otra aseveración es que Random Forests *no pueden sobreajustarse* a los datos. Ciertamente es cierto que aumentar B no hace que la secuencia de bosque aleatoria se sobreajuste; como el bagging, la estimación aleatoria del bosque (2.11) se aproxima a la expectativa

$$\hat{f}_{rf}(x) = E_{\Theta} T(x;\Theta) = \lim_{B\to\infty} \hat{f}(x)_{rf}^{B}.$$

con un promedio sobre B realizaciones de Θ. La distribución de aquí depende de los datos de entrenamiento. Sin embargo, este límite puede sobre ajustarse a los datos; el promedio de los árboles con un desarrollo completo da como resultado un modelo exorbitantemente bueno e incidir en una variación no necesaria, [35] explica que al controlar la profundidad uno obtiene ganancias en el rendimiento de los bosques aleatorios. La experiencia nos ha demostrado que el usar árboles adultos tiene la ventaja de darnos un costo computacional bajo, dándonos como resultado un parámetro de ajuste bajo.

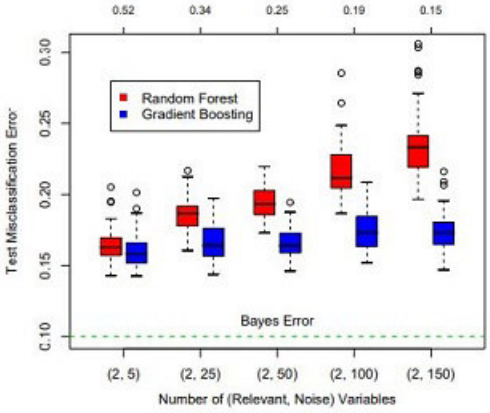


FIGURA 2.14. Una comparación de bosques aleatorios y aumento de gradiente en problemas con un número creciente de variables de ruido. Fuente [38]

En cada caso la verdadera decisión, el límite depende de dos variables y de un número creciente de variables de ruido. Los bosques aleatorios utilizan su valor predeterminado $m = \sqrt{p}$. En la parte superior de cada par la probabilidad de que se elija una de las variables relevantes en cualquier división. Los resultados se basan en 50 simulaciones para cada par, con una muestra de entrenamiento de 300 y una muestra de testeo de 500.

2.4.7. Naïve Bayes

Esta es una técnica que se a mantenido ha lo largo de los años, a pesar de su nombre. El Naïve Bayes es un algoritmo de clasificación de texto simple y efectivo que usa las probabilidades conjuntas de palabras y categorías para estimar las probabilidades de categorías dado un documento [6]. El supuesto de independencia condicional se puede expresar formalmente como:

$$P(A \mid C = c) = \prod_{i=1}^{n} P(A_i \mid C = c),$$

donde cada conjunto de atributos o conjunto de características $A = \{A_1, A_2, ..., A_n\}$ consta de n valores de atributo. Con el supuesto de independencia condicional, en lugar de calcular la probabilidad condicional de clase para cada agrupación de A, solo estime la probabilidad condicional de cada A_i, dado C. Esto hace que el enfoque Naïve Bayes sea más práctico porque no requiere un conjunto de entrenamiento muy grande para obtener una buena estimación de la probabilidad. Además, el clasificador puede manejar fácilmente los valores de atributos faltantes omitiendo la probabilidad al calcular las probabilidades de pertenencia a cada clase. Para clasificar una muestra de prueba, el clasificador NB calcula la probabilidad posterior para cada clase C como:

$$P(C \mid A) = \frac{P(C) \prod_{i=1}^{n} P(A_i \mid C)}{P(A)}. \tag{2.12}$$

La ecuación (2.12) indica que al observar el valor de una característica particular, A_i, la probabilidad previa de una categoría particular, C_i, $P(C_i)$ se puede convertir a la probabilidad posterior, $P(C_i \mid A_i)$, que representa la probabilidad de una característica en particular, A_i siendo una categoría particular, C_i. El algoritmo 5 Clasificador Bayes Naïve(NBC), que se encuentra disponible en el Apéndice 5-B, muestra cómo funciona el clasificador NB según

el modelo construido para la clasificación de phishing.

2.5. Evaluación y Selección del modelo

Para esta discusión, nos enfocamos en la regresión y la pérdida de error al cuadrado, ya que esto llega a los puntos principales, el sesgo y la varianza son más complejos con una pérdida de 0 a 1. Además, incluso en el caso de un problema de clasificación, podemos considerar el promedio de Random Forests como una estimación de las probabilidades posteriores de la clase, para las cuales el sesgo y la varianza son descriptores apropiados.

2.5.1. Varianza y efecto de descorrelación

La forma limitante $(B \to \infty)$ del estimador de regresión forestal aleatoria es

$$\hat{f}_{rf}(x) = E_{\Theta|Z} T(x;\Theta(Z)),$$

donde hemos hecho explícita la dependencia de los datos de entrenamiento Z. Aquí consideramos la estimación en un solo punto x. De 2.10 vemos que

$$Var\hat{f}_{rf}(x) = \rho(x)\sigma^2(x). \tag{2.13}$$

Donde

- $\rho(x)$ es el muestreo correlación entre cualquier par de árboles utilizados en el promedio:
$$\rho(x) = corr[T(x;\Theta_1(Z)), T(x;\Theta_2(Z))], \tag{2.14}$$
donde $\Theta_1(Z)$ y $\Theta_2(Z)$ son un par de árboles forestales extraídos aleatoriamente cultivados para el muestreo aleatorio Z;

- $\sigma^2(x)$ es la varianza muestral de cualquier árbol extraído al azar,
$$\sigma^2(x) = VarT(x;\Theta(Z)). \tag{2.15}$$

Es fácil confundir $\rho(x)$ con la correlación promedio entre árboles ajustados en un conjunto dado de bosque aleatorio; es decir, piense en los árboles ajustados como $N - vectores$, y calcular la correlación promedio por pares entre estos vectores, condicionada a los datos. Este no es el caso; esta correlación condicional no es directamente relevante en el proceso de promediado, y la dependencia en x en $\rho(x)$ nos advierte de la distinción.

Más bien, $\rho(x)$ es el valor teórico de la correlación entre un par de árboles forestales aleatorios evaluados en x, inducida haciendo repetidamente extracciones de la muestra de entrenamiento Z de la población, y luego dibujando un par de árboles al azar. En jerga estadística, esta es la correlación inducida por la distribución muestral de Z y Θ. Más precisamente, la variabilidad promediada en los cálculos en (2.14) y (2.15) es:

- condicionado a Z: debido al muestreo bootstrap y al muestreo de características en cada división, y

- como resultado de la variabilidad muestral de Z sí mismo.

De hecho, la covarianza condicional de un par de árboles se ajusta a x es cero, porque el Bootstrap y el muestreo de características son iid. En la siguiente grafica se observa las correlaciones entre pares de árboles dibujados por un algoritmo de regresión de bosque aleatorio

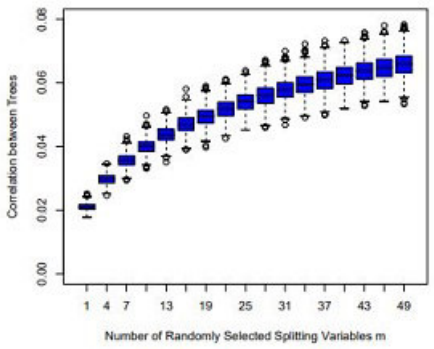


FIGURA 2.15. Correlaciones entre pares de árboles dibujados por un algoritmo de regresión de Random Forests, en función de m. Los diagramas de caja representan las correlaciones en 600 puntos de predicción elegidos al azar x. Fuente [38]

Las siguientes demostraciones se basan en un modelo de simulación

$$Y = \frac{1}{\sqrt{50}} \sum_{j=1}^{50} X_j + \varepsilon,$$

con todo el X_j y ε *iid* gaussiano. Usamos 500 conjuntos de entrenamiento de tamaño 100 y un único conjunto de ubicaciones de prueba de tamaño 600. Dado que los árboles de regresión no son lineales en Z, los patrones que vemos a continuación, diferirán algo dependiendo de la estructura del modelo.

La figura 2.15 muestra cómo la correlación (2.14) entre pares de árboles disminuye a medida que m disminuye: pares de predicciones de árboles en x para diferentes conjuntos de entrenamiento Z es probable que sean menos similares si no utilizan las mismas variables de división.

En el panel izquierdo de la Figura 2.16 consideramos las varianzas de predictores de un solo árbol, $VarT(x;\Theta(Z))$ (promedió más de 600 puntos de predicción x extraído al azar de nuestro modelo de simulación). Esta es la varianza total y puede ser descompuesto en dos partes usando argumentos de varianza condicional estándar:

$$\begin{aligned} Var_{\Theta,Z}T(x;\Theta(Z)) &= Var_Z E_{\Theta|Z}T(x;\Theta(Z)) + E_Z Var_{\Theta|Z}T(x;\Theta(Z)), \\ Varianza\_Total &= Var_Z \hat{f}_{rf}(x) + Varianza(dentro - Z), \end{aligned} \tag{2.16}$$

el segundo término es Varianza(dentro-Z): un resultado de la aleatorización, que aumenta a medida que m disminuye. El primer término es, de hecho, la varianza muestral del conjunto aleatorio (que se muestra en el panel derecho), que disminuye a medida que m disminuye. La varianza de los árboles individuales no cambia apreciablemente en gran parte del rango de m, por lo tanto, a la luz de (2.13), la varianza del conjunto es dramáticamente más baja que la varianza de este árbol.

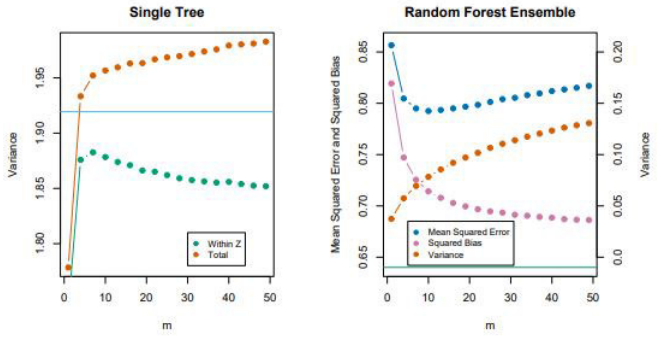


FIGURA 2.16. Resultados de la simulación. El panel de la izquierda muestra la varianza promedio de un solo árbol aleatorio, en función de m. Dentro de Z se refiere a la contribución promedio dentro de la muestra a la varianza, resultante del muestreo Bootstrap y el muestreo de variables divididas (2.13). La variabilidad muestral de Z. La línea horizontal es la varianza promedio de un solo árbol completamente desarrollado (sin muestreo bootstrap). El panel derecho muestra el error cuadrático medio promedio, el sesgo cuadrático y la varianza del conjunto, en función de m. Tenga en cuenta que el eje de varianza está a la derecha (misma escala, nivel diferente). La línea horizontal es el sesgo cuadrático medio de un árbol completamente desarrollado [38].

2.5.2. Sesgo

Al igual que en el ensacado, el sesgo de un bosque aleatorio es el mismo que el sesgo de cualquiera de los árboles muestreados individualmente $T(x;\Theta(Z))$:

$$\begin{aligned} Sesgo(x) &= \mu(x) - E_Z \hat{f}_{rf}(x), \\ &= \mu(x) - E_Z E_{\Theta|Z} T(x;\Theta(Z)). \end{aligned}$$

Esto también suele ser mayor (en términos absolutos) que el sesgo de un árbol sin podar cultivado a Z, ya que la aleatorización y el espacio muestral reducido imponen restricciones. Por lo tanto, las mejoras en la predicción obtenidas por ensacado o bosques aleatorios son únicamente como resultado de la reducción de la varianza. Cualquier discusión sobre el sesgo depende de la función verdadera desconocida. La figura 2.16 (panel derecho) muestra el sesgo al cuadrado para nuestra simulación de modelo aditivo (estimado a partir de las 500 realizaciones).
Aunque para diferentes modelos la forma y la tasa de las curvas de sesgo pueden diferir, la tendencia general es que a medida que m disminuye, el sesgo aumenta. En la figura se muestra el error cuadrático medio, y vemos un equilibrio clásico entre sesgo y varianza en la elección de m. Para todo m el sesgo al cuadrado del bosque aleatorio es mayor que el de un solo árbol (línea horizontal). Estos patrones sugieren una similitud con la regresión de crestas.

La regresión de crestas es útil (en modelos lineales) cuando se tiene una gran cantidad de variables con coeficientes de tamaño similar; *Ridge* reduce sus coeficientes hacia cero y los de las variables fuertemente correlacionadas entre sí. Aunque el tamaño de la muestra de entrenamiento podría no permitir que todas las variables estén en el modelo, esta regularización a través de la cresta estabiliza el modelo y permite que todas las variables tengan voz (aunque disminuya). Bosques aleatorios con pequeños m realizar un promedio similar. Cada una de las variables relevantes tiene su turno para ser la división primaria, y el promedio de

conjunto reduce la contribución de cualquier variable individual.

2.5.3. Validación Cruzada

Dado un conjunto de datos, se justifica el uso de un método de aprendizaje estadístico en particular si da como resultado un error de prueba bajo. El error de prueba se puede calcular fácilmente si el equipo de prueba designado está disponible. Desafortunadamente, este no suele ser el caso. Por el contrario, el error de entrenamiento se puede calcular fácilmente aplicando el método de aprendizaje estadístico a las observaciones utilizadas en su entrenamiento. Pero, la tasa de error de entrenamiento a menudo es bastante diferente de la tasa de error de prueba, y en particular el primero puede subestimar dramáticamente este último.

En ausencia de un conjunto de prueba designado muy grande que pueda usarse para estimar directamente la tasa de error de la prueba, se pueden utilizar varias técnicas estimar esta cantidad utilizando los datos de entrenamiento disponibles. Algunos métodos para hacer un ajuste matemático a la tasa de error de entrenamiento y para estimar la tasa de error de la prueba. En esta sección, consideramos una clase de métodos que estiman la tasa de error de la prueba manteniendo un subconjunto de las observaciones de entrenamiento del proceso de ajuste, y luego aplicando el método de aprendizaje estadístico a aquellas observaciones. Para esta subsección se utilizará las fuentes [3], [31] y [39]

2.5.4. El enfoque del conjunto de validación

Piense que, quisiéramos estimar el error de prueba relacionado con el ajuste en un conjunto de observaciones. La perspectiva del conjunto de validación, que se observa en la Figura 2.17, es una estrategia sencilla para esta labor. Involucra fraccionar aleatoriamente el conjunto de observaciones en dos partes, siendo estas el conjunto de entrenamiento y de validación. El modelo se ajusta en el conjunto de entrenamiento, una vez que el modelo esta ajustado este se usa para predecir las respuestas, para las observaciones que se utilizaran en el conjunto de validación.

La validación de la Tasa de error resultante establecida: de forma general se evalúa mediante MSE en el caso de tener una evaluación cuantitativa, dándonos como respuesta: la estimación de la tasa de error de la prueba. Ilustramos el enfoque del conjunto de validación en el conjunto de datos *Auto*, que parece haber una relación no lineal entre *mpg* y *horsepower*, y que un modelo que predice *mpg* usando *horsepower* y $horsepower^2$ da mejores resultados que un modelo que usa solo un término lineal. Es natural preguntarse si un ajuste cúbico o de orden superior podría proporcionar incluso mejores resultados.

Validación cruzada de K-fold

FIGURA 2.17. Una representación esquemática del enfoque del conjunto de reserva, las n observaciones se dividen aleatoriamente en un conjunto de entrenamiento (mostrado en azul, que contiene las observaciones 7, 22 y 13, entre otras) y un conjunto de validación (mostrado en beige, que contiene la observación 91, entre otros). Fuente [14]

Respondemos a esta pregunta examinando los p valores asociados con un término cúbico y un polinomio de orden superior en una regresión lineal. Pero también podríamos responder a esta pregunta usando el método de validación, dividimos aleatoriamente las 392 observaciones en dos conjuntos, un conjunto de entrenamiento que contiene 196 puntos de los datos y un conjunto de validación que contiene las 196 observaciones restantes. Las tasas de error del conjunto de validación que resultan de ajustar varios modelos de regresión en la muestra de entrenamiento y evaluando su desempeño en la muestra de validación, utilizando MSE como la medida del error del conjunto de reserva, se visualiza en el panel de la izquierda de la Figura 2.18.

El conjunto de validación MSE para el ajuste cuadrático es considerablemente menor para el ajuste lineal. Sin embargo, el conjunto de validación MSE para el ajuste cúbico es en realidad un poco más grande que para el ajuste cuadrático. Esto implica que incluir un término cúbico en la regresión no conduce a una mejor predicción que simplemente usar un término cuadrático. Recuerde que para crear el panel de la izquierda de la Figura 2.18, dividimos aleatoriamente el conjunto de datos en dos partes, un conjunto de entrenamiento y un conjunto de validación. Si repetimos el proceso de dividir aleatoriamente el conjunto de muestras en dos partes, obtendremos una estimación algo diferente para el MSE de prueba. como en la ilustración, el panel de la derecha de la Figura 2.17 muestra diez curvas de MSE de conjunto de validación diferentes del conjunto de datos *Auto*, producidas usando diez divisiones aleatorias de las observaciones en conjuntos de entrenamiento y validación, las diez curvas indican que el modelo con un término cuadrático tiene un dramáticamente conjunto de validación más pequeño MSE que el modelo con solo un término lineal.

