ANÁLISIS DE UN MODELO PARA IDENTIFICAR ALERTAS TEMPRANAS ANTE ATAQUES DE PHISHING

JOSE ALFONSO VALENCIA RODRIGUEZ

ESCUELA COLOMBIANA DE INGENIERÍA JULIO GARAVITO
FACULTAD DE INGENIERIA
MAESTRÍA EN GESTIÓN DE INFORMACIÓN
BOGOTA D.C.
2016
ANÁLISIS DE ARBOLES DE DECISION PARA IDENTIFICAR ALERTAS TEMPRANAS ANTE ATAQUES DE PHISHING

JOSE ALFONSO VALENCIA RODRIGUEZ

TRABAJO DE GRADO

Director de proyecto

Ing. Ignacio Pérez Vélez PhD.

ESCUELA COLOMBIANA DE INGENIERÍA JULIO GARAVITO
FACULTAD DE INGENIERIA
MAESTRÍA EN GESTIÓN DE INFORMACIÓN
BOGOTA D.C.
2016

Nota de Aceptación

_____

_____

_____

_____

_____

_____

Firma Presidente del Jurado

_____

Firma del Jurado

_____

Firma del Jurado

Bogotá D.C. 16 Mayo de 2016

CONTENIDO

LISTA DE TABLAS

Pág.

LISTA DE FIGURAS

ANEXOS

pág.

1. INTRODUCCION

La comunidad cibernauta es víctima continua de ataques informáticos basados en suplantación y robo de identidad, según estudios realizados por Symantec[1], donde se indica el incremento de incidentes de seguridad en la Web, afectando con esto la confianza de los usuarios que se conectan diariamente. La información presenta riesgos, amenazas, secuestro de información (*ransomware*) y ataques de Phishing. Según Gómez Vieites[2], los tipos de ataques informáticos son: "Activos, los cuales producen cambios en la información y los recursos del sistema. Los Pasivos se limitan a registrar el uso de los recursos y acceder a la información guardada o transmitida por el sistema. Además existen incidentes como los *Data Breach* o violación de datos, que se originan cuando la información es robada u obtenida de un sistema, sin el conocimiento o autorización de su propietario[3]. El Phishing, busca obtener información confidencial de los usuarios, y es el tema analizado en este trabajo. El Malware es software que intercepta y genera daños en los recursos informáticos y se usa para materializar phishing porque modifica, altera o suplanta información de los usuarios y procesos, mediante algunas modalidades como *backdoors*, virus, troyanos, *Hijackers* y *botnets*. El phishing, se define como una forma de robo de identidad, que sucede a partir de una serie de técnicas de ingeniería social y mediante el uso de código malicioso, donde el delincuente logra recolectar la mayor cantidad de información confidencial de usuarios incautos, para luego cometer fraude, se presenta cuando un delincuente envía un correo electrónico, que contiene enlaces dirigidos a una página falsa, que suplanta la identidad de una empresa o institución, buscando que la víctima ingrese sus datos privados, y luego ser víctima de suplantación. Otra forma es cuando un usuario

[1] http://www.symantec.com/security_response/publications/threatreport.jsp. Consultado Junio 2015

[2] Gómez Vieites, Álvaro. Enciclopedia de la Seguridad Informática. 2ª Edición Actualizada. ISBN978-84-9964-036-5. Ed. Alfa Omega. México DF. 2011. 825 p.

[3] Diaz-Gomez, Pedro, Valencia Rodríguez José Alfonso and Gómez Luis E. H. Management - An Achilles Heel of Information Assurance Security: A Case Study of Verizon's Data Breach Reports. Cameron University, Lawton, Oklahoma, USA. Universidad Piloto de Colombia. RRP. http://search.proquest.com/openview/82859422c7e73b4b71ec287f2a8622d2/1?pq-origsite=gscholar

recibe mensajes de texto enviados desde redes sociales con vínculos fraudulentos. En algunos casos se usa la dirección IP y algunos códigos de programación que ocultan la barra de direcciones del browser. Para identificar este tipo de ataque se analizó la estructura de la URL, que contiene características para identificar su legitimidad. La pregunta que se buscó resolver fue: *¿Cómo identificar alertas tempranas ante ataques de Phishing?* Como solución al problema se analizaron algunos algoritmos basados en arboles de decisión empleados en minería de datos, usando conjuntos de datos (*dataset*) extraídos de comunidades orientadas a identificar ataques de Phishing, mediante un proceso de Crawling. Para procesar el análisis se implementó un prototipo[4] de software, el cual genero la entrada de datos. El análisis permitió realizar una guía de buenas prácticas, aportando recomendaciones de tipo técnico.