Además, las diez curvas indican que no hay mucho beneficio en incluir términos polinómicos cúbicos o de orden superior en el modelo. Pero vale la pena señalar que cada una de las diez curvas da como resultado una estimación de MSE de prueba diferente para cada uno de los diez modelos de regresión considerados. Y no hay consenso entre las curvas en cuanto a qué modelo da como resultado el conjunto de validación más pequeño MSE. En base a la variabilidad entre estas curvas, todo lo que podemos concluir con confianza es que el ajuste lineal no es adecuado para estos datos. La perspectiva del conjunto de reserva es teóricamente sencillo y fácil de implementar. Pero presenta dos inconvenientes fuertes:

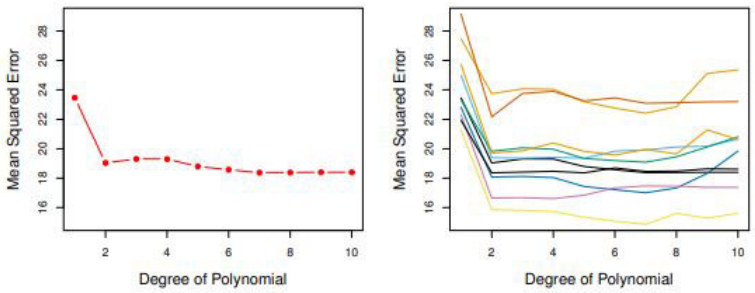


FIGURA 2.18. La perspectiva del conjunto de reserva se utilizó en el conjunto de datos *Auto* para estimar el error de prueba, que resulta de predecir *mpg* usando funciones polinomiales de los caballos de fuerza. Izquierda: Son las estimaciones del error de validación para una sola división para los conjuntos de entrenamiento y validación. Derecha: El método de validación se repitió diez veces, cada vez usando una división aleatoria diferente de las observaciones en un conjunto de entrenamiento y un conjunto de validación. Esto ilustra la variabilidad en el MSE de prueba estimado que resulta de esta perspectiva. Fuente [14]

1. Como se muestra en el panel derecho de la Figura 2.18, la estimación de validación de la tasa de error de la prueba puede ser muy variable, y esto depende de las observaciones que se incluyen en el conjunto de entrenamiento y las observaciones se incorporan en el conjunto de reserva.

2. En la perspectiva de validación, solo una submuestra de las observaciones que se incorporaron en el conjunto de entrenamiento y que no se incluyeron en el conjunto de reserva: se utilizan para ajustar el modelo, dado que los métodos estadísticos tienden a funcionar peor cuando se entrenan con menos observaciones, esto sugiere que la tasa de error en el conjunto de validación podría presentar una sobrestimación en la tasa de error de prueba para el modelo que se ajusta a toda la base de datos.

En las próximas subsecciones, presentaremos la validación cruzada, tomando en cuenta el refinamiento del enfoque del conjunto de validación que aborda estos dos problemas.

2.5.5. Validación cruzada Leave-One-Out

Para la validación cruzada que deja uno fuera (LOOCV) es estrechamente relacionada con el enfoque del conjunto validación de la subsección anterior, pero intenta abordar el enfoque de los inconvenientes de ese método. Al igual que la perspectiva del conjunto de validación, LOOCV involucra dividir el conjunto de observaciones en dos partes. No obstante, en vez de construir dos subconjuntos del mismo tamaño, se usa una observación (x_1, y_1) para el conjunto de reserva, y los restantes $\{(x_2, y_2), ..., (x_n, y_n)\}$ componen el conjunto de entrenamiento. El método se ajusta a las $n-1$ observaciones del entrenamiento, para posteriormente hacer una predicción $\hat{y}_1$ para la observación apartada, utilizando su valor x_1.

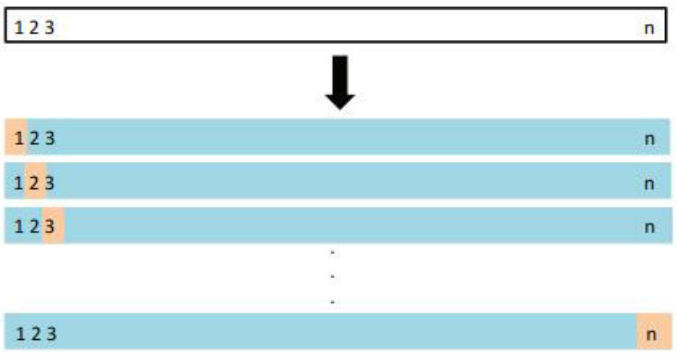


FIGURA 2.19. Una visualización esquemática de LOOCV. Un conjunto de n puntos se fracciona de forma recurrente en un conjunto de entrenamiento (mostrado en azul) que contiene todas las observaciones excepto una, y el conjunto de validación que solo tiene una observación (mostrado en beige). Luego se estima el error de prueba promediando los n resultantes MSE. El primer conjunto de entrenamiento, incluye todo menos la observación 1, el segundo conjunto de entrenamiento incluye todo menos la observación 2, y así consecutivamente. [14].

Dado que (x_1, y_1) no se usó en el proceso de ajuste, $MSE_1 = (y_1 - \hat{y}_1)^2$ proporciona una estimación aproximadamente imparcial del error de prueba. Pero a pesar de que MSE_1 no está sesgado por el error de prueba, es una estimación pobre porque es muy variable, dado que solo tiene una sola observación (x_1, y_1).

Se va a repetir el proceso elegido (x_2, y_2) para los datos de validación, entrenando el proceso en las $n-1$ observaciones $\{(x_1, y_1), (x_3, y_3), ..., (x_n, y_n)\}$, para después calcular $MSE_2 = (y_2 - \hat{y}_2)^2$. Reproducir esta perspectiva n veces da como resultado n errores al cuadrado, $MSE_1, ..., MSE_n$. La estimación LOOCV para el MSE de prueba es la media de estos n errores estimados:

$$CV_{(n)} = \frac{1}{n} \sum_{i=1}^{n} MSE_i.$$

En la Figura 2.19 se visualiza un esquema del enfoque LOOCV, presenta unas ventajas significativas acerca del enfoque del conjunto de reserva. Antes de nada, presenta menos sesgo. En LOOCV, se ajusta el método de forma repetitiva usando los conjuntos de entrenamiento que incluyen $n-1$ observaciones, pudiendo notar que son casi todas las observaciones del conjunto original. A comparación de la perspectiva del conjunto de validación, donde la base de entrenamiento contiene el 50 % de la base total. Por consiguiente, el enfoque LOOCV no sobrestima la tasa de error, a diferencia que lo hace en el enfoque del conjunto de reserva. Como segundo punto, en comparación de la perspectiva de validación que creará resultados diferentes al aplicar el proceso en forma repetitiva, esto se debe a la aleatoriedad en la base de entrenamiento/validación se divide, LOOCV diversas veces y si da los mismos resultados: quiere decir que la aleatoriedad no se está aplicando en la división de la base de entrenamiento/validación.

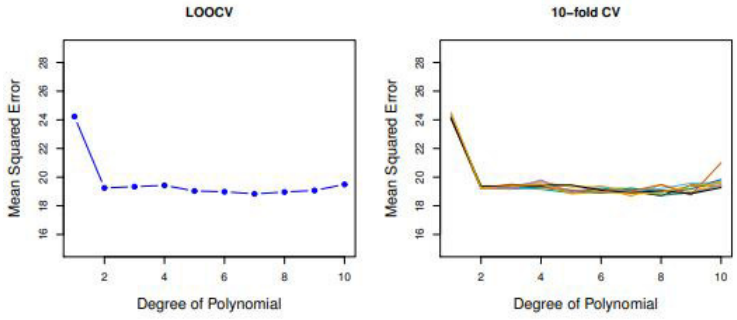


FIGURA 2.20. Se usó validación cruzada en el conjunto de datos *Auto* para estimar el error de prueba que resulta de predecir *mpg* usando funciones polinómicas de caballos de fuerza. Izquierda: La curva de error LOOCV. Derecha: CV de 10 veces se ejecutó en nueve tiempos separados, cada uno con una división aleatoria diferente de los datos en diez partes. En la figura se observa las nueve curvas de error CV con una ligera diferencia. [14].

Usamos LOOCV en el conjunto de datos *Auto* para obtener una estimación del conjunto de prueba MSE que resulta de ajustar un modelo de regresión lineal a predecir millas por galón usando funciones polinómicas de caballos de fuerza. Los resultados se observan en la Figura 2.20 del lado izquierdo. LOOCV tiene el potencial de ser costoso de implementar, ya que el modelo tiene que estar en forma n veces. Esto puede llevar mucho tiempo si n es grande y si cada modelo individual es lento para adaptarse. Con mínimos cuadrados lineales o regresión polinomial, un atajo increíble hace que el costo de LOOCV sea el mismo que de un solo modelo en forma. Se cumple la siguiente fórmula:

$$CV_{(n)} = \frac{1}{n} \sum_{i=1}^{n} \left(\frac{y_i - \hat{y}_i}{1 - h_i}\right)^2, \tag{2.17}$$

donde $\hat{y}_i$ es el $i-simo$ valor ajustado del ajuste original por mínimos cuadrados, e h_i es el apalancamiento. Esto es como el MSE ordinario, excepto que el $i-simo$ residuo se divide por $1-h_i$. El apalancamiento se encuentra entre $1/n$ y 1, y refleja la cantidad en que una observación influye en su propio ajuste.

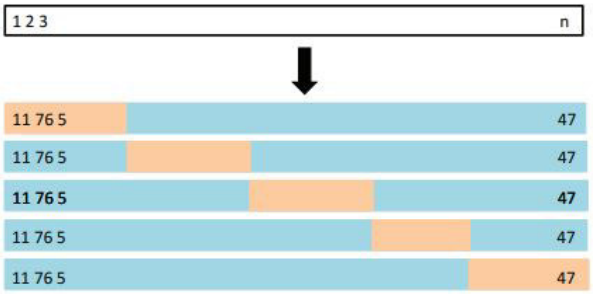


FIGURA 2.21. Una pantalla esquemática de CV de 5 veces. Un conjunto de n observaciones se divide al azar en cinco grupos que no se sobreponen. Cada conjunto es un grupo para la validación (mostrado en beige), y el restante como entrenamiento (mostrado en azul). El error de prueba se estima realizando el promedio de las cinco estimaciones MSE resultantes. [14].

Por lo tanto, los residuos de los puntos de alto apalancamiento están inflados en esta fórmula por exactamente la cantidad correcta para que se mantenga esta igualdad. LOOCV es un proceso muy general y se puede usar para todo tipo de modelado predictivo. Por ejemplo, podríamos usarlo con regresión logística o análisis discriminante lineal. La fórmula mágica (2.17) no se cumple en general, en cuyo caso el modelo tiene que ser reajustado n veces.

2.5.6. Validación cruzada de k-Fold

Una opción a LOOCV es la validación cruzada de *k-fold*. La perspectiva consiste en fraccionar aleatoriamente k veces el conjunto de observaciones en k grupos, del mismo tamaño. El primer *fold* se trata como un conjunto de reserva, y el método se ajusta a los $k-1$ *fold* restantes. El error cuadrático medio, MSE_1, luego se calcula sobre las observaciones en el *fold* retenido. Este procedimiento es repetido k veces; en cada iteración se tiene un conjunto diferente como un conjunto de reserva. El procesamiento da como resultado k estimaciones del error de prueba, $MSE_1, MSE_2, ..., MSE_k$. La estimación de la CV de k veces se lo calcula como el promedio de estos valores,

$$CV_{(k)} = \frac{1}{k}\sum_{i=1}^{k} MSE_i.$$

La figura 2.21 ilustra el enfoque CV de k veces. Es sencillo observar que LOOCV es un caso particular de *k-fold* en el que k y n se establece de manera idéntica. La experiencia dice que, generalmente se efectúa *k-fold* utilizando $k = 5$ o $k = 10$. Esto se realiza por la ventaja computacional. LOOCV precisa ajustar el método n veces. Dando un proceso de costo computacional elevado (excepto para modelos lineales ajustados por mínimos cuadrados, en cuyo caso se puede utilizar la fórmula (2.17). Donde CV tiene una perspectiva habitual que se puede emplear a casi cualquier método. Algunos métodos tienen un uso computacional agotador, como: procedimientos de ajuste, por lo tanto, usar LOOCV puede dar problemas en el ámbito computacional, de manera especial si n es demasiado grande.

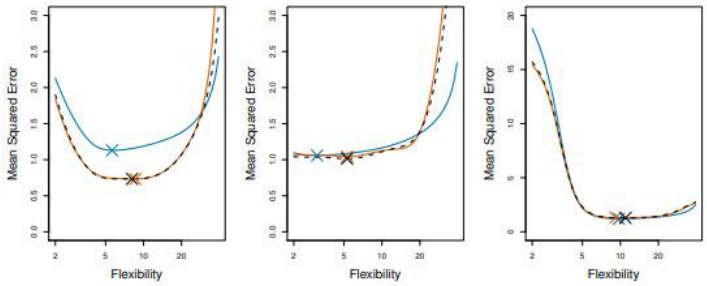


FIGURA 2.22. MSE de prueba verdadera y estimada para los conjuntos de datos simulados en las Figuras de la (izquierda) 2,9, del (centro) 2,10 y la (derecha) 2,11. La verdadera prueba MSE se muestra en azul, la estimación LOOCV se muestra como una línea discontinua negra y el CV de 10 veces la estimación se muestra en naranja. Las cruces indican el mínimo de cada uno de las curvas MSE. Fuente [14]

Por el contrario, realizar 10 veces CV requiere ajustar el proceso de aprendizaje diez veces, lo que es mucho más práctico. Como observamos, existen otras ventajas que no son tiene que ver con el costo computacional, para ejecutar CV de 5 o 10 veces, que implican el compromiso sesgo-varianza. El panel de la derecha de la Figura 2.20 muestra nueve CV diferentes de 10 veces, estimaciones para el conjunto de datos *Auto*, cada uno resultante de un proceso aleatorio diferente en la división de las observaciones en diez Fold. Como podemos ver en la figura, hay cierta variabilidad en las estimaciones de CV como resultado de la variabilidad en cómo las observaciones se dividen en diez Fold. Pero esta variabilidad es típicamente mucho menor que la variabilidad en las estimaciones de error de prueba que resulta del enfoque del conjunto de validación (panel derecho de la Figura 2.18).

Cuando examinamos datos reales, no conocemos el verdadero MSE de prueba, y por lo que es difícil determinar la precisión de la estimación de validación cruzada. Sin embargo, si examinamos los datos simulados, podemos calcular la verdadera prueba MSE, y por lo tanto puede evaluar la precisión de los resultados de nuestra validación cruzada. En la Figura 2.22, trazamos las estimaciones de validación cruzada y la prueba verdadera de tasas de error que resultan de aplicar splines de suavizado al simulado de conjuntos de datos ilustrados. La verdadera prueba MSE se muestra en azul. Las líneas discontinuas negras y naranjas sólidas respectivamente muestren el LOOCV estimado y las estimaciones de CV de 10 veces. En las tres regiones, las dos estimaciones de CV son muy parecidas. En el panel de la derecha de la Figura 2.22, el verdadero MSE de prueba y las curvas de validación cruzada son casi idénticos. En el panel central de la 2.22, los dos conjuntos de curvas son similares en los grados más bajos de flexibilidad, mientras que las curvas CV sobrestiman la prueba MSE para obtener mayores grados de flexibilidad. En el panel izquierdo de la Figura 2.22, las curvas CV tienen la forma general correcta, pero subestiman la prueba verdadera MSE.

Cuando realizamos una validación cruzada, nuestro objetivo podría ser determinar que tan bien se puede esperar que un determinado procedimiento de aprendizaje estadístico funcione en datos independientes; en este caso, la estimación verdadera del MSE de prueba es importante. Pero en otras ocasiones sólo nos interesa la ubicación del punto mínimo en la curva MSE de prueba estimada. Esto se debe a que podría estar realizando una validación cruzada en una serie de métodos de aprendizaje estadístico, o en un solo método usando diferentes niveles de flexibilidad, para identificar el método que da como resultado el error de prueba más bajo. Para este propósito, la ubicación del punto mínimo en la curva MSE de prueba estimada es importante, pero el valor real del MSE de prueba estimado no lo es.

Encontramos en la Figura 2.22 que a pesar de que a veces subestiman la verdadera prueba MSE, todas las curvas de CV se acercan a identificar el correcto nivel de flexibilidad, es decir, el nivel de flexibilidad correspondiente a la menor prueba MSE.

2.5.7. Compensación de sesgo y varianza para la validación cruzada de k-Fold

Mencionamos que CV de *k-fold* con $k < n$ tiene una ventaja computacional para LOOCV. Sin pesar en los problemas del enfoque computacional, una ventaja poco perceptible, pero de mayor importancia de *k-fold* es que brinda estimaciones precisas de la tasa de error de prueba a comparación de LOOCV. La razón es que el sesgo-varianza son compensados. El enfoque de la base de validación nos dirige a sobreestimaciones en la tasa de error de prueba, ya que en esta perspectiva en el conjunto de entrenamiento se usa para ajustarse al método de aprendizaje estadístico que incluye la mitad las observaciones de todo el conjunto original. Con este razonamiento, es sencillo observar que LOOCV brindara estimaciones prácticamente ecuánimes del error de prueba, dado que cada conjunto de entrenamiento incluye $n-1$ observaciones, que tienden a ser todo el conjunto original. Y realizando CV de *k-fold* para, $k = 5$ o $k = 10$ nos dirigirá a un sesgo promedio, dado que cada conjunto de entrenamiento incluye alrededor de $(k-1)n/k$ observaciones, menos que en el enfoque LOOCV, pero más que en la perspectiva de conjunto de validación. Por esta razón, desde el enfoque de reducción del sesgo, es obvio que LOOCV debe escogerse en vez de *k-fold*.

No obstante, el sesgo no es la en lo que debemos preocuparnos en un proceso de estimación; hay que considerar la varianza del procedimiento. Sabemos que LOOCV tiene una varianza más alta que CV de *k-fold* con $k < n$. Porque es este el caso. Cuando ejecutamos LOOCV, promediamos las salidas de n modelos acoplados, donde cada conjunto está entrenado en un grupo casi parecido de observaciones; por lo tanto, los resultados tienen una correlación positiva alta entre sí. En cambio, cuando ejecutamos *k-fold* con $k < n$, se procede a promediar las salidas de k modelos adaptados que su correlación no es tan alta entre sí, ya que la sobreposición entre los grupos de entrenamiento es pequeña en cada modelo. Puesto que el promedio de cantidades con una correlación alta, presentara una varianza mucho más alta que la media de varias cantidades que no tienen una correlación alta, la estimación del error de prueba resultante de CV de *k-fold* tiende a tener una varianza más baja, que la estimación resultante de LOOCV.

En resumen, el sesgo-varianza tiene una compensación ligada con la selección de k en la CV de *k-fold*, generalmente al tener estas consideraciones, uno realiza una validación cruzada de k veces usando $k = 5$ o $k = 10$, producen un sesgo y varianza baja en las estimaciones de tasa de error de prueba.

2.5.8. Validación cruzada en problemas de clasificación

Hasta ahora, en esta sección, hemos ilustrado el uso de la validación cruzada en el ajuste de regresión donde el resultado Y es cuantitativo, por lo que han usado MSE para cuantificar el error de prueba. Pero la validación cruzada también puede ser un enfoque muy útil en el entorno de clasificación cuando Y es cualitativo. En esta configuración. La validación cruzada funciona tal como se describió anteriormente, excepto que en lugar de usar MSE para cuantificar el error de prueba, en su lugar usamos el número de observaciones clasificadas de manera errónea. Por ejemplo, en el marco de la clasificación, en la tasa de error LOOCV toma la forma

$$CV_{(n)} = \frac{1}{n}\sum_{i=1}^{n} Err_i,$$

donde $Erri = I(y_i \neq \hat{y}_i)$. La tasa de error CV de k-fold y el error del conjunto de validación, las tasas se definen de manera análoga. *i.e.*, ajustamos varios modelos de regresión logística en los datos de clasificación bidimensional. En el tablero superior izquierdo de la Figura 2.23, la línea continua negra muestra el límite de decisión estimado que resulta de adaptar un modelo de regresión logística estándar a estos registros. Dado que se trata de datos simulados, podemos calcular la verdadera tasa de error de prueba, que toma un valor de 0,201 y por lo tanto es sustancialmente mayor que la tasa de error de Bayes de 0,133. Claramente, la regresión logística no tiene suficiente flexibilidad para modelar el límite de decisión de Bayes en este entorno. Podemos fácilmente extender la regresión logística para obtener un límite de decisión no lineal usando funciones polinómicas de los predictores, como hicimos en el escenario de regresión. *i.e.*, podemos ajustar un modelo de regresión logística cuadrática, dada por

$$\log\left(\frac{p}{1-p}\right) = \beta_0 + \beta_1 X_1 + \beta_2 X_1^2 + \beta_3 X_2 + \beta_4 X_2^2.$$

El panel superior derecho de la Figura 2.23 muestra el límite de decisión resultante, que ahora es curvo. Sin embargo, la tasa de error de prueba ha mejorado solo ligeramente, a 0,197. Una mejora mucho mayor es evidente en el panel inferior izquierdo de la figura 2.23, en la que hemos ajustado un modelo de regresión logística que implica polinomios cúbicos de los predictores.

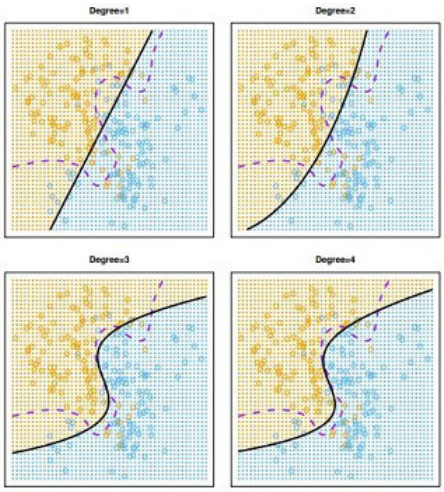


FIGURA 2.23. La regresión logística se ajusta a los datos de clasificación bidimensional. El límite de decisión de Bayes se representa mediante una línea discontinua púrpura. Límites de decisión estimados lineal, cuadrático, cúbico y las regresiones logísticas cuárticas (grados 1 a 4) se muestran en negro, el error de la prueba las tasas para los cuatro ajustes de regresión logística son respectivamente 0,201, 0,197, 0,160 y 0,162, mientras que la tasa de error de Bayes es 0,133. [14].

Ahora la tasa de error de prueba ha disminuido a 0,160. Ir a un polinomio cuártico (abajo a la derecha) aumenta ligeramente el error de la prueba. En la práctica, para datos reales, se desconocen el límite de decisión de Bayes y las tasas de error de prueba. Entonces, ¿cómo podemos decidir entre los cuatro métodos logísticos? Los modelos de regresión que se muestran en la figura 2.23, podemos usar la validación cruzada para tomar esta decisión. El panel de la izquierda de la Figura 2.24 muestra en negro las tasas de error CV de 10 veces

que resultan de ajustar diez modelos de regresión logística a los datos, usando funciones polinómicas de los predictores arriba del décimo orden.

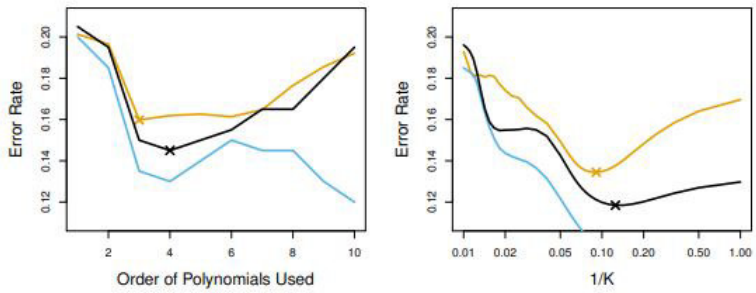


FIGURA 2.24. Error de prueba (marrón), error de entrenamiento (azul) y error CV de 10 veces (negro) en los datos de clasificación bidimensional que se muestran en la Figura 2.23. Izquierda: Regresión logística utilizando funciones polinómicas de los predictores. El orden de los polinomios utilizados se muestra en el eje x. Derecha: El clasificador KNN con diferentes valores de K, el número de vecinos utilizados en el clasificador KNN. Fuente [14]

Los verdaderos errores de prueba se muestran en marrón, y en el entrenamiento los errores se muestran en azul. Como hemos visto anteriormente, el error de entrenamiento tiende a disminuir a medida que aumenta la flexibilidad del ajuste. (La figura indica que, aunque la tasa de error de entrenamiento no disminuye monótonamente, tiende a disminuir en general a medida que aumenta la complejidad del modelo). En cambio, el error de prueba muestra una forma de U-característica. El CV de 10 veces en la tasa de error proporciona una aproximación bastante buena a la tasa de error de prueba.

Si bien subestima un poco la tasa de error, alcanza un mínimo cuando se utilizan polinomios de cuarto orden, que está muy cerca del mínimo de la curva de prueba, lo que ocurre cuando se utilizan polinomios de tercer orden. De hecho, el uso de polinomios de cuarto orden probablemente conduciría a una buena prueba, establecer el rendimiento, ya que la verdadera tasa de error de prueba es aproximadamente la misma para polinomios de tercer, cuarto, quinto y sexto orden.

El panel de la derecha de la Figura 2.24 muestra las mismas tres curvas utilizando el enfoque KNN para la clasificación, en función del valor de K (que en este contexto indica el número de vecinos utilizados en el clasificador KNN, en lugar del número de CV de *Fold* utilizados). De nuevo la tasa de error en el entrenamiento disminuye a medida que el método se vuelve más flexible, por lo que vemos que la tasa de error de entrenamiento no se puede utilizar para seleccionar el valor óptimo de K. Aunque la curva de error de validación cruzada subestima ligeramente la prueba tasa de error, toma un mínimo muy cercano al mejor valor para K.

2.6. Evaluación de desempeño del modelo de clasificación

Para finalizar, en esta sección se analiza las medidas que existen para evaluar el rendimiento del procesamiento para los problemas de clasificación. Lo que pretendemos es cuantificar la calidad del modelo, y posteriormente para efectuar la comparación entre varios modelos. Para evaluar los modelos de clasificación, las medidas de rendimiento se calculan equiparando las predicciones resultantes de la modelización para la base de validación. Por este razonamiento, explicaremos las siguientes medidas, descritas por [15].

2.6.1. Matriz de Confusión

La matriz de confusión es una medida que se utiliza para resolver problemas de clasificación. Puede aplicarse tanto a la clasificación binaria como a los problemas de clasificación multiclase. Un ejemplo de una matriz de confusión para la clasificación binaria se muestra en la figura 2.25.

		Predicción	
		Positivos	**Negativos**
Observación	**Positivos**	Verdaderos Positivos (**TP**)	Falsos Negativos (FN)
	Negativos	Falsos Positivos (FP)	Verdaderos Negativos (**TN**)

FIGURA 2.25. Matriz de Confusión

Las matrices de confusión representan recuentos de valores de predicción y reales. Consta de cuatro características básicas (números) que se utilizan para definir las métricas de medición del clasificador. Estos cuatro números son:

1. **TP (Verdadero Positivo):** TP indica el número de positivos clasificados con precisión.
2. **TN (Verdaderos Negativo):** TN que muestra el número de negativos clasificados con precisión.
3. **FP (Falso positivo):** FP es el número de negativos reales clasificados como positivos. FP también es conocido como error de tipo I.
4. **FN (Falso negativo):** FN significa un valor falso negativo que es el número de positivos reales clasificados como negativos. FN también es conocido como error de tipo II.

Las métricas de rendimiento de un algoritmo son exactitud, precisión, recuperación y puntuación F1, que se calculan sobre la base de los valores TP, TN, FP y FN indicados anteriormente.

Una métrica muy utiliza al realizar la clasificación es la exactitud. La exactitud de un modelo (a través de una matriz de confusión) se calcula utilizando la fórmula dada a continuación.

$$Accuracy = \frac{(TP+TN)}{TP+FP+FN+TN}.$$

La precisión son los casos positivos predichos correctamente por el clasificador. Se mide mediante la ecuación dada:

$$Precisin = \frac{(TN)}{FN+TN}.$$

La métrica de sensibilidad se calcula tomando la proporción de entradas positivas identificadas correctamente. Es la tasa de TP y se mide mediante la ecuación dada:

$$Recall = \frac{(TN)}{FP+TN}.$$

La puntuación F1 o conocida como la medida F, es una medida de la precisión de la prueba. Se define como una media ponderada de precisión y recuerdo(sensibilidad). Tiene su valor máximo en 1 y el peor en 0.

$$F_1 score = \frac{2\text{xPrecisionxRecall}}{\text{Precisión+Recall}}.$$

2.6.2. Curva ROC

La curva ROC[6] representa la sensibilidad en función de 1-especificidad para todos los valores umbral posibles del marcador estudiado. La sensibilidad es la capacidad para detectar bien y la especificidad es la capacidad para detectar bien a los que no son. La figura 2.26 muestra una curva ROC típica.

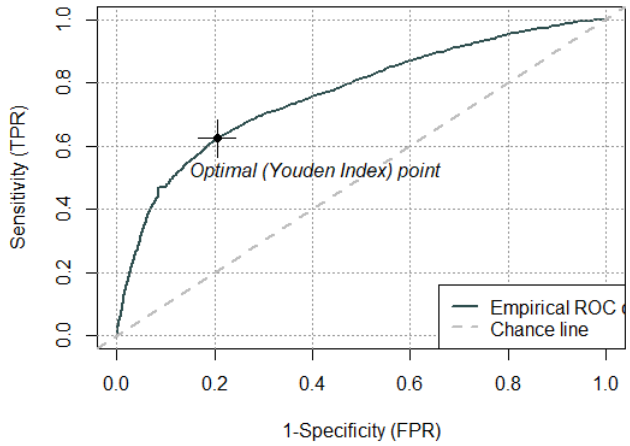


FIGURA 2.26. Curva ROC

Para calcular los puntos en una curva ROC, se lo hace de forma eficiente con un algoritmo basado en clasificación que proporciona esta información, llamado AUC.

2.6.3. AUC: área bajo la curva ROC

El AUC se puede interpretar como la probabilidad de que, entre dos correos para detección de phishing elegidos al azar, uno tenga presencia de phishing y el otro no. Así, un AUC de 0,5 (50 %) indica que la etiqueta no es informativa. Un aumento en AUC indica una mejora en las habilidades discriminatorias, con un máximo de 1.0 (100 %).

El AUC es recomendable por estas razones:

- El AUC es no varía en la escala. Mide que tan buenas son las predicciones, en lugar de sus valores absolutos.
- El AUC no varía en el umbral de clasificación. Mide la calidad de las predicciones, independientemente del umbral de clasificación del modelo elegido.

[6]ROC:Receiver Operating Characteristic

Capítulo 3
MARCO METODOLÓGICO

En este capítulo nos centraremos en la segunda etapa de la metodología CRISP-DM que se basa en la recopilación de datos para poder abordar el problema, como siguiente paso el análisis de mismos para llegar a verificar la calidad de los datos, en la siguiente etapa se procede a la construcción del vector de características esenciales para la generación del modelo. Describir la metodología empleada es una parte fundamental en la construcción del modelo predictivo, el cual tiene el propósito de detectar y moderar ataques cibernéticos en particular (Phishing) en correos electrónicos. Como fuente principal para este capítulo, véase, [27], [34] y [41].

Se ha utilizado el método Feature Selection, que permite preprocesar las características de las URL_S y correos electrónicos, teniendo una peculiaridad que es imputar los innecesarios sin correr el riesgo de pérdida de información. Además, se utilizó Random Forests y Naive Bayes para la construcción del vector de aprendizaje automático. A continuación, en la figura 3.1 se ilustra el diagrama de flujo del modelo a realizar.

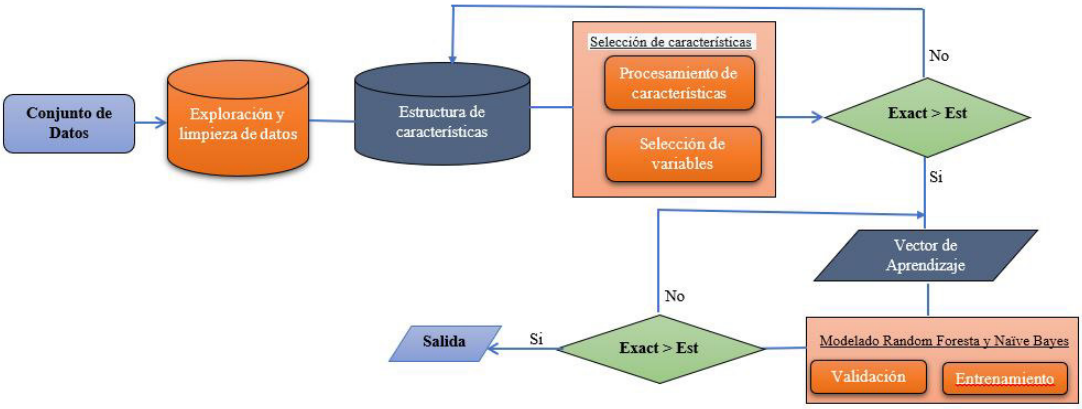


FIGURA 3.1. Diagrama de flujo del modelo de detección y mitigación de phishing

Es de vital importancia mencionar que para este proyecto se seleccionó los lenguajes de programación Java para la construcción del vector de características y el software Python para la implementación de los modelos predictivos, la razón de por qué se escogió estos lenguajes es porque proporciona librerías eficientes en la manipulación y tratamiento de este tipo de registros, ya que, al ser softwares relativamente libres y multiplataformas, podemos extraer y procesar los datos de una manera sencilla, razón por la cual se eligió estos lenguajes de programación.

En la siguiente sección se va a describir los datos descargables con que se trabajará para el proyecto de investigación.

3.1. Descripción del conjunto de datos

Se utilizan tres conjuntos de datos que se encuentran disponibles al público, estas bases contienen URL malignas y reales, correos con presencia de phishing y correos benignos para

evaluar el rendimiento de la arquitectura de detección de phishing propuesta. Los conjuntos experimentales se obtienen de un corpus de datos extraídos de [29], [33] y [40].

El número de datos recolectados en cada conjunto es: 14200 en PhishTank, 3700 en Monkey y para Enron se tiene 16800, dándonos un total general de 31084 estos datos posteriormente se los tratara para obtener una data limpia lista para realizar análisis y visualizar algunos resultados.

La información de las bases de datos y la descripción de las variables de las mismas se encuentran disponibles en el Apéndice C, ahora bien, estas tres fuentes de datos se eligen porque contienen conjuntos de datos verificados que se utilizan en la mayoría de las investigaciones anti-phishing para comparar los resultados de la evaluación. Como la vida de los sitios de phishing suele ser breve, rastreamos solo 14200 URL y de correos entre infectados/reales 20500 obteniendo un conjunto de datos de disponible con fines experimentales. Estos sitios fueron seleccionados al azar de todo el corpus de datos de investigación con fin de experimentar y evaluar. Cabe mencionar que cada entrada del corpus de datos experimentado es única.

Por lo tanto, el corpus consta de sitios populares comúnmente dirigidos a ataques de phishing. Estos sitios son fácilmente dirigidos porque los usuarios desprevenidos son engañados por la apariencia de cualquier página de phishing que imite páginas genuinas. Un simple ataque homomórfico usando un carácter similar puede hacer que este sea posible. Mediante una revisión de literatura, acerca de la problemática presentada en Ecuador sitios incluyen sitios de comercio electrónico conocidos, casas financieras, sitios de redes sociales, agencias gubernamentales, etc. Estos sitios muestran una representatividad alta en la suplantación de identidad.

3.2. Tratamiento de datos

Ya habiendo identificado los datos recogidos, seguimos con la exploración y análisis de datos, que se lleva a cabo con el uso de técnicas estadísticas, que nos dirán si los datos extraídos son válidos. En esta etapa se recomienda desarrollar tablas y gráficos para una visualización adecuada de los resultados.

Esta fase sirve para identificar si los datos recogidos tienen la calidad necesaria y con ello podamos cumplir con el fin establecido. La etapa de exploración se hizo a través de búsquedas básicas de los Dataset con extensión M.box[1] y CSV.

3.2.1. Depuración de Datos

En esta parte del documento se procede a preparar los datos para posteriormente utilizarlos en los modelos de clasificación. Hay que tener en cuenta que esta fase es una de las más importante del proceso dado que mediante este, debido a que se necesita unir y elegir de forma correcta los datos extraídos que serán procesados. Se inició con la limpieza del conjunto de datos que se obtuvieron y que tienen la información que corresponde a correos electrónicos con presencia de phishing y sin ella, de este Dataset se eligió un conjunto de muestra para limpiar y de esta manera tener un proceso de extracción de características de calidad en el formato que requerimos para tener éxito en su procesamiento al momento de modelar.

[1] **M.box** es un término común para un conjunto de formatos que se usan para guardar conjuntos de correos electrónicos. Todos los mensajes de un buzón de correo (mailbox) quedan unidos en un archivo único.

3.2.2. Limpieza de Datos

Los documentos descargables son los que contienen las URL y correos electrónicos con Phishing y sin ello, por ello no requiere de una discriminación entonces se lleva a cabo un proceso aparte, debido a que, aunque se cuenta con Datasets completamente diferentes en sus categorías, y sus variables son parecidas. Recordando que el formato tiene una extensión M.box, que tiene una estructura determinada para la cabecera. Se eligió estos datos para hacer el análisis.

Nota: Hay que tener en cuenta que el procesamiento en el que extraemos las características es idéntico en las dos datas.