La propuesta se estructuró de la siguiente manera. En el Capítulo 1: presenta el alcance, el impacto, la justificación y los objetivos. El capítulo 2 contiene el marco teórico, el estado del arte del problema objeto de la investigación y la situación actual. El Capítulo 3 desarrolla el escenario de análisis usando minería de datos con los diferentes algoritmos basados en arboles de decisión. El Capítulo 4 presenta la comparación y análisis de resultados obtenidos. Luego las conclusiones y trabajos futuros. Finalmente se presenta la bibliografía y anexos. La guía de buenas prácticas será un anexo.

ANTECEDENTES

Algunos estudios de impacto mundial muestran el impacto de los ataques informáticos, como el realizado por la firma Symantec, titulado Internet Security Threat Report 2014, que reveló los siguientes datos:

[4] Pressman, Roger. Ingeniería de Software. Un enfoque práctico. Tercera Edición. Mc Graw-Hill. España. 1994. Pág. 199

- Más de 552 millones de identidades fueron expuestos a través de violaciones en 2013.
- 23 vulnerabilidades de día cero se descubrieron.
- 1 de cada 392 correos electrónicos contienen un ataque de Phishing.
- 1 de cada 8 sitios web legítimos tienen una vulnerabilidad crítica.

Así mismo en el Informe 2015 *Internet Security* se enfocó a visualizar los aspectos más destacados de la amenaza y como los cibercriminales atentan contra la seguridad de los usuarios. Según el informe, algunos hallazgos fueron:

- Los ciberataques vulneran las defensas de las organizaciones.
- Los ataques *ransomware* se incrementaron en 2014.
- Las redes sociales y sus aplicaciones permiten la acción de los ciberdelincuentes.
- Los mecanismos de defensa de las organizaciones fallan frente a los ataques de los ciberdelincuentes.
- Los ataques a los dispositivos orientados al Internet de las cosas generan mayores impactos.

Otros estudios[5] , visualizan los aspectos más destacados de las amenazas y como los cibercriminales están vulnerando la seguridad de los usuarios. Se hace énfasis en las vulnerabilidades de las redes sociales como la extorsión digital. El informe titulado "Predicciones de seguridad para 2015" realizado por PandaLabs[6], muestra algunos de los más frecuentes ataques dirigidos, entre los que se encontraron los siguientes:

- Cryptolocker: Consiste en ingresar a un equipo víctima, cifrando toda su información, y cobrando por su rescate bitcoins, para no ser rastreados.

[5] http://www.symantec.com/security_response/publications/threatreport.jsp. Consultado Junio 2015
[6] http://www.pandasecurity.com/spain/mediacenter/noticias/predicciones-seguridad-2015/. Realizado por Marta López. Diciembre 30, 2014

- Terminales de punto de venta: Generan robo de información en tarjetas de crédito de los clientes.
- APT: (Advanced Persisten Threats): Ataque dirigido a empresas o instituciones estratégicas, sin ser detectado fácilmente.

Otro antecedente fue publicado por la revista portafolio del día 16 de octubre de 2015, titulado: "En américa latina el 98,5% de los riesgos bancarios son informáticos: Los riesgos más frecuentes son la clonación de tarjetas de crédito, suplantación de identidad en compras no presenciales y "Phishing".[7]

OBJETIVOS DEL PROYECTO

OBJETIVO GENERAL

Desarrollar un análisis de los modelos de árboles de decisión usados en minería de datos para identificar ataques de phishing de forma temprana.

OBJETIVOS ESPECÍFICOS

- Describir diversos métodos para identificar ataques de Phishing.
- Desarrollar un prototipo de Software para generar entradas a los modelos analizados.
- Comparar los resultados del análisis de los modelos basado en árboles de decisión para identificar ataques de Phishing.
- Desarrollar guía de buenas prácticas.

JUSTIFICACION

Según Mohamed y otros[8], en la actualidad muchas compañías presentan vulnerabilidades en su proceso de pagos mediante comercio electrónico. El phishing

[7] Estudio presentado por Felaban durante el Congreso de Seguridad Bancaria en Panamá. Consultado el día 16 de Octubre de 2015.