1). header['From']

2). header['Subject']

3). header ['Content-Type']

4). header['Date']

5). header['Body']

Siguiendo con este proceso limpiamos los datos únicamente para las variables seleccionadas, ya que la información contenida en estas columnas puede contener información no adecuada en su estructura o no contener información en algunos de los campos, es por esta razón que es lo primero que se va a corregir analizando el porcentaje de valores nulos. Es así que se tiene los siguientes resultados.

Bases	Valores nulos	Porcentaje de Na_s
PhishTank	838	5,9 %
Monkey	400	10,8 %
Enron	1680	10 %

CUADRO 3.1. Porcentaje de Na_s

Observamos en el cuadro 3.1, que el porcentaje de valores nulos es aceptable dado que solo representa el 10 %, por tanto, no presenta un riesgo al momento de tener un resultado. Para los patrones de información que no aportan al estudio, se requiere de un tratamiento especial, notando que depende en gran medida de la finalidad del análisis y de la fuente de la que proceda el texto. En este caso el uso elevado de abreviaturas y signos de puntuación, nos indica que la mejor manera es tokenizar, se usó la librería *re-Regular expression operation* del software libre Python, ésta nos deja suprimir partes determinadas del archivo sin cambiar el origen de los datos, pero si su constitución. Los patrones que se eliminó para este experimento son:

- **Espacios en blanco**
- **Tabulaciones**
- **Salto de línea**

En el cuadro 3.2. se presenta el número de datos tras el proceso de limpieza.

Bases	Registros antes de la limpieza	Registros después de la limpieza
PhishTank	14200	13362
Monkey	3700	2602
Enron	16800	15120

CUADRO 3.2. Datos obtenidos después de la limpieza

Con los datos que se obtuvo de este análisis, se podrá determinar que dominios predominan en la falsificación de identidad. Seguidamente, se visualiza estos resultados de las principales fuentes utilizadas para este estudio que son: PhishTank y Monkey.

Dominios	PhishTank	R.PhisT	Monkey	R.Monkey
Bank	1300	0,1831	227	0,0289
Gmail	1200	0,1690	3512	0,4472
Run escape	1039	0,1463	0	0,0000
Google	620	0,0873	231	0,0294
PayPal	609	0,0858	1020	0,1299
eBay	388	0,0546	1	0,0001
Facebook	345	0,0486	25	0,0032
office	259	0,0365	138	0,0176
Microsoft	217	0,0306	124	0,0158
Halifax	124	0,0175	0	0,0000
Amazon	119	0,0168	24	0,0031
Android	99	0,0139	393	0,0500
Apple	99	0,0139	695	0,0885
Netflix	93	0,0131	100	0,0127
Adobe	84	0,0118	13	0,0017
WhatsApp	75	0,0106	1	0,0001
YouTube	60	0,0084	54	0,0069
Steam	57	0,0080	0	0,0000
Yahoo!	55	0,0077	993	0,1264
Outlook	52	0,0073	61	0,0078
LinkedIn	40	0,0056	0	0,0000
Instagram	38	0,0054	9	0,0011
Virus total	34	0,0048	0	0,0000
Twitter	29	0,0041	17	0,0022
JPMorgan Chase and Co.	22	0,0031	0	0,0000
Hotmail	12	0,0017	56	0,0071
American express	10	0,0014	100	0,0127
Vodafone	9	0,0013	0	0,0000
HSBCgruop	7	0,0010	0	0,0000
Windows	6	0,0008	60	0,0076

Cuadro 3.3: Dominios detectados como los más utilizados en la suplantación de identidad (Phishing) en PhishTank

Se observa en el cuadro 3.3 que el dominio Bank presenta la frecuencia más alta en el contexto de ser víctima de Phishing para la data de PhishTank con un porcentaje de ocurrencia del 18 % le sigue Gmail con el 16 %. Ahora bien, para la base Monkey el dominio predominante es Gmail con una representatividad del 45 % y esto se sobreentiende dado que esta data en particular contiene correos infectados. De manera general los phishers apuntan a tener una buena ganancia con el menor tiempo de ejecución.

Analizando los dominios en una forma general los que predominan en este tipo de ataque son los que normalmente las personas usan a diario tendiendo el uso de al menos 2 horas diarias. Como conclusión se puede decir que siempre el atacante realizará un estudio sobre la tendencia de visitas que se ejecutó en un intervalo de tiempo concreto en los diversos sitios para proceder con el ataque teniendo en cuenta la vulnerabilidad que presenta y además la confiabilidad que se tiene de estos sitios.

A continuación, en la figura 3.2 se tiene la frecuencia absoluta entre los dominios de PhishTank y Monkey, visualizando que Gmail es el que tiene una presencia alta en las dos bases de datos

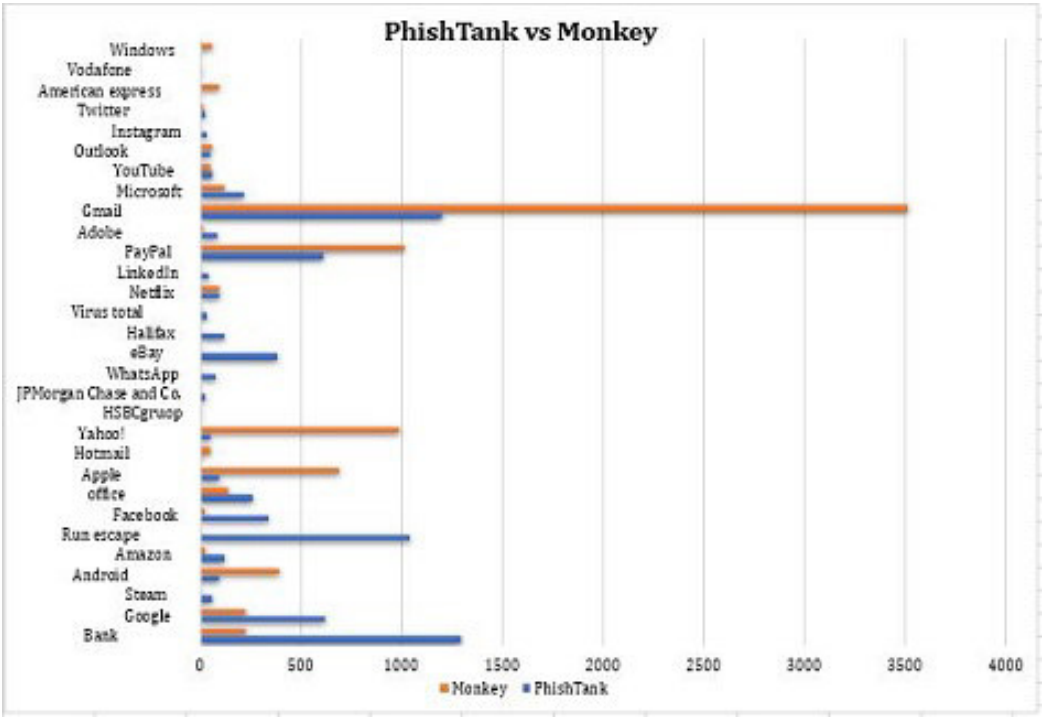


FIGURA 3.2. Frecuencia de los Dominios PhishTank vs Monkey

Posteriormente se procede a identificar las palabras más usadas tanto en los emails con presencia de phishing y los reales. Mediante el método de exploración de texto (Text Mining) en las bases, esta técnica ayudó a identificar las palabras que presentan una alta frecuencia en los correos electrónicos con presencia de Phishing ó benignas, Una vez obtenido el conjunto de palabras se las tomará como variables en el modelo de detección. Ahora veamos las palabras identificadas, la frecuencia absoluta y la frecuencia relativa en los emails, lo que permite observar la diferencia para la detección con más claridad.

Nota: Se utilizo esta técnica dado que se puede explorar y descubrir relaciones ocultas

dentro de datos no estructurados. Dado que el 80 % de los datos en el mundo residen en un formato no estructurado, la minería de texto es una práctica extremadamente valiosa dentro de este tipo de procesos.

Término	Phishing	F.Rphis	No-Phishing	F.R.Nphis
Actualizar	233	0,006	746	0,073
Confirmar	121	0,003	197	0,019
Usuario	244	0,006	220	0,021
Cliente	45	0,001	245	0,024
Querido	112	0,003	65	0,006
Miembro	44	0,001	258	0,025
Restringir	365	0,009	257	0,025
Sostener	120	0,003	871	0,085
Verificar	242	0,006	18	0,002
Cuenta	402	0,010	179	0,017
Notificación	343	0,009	80	0,008
Login	236	0,006	12	0,001
Sesión	500	0,012	81	0,008
Clic aquí	1200	0,030	194	0,019
Contraseña	930	0,023	81	0,008
Felicidades	700	0,017	38	0,004
Felicitaciones	523	0,013	22	0,002
Ganaste	189	0,005	0	0,000
Gratis	1200	0,030	934	0,091
Seguridad	118	0,003	40	0,004
Importante	1500	0,037	125	0,012
Aviso	47	0,001	135	0,013
Crédito	5800	0,144	833	0,081
Banco	14500	0,360	219	0,021
En línea	124	0,003	818	0,080
Enviar	448	0,011	593	0,058
Transferir	945	0,023	2057	0,201
Acceso	743	0,018	369	0,036
Contagio	38	0,001	3	0,000
Suspender	68	0,002	26	0,003
Tarjeta	6054	0,150	278	0,027
Sospechosa	435	0,011	89	0,009
Financiera	657	0,016	56	0,005
Actividad	767	0,026	99	0,010
Covid-19	25750	0,597	7576	0,191

Cuadro 3.4: Frecuencia absoluta de los términos que se identificaron como características en la detección de Phishing.

Con los resultados obtenidos y presentados en el cuadro 3.4, se observa la influencia del uso de estas palabras en los correos falsos y verídicos. A continuación, se visualiza una gráfica de las frecuencias obtenidas.

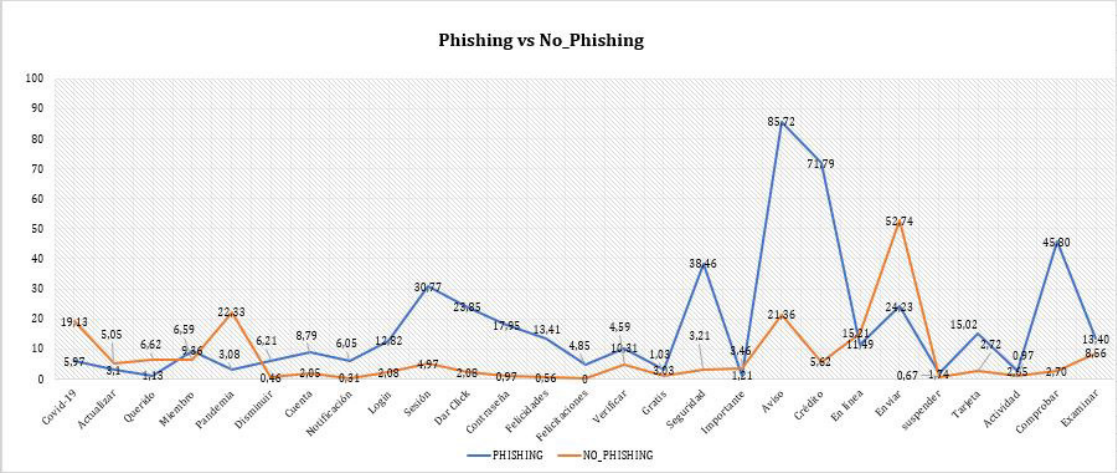


FIGURA 3.3. Número de términos usados en los emails con Phishing y sin Phishing

Verificación la calidad de los datos

Luego de realizar el análisis de datos, se concluye que los Datasets obtenidas son adecuadas, dado que nos brindan la información requerida para la extracción de características, las mismas que contribuyen para la modelación eficiente, pretendiendo obtener una buena precisión. Con respecto a los correos electrónicos, provienen de una fuente fidedigna, hay que notar que los registros están actualizados, por lo que podemos asegurar que los resultados son un buen punto de referencia para nuevos estudios.
El conjunto de datos que contienen correos no infectados, ayuda a balancear los datos y con ello obtener un mejor resultado en la predicción del modelo de aprendizaje, dando un mejor pronóstico en la identificación de correos electrónicos infectados, debido a que nos brinda información para determinar las características que ayudan a determinar entre un correo infectado y otro no.

PhishTank es una fuente confiable mencionada en varios artículos referentes a este tema en especial [30], la información proporcionada nos ayuda en demasía en el análisis de URL_s y de dominios. Los registros recopilados, brindan datos muy relevantes y precisos, lo que garantiza la ausencia de riesgo de tener un porcentaje alto de ruido en el procesamiento, y que los resultados serán veraces.

La preparación de los datos es un proceso para ajustarlos de forma adecuada para el procesamiento del modelo. Esta etapa es una de las principales y frecuentemente conlleva más tiempo, debido a que es necesario agrupar y elegir de manera correcta las variables que van a ser procesadas.

3.3. Construcción del vector de características

Para el desarrollo de esta parte del proyecto, se seleccionó el lenguaje de programación Java 11, dado que tenemos datos no estructurados en formato M.box y se procesan para determinar las características para la detección de Phishing, para lo cual se utiliza librerías para parsear el HTML y extraer los datos necesarios.

Con esto se exporta la base de mensajes completa del formato JSON a CSV. El código se encuentra desarrollado Apéndice D, consta de varios paquetes entre los que se destacan.

- Srv: se presenta la lógica de negocio y procesamiento de los emails.

- Dao: es una interfaz que permiten interactuar con la base de datos.
- Dto: es la representación de las colecciones de la Dataset en Entidades.
- Ctrl: se exponen endpoints REST para interactuar con los servicios.
- Util: son varias clases con métodos estadísticos que presentan varias utilidades.

3.3.1. Selección de características

Como se expuso al inicio de este capítulo para el diseño del modelo, hay que seleccionar las características apropiadas, basándonos en estudios referentes a la temática y el análisis que se realizó en el capítulo anterior. Además, se utiliza una biblioteca Java llamada Secure Socket Extension para extraer información de terceros relacionada con un dominio en particular durante el proceso de extracción de características. Esto proporciona una forma eficaz de examinar todas las características y etiquetas relevantes de la página analizada para examinar su estado.

Para las funciones de URL, se extraen 7 funciones. En las características del documento web, se extraen otras 12 características, que se extraen para mejorar la detección de phishing; para el corpus del email se tiene 9 grupos de palabras, estas funciones se pueden visualizar en los cuadros 3.5 y 3.6. Aunque algunas otras características todavía están disponibles, elegimos especialmente esas características porque las que se omiten se puede deducir de las elegidas, (*e.g.*, el número de puntos en la mayoría de las URL) de phishing está asociado con nombres de dominio alargados.

Se designa un nombre a las características extraídas. La fase del clasificador de aprendizaje automático utilizó las características elegidas para entrenar los algoritmos (RF y NB) de aprendizaje automático y seguir con el proceso de validación.

	Detalle	**Variable**
1	Presencia de links en el email	numLinks
2	Presencia de Http en la URL	numSecureLinks
3	Detección de dominio distinto al email que envia (raiz no subdominio).	numNotSameDomainLinks
4	Detección de text link que tienen el texto igual al que redirecciona	numNotSameTextLinks
5	Detección de direcciones IP en URL.	numIpDomainLinks
6	Existencia de links inválidos en el cuerpo del correo.	numInvalidLink
7	Dirección del correo (from) es diferente del dominio principal del link	difDomainAddress
8	Si tiene un TLD en medio de la dirección de correo from	hasTLDAddress
9	Detección de link que tienen un TLD en medio	numTLDLinks
10	El dominio ha sido registrado recientemente	isRecentDomain
11	El from es igual al to	isFromEqualTo
12	Uso de links acortados	numShortLinks
13	Símbolo @ en URL	simarrob
14	URL basada en IP	URL-IP

15	URL con código hexadecimal	hexcodig
16	Longitud de URL larga	longURL
17	Varias barras en la ruta de la URL	barrasURL
18	Verificación del nombre de dominio	verfdom
19	Código malicioso descargable	codmaldeg

Cuadro 3.5: Características que se analizó e investigo para la detección Phishing

Para el entendimiento sobre los patrones que tiene un correo electrónico con presencia de Phishing o a su vez un correo legítimo, se realizó una revisión exhaustiva de documentación relacionada con el tema de estudio, con esto se procede a realizar una comparación de palabras entre correos electrónicos benignos (reales) e infectados mediante la técnica de text mining. Este proceso permitió la correcta caracterización en palabras que se presentan en email con presencia de un ataque cibernético o uno legítimo. La formación de características, basadas en los términos textuales son unidas y esquematizadas de tal forma que cada palabra tiene homogeneidad entre ellas y así formar un grupo de estas. Se debe mencionar que esta etapa es una parte primordial para la construcción del Dataset que posteriormente se usará para el modelamiento y la validación. Los grupos de palabras se presentan en el siguiente cuadro.

	Detalle	**Variable**
20	1. Actualizar; Renovar; Verificar; Enviar	VP14
21	2. Beneficiario; Cliente; Querido; Miembro	VP15
22	3. Suspender; Disminuir; Pandemia; Mantener; Covid-19; Contagio.	VP16
23	4. Comprobar; Notificación; Examinar	VP17
24	5. Iniciar; Comenzar; Sesión; Contraseña; Dar clic	VP18
25	6. Ganancia; Sorteo; Felicitaciones; Premio; Ganaste; Gratis	VP19
26	7. Cuenta; Seguridad;	VP20
27	8. Importante; Aviso; Verificar; Verificación ; Sospechosa	VP21
28	9. Crédito; Banco; Banca web, Transferir	VP22
29	10. Estimado miembro, Estimado cliente, Estimado usuario	VP23
30	11. Tarjeta; Financiera; Actividad; Débito.	VP24

CUADRO 3.6. Agrupación de términos que presentan una frecuencia alta en correos con presencia de Phishing y correos benignos

3.3.2. Definición de la variable dependiente

La variable dependiente y representa, la identificación de correos mediante ciertas características, por tanto, será una variable binaria que toma los valores de 1 para los correos con etiqueta de Phishing y 0 para los correos con etiqueta de benignos: de modo que la etiqueta y es una clase binaria representada como:

$$y = \begin{cases} 1, & \text{si es phishing,} \\ 0, & \text{si es benigna.} \end{cases} \tag{3.1}$$

3.3.3. Unificación de Datos para el modelamiento

Para este proceso no se tuvo que generar atributos, debido a que la información requerida se encuentra en las características elegidas. Como siguiente paso integramos los datos.

Con las características obtenidas, procedemos a consolidar en una matriz, donde cada columna contiene las variables extraídas, mientras que en cada fila está conformada por un registro, estos elementos tienen una forma binaria donde 1 representa que es phishing y 0 es benigna. Una vez clasificados de forma adecuada son almacenados en un archivo con extensión CSV, este documento será utilizado para la modelación y validación.

3.4. Construcción del modelo Predictivo

En esta sección describiremos el modelo utilizado para el cumplimiento del objetivo planteado, por lo que se utilizará las métricas que indicarán el porcentaje de precisión de los algoritmos de clasificación, en la siguiente subsección, evaluaremos estos modelos y se observará si se cumple con los criterios establecidos para los datos de prueba y locales.

El algoritmo de Random Forests (RF) descrito en el Capítulo 2 es uno de los escogidos como modelo de predicción de la variable Y. El RF se basa en la técnica Ensemble Learning que ha demostrado ser exitoso en muchos campos. El concepto principal del RF es la construcción de un conjunto de árboles sucesivos de profundad pequeña y por ende débiles, en el que cada árbol de aprendizaje y mejora con relación al anterior. Al combinar estos árboles se obtendrá un potente clasificador que frecuentemente es muy estable.

Ventajas y Desventajas del algoritmo Random Forests

Ventajas

1. Random Forest se fundamenta en el algoritmo de Bagged explicado a profundidad en el anterior capítulo. Tiene como objetivo crear tantos árboles en el subconjunto de datos y acopla la salida de estos árboles. De este modo, se reduce el sobreajuste en los árboles de decisión y a su vez disminuye la varianza y, en consecuencia, la precisión mejora

2. Random Forest se puede utilizar para resolver problemas de clasificación y regresión.

3. Random Forest puede manejar automáticamente los valores perdidos.

4. No se requiere escalado de características: No se requiere escalado de características (estandarización y normalización) en el caso de Random Forest, ya que utiliza un enfoque basado en reglas en lugar del cálculo de distancia.

5. Maneja los parámetros no lineales de manera eficiente: los parámetros no lineales no afectan el rendimiento de un bosque aleatorio a diferencia de los algoritmos basados en curvas. Por lo tanto, si existe una alta no linealidad entre las variables independientes, Random Forest puede tener un rendimiento superior en comparación con otros algoritmos basados en curvas.

6. El algoritmo de Random Forest es muy estable. También al introducir nuevos datos en el conjunto, el algoritmo no se afecta, la razón es que, aunque afecte a un árbol, no lo hará con todos los árboles.

Desventajas

1. **Complejidad:** Random Forest crea muchos árboles (a diferencia de un solo árbol en el caso del árbol de decisión) y combina sus resultados. De forma predeterminada, crea 100 árboles en la biblioteca sklearn de Python. Para ejecutarlo, este algoritmo necesita recursos computacionales potentes. Por otro lado, el árbol de decisiones es sencillo y no requiere medios computacionales altos.

2. **Período de entrenamiento más largo:** Random Forest necesita de mucho tiempo para entrenar, mientras que los árboles de decisión presentan un tiempo de entrenamiento corto, y esto se debe a la generación de muchos árboles (en lugar de un árbol en el caso del árbol de decisión) y toma decisiones por mayoría de votos.

Naïve Bayes (NB) es un algoritmo de aprendizaje automático abordado en el Capítulo 2 es otro de los escogidos como modelo de predicción de la variable Y. NB se utilizamos para resolver problemas de clasificación. Se basa en el teorema de Bayes. Es uno de los algoritmos ML más simples pero potentes que se utilizan y encuentra aplicaciones en muchas industrias.

Ventajas

1. Este algoritmo funciona de una forma rápida y por esta razón el tiempo de ejecución es corto.

2. NB es apropiado para resolver problemas de predicción de clases múltiples.

3. Si las variables son independientes, el funcionamiento del modelo es mejor que otros y sin tantos datos en el entrenamiento.

4. Si las variables de entrada son categóricas es apropiado usar NB.

Desventajas

1. NB asume que todos los predictores (o características) son independientes y rara vez ocurren en la vida real. Esto restringe la aplicabilidad del algoritmo en casos de uso en el mundo real.

2. Este algoritmo enfrenta el problema de frecuencia cero donde asigna probabilidad cero a una variable categórica cuya categoría en el conjunto de datos de prueba no estaba disponible en el conjunto de datos de entrenamiento. Sería mejor si utilizara una técnica de suavizado para superar este problema.

3. Sus estimaciones pueden ser incorrectas en algunos casos, por lo que no debe tomarse muy en serio sus resultados de probabilidad.

3.4.1. Implementación del Algoritmo Predictivo.

Haciendo uso del software Python explicado a profundidad en una sección anterior, en esta implementación se utilizará las siguientes librerías y módulo para la construcción del Algoritmo Predictivo.

1) **sklearn:** Es una biblioteca de Python para trabajar con datos complejos. Scikit-learn es una biblioteca de código abierto que admite aprendizaje automático. Admite diversos algoritmos supervisados y no supervisados como: regresión lineal, clasificación, agrupamiento, etc. Esta biblioteca está asociada a Numpy y SciPy.

2) **sklearn.metrics:** es un módulo de funciones que evalúan el error de predicción para propósitos específicos. Estas métricas se detallan en las secciones Métricas de clasificación, Métricas de clasificación multietiqueta, Métricas de regresión y Métricas de agrupamiento.

3) **matplotlib.pyplot:** esta biblioteca es responsable de graficar datos numéricos. Y es por eso que se utiliza en el análisis de datos. También es una biblioteca de código abierto y traza figuras de alta definición como gráficos circulares, histogramas, diagramas de dispersión, gráficos, etc.

4) **numpy:** El nombre Numpy significa Numerical Python. Es la biblioteca de uso común. Es una biblioteca popular de aprendizaje automático que admite matrices grandes y datos multidimensionales. Consta de funciones matemáticas integradas para facilitar los cálculos. Incluso bibliotecas como TensorFlow usan Numpy internamente para realizar varias operaciones en tensores. Array Interface es una de las características clave de esta biblioteca.

6) **sklearn.ensemble:** es un módulo que acopla predicciones de diversos estimadores compuestos con un algoritmo de aprendizaje que permiten una mejora en la generalización/robustez.

Detalles de la implementación:
El código de la implementación se encuentra disponible en el Apéndice D, a continuación, se muestran los detalles y observaciones del mismo.

En el software Python cargamos el paquete donde se encuentran las librerías, de NB, Arboles de decisión y RF son los que se utilizará para la modelización de esta problemática problemática que tiene como punto fundamental predecir si un correo tiene presencia o no de este ataque.

En la elaboración del modelo se toma como entrada el documento que contiene la matriz de 0_s y 1_s, estos datos se dividirán mediante un muestreo aleatorio simple, en dos submuestras aleatorias, una para el desarrollo del modelo (train) y la otra para su validación (testeo).

En la muestra de modelamiento, la cual corresponde aproximadamente al 80 % de la muestra original. En cambio, la muestra de validación corresponde aproximadamente al 20 % de la muestra original, siguiendo el principio de Pareto. Teniendo como objetivo principal detectar la presencia de phishing, las entradas que se consideran para los algoritmos de clasificación son las que se muestran a continuación.

- train_inputs: Variable que abarca el 80 % de los datos para la caracterización de un email con presencia o no de Phishing.
- test_inputs: Variable que abarca el 80 % de los datos para la caracterización de un email con presencia o no de Phishing
- train_outputs: Variable que abarca el 20 % de los datos para la caracterización de un email con presencia o no de Phishing.
- test_outputs: Variable que abarca el 20 % de los datos para la caracterización de un email con presencia o no de Phishing.

3.4.2. Modelo Random Forests

El módulo sklearn.ensemble incorpora dos algoritmos para obtener el promedio, basados en árboles de decisión aleatorios: el algoritmo Random Forest y el método Extra-Trees. Los dos algoritmos son técnicas de perturbación y combinación estructuradas precisamente para árboles. Esto implica la construcción de un conjunto con diferentes clasificadores, por medio de la introducción de aleatoriedad en la creación del clasificador. La predicción del conjunto es la predicción promedio de los clasificadores individuales. [32]

Como otros clasificadores, los clasificadores de bosque deben estar equipados con dos matrices: una matriz X de forma dispersa o densa que contiene las muestras de entrenamiento, y una matriz Y de forma que contiene los valores objetivo, para este modelo se utilizara la librería RandomForestClassifiery, donde cada árbol en el conjunto se crea a partir de una proporción extraída con reemplazo del conjunto de entrenamiento. Sin embargo, RandomForestRegressor usa un número predeterminado de árboles de 100, que normalmente no es suficiente. En consecuencia, esta se subió hasta 1000 árboles en primera instancia para posteriormente dejarlo en 3000 árboles. La profundidad predeterminada de cada árbol (max_depth) es 5, lo que significa que estamos ensamblando árboles con profundidad máxima de 5.

Para determinar los parámetros óptimos primero se ha ejecutado la forma predeterminada, es decir sin cambiar lo que por defecto está determinado para observar los resultados que se obtienen, posteriormente se va jugando con los parámetros para encontrar los que hacen mínima la tasa de mal clasificados.

El propósito de usar Random Forest es que se consigue una variación pequeña al combinar vario árboles, ocasionalmente a costa de un incremento pequeño en el sesgo. En la práctica, la reducción de la varianza es a menudo significativa, por lo que se obtiene un mejor modelo.

3.4.3. Modelo Naive Bayes

El módulo sklearn. ensemble tiene la librería MultinomialNB que se usara para la implementación del algoritmo de Bayes para datos distribuidos multinomialmente, siendo estas las dos variaciones clásicas de NB, para el uso en la clasificación de texto en el que los datos son recuentos de vectores de palabras normalmente, aunque también se sabe que los vectores tf-idf funcionan bien. La distribución está parametrizada por vectores $\theta_y = (\theta_{y_1}, ..., \theta_{y_n})$ para cada clase y, donde n es el número de características (en la clasificación del texto, el tamaño de la palabra) y θ_{y_i} es la probabilidad $P(x_i \mid y)$ de característica i apareciendo en una porción perteneciente a la clase y.

Para esta implementación no fue necesario utilizar (partial_fit) dado que el conjunto de entrenamiento completo no presento ningún inconveniente en la memoria.

De forma general el tiempo de ejecución de este modelamiento depende de los datos que utilicemos *e.g.*, para el conjunto de datos descargables en la construcción de características tuvo una hora y media de procesamiento mientras que para el entrenamiento se demoró alrededor de una hora, ahora bien, para los datos locales (cuentas personales) el tiempo vario sustancialmente dado que era necesario construir un programa en JAVA 11 para indexar los mensajes de las cuentas, por tal razón, se requirió de cuatro horas para completar el proceso. Hay que mencionar que este periodo de tiempo varía dependiendo del equipo que se utilice en la modelización.

3.4.4. Evaluación el modelo

En la fase de evaluación vamos a comparar el rendimiento del sistema propuesto a través de un experimento de validación cruzada de 10 veces antes del proceso de evaluación en el conjunto de datos de prueba. Esto implica la división aleatoria del conjunto de datos de prueba en diez submuestras iguales, de las cuales una sola submuestra se usa para la validación final del modelo, mientras que las otras submuestras son usadas para el entrenamiento del sistema. Por tanto, el modelo predictivo propuesto se basó en el 80 % del conjunto de datos y se validó en el 20 % restante.

Las razones para usar la validación cruzada en este modelo son para

i. Verificar el comportamiento del error del modelo predictivo. En este caso, los errores asociados con el modelo predictivo de RF y NB en la detección de phishing.

ii. Validar el Dataset de entrenamiento mediante la validación de cada subconjunto. Esto es para tener un nivel alto de confianza en el modelo entrenado.

iii. Evaluar cómo los resultados del modelo predictivo se generalizarán a conjuntos de datos independientes sobre sitios tanto benignos como de phishing.

En el experimento se usara tanto conjuntos de datos de prueba como los locales, estas datas contienen sitios web legítimos y de phishing no superpuestos que se procesara previamente para posteriormente obtener un archivo CVS, que contiene el vector de características finales para procesar en el modelo predictivo. A continuación, explicaremos como se obtuvo los datos locales en forma resumida sin dejar de lado lo primordial.

Indexado de mensajes

Dado que el enfoque de esta investigación es a nivel local del país los dataset disponibles en la red pública son demasiado antiguos o en otro idioma, por tal motivo se opta por utilizar los correos de 3 cuentas, una cuenta de Gmail personal, Outlook personal y una institución educativa nacional.

- Se implementa un programa en JAVA 11 que permite ejecutar un cron que se encargara de indexar todos los emails de las cuentas mencionadas para obtener tanto correos categorizados como verdaderos, así como de la carpeta de spam mediante la lectura directa del buzón de mensajes con el uso del protocolo POP3; el código se encuentra disponible en el Apéndice D.

- Esos registros se guardan en una Dataset no estructurada (mongodb) en formato JSON y se procesan para determinar las características mencionadas en los cuadros 3.5 y 3.6, para lo cual se utiliza librerías como JSOUP para parsear el HTML y extraer los datos necesarios.

Con esto se exporta la base de mensajes completa del formato JSON a CSV, las librerías que se utilizó para este proceso están descritas en la sección Çonstrucción de características"; En el Servicio "MensajeSrv"se encuentra la lógica descrita en Indexado de mensajes.

Posteriormente realizamos el análisis respectivo, para esto procedemos a dividir el proceso en tres etapas, la primera el entrenado del modelo, como siguiente fase tenemos la predicción, y para finalizar los resultados conseguidos al ejecutar el modelo en un entorno

moderado para la detección de phishing en los emails en las cuentas personales antes mencionadas.
A continuación, se detalla los resultados y el análisis del modelo.

Etapa uno: Entrenado del modelo Para esta etapa tenemos alrededor de 7043 correos para el entrenamiento del modelo, teniendo 3620 con presencia de phishing y 3423 correos reales.

Nota: Estos datos representan el 80 % del dataset obtenido.

Hay que mencionar, antes de seguir con la segunda etapa, que en la data de testeo se tiene 1761 emails teniendo 920 emails con phishing y 841 emails sin presencia de phishing, estos datos se utilizaran para la predicción, teniendo un total de 8804 correos en la base original.

Segunda etapa: Predicción del modelo Para la fase de predicción, se trabajará con 1629 emails para esta etapa, se hará uso de los modelos de clasificación RF y NB. Con el fin de obtener las métricas adecuadas para la evaluación de los resultados.

Estas consisten en:

- Verdadero Positivo (TP): que señala el número de correos identificados como phishing, siendo estos phishing.
- Verdadero Negativo (TN): son los señalados como benigno, siendo estos phishing.
- Falsos Positivos (FP): son los señalados como phishing, siendo estos benignos/reales.
- Falsos Negativos (FN): son señalados como benignos, siendo estos benignos.

Además, calculamos el coeficiente de correlación de Mathew (MCC) para determinar la calidad del modelo de predicción. Entendiendo que cuando MCC se acerca a la unidad, indica que el sistema tiene una predicción casi perfecta y, por lo tanto, es un sistema de detección confiable. La siguiente ecuación representa el MCC

$$MCC = \frac{\text{TPRxTNR-FPRxFNR}}{\sqrt{(TPR+FPR)(TPR+FNR)(TNR+FPR)(TNR+FNR)}}$$

En el siguiente capitulo se presenta los resultados para los dos conjuntos de datos, adicionalmente se presenta la comparación con otros trabajos que siguen la misma línea de investigación y de esta manera observar el nivel de predictibilidad del modelo.

Capítulo 4
RESULTADOS Y DISCUSIÓN

El rendimiento del sistema propuesto se evalúa utilizando cinco parámetros estándar y la validación cruzada esto se menciona en el anterior capítulo. Para el proceso de VC se repitió 10 veces y después de la validación, se calcula una sola estimación. Esta estimación es el promedio de las diez iteraciones.

El incentivo para aplicar el experimento de validación cruzada es ajustar el rendimiento de un modelo fuera del conjunto de entrenamiento, a continuación, se analizará los resultados del conjunto de datos descargables.

4.1. Resultado para datos descargables

Métricas \Modelo	Random Forests	Naïve Bayes	Trees
Accuracy	97,70 %	92,53 %	94,50 %
Presición	97,24 %	90,57 %	94,30 %
Recall_score	98,55 %	93,06 %	95,70 %
Puntuación F1	97,54 %	92,06 %	95,92 %
Roc_auc	97,50 %	93,50 %	94,60 %
Coef. Mathew	0,971	0,923	0,959

CUADRO 4.1. Tabla de resultados para el conjunto de datos (PhishTank, Monkey y Enron)

Procedemos a interpretar los datos del cuadro 4.1, se visualiza que el mejor modelo es Random Forests a comparación de Naïve Bayes como un plus se calculó para árboles de decisión, teniendo como resultado que RF presenta el que mejor nivel de predicción para este investigación, ahora bien analicemos las métricas obtenidas, teniendo la exactitud que representa el porcentaje en el cual el modelo ha acertado, obteniendo un valor de 97,70 % de exactitud, el valor obtenido para la precisión es de un 97,24 %. Dado que se tiene un conjunto relativamente balanceado se puede decir que la métrica Accuracy proporciona un resultado confiable para nuestro trabajo

En el caso de la métrica de Recall es la capacidad del modelo para detectar los casos significativos. En nuestro caso se tiene 98,55 % es claramente un valor muy bueno para una métrica. Podemos decir que nuestro algoritmo de clasificación es muy sensible.

Para la Puntuación F1 se observa un porcentaje de 97,54 % dado que nos esquematiza la precisión y sensibilidad en una sola métrica, dándonos una gran ayuda cuando la distribución es desigual, así al tener una alta precisión y alto recall, implica que, el modelo elegido maneja perfectamente esa clase.

En el caso del coeficiente de correlación de Mathew se tiene 0,971 teniendo una buena calidad para el modelo de predicción RF.

A continuación, en la figura 4.1 se presenta la curva ROC para los datos de testeo, la cual vamos a interpretar. Dado que el valor es cercano a uno, podemos decir que el rendimiento del modelo es bastante bueno. Así, hemos encontrado un clasificador con un rendimiento muy bueno sin tener el riesgo de sobreajuste en los datos de las bases: PhishTank, Monkey y Enron.