[8] Mohammad, Rami, McCluskey, and Thabtah, Fadi (2012) An Assessment of Features Related to Phishing Websites using an Automated Technique. In: International Conference For Internet Technology And Secured Transactions. ICITST 2012. IEEE, London, UK, pp. 492-497. ISBN 978-1-4673-5325-0. Disponible en

suplanta el sitio web de una empresa, para obtener de forma ilegal la información confidencial de sus clientes. Existen actualmente estudios sobre las características ubicadas en las URL's que contienen phishing, como: presencia de prefijo y sufijo de dominio, la longitud y la presencia de IP dentro de ella. Otro estudio[9], concluyó que identificando características, se detectan amenazas de suplantación. Algunos autores sugieren el uso de minería de datos con algoritmos de clasificación para extraer nuevas reglas, modificar las existentes y añadir algunas que permitan identificar Phishing. Thabtah[10] y otros investigadores, validaron el uso de la minería de datos, afirmando que la presencia de phishing en Internet genera vulnerabilidades a los usuarios, originando pérdidas financieras, robo de identidad, daño de información privada, perdida de reputación y de confianza en procesos de comercio electrónico. Otro de sus estudios[11] busco predecir ataques de phishing, a partir de reglas de clasificación por inducción, las cuales demostraron ser confiables.

Existen algunas herramientas colaborativas, que permiten a los usuarios identificar sitios fraudulentos, con solo ingresar la dirección electrónica, para determinar si está afectada o no por Phishing, las cuales ofrecen base de datos de victimas de fraude y algunas API para interactuar con otras herramientas de seguridad. Lo anterior permitió realizar análisis sobre este tipo de ataque y proponer mecanismos que permitieran identificar alertas tempranas de phishing a través de las URL's, usando varios modelos basados en árboles de decisión, porque son fáciles de entender e

URL:http://ieeexplore.ieee.org/xpl/login.jsp?tp=&arnumber=6470857&url=http%3A%2F%2Fieeexplore.ieee.org%2Fxpls%2Fabs_all.jsp%3Farnumber%3D6470857

[9] An Assessment of Features Related to Phishing Websites using an Automated Technique. Mohammad, Rami, McCluskey, T.L. and Thabtah, Fadi Abdeljaber (2012) In: International Conference for Internet Technology and Secured Transactions. ICITST 2012. IEEE, London, UK, pp. 492-497. ISBN 978-1-4673-5325-0

[10] Mohammad, Rami, Thabtah, Fadi Abdeljaber and McCluskey, T.L. (2014) Predicting phishing websites based on self-structuring neural network. Neural Computing and Applications, 25 (2). pp. 443-458. ISSN 0941-064. Disponible en URL: http://link.springer.com/article/10.1007/s00521-013-1490-z

[11]Mohammad, Rami, McCluskey, T.L. and Thabtah, Fadi Abdeljaber (2014) Intelligent Rule based Phishing Websites Classification. IET Information Security, 8 (3). pp. 153-160. ISSN 1751-8709. Disponible en URL: http://ieeexplore.ieee.org/xpl/login.jsp?tp=&arnumber=6786863&url=http%3A%2F%2Fieeexplore.ieee.org%2Fxpls%2Fabs_all.jsp%3Farnumber%3D6786863

interpretar, requieren poca preparación de los datos y se pueden validar usando técnicas estadísticas. Este tema ha sido un problema de estudio para muchos investigadores, además se han registrado varios adelantos aplicando minería de datos obteniendo valores de precisión altos, con el uso de métodos analíticos eficaces y eficientes. A partir de lo anterior, se realizó este estudio para identificar alertas tempranas y proponer buenas prácticas, dando respuesta a la pregunta ¿Cómo puedo identificar alertas tempranas ante los ataques de Phishing? [12].

IMPACTO Y PERTINENCIA DE LA INVESTIGACIÓN

El impacto que proporciona el desarrollo del proyecto, es desarrollar un análisis con métodos de clasificación basado en árboles de decisión usados en minería de datos para identificar phishing a partir de algunas características contenidas en las URL´s, y recomendar buenas prácticas que eviten a los usuarios ser víctimas de actos fraudulentos. Lo pertinente se enfoca en identificar características para reducir la cantidad de víctimas de phishing. Su relevancia está dada en el hecho que es un tema de actualidad, como lo indican algunos autores que han desarrollado investigación sobre la temática y han aportado recomendaciones, buenas prácticas y mecanismos de solución a los usuarios.

RESULTADOS POSIBLES Y POTENCIALES BENEFICIARIOS

El análisis de modelos de minería de datos que permita identificar alertas tempranas en ataques phishing, servirá para que usuarios y organizaciones identifiquen en las URL’s características que representan amenazas y pongan en riesgo su información confidencial. Los potenciales beneficiarios serán toda la comunidad cibernauta y las asociaciones que investigan medidas de protección a los usuarios de la web.

[12] Hernández Orallo, José y Ramírez Quintana María José. Introducción a la minería de datos. Pearson Prentice Hall. Madrid. 2004. Pág. 5.

ALCANCE

Con el desarrollo del presente proyecto se busca analizar características contenidas en las URL´s que pueden llevar a ataques de Phishing, identificando alertas tempranas para evitar su incidencia a partir de un modelo de minería de datos.

PLANTEAMIENTO DEL PROBLEMA.

A partir de la pregunta: *¿Cómo identificar alertas tempranas ante los posibles ataques de Phishing?* , se buscó definir una alternativa para identificar características y tomar acciones oportunas. Se contempló el uso de un método de clasificación, a partir de los árboles de decisión. Algunos referentes son estudios que buscan identificar[13] y generar metodologías[14] para evitar este ataque.

[13] http://wenke.gtisc.gatech.edu/papers/esorics_paper_2004.pdf. Discovering Novel Attack Strategies from INFOSEC Alerts. Xinzhou Qin and Wenke Lee. College of Computing. Georgia Institute of Technology. Atlanta, GA 30332, USA. xinzhou, wenke@cc.gatech.edu. 2004. Consultado el 01 de Octubre de 2015

[14] Efforts and Methodologies used in Phishing Email Detection and Filtering: A Survey. Ripsa Khadir y Sony P. http://bibliotecavirtual.escuelaing.edu.co:2181/ehost/detail/detail?vid=4&sid=7619765f-5ecb-4afa-b47c-79fbb9abd2f8%40sessionmgr4004&hid=4214&bdata=JnNpdGU9ZWhvc3QtbGl2ZQ%3d%3d#AN=102304620&db=aci. Consultado 18 Octubre de 2015.

2. MARCO TEORICO

ESTADO DEL ARTE

Identificar ataques Phishing, es un problema complejo de analizar, por lo cual se usan diversas técnicas como la minería de datos, que identifica nuevos patrones y comportamientos con la información, usando métodos de clasificación basado en árboles de decisión, buscando identificar características que llevan a materializar phishing. Un estudio basado en este método lo realizaron *Guang-Gang Geng, Xiao-Dong Lee and Yan-Ming Zhang*[15], con datos extraídos de www.phishtank.com y www.apwg.com, identificando verdaderos positivos en una tasa del 98,8% y falsos positivos, es decir URL's legítimas clasificadas como afectadas por phishing en una proporción del 0,09% concluyendo que características como presencia de favicon, logotipos y avisos de copyright, se usan dentro de las URL's para engañar a las víctimas y materializar el ataque. Una recomendación dada es realizar más experimentos con otros datos. Otro estudio titulado: "*Methods of the data mining and machine learning in computer Security*"[16] basados en métodos de clasificación con máquinas de aprendizaje, lograron distinguir ciertos patrones anormales de tráfico, a partir de Naives Bayes y Árboles de Decisión con algoritmos C4.5 y CART con datos extraídos de KDD CUP de 1999, implementaron una aplicación usando R (software libre), concluyendo que con una correcta elección de atributos, parámetros de configuración y pesos, se obtienen mejores resultados. Las URL's que se analizaron presentaron cierta volatilidad, ocasionando que algunas fueran

[15] Guang-Gang Geng, Xiao-Dong Lee and Yan-Ming Zhang. Combating phishing attacks via brand identity and authorization features. Computer Network Information Center, Chinese Academy of Sciences, Beijing, 100180, China. Institute of Automation, Chinese Academy of Sciences, Beijing, 00180, China