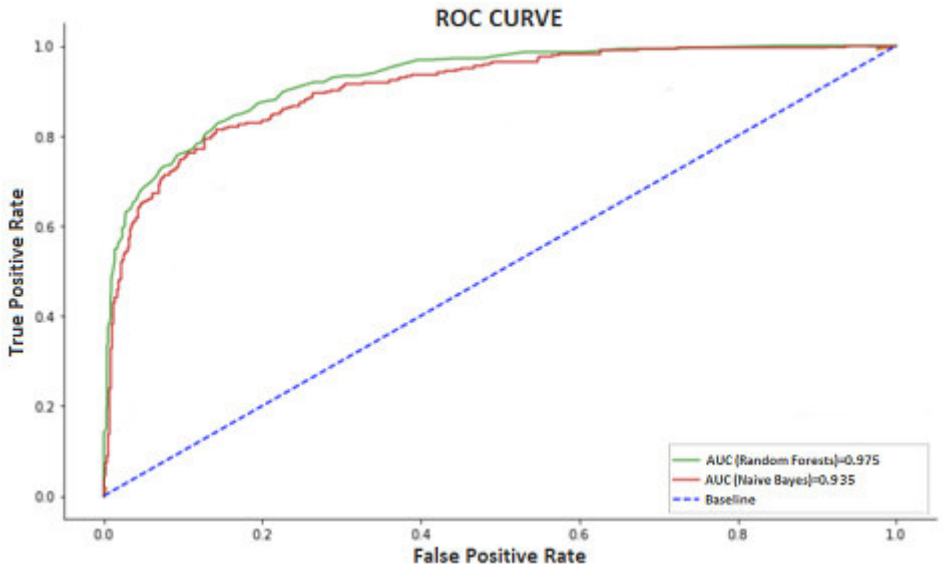


FIGURA 4.1. Curva Roc y AUC de datos descargables

Ahora procedemos a calcular el valor real del error y el porcentaje de error que tiene el modelo, en la ecuación 4.1 determina el valor real (ERR) y en la ecuación 4.2 la tasa (TERR), es otras palabras, el porcentaje el modelo no etiqueto los correos de correctamente.

$$ERR = \frac{\text{FN+FP}}{\text{TN+FN+FP+TP}}, \tag{4.1}$$

donde, la tasa de error (TERR) se calcula de la siguiente manera

$$TERR = \frac{\text{FN+FP}}{\text{TN+FN+FP+TP}} \cdot 100, \tag{4.2}$$

$$= 2,3\,\%.$$

Finalmente, de los resultados que se obtuvo, se observa que, de los 6217 correos analizados, el 97,7 % que corresponde a 6074 emails fueron clasificados correctamente, por el contrario, el 2,3 % correspondiente a 143 correos fueron etiquetados de forma incorrecta. Por consiguiente, se puede decir que el modelo presenta una predictibilidad alta, teniendo como consecuencia resultados fiables en la clasificación de phishing.

Para el conjunto de prueba se tendrá datos locales teniendo un enfoque aplicado a mails, el algoritmo utiliza los parámetros para identificación de mail presentados en sección anterior. Hay que tomar en cuenta que es posible usar el mismo algoritmo y metodología para phishing web, pero se deben considerar los parámetros para Phishing web.

4.2. Resultado para datos Locales

En esta sección, analizaremos los resultados para los datos extraídos de las tres cuentas personales, el conjunto de prueba contiene registros de correos con presencia de phishing y benignos por lo tanto las variables que se usaran para la detección son un subconjunto de los cuadros 3.5 y 3.6. A continuación, se presenta el cuadro 4.2 que contienen las métricas obtenidas en el modelo con este conjunto de datos.

La motivación para aplicar el experimento de validación cruzada es ajustar el rendimiento de un modelo fuera del conjunto de entrenamiento, a continuación, se analizará los resultados del conjunto de datos locales.

Métricas \Modelo	Random Forests	Naïve Bayes	Trees
Accuracy	95,40 %	89,35 %	91,50 %
Presición	92,14 %	85,98 %	89,03 %
Recall_score	96,55 %	89,62 %	92,70 %
Puntuación F1	94,74 %	87,08 %	91,22 %
Roc_auc	94,90 %	90,20 %	91,26 %
Coef. Mathew	0,967	0,932	0,954

CUADRO 4.2. Tabla de resultados para el conjunto de datos Locales (Cuentas personales: Gmail, Outlook e Institucional)

De la misma forma que se hizo en la sección anterior para los registros descargables se procede a interpretar los datos de el cuadro 4.2 obtenidas, se puede visualizar que el mejor modelo es Random Forests al igual que en el caso anterior, ahora bien analicemos las métricas obtenidas, teniendo la exactitud que representa el porcentaje de predicciones correctas frente al total por tanto se tiene 95,40 % de exactitud para este modelo, el valor obtenido para la precisión es de un 92,14 %. Por tanto, nuestro modelo es más preciso que exacto, coincidiendo con el experimento anterior.

En el caso de la métrica de Recall (Sensibilidad) es la habilidad del modelo para detectar los casos relevantes. En nuestro caso se tiene 96,55 % es claramente un valor bueno para una métrica. Podemos decir que nuestro algoritmo de clasificación es sensible.
Para la Puntuación F1 se observa un porcentaje de 94,74 %, dado que nos esquematiza la precisión y sensibilidad en una sola métrica, dándonos una gran ayuda cuando la distribución es desigual, así al tener una alta precisión y alto recall, implica que, el modelo elegido maneja perfectamente esa clase.

En el caso del coeficiente de correlación de Mathew se tiene 0,967 teniendo una buena calidad para el modelo de predicción Random Forests.
De misma manera que se visualizó en los datos de experimentales, en la figura 4.2 se presenta la curva ROC para los datos de testeo, la cual vamos a interpretar. Dado que el valor es cercano a uno, podemos decir que el rendimiento del modelo es bastante bueno. Así, hemos encontrado un clasificador con un rendimiento muy bueno sin tener el riesgo de sobreajuste en los datos locales.

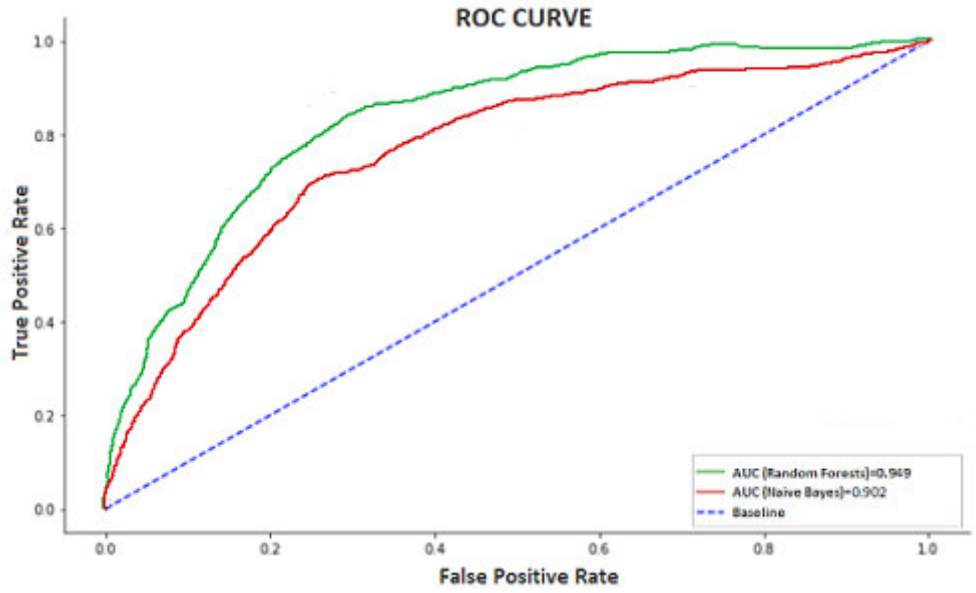


FIGURA 4.2. Curva Roc y AUC de los datos locales

Al igual que sección anterior se determina el porcentaje de error presente en el modelo, y esto nos proporciona una validez buena para el análisis de resultados.

Ahora procedemos a calcular el valor real del error y el porcentaje de error que tiene el modelo, en la ecuación 4.1 determina el valor real (ERR) y en la ecuación 4.2 la tasa (TERR), es otras palabras, el porcentaje el modelo no etiqueto los correos de correctamente.

$$ERRL = \frac{\text{FN+FP}}{\text{TN+FN+FP+TP}}, \tag{4.3}$$

donde, la tasa de error (TERRL) se calcula de la siguiente manera

$$TERRL = \frac{\text{FN+FP}}{\text{TN+FN+FP+TP}} \cdot 100, \tag{4.4}$$

$$TERRL = 4,6\,\%.$$

De los resultados alcanzados, se observa que, de 1761 correos analizados, el 95,4 % que corresponde a 1680 correos se clasifico de forma correcta, mientras que el 4,6 % correspondiente a 81 correos fueron clasificados de forma errada. Por lo motivo, se determina que el modelo tiene un porcentaje de predicción relativamente alto, dando como consecuencia resultados medianamente fiables en la clasificación de phishing.

4.3. Comparación de resultados obtenidos

Resumiendo, de todos los resultados obtenidos en los dos conjuntos de datos podemos observar que, el modelo construido logró un 97,7 % en determinar la existencia de phishing en URL_s y en los correos electrónicos para los datos experimentales como son: PhishTank, Monkey y Enron. Teniendo como observación principal que al variar el conjunto de características de otros trabajos enfocados en la misma temática se puede obtener un mayor o menor

porcentaje de predicción, pero esto se debe a la actualización de técnicas de ataque en este tipo de delito siempre el vector de características debe ir actualizándose.

Obteniendo que la aplicación de un modelo ayuda a la detección de correos electrónicos con presencia de phishing. Pese a esto también se debe mencionar que la posibilidad de tener el 2,3 % de error al momento de la clasificación, aunque es pequeño no deja de ser un dato que hay que tomar mucho en cuenta para futuros trabajos.

Para los resultados obtenidos en el conjunto de datos locales podemos visualizar que, en diversas métricas aplicadas, el modelo creado logró un 95,4 % en la determinación de existencia de phishing en URL_s y en los correos electrónicos. Los porcentajes tienen una gran diferencia entre los datos experimentales y los locales, dado que los datos que se tiene en nuestro país no son los mejores o como se menciona anteriormente la detección de delitos cibernéticos en Ecuador está dando comenzando, es por esta razón que los datos no son tan buenos o algunos no presentan ciertas características para tener un buen análisis y por ende obtener un porcentaje mayor.

La tasa de error es de 4,6 % dándonos como indicativo que, de 1000 correos analizados, 46 de ellos no se clasificaran de forma adecuada dando un margen de perdida de información o ser víctimas de phishing.

En la siguiente subsección, se mostrará los resultados de trabajos anteriores referentes a esta temática, al aplicar técnicas de ML para la detección de este tipo de delito.

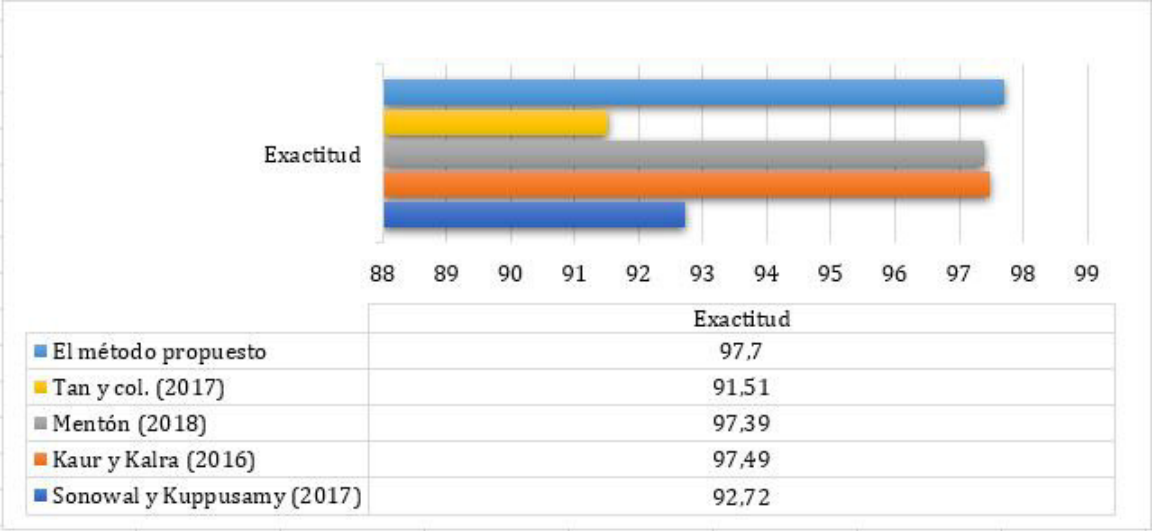
4.3.1. Resultados para el sistema y los trabajos anteriores

El enfoque propuesto en este trabajo de investigación se aborda las limitaciones de otras técnicas anti-phishing en términos de ciertas mediciones de parámetros como la eficiencia computacional y la robustez. En esta sección, la evaluación del desempeño en parámetros de evaluación como La tasa de verdaderos positivos, la tasa de falsos positivos y la precisión se comparan con otras técnicas existentes. Esta comparación se basa en los trabajos relacionados a técnicas anti-phishing. El Cuadro 4.3.1 presenta las estadísticas de rendimiento que indican que el método propuesto es uno de los mejores modelos anti-phishing existente en los anteriores trabajos.

S/N	Trabajo	Datos phish	Datos benig	Total	TPR	FPR	Exactitud
1	Sonowal y Kup [36]	667	995	1.662	90,54	5,82	92,72
2	Kaur y Kalra [24]	1.078	846	1.924	99,44	0,56	97,51
3	Chin [9]	3.718	1.185	4.903	96,9	0,03	97,39
4	Tan y col. [37]	500	500	1.000	99,2	7,8	91,41
5	El método propuesto	15.964	15.120	31.084	98,7	0,79	97,7

CUADRO 4.3. Comparación de trabajos relacionados con el método propuesto

La Fig. 4.3 presenta las ilustraciones gráficas, en términos de precisión de detección del método propuesto en comparación con otros métodos.



	Exactitud
El método propuesto	97,7
Tan y col. (2017)	91,51
Mentón (2018)	97,39
Kaur y Kalra (2016)	97,49
Sonowal y Kuppusamy (2017)	92,72

FIGURA 4.3. Comparación gráfica de la precisión del esquema propuesto con otros métodos.

4.3.2. Discusión

Los esfuerzos que se debe tener para la detección de phishing deben ser aún mayor, dado la ola de tecnología que impulsa la necesidad de ser más precavidos en la web, por esta razón se examinó: las limitaciones, el éxito y el impacto de esta investigación a nivel práctico y teórico. En un enfoque práctico, las personas que revisen este trabajo pueden tener una idea del cómo se construyó el vector de características y se aplicó en el modelo, para posteriormente mejorar su resiliencia en la seguridad cibernética al tratar de explicar cómo funciona este delito y prepararse para evitar caer en el phishing. Desde el punto de vista teórico, esta investigación combina dos mundos el práctico y académico donde los informes de inteligencia de amenazas cibernéticas y los trabajos de investigación académica se utilizó para aprender y desarrollar la comprensión de las capacidades, tácticas, técnicas y procedimientos de los atacantes.

Las limitaciones de esta investigación están relacionadas con los datos que se tiene para la ejecución de este modelo, dado que son registros antiguos o que tiene una alta presencia de *NaN* y esto no permite tener los resultados que se tenía en mente. Además, hay que recalcar que los datos utilizados para la evaluación son de 3 cuentas personales, dando como resultado un conjunto de muestra pequeña para el entrenamiento de los modelos RF Y NB obteniendo un desempeño menor al que se obtuvo con los datos de experimentales. Las limitaciones adicionales son que las organizaciones son diferentes en todos los campos, por ende, los resultados y hallazgos de esta tesis de pregrado no son aplicables a todos. Además, gran parte de los datos que se utilizó en esta investigación se recopiló a partir de informes de filtración de datos e inteligencia sobre amenazas disponibles públicamente. Donde se pudo notar, que no se está realizando una investigación académica a profundidad que esté relacionada con las técnicas modernas de ataque, lo que puede tener un impacto en la calidad académica de este trabajo. Sin embargo, durante este trabajo se procuró analizar las características que se están presentando actualmente en los ataques de phishing y así tener un mejor vector final de características, sin dejar de lado el escoger un buen modelo predictivo que fue de gran ayuda para visualizar el camino que se debe escoger para protegerse contra este delito.

El éxito de esta investigación se basa principalmente en el análisis general de técnicas, tácticas y procedimientos de atacantes modernos que son comúnmente utilizados por los ac-

tores de amenazas del mundo real que se obtuvieron a través de la revisión de la literatura. Estos proporciono más información sobre las acciones específicas y qué capacidades técnicas existen para eludir los controles de seguridad modernos, como el uso del candado de seguridad *Https* para el robo de datos personales. Aunque esta tesis analiza diferentes modelos para visualizar el que nos otorga un mejor nivel de productividad, exactitud y precisión y así dar una pauta a la hora de diseñar e implementar controles de seguridad de compensación, detecciones y procedimientos de respuesta para proteger los datos críticos.

Los resultados y el contenido de este trabajo podrían ser utilizados por estudiantes que deseen aprender más sobre este delito cibernético; las técnicas, tácticas y procedimientos que usan comúnmente los actores de amenazas. Esto también proporciona capacidades para que construyan sus propias características de detección y mejoren sus capacidades de respuesta en caso de que sean víctimas de un ataque de phishing.

Capítulo 5
CONCLUSIONES Y RECOMENDACIONES

5.1. Conclusiones

(a) Según el análisis del material recopilado para esta tesis, los actores de amenazas modernos se basan más comúnmente en técnicas de phishing para obtener acceso inicial. Algunos de los medios más comunes para obtener el acceso parecen ser la implementación de malware en un documento de Office, que luego se ejecuta una vez que el usuario habilita las macros. Los atacantes también utilizan cada vez más herramientas y marcos de seguridad ofensivos disponibles públicamente en sus campañas. Otro método que los atacantes parecen estar usando constantemente es el abuso de vulnerabilidades conocidas públicamente, ya que es mucho más rentable para ellos en lugar de dedicar tiempo y recursos a descubrir vulnerabilidades previamente desconocidas.

(b) En este trabajo, consideramos el problema de la detección de phishing utilizando enfoques de aprendizaje automático. En nuestro primer intento, llamado conjunto de prueba, identificamos varias características interesantes al analizar el problema desde el punto de vista estadístico. Se pudo visualizar que ver el problema desde un punto de vista puramente estadístico es insuficiente para resolverlo de manera efectiva. La intención del atacante de phishing también debe tenerse en cuenta para una solución eficaz.

(c) Describimos un enfoque hacia el diseño de Funciones basadas en nombres de dominio, análisis tanto de URL como el corpus del email para la detección de phishing mediante aprendizaje automático. Nuestro diseño de funciones hizo hincapié en la eliminación del posible sesgo en la clasificación debido a conjuntos de datos de phishing y páginas legítimas elegidos. Nuestro enfoque difiere de otros trabajos en este espacio, ya que explora la relación del corpus del email con su intención de phishing. Con un conjunto de características balanceado, podemos lograr una tasa de clasificación de 97 % con datos de validación cruzada. Además, pudimos mostrar una tasa de detección de 97-97.7 % para URL activas en la lista negra.

(d) Nuestros tiempos de extracción y clasificación de características son muy bajos y muestran que nuestro enfoque es adecuado para la implementación en tiempo real. Es probable que nuestro enfoque sea muy eficaz en las estrategias de phishing modernas, como el phishing extremo, que están diseñadas para engañar incluso a los usuarios experimentados.

(e) El desafío que tiene que afrontar Ecuador en el ámbito de la seguridad informática y la fiabilidad adecuada en una situación de amenaza que cambia cada vez más rápido es enorme, pero solucionable. Con respecto a la Industria, finalmente se deben tomar juntos pasos concretos para mejorar la situación de seguridad en el país, que es crítica en muchas áreas y elevarlo a un nivel razonable.

(f) El uso generalizado de computadoras, dispositivos móviles y sistemas de red ha aumentado la inserción en el mercado en línea de la mayoría de las empresas. Cada día, los canales de venta se trasladan gradualmente a Internet. Esta transformación ofrece numerosas facilidades en términos de adquisición en productos / servicios, etc. Sin embargo, estas oportunidades se ven desafiadas continuamente por la creciente tasa de delitos cibernéticos. Por tal razón la concientización de analizar los sitios de donde se quiere obtener cualquier tipo de servicio es primordial.

5.2. Recomendaciones

Las recomendaciones que se presentan en la sección final son de ayuda para trabajos futuros, teniendo en cuenta que esta investigación es un pequeño aporte a este campo tan extenso y relativamente nuevo.

(a) Los phishers buscan vulnerabilidades constantemente para atacar y a su vez no ser detectados al momento de sus ataques, por tal razón se recomienda ejecutar un análisis constante de las técnicas empleadas por estos atacantes, y así determinar características nuevas o que ayuden a mejorar el vector final y por ende obtener un modelo preciso con un nivel de predictibilidad más elevado.

(b) Se recomienda utilizar Datasets actuales para tener una mejor visión de este tipo de delitos cibernéticos, pero este punto es primordial dado que el tiempo de vida de estos dominios y URL_s es muy corto, por tal razón el conjunto de datos debe ser actualizado al menos cada 6 meses. Así se obtendrá resultados confiables y actuales.

(c) Revisar el trabajo de Orunsolu, Sodiya y Akinwale, en [30], dado que tiene una buena revisión de literatura para este tema de investigación.

BIBLIOGRAFÍA

[1] ABURROUS, M., HOSSAIN, M., DAHAL, K. AND THABTAH, F. (2010). Experimental case studies for investigating e-banking phishing techniques and attack strategies.

[2] ADEBOWALE, M., LWIN, K., SANCHEZ, E. AND HOSSAIN, M. (2018). Intelligent web-phishing detection and protection scheme using integrated features of images, frames and text.

[3] AMAT, R. J. Validación de modelos predictivos: Cross-validation, oneleaveout, bootstraping. Obtenido de: *https://www.cienciadedatos.net/.* (Julio, 2016).

[4] AMAT, R. J. Text mining con r: ejemplo práctico twitter. *https://www.cienciadedatos.net/.* (2017).

[5] AMAT, R. J. Análisis de texto (text mining) con python. *https://www.cienciadedatos.net/.* (Diciembre, 2020).

[6] ANWAR, T., ABU-KRESHA M. AND BAKRY, A. (2017). An efficient method for web page classification based on text. *J. Eng. Comput. Sci.*

[7] BREIMAN, L. (2001). *Random forests, Machine Learning*.

[8] CALVA YAGUANA, K. P. (2020). *Modelo de predicción del rendimiento académico para el curso de nivelación de la Escuela Politécnica Nacional a partir de un modelo de aprendizaje supervisado automatizado en R.* [Tesis de pregrado, Escuela Politécnica Nacional]. Repositorio institucional de la Escuela Politécnica Nacional. *https://bibdigital.epn.edu.ec/*

[9] CHIN, T. (2018). Phishlimiter: A phishing detection and mitigation approach using software-defined networking. *IEEE Access*, 6, 42516-42531, 10.1109/ACCESS.2018.2837889

[10] CORTINA, V. (2015). *Aplicación de la metodología CRISP-DM a un proyecto de minería de datos en el entorno universitario.* [Universidad Carlos III de Madrid]. Departamento de Informática.

[11] CRESWELL. (2015). Educational research. planning, conducting and evaluating quantitative and qualitative research. U.S.A.

[12] DHANALAKSHMI, R. & CHELLAPPAN, C. (2013). *Detecting Malicious URLs in E-mails- An Implementation*, vol. 4. AASRI.

[13] GANSTERER,W. N. & POLZ, D. (2009). *E-mail classification for phishing defense, in Advances in Information Retrieval*. Springer Berlin Heidelberg.

[14] GARETH JAMES, DANIELA WITTEN, T. H. AND TIBSHIRANI, R. (2009). An introduction to statistical learning with applications in r.

[15] GIRONÉS ROIG, J. & CASAS ROMA, J. (2017). *Minería de datos Modelos y algoritmos*. Editorial UOC.

[16] GOWTHAM, R. & KRISHNAMURTHI, I. (2014). *PhishTackle-a web services architecture for anti-phishing*. Cluster Compt.

[17] GUPTA, B., TEWARI, A., JAIN, A. K. AND AGRAWAL, P. (2017). Fighting against phishing attacks. *Communications in Partial Differential Equations*.

[18] HARÁN, J. M. (2020). En 2021 se registró pico histórico en la cantidad de sitios de phishing. *Welivesecurity by eseT*.

[19] HOTA, H.S., SHRIVAS A.K. AND HOTA, R. (2016). *An ensemble model for detecting phishing attack with proposed remove replace feature selection technique*. Procedia Comput. Sci.

[20] ISA, D., LEE, L., KALLIMANI, V. AND RAJKUMAR, R. (2008). Text document preprocessing using bayes formula for classification based on the vector space model. *Comput. Informat. Sci. J.*

[21] JAIN, A.K. & GUPTA, B.B. (2016). *A novel approach to protect against phishing attacks at client side using auto-updated white-list EURASIP*. J. Inf. Secur.

[22] JAMES, G. AND COL. (2020). An introduction to statistical learning. *DOI:10.1007/978-1-4614-7138-7*

[23] JAN, ZIZKA., FRANTISEK DARENA. AND SVOBODA, A. (2020). *Text Mining with Machine Learning Principles and Techniques*. Taylor & Francis Group. A SCIENCE PUBLISHERS BOOK.

[24] KAUR, D. & KALRA, S. (2016). Five-tier barrier anti-phishing scheme using a hybrid approach. *Inform. Secur. J.A Global Perspect.*

[25] KITTLER, J., HATEF, M., AND DUIN, R.P W. (1998) *On Combining Classifiers. Transactions on pattern analysis and machine intelligence*, vol. 20. IEEE.

[26] KUNCHEVA, L. (2004). Combining pattern classifiers. methods and algorithms. *John Wiley & Sons, New Jersey*.

[27] MARTÍNEZ, M. B. (2018). *Minería de Datos*. web: *http://bbeltran.cs.buap.mx/NotasMD.pdf.*

[28] MOGHIMI, M. & VARJANI, A. (2016). *New rule-based phishing detection method.* J.Exp.Syst.Appl.

[29] MONKEY.ORG. Data de correos electrónicos de monkey, (2020).

[30] ORUNSOLU, A.A., SODIYA, A.S. AND AKINWALE, A. (2019). A predictive model for phishing detection. *Journal of King Saud University - Computer and Information Sciences*

[31] PATRO, R. (2021). Cross-validation: K-fold vs monte carlo. *Towards Data Science*

[32] PEDREGOSA. (2011). *Scikit-learn: Machine Learning in Python*, vol. 12. JMLR, 2011.

[33] PHISHTANK. Base de datos de url, (2020).

[34] ROSERO GOMEZCOELLO, J. M. (2020). *Detección y mitigación de ataques de ingeniería social tipo Phishing utilizando minería de datos*. [Tesis de pregrado, Escuela Superior Politécnica del Ejercito]. Repositorio institucional de la Escuela Superior Politécnica del Ejercito. *http://repositorio.espe.edu.ec/*

[35] SEGAL, M. (2004). Machine learning benchmarks and random forest regression. [Tesis, University of California].

[36] SONOWAL, G. & KUPPUSAMY, K. (2017). Phidma-a phishing detection model with a multi-filter approach. *Information and Computer Security*.

[37] TAN C., CHIEW L. AND SZE, N. (2017). Phishing webpage detection using weighted url tokens for identity keywords retrieval.

[38] TREVOR, H., TIBSHIRANI, R., AND FRIEDMAN, J. (2017). The elements of statistical learning: data mining, inference, and prediction.

[39] WIKIPEDIA. Validación cruzada. (2020)

[40] WILLIAM, W. & COHEN, MLD.C. (2018). Base de datos de correos electrónicos de enron.

[41] ZHAO, J., WANG, N., MA, Q. AND CHENG, Z. (2019) *Classifying Malicious URLs Using Gated Recurrent Neural Networks*. Springer, Cham.

[42] ZOUINA, M. & OUTTAJ, B. (2019). A novel lightweight url phishing detection system using svm and similarity index.

APÉNDICE

Apéndice A

Proceso CRISP-DM

El estándar CRISP-DM fue desarrollado en 1996 por un consorcio de cinco empresas con experiencia personal significativa en el uso de tecnologías analíticas: Integral Solutions, Teradata, Daimler AG, NCR Corporation y OHRA.

CRISP-DM se ha convertido en la forma más utilizada para minería de datos. La desventaja de este modelo es la falta de apoyo a la gestión de proyectos. La ventaja de CRISP-DM es que es neutral en cuanto a dominios, herramientas y aplicaciones.

El estándar CRISP-DM presenta seis fases en el proceso de analizar problemas, datos y dar una idea o conocimientos para mejorar el desempeño del modelo como se observa en el cuadro 5.1. Fuente [10]

Fase	Breve descripción
Comprensión del problema	Se debe evaluar los recursos disponibles y necesarios. La determinación del objetivo. Primero se debe explicar el tipo de técnica que se pretende usar (Ej. Clasificación) y los criterios de éxito (como precisión). Adicionalmente se debe crear un plan del proyecto.
Comprensión de datos	Recopilar datos de fuentes confiables, explorarlos, describirlos y verificar la calidad de los datos son tareas esenciales en esta fase. Para hacerlo de forma más concreta, se realiza la descripción de datos utilizando análisis estadístico y determinando atributos.
Preparación de los Datos	La selección de datos debe realizarse mediante la definición de criterios de inclusión y exclusión. La mala calidad de los datos requiere la limpieza de los mismos. Dependiendo del modelo utilizado, se deben construir atributos derivados.
Modelado	La fase de modelado consiste en seleccionar la método de modelización, para esto se debe construir el caso de testeo y el modelo. En general, la elección depende del problema y el tipo de datos. Para construir el modelo, se deben establecer parámetros específicos. Para evaluar el modelo es apropiado evaluar el modelo contra los criterios de evaluación y seleccionar los mejores.
Evaluación	En la fase de evaluación, los resultados se comparan con los objetivos definidos. Por lo tanto, los resultados deben de interpretar y definir otras acciones.
Desarrollo	En fase de implementación se tiene con fin redactar un informe final o tener un componente de software.

CUADRO 5.1. Descripciones del proceso CRISP-DM. Fuente [10]

Apéndice B

En este Apéndice podemos encuentra los algoritmos en escritura de seudocódigo que se menciona en el capítulo dos (Marco Teórico), para estos algoritmos tomamos como fuente [14], [30] y [38]

1-B

Algorithm 1 Análisis de frecuencia de evaluación de características en URL

Entrada: Corpus de phishing actualizado, d_{ph}, URL, d_{be}, Valor umbral predefinido,θ.
Salida: Vector de dimensión de características basado en URL S_m

Empezar
1. Para $i = 1$ hasta n:
2. $F_{URL(n)} \leftarrow$ el conjunto de todas las n funciones de URL
3. Si $F_{url\_i} \in d_{ph}$ ó d_{be} Luego
4. $S \leftarrow$ nueva lista de funciones
5. Calcular la frecuencia de $F_{url\_i} \in d_{ph}.d_{be}$
6. Calcular la información de frecuencia,FI, de F_{url_i}
7. Si $FI_{F_{url\_i}} > \theta$, Luego
8. Adjuntar F_{url_i} a S
9. **Caso Contrario**
Rechazar F_{url_i}
10. Siguiente i
Continuar
11. Rango $F_{url\_i} \in S$
12. Seleccionar la parte superior F_{url_i} características $\in S$
13. Obtener una medida de desempeño de S
14. Identificar la mejor medida de desempeño como las mejores características
15. $S_m \leftarrow$ mejores características
16. Fin

El Cuadro 5.2 presenta el significado de las notaciones utilizadas en el algoritmo 1.

Notaciones	Descripciones
n	Número de funciones para el análisis de frecuencia
F_{url_i}	Una instancia de la función de URL
d_{ph}	Una base de datos de URL de phishing confirmadas
d_{be}	Una base de datos de URL legítimas confirmadas
θ	El umbral para el análisis de características
p	Tamaño del conjunto de datos
S_i	Categoría de función para la función F2
H_P	Funciones de phishing de alto impacto
$CFS(s)$	Función de selección de característica de correlación para s
f	Una instancia de característica en una categoría de característica particular
t	Factor de correlación (incertidumbre de simetría o correlación Coeficiente de Pearson)
a	Encimera
$f(s)$	El conjunto de características de alto impacto seleccionadas
$m(fs)$	El subconjunto de funciones de alto impacto seleccionadas para la detección de phishing

CUADRO 5.2. Lista de notaciones y sus significados. Fuente [30]

2-B

Algorithm 2 Análisis de frecuencia de evaluación de características

Entrada: Tamaño de datos(p), conjunto de características original(n), umbral(θ), clase(C).
Salida: subconjunto de características principales de dimensión $m(fs)$

Empezar

1. Para $i = 1$ hasta n:
2. Para $j = 1$ hasta p:
3. a=1
4. Seleccioanr $s_i \in H_P$
5. Calcular CFS usando {
6. $s(f_i, f_2, ..., f_n, c)$ como entrada
7. Para $i = 1$ hasta n:
8. Inicializar el factor de correlación apropiado, t
9. $r = calcular\_correlacion(f, c)$
10. Si $(t > r)$ luego
11. Adjuntar $f_i \in s_n$ en m
12. Fin
13. Ordenar m en valor descendente de t
14. Quitar f con rango inferior
15. Devuelver f predominante como $f(s)$
16. Fin}
17. Si $(f_i(s) > \theta)$ luego agregar a $m(fs)$
18. $c = c + 1$
19. Si $(a \leq n)$ volver a 3
20. Fin para
21. Fin para
22. Conjunto de características de retorno $m(fs)$
23. Fin

3-B

Algorithm 3 Construcción de un árbol de regresión

1. Utilizar la división binaria recursiva para el crecimiento de un árbol en los datos del conjunto de entrenamiento, parando siempre y cuando cada nodo terminal posee un mínimo número de observaciones.
2. Podar el árbol mediante la complejidad de costos para conseguir una cadena de subárboles, en función de α.
3. Mediante la validación cruzada de K-fold se elige α. para fraccionar las K-fold del conjunto de entrenamiento. Para cada $k = 1, ..., K$:
3a. Repetir los pasos 1 y 2 en todos menos el $k - simo$ pliegue de los datos de entrenamiento.
3b. Evaluar el error de predicción cuadrático medio de los datos en el $k - simo$ pliegue omitido, en función de α.
Obtener el promedio de los resultados para cada uno de los valores α y elegir α para reducir el error medio.
4. Reponer el subárbol del segundo paso, que corresponde al valor elegido de α.

4-B

Algorithm 4 Random Forests para regresión o clasificación.

1. Para $b = 1$ hasta B:
2. Seleccionar una muestra de Bootstrap Z^* de tamaño N a partir de los datos de entrenamiento.
3. Generación de Random Forests T_b hacia los datos de Bootstrapped, repitiendo recursivamente los siguientes pasos para cada nodo terminal del árbol, hasta el mínimo nodo n_{min} es alcanzado.
4. Seleccionar m variables al azar de las p variables de entrada.
5. Elegir la mejor variable / punto de división de m.
6. Dividir el nodo en dos nodos secundarios.
7. Generar el conjunto de árboles $T_b{}_1^1$
8. Para hacer una predicción en un nuevo punto x:
9. Regresión: $\hat{f}_{rf}^B(x) = \frac{1}{B}\sum_{b=1}^{B} T_b(x)$
10. Clasificación: Dado $\hat{C}_B(x)$ la predicción de clase del B_{th} bosque aleatorio. Entonces

$$\hat{C}_{rf}^B(x) = \text{mayoría de votos}\hat{C}_b(x)_1^B$$

.

5-B

Algorithm 5 Clasificador Bayes Naïve(NBC). Fuente [30]

Entrada: vector de características, F_v; conjunto de datos de entrenamiento (D)
Salida: L, el resultado de la clasificación
Empezar

1. Inicializar cada $F \in F_v$ extraído de W
2. Mientras que $F_v \in W \neq 0$ hacer {
3. Calcular la probabilidad previa $P(C_k)$ para cada clase C_k
4. Calcular la probabilidad condicional de $P(A_i \mid C_i)$ para cada $F \in F_v$
5. Clasificar cada ejemplo de entrenamiento, $e_i \in D$, con probabilidades posteriores máximas
6. Actualizar las probabilidades de cada $e \in D$ basado en la probabilidad de clasificación
7. Clasificar de nuevo $e \leftarrow L$}
8. Terminar Mientras
9. Fin

Apéndice C

Las fuentes que se utilizamos para este Apéndice son: [29], [33] y [40]

Descripción del conjunto de datos PhishTank, Monkey y Enron

- **Base de datos PhishTank**
  Se partió de este conjunto de datos, dado que PhishTank es un servicio de seguridad basado en la comunidad que contiene una base de datos de URL de phishing verificadas y etiquetadas informadas por una comunidad de voluntarios humanos. Los usuarios de Internet envían sitios web que sospechan que son estafas de phishing y otros votan si están de acuerdo o no con el remitente. El resultado son los mejores datos de phishing disponibles, tanto los más oportunos como los más precisos. Además, brinda la facilidad para descargar repositorios de datos o código para la identificación registro concreto de Phishing.