[16] Norbert ÁDÁM, Branislav MADOŠ, Marek ČAJKOVSKÝ, Ján HURTUK, Tomáš TOMČÁK. METHODS OF THE DATA MINING AND MACHINE LEARNING IN COMPUTER SECURITY. Acta Electrotechnica et Informatica, Vol. 14, No. 2, 2014, 46–50, DOI: 10.15546/aeei-2014-0017. ISSN 1335-8243 (print) © 2014 FEI TUKE ISSN 1338-3957(online), www.aei.tuke.sk. Department of Computers and Informatics, Faculty of Electrical Engineering and Informatics, Technical University of Košice, Letná 9, 042 00 Košice, Slovak Republic, tel. +421 55 602 3023, e-mail: {norbert. adam, branislav. mados, marek.cajkovsky, jan.hurtuk}@tuke.sk, tomas.tomcak@student.tuke.sk

bloqueadas o no existieran en el momento del análisis. Algunos atributos empleados en el desarrollo del presente trabajo fueron: atributos basados en la barra de direcciones, atributos anormales, características basadas en HTML y JavaScript y características basadas en información sobre dominios como vigencia, registro DNS, rango de páginas, presencia de índices y presencia de referencias.

2.1. MECANISMOS PARA IDENTIFICACIÓN DE PHISHING

Los atacantes continuamente emplean nuevas estrategias, para pasar desapercibidos, por lo cual existen algunas técnicas además de la minería de datos, para identificar presencia de phishing, como los sistemas de detección de intrusos.

2.1.1. Técnicas Tradicionales.

Estas técnicas emplean herramientas que reportan alarmas de posibles sospechas, estos sistemas de detección de intrusos[17], se clasifican en:

- Host-Based IDS: operan en un host para detectar actividad maliciosa en sus procesos.
- Network-Based IDS: operan sobre los flujos de información intercambiados en una red.
- Knowledge-Based IDS: Sistemas basados en Conocimiento.
- Behavior-Based IDS: Sistemas basados en Comportamiento.

Cada una de estas asume que una intrusión se puede detectar observando una desviación respecto del comportamiento normal dentro del entorno en el cual los usuarios se desempeñan en el sistema. Estas técnicas no se analizan en este documento.

[17] Northcutt, Stephen and Novak Judy. Network Intrusion Detection. Ed. SANS New Riders Publishing. Third Edition. United States. 2003.

2.1.2. Clasificación estadística.

Realiza análisis de regresión de datos, y se usan para detectar comportamientos anormales dentro de un proceso, a partir de una serie de acciones ejecutadas por el usuario, durante un periodo de tiempo determinado. También se usa el análisis de relaciones, que permite hallar relaciones entre elementos de información como operaciones, procesos y usuarios participantes. Se requiere de un esquema supervisado[18].

Existen listas de personas, instituciones u objetos que deben ser discriminados en alguna forma con respecto a los que no están en la lista, estas se denominan listas negras[19]. Esta discriminación puede ser social, técnica o de otra forma; cuando es positiva se habla de lista blanca.

2.2. METODO DE CLASIFICACION EN MINERÍA DE DATOS.

Algunos investigadores dieron origen a campos de investigación como el descubrimiento de conocimiento en bases de datos (*Knowledge Discovery in Database - KDD*), usados en Minería de Datos (Data Mining). La minería de datos posee algunas técnicas que identifican situaciones desconocidas, mediante modelos clasificados en:

- Modelos de datos inusuales. Buscan detectar comportamientos extraños en un dato con respecto al grupo. Emplea técnicas de análisis de Agrupamiento (*Clustering*), además de análisis de “Outliers".
- Modelos de relaciones inexplicables. Se orientan a encontrar relación de registros que tienen valores similares para determinados campos, usando técnicas de agrupamiento (*Clustering*) para encontrar grupos sospechosos y reglas de asociación.

[18] R. J. Bolton and D. J. Hand, \Statistical fraud detection: A review," Statistical Science, vol. 17, no. 3, pp. 235 {249, Aug. 2002, Article Type: primary article /Full publication date: Aug., 2002 / Copyright 2002 Institute of Mathematical Statistics. Disponible: http://www.jstor.org/stable/3182781

[19] https://sucuri.net/es/seguridad-de-website/entender-google-website-blacklist. Consultado Julio 2016.