- **Base de datos correos Monkey**
  Es una comunidad, donde un conjunto de personas que se dedican al estudio de robo de información, en correos electrónicos, en este sistema se suben diariamente correos infectados, a un Dataset que está libre para todos los que necesiten realizar investigaciones acerca de este tipo de ataque en base a los correos guardados en esta data.

- **Base de datos correos Enron** De forma similar al item anterior la base está compuesta por correos electrónicos reales en otras palabras no tienen presencia de phishing. Esta data, consta de aproximadamente 5000 registros. Este tipo de datos nos permite balancear para obtener una buena clasificación y aprendizaje del modelo.

  Esta data fue recopilada y preparada por el proyecto CALO[1].

A continuación, se describirá las variables presentes en cada data anteriormente descritas:

Descripción de las Variables

Comenzamos el conjunto de datos de PhishTank, que se puede visualizar en el cuadro 5.3, tengamos en cuenta que esta base solo contiene URL_s siendo este el paso inicial para este estudio.

Variable	Descripción
phish_id	Es el número de identificación con el que Phishtank se refiere a un envío de phish. Todos los datos de PhishTank están vinculados a esta identificación. Siempre será un número entero positivo.
phish_detail_	URL detallada de PhishTank para la suplantación de identidad, donde puede ver datos sobre la suplantación de identidad, incluida una captura de pantalla y los votos de la comunidad.

[1] CALO: Asistencia cognitiva para aprendizaje y organización

url	La URL de suplantación de identidad. Siempre es una cadena, y en las fuentes XML puede haber un bloque CDATA.
submit_time	La fecha y hora en las que se informó a Phishtank de este phish. Esta es una fecha con formato ISO 8601.
verificado	Si nuestra comunidad ha verificado o no está suplantación de identidad. En estos archivos de datos, siempre será la cadena "sí", ya que solo proporcionamos phishing verificados en estos archivos.
verify_time	La fecha y hora en las que nuestra comunidad verificó la suplantación de identidad como válida. Esta es una fecha con formato ISO 8601.
online	Si la suplantación de identidad está online y operativa. En estos archivos de datos, siempre será la cadena "sí", ya que solo proporcionamos phishing en línea en estos archivos.
target	El nombre de la empresa o marca a la que se hace pasar el phish, si se conoce.

Cuadro 5.3: Descripción de las variables de Phish-Tank. Fuente [33]

A continuación, se presenta listará las variables que se tienen tanto para la base de datos **Monkey** como **Enron** cabe mencionar que la extensión de estas datas son M.box, observemos estas variables en el cuadro 5.4.

Variable	Descripción
Return-Path	Es la dirección de donde procede del correo.
X-Original-To	En esta variable se tiene la dirección de destino
Delivered-To	Dirección de destino final del correo
Received	Dominio del receptor.
From	Nombre y correo del emisor.
To	Nombre y correo del receptor.
Subject	Asunto del correo.
Sender	Constatación del cibernauta que envía el mensaje.
User-Agent	Identificación del sistema operativo y el proveedor.
X-Priority	Grado de prioridad que se da al mensaje.
MIME-Version	Versión estándar de formato del mensaje.

Content-Type	Esta variable permite analizar el corpus del correo electrónico.
Message-Id	Código único de identificación del correo.
Date	Fecha de recepción del correo.
Content-Length	Tamaño del mensaje.
Lines	Número de líneas en el email.
Status	Estado del correo.

Cuadro 5.4: Descripción de las variables de Monkey y Enron. Fuente [29] y [40]

Apéndice D

Algoritmo para detección de características implementado en Java

```
"""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""
DIRECTORIO DE TRABAJO
"""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""
"""EPN
"""AUTOR: FERNANDA ALBÁN
"""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""
EXTRACCIÓN DE DATOS DE LAS DIVERSAS CUENTAS

"""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""
"""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""
MONGO

"""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""
package com.alban.fer.spam.conf;

import lombok.NonNull;
import org.bson.types.Decimal128;
import org.springframework.context.annotation.Bean;
import org.springframework.context.annotation.Configuration;
import org.springframework.core.convert.converter.Converter;
```

```
import org.springframework.data.convert.ReadingConverter;
import org.springframework.data.convert.WritingConverter;
importorg.springframework.data.mongodb.core.convert.MongoCustomConversions;

import java.math.BigDecimal;
import java.math.RoundingMode;
import java.time.ZonedDateTime;
import java.util.Arrays;
import java.util.Date;

import static java.time.ZoneId.systemDefault;
import static java.time.ZonedDateTime.ofInstant;


@Configuration
public class MongoConf {
    @Bean
    public MongoCustomConversions customConversions() {
        return new MongoCustomConversions(Arrays.asList(
                new DateToZonedDateTimeConverter(),
                new ZonedDateTimeToDateConverter(),
                new BigDecimalDecimal128Converter(),
                new Decimal128BigDecimalConverter()
        ));
```

```
}

class DateToZonedDateTimeConverter implements Converter<Date, ZonedDateTime> {

    @Override
    public ZonedDateTime convert(Date source) {
        return source == null ? null : ofInstant(source.toInstant(), systemDefault());
    }
}

class ZonedDateTimeToDateConverter implements Converter<ZonedDateTime, Date> {

    @Override
    public Date convert(ZonedDateTime source) {
        return source == null ? null : Date.from(source.toInstant());
    }
}

@WritingConverter
private static class BigDecimalDecimal128Converter implements
 Converter<BigDecimal, Decimal128> {

    @Override
    public Decimal128 convert(@NonNull BigDecimal source) {
```

```
            if (source.scale() > 9) {
                source = source.setScale(10, RoundingMode.CEILING);
            }
            return new Decimal128(source);
        }
    }

    @ReadingConverter
    private static class Decimal128BigDecimalConverter implements
     Converter<Decimal128, BigDecimal> {

        @Override
        public BigDecimal convert(@NonNull Decimal128 source) {
            return source.bigDecimalValue();
        }

    }

}

""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""
WEB

""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""
```

```
package com.alban.fer.spam.conf;

import org.springframework.context.annotation.Configuration;
import org.springframework.web.servlet.config.annotation.CorsRegistry;
import org.springframework.web.servlet.config.annotation.WebMvcConfigurer;

@Configuration
public class WebConf implements WebMvcConfigurer {

    @Override
    public void addCorsMappings(CorsRegistry registry) {
        registry.addMapping("/**")
                .allowedOrigins("*").allowCredentials(true)
                .allowedMethods("*")
                .maxAge(3600);
    }
}
"""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""
WEBSOCKETSTOMP

"""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""
package com.alban.fer.spam.conf;

import org.springframework.beans.factory.annotation.Value;
```

```
import org.springframework.context.annotation.Configuration;
import org.springframework.messaging.simp.config.MessageBrokerRegistry;
import org.springframework.web.socket.config.annotation.EnableWebSocketMessageBroker;
import org.springframework.web.socket.config.annotation.StompEndpointRegistry;
import org.springframework.web.socket.config.annotation.WebSocketMessageBrokerConfigurer;


@Configuration
@EnableWebSocketMessageBroker
public class WebSocketStompConfig implements WebSocketMessageBrokerConfigurer {

    @Value("${sec.origins}")
    String origins;

    @Override
    public void registerStompEndpoints(StompEndpointRegistry registry) {
        registry.addEndpoint("/ws").setAllowedOrigins(origins.split(","));

    }

    @Override
    public void configureMessageBroker(MessageBrokerRegistry registry) {
        registry.enableSimpleBroker("/topic");
        registry.setApplicationDestinationPrefixes("/app");
```

```
    }

}

"""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""
APLICACIÓN PARA DETECTAR SPAM (SpamDetectorApplicationTests)

"""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""
package com.alban.fer.spam.spamdetector;

import com.alban.fer.spam.srv.CronMailSrv;
import com.alban.fer.spam.srv.DomainSrv;
import com.alban.fer.spam.srv.MensajeSrv;
import com.google.common.net.InternetDomainName;
import org.junit.jupiter.api.Test;
import org.nibor.autolink.LinkExtractor;
import org.nibor.autolink.LinkSpan;
import org.nibor.autolink.LinkType;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.boot.test.context.SpringBootTest;

import java.net.MalformedURLException;
import java.net.URL;
import java.time.ZonedDateTime;
```

```
import java.util.ArrayList;
import java.util.EnumSet;
import java.util.List;


@SpringBootTest(webEnvironment = SpringBootTest.WebEnvironment.RANDOM_PORT)
class SpamDetectorApplicationTests {

    @Autowired
    CronMailSrv cronMailSrv;

    @Autowired
    MensajeSrv mensajeSrv;

    @Autowired
    DomainSrv domainSrv;

    @Test
    void contextLoads() throws InterruptedException {
        cronMailSrv.leerMails();
    }

    @Test
    void testMensaje() {
```

```
    var doc = mensajeSrv.findOne("603935e461d69f66f21ec565");
    // todo
    mensajeSrv.doProcessFields(doc);
    System.out.println(doc);
}

@Test
void testDomainSrv() {
    domainSrv.checkIfIndexed("qorimed.com");
    domainSrv.checkIfIndexed("app.qorimed.com");
    domainSrv.checkIfIndexed("qorimedacsfg.com");
}

@Test
void testRegisterDate() {
    var now = ZonedDateTime.now();
    System.out.println(domainSrv.checkRegisterDate("qorimed.com", now));
    System.out.println(domainSrv.checkRegisterDate("app.qorimed.com", now));
    System.out.println(domainSrv.checkRegisterDate("qorimedacsfg.com", now));
    System.out.println(domainSrv.checkRegisterDate("yanibot.jedai.group", now));
}

@Test
void testUrl() {
```

```
    var msm = "hola http://qorimed.com http://192.168.12.4:8080/xxx";

    LinkExtractor linkExtractor = LinkExtractor.builder()
            .linkTypes(EnumSet.of(LinkType.URL, LinkType.WWW))
            .build();
    Iterable<LinkSpan> links = linkExtractor.extractLinks(msm);

    List<String> result = new ArrayList<>();
    links.forEach(span -> result.add(msm.substring(span.getBeginIndex(),
    span.getEndIndex())));
    System.out.println(result);
}

@Test
public void xxxx() {
    var domain = "bancopichincha.com.godaddy.com";
    System.out.println(domainSrv.checkHasMultipleTld(domain, getDomain(domain)));
    domain = "bancaweb.bancopichincha.com";
    System.out.println(domainSrv.checkHasMultipleTld(domain, getDomain(domain)));
    domain = "app.bancaweb.bancopichincha.com";
    System.out.println(domainSrv.checkHasMultipleTld(domain, getDomain(domain)));
    domain = "www.bancopichincha.com.ec";
    System.out.println(domainSrv.checkHasMultipleTld(domain, getDomain(domain)));
```

```
    }

    private String getDomain(String url) {
        return InternetDomainName.from(parseURL(url).getHost()).topPrivateDomain()
        .toString();
    }

    public URL parseURL(String url) {
        try {
            if (!url.startsWith("http")) {
                url = "https://" + url;
            }
            return new URL(url);
        } catch (MalformedURLException e) {
            return null;
        }
    }
}
```

Algoritmo RF y NB implementado en Python

```
""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""
RANDOM FORESTS AND NAIVE BAYES MODEL
""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""
LIBRERIAS UTILIZADAS
```

```
"""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""
from sklearn.ensemble import RandomForestRegressor
from sklearn.naive_bayes import MultinomialNB
from sklearn import tree
from sklearn.metrics import accuracy_score, precision_score, confusion_matrix
from sklearn.metrics import f1_score, recall_score, roc_auc_score, roc_curve
import matplotlib.pyplot as plt
import numpy as np

"""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""
CARGA DEL ARCHIVO CON LAS CARACTERISTICAS SELECCIONADAS
"""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""


def load_data():
    """
    Esta función auxiliar carga el conjunto de datos guardado en el archivo
    CSV y devuelve 4 matrices que contienen las entradas del conjunto de
    entrenamiento y etiquetas, y las entradas y etiquetas del conjunto de
    pruebas.
    """

    # Cargar los datos de entrenamiento del archivo CSV
    training_data = np.genfromtxt('datasetprueba.csv', delimiter=',', dtype=np.int32)
```

```
    """
   Cada fila del archivo CSV contiene las características recopiladas si se tuvo
   suplantación de identidad o no. Ahora separamos las entradas (características
   recopiladas) de las etiquetas de salida
    """

    # Extraccion de las entradas de la matriz de datos de entrenamiento
    #(todas las columnas menos la última)
    inputs = training_data[:,:-1]

    # Extraccion de los resultados de la matriz de datos de entrenamiento
    #(última columna)
    outputs = training_data[:, -1]

"""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""
ENTRENAMIENTO Y TESTEO
"""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""
    # Entrenamiento y los datos de prueba
    training_inputs = inputs[:10502]
    training_outputs = outputs[:10502]
    testing_inputs = inputs[10502:]
    testing_outputs = outputs[10502:]

    # Devuelve las cuatro matrices
```

```
    return training_inputs, training_outputs, testing_inputs, testing_outputs

"""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""
ENTRENAMIENTO DE LOS MODELOS
"""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""
if __name__ == '__main__':
    print (" Entrenamiento de los modelos para detectar phishing")

    # Carga de los datos de entrenamiento
    train_inputs, train_outputs, test_inputs, test_outputs = load_data()
    print ("Datos de entrenamiento cargados.")

    # Crear un modelo de clasificacion "naive bayes" usando scikit-learn
    model = RandomForestRegressor()
    model.fit(train_inputs, train_outputs)

    print ("Se creó el modelo de clasificacion "Random Forests")

    # Crear un modelo de clasificacion "naive bayes" usando scikit-learn
    model = MultinomialNB()
    model.fit(train_inputs, train_outputs)

    print ("Se creó el clasificador bayesiano.")
    # Crear un modelo de clasificacion "árbol de decisión" usando scikit-learn
```

```
    classifier = tree.DecisionTreeClassifier()
    print ("Se creó el clasificador de árbol de decisión.")

    print ("Entrenamiento de modelos.")
    # Entrenar el clasificador "árbol de decisión"
    classifier.fit(train_inputs, train_outputs)
    print ("Formación de modelos completo.")

    # Utilizar el clasificador entrenado para hacer predicciones sobre
    #los datos de prueba
    predictions = classifier.predict(test_inputs)
    print ("Predicciones sobre los datos de prueba calculados." )

"""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""
METRICAS DEL MODELO
"""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""

    # Imprima la matriz de confusión
    mc=confusion_matrix(test_outputs, predictions)
    print ("La matriz de confusión sobre los datos de prueba es: " + str(mc))

    # Imprima la exactitud (porcentaje de sitios web de phishing predicho correctamente)
    accuracy = 100.0 * accuracy_score(test_outputs, predictions)
    print ("La exactitud del Random Forests sobre los datos de prueba es: "
```

```
      + str(accuracy))

# Imprima la precisión (porcentaje de sitios web de phishing predicho correctamente)
precision = 100.0 *precision_score(test_outputs, predictions)
print ("La precisión del Random Forests sobre los datos de prueba es: "
      + str(precision))

# Imprima el recall/recuperación
recall = 100.0 *recall_score(test_outputs, predictions)
print ("La recall/recuperación del Random Forests sobre los datos de prueba es: "
      + str(recall))

# Imprima la puntuación F1
f1 = 100.0 *f1_score(test_outputs, predictions)
print ("La puntuación F1 del Random Forests sobre los datos de prueba es: "
      + str(f1))

# Imprima el valor roc
roc_auc = 100.0 *roc_auc_score(test_outputs, predictions)
print ("El valor roc del Random Forests sobre los datos de prueba es: "
      + str(roc_auc))
```

Apéndice E

Tablas de Siglas y Acrónimos

En los siguientes cuadros se presenta el significado de siglas y palabras técnicas empleadas en el trabajo de investigación para tener un mejor entendimiento al momento de leer el documento.

Siglas	
APWG	Anti-Phishing Working Group
CSO	Chief Security Officer
APS	Anti-Phishing System
CRISP-DM	Cross Industry Standard Process for Data Mining
FSM	Módulo de selección de características
TF-IDF	Término-Frecuencia Documento Frecuencia Inversa
ROC	Receiver Performance Characteristic
URL	Uniform Resource Locator
TP	Verdadero Positivo
TN	Verdadero Negativo
FP	Falsos Positivos
FN	Falsos Negativos

CUADRO 5.5. Abreviaturas y Siglas. Fuente [30] y [34]

	Acrónimo
1	**Ataque:** acción mediante un equipo electrónico para el robo de información.
2	**Ciberseguridad:** llamado también seguridad informático tiene como principio la protección y el procesamiento de la información.
3	**Phishing:** es un ataque que tiene su principal función el robo de información confidencial (suplantación).
4	**Malware:** es un término utilizado para los softwares maliciosos (virus).
5	**URL:** es una dirección que apunta a un recurso único en la Web.
6	**Corpus del email:** es el cuerpo textual del correo electrónico.
7	**Dominio:** es una dirección web compuesta por un nombre y una extensión de un sitio especifico.
8	**Ruta:** es una forma de referenciar un archivo o directorio a un sistema.
9	**Direccion IP:** es una etiqueta numérica, que identifica a una interfaz en la red de un dispositivo inteligente que usa internet.

10	**Anti-Phishing System:** es un sistema que permite vigilar el tráfico de aplicaciones para detectar intentos de ataques cibernéticos.
11	**Ransomware:** es una manera diferente de decir malware de rescate.
12	**Phisher:** persona que ataca mediante la técnica de robo de información (Phishing).
13	**Machine Learning:** en español aprendizaje automático, es una rama de la inteligencia artificial.
14	**Fiabilidad:** es un dispositivo que cumple con determinada función bajo ciertas condiciones.
15	**Http:** es el protocolo de red que aprueba la transferencia de documentos en la red, de manera general entre un navegador y un servidor.
16	**Código Malicioso:** son softwares no legítimos, que ayudan al robo de información.
17	**Lista negra:** es una base de direcciones IP y dominios censurados, contienen spam o virus.
18	**M.box:** es una extensión usada específicamente para conjuntos con un formato de correo electrónico.

Cuadro 5.6: Acrónimos. Fuente [30], [34] y [41]