La clasificación es una de las tareas más comunes de Minería de Datos, ya que permiten examinar las características de algunas variables dentro de un conjunto de datos, que se presentan en forma de registros, para luego asignarlas a otro elemento de clasificación, agrupados en una clase específica, que se usan en el entrenamiento. El objetivo de este tipo de técnicas, es clasificar un valor particular de un atributo, llamado "clase" o variable dependiente, a partir de otro. Los atributos usados para hacer predicción se llaman variables independientes. Las técnicas más comunes de la minería de datos[20] , además de la clasificación son la predicción, en que los registros se clasifican de acuerdo a un comportamiento o valor futuro. La asociación, que busca encontrar atributos que se relacionan entre sí. También está la descripción, que describe cuales eventos están ocurriendo con los datos analizados, para conocer cuales factores dieron su origen. Los árboles de decisión permiten dar explicaciones entendibles de los resultados obtenidos. Existen otras técnicas de minería de datos como Visualización, Arboles de decisión y Redes neuronales. Para el presente trabajo se utilizaron los arboles de decision, que se usan en la clasificación y predicción, basados en el criterio de dividir los datos en ramas del árbol. Son estructuras que representan conjuntos de decisiones, las cuales generan reglas para clasificar un conjunto de datos. Se usa con atributos discretos y continuos, manejando atributos no significativos, valores faltantes y datos incongruentes. Son eficientes y permiten la clasificación, pero son inestables ante variación de muestras. Su objetivo es describir las variables entre sí, por ser un método supervisado, usando modelos de máquinas de aprendizaje. El aporte dado por Webber y otros[21], a partir de la identificación de características usadas para detectar ataques de Phishing mediante técnicas de minería de datos y su proceso de evaluación y comparación con el uso de redes neuronales y árboles de decisión, proporcionaron criterios semánticos y sintácticos para el análisis de

[20] Clark, P. and Boswell, r. Datamining. Practical Machine Learning tools and Technicals with Java Implementations. Morgan Kaufmann Publishers, 2000.

[21] Carine G. Webber, Maria de Fátima W. do Prado Lima, and Felipe S. Hepp. Testing Phishing Detection Criteria and Methods. University of Caxias do Sul, Information Technology Centre Rua Francisco Getúlio Vargas, 1130 – CEP 95060-570. Caxias do Sul, RS, Brazil.

correos electrónicos suplantados o no. Al usar el algoritmo del perceptrón multicapa, se logró una precisión de 96,5%, generando solo 7 falsos positivo (mensajes legítimos clasificados como suplantados) y 11 falsos negativos (mensajes suplantados clasificados como mensajes legítimos). Con el uso del algoritmo J48 en árboles de decisión, se logró una precisión de 94,94%, con ID3 se obtuvo una menor precisión, el número de falsos negativos disminuyo. La clasificación de falsos positivos género menos impacto en los usuarios de Internet. De esto se logró concluir que el algoritmo del perceptrón multicapa garantiza una clasificación adecuada.

El análisis de identificación de ataques de phishing a desarrollar, estuvo basado en métodos de clasificación, donde a partir de análisis de características identificadas en las URL´s se lograron identificar posibles amenazas, usando un árbol de decisión y algunas reglas de clasificación para definir patrones que identifiquen ataques de phishing, permitiendo a la comunidad cibernauta examinar y actuar rápidamente, evitando ser víctimas de acciones fraudulentas. El modelo presentó un conjunto de fases que permitió la preparación de datos, extraer y presentar el conocimiento, como parte de cualquier proceso de minería de datos. Adicionalmente se hizo énfasis en el proceso de aprendizaje mediante un árbol de decisión.

Los algoritmos usados en máquinas de aprendizaje, inducen a clasificadores que dependen de los datos de entrenamiento y de las pruebas estadísticas, usadas para evaluar el error esperado, para luego compararlos y evaluar el mejor. No se puede decidir con el error del conjunto de entrenamiento, porque siempre es menor que el error del conjunto de pruebas. El conjunto de validación es diferente al que se usa en el entrenamiento. Se tuvo que evaluar varias veces el conjunto de validación y entrenamiento para obtener resultados fiables, debido a que eran pequeños y presentan casos excepcionales como ruido y valores atípicos, que generaban resultados erróneos. El método de aprendizaje puede depender de otros factores

aleatorios que afectan a la generalización, por lo tanto se debe emplear un método de validación cruzada, para medir mejor el error (i.e matriz de confusión).

2.2.1. Método de validación cruzada. Este método toma el conjunto de datos originales, para luego dividirlo en K partes de manera aleatoria, una para entrenamiento y otra para validación.

Existen algunos métodos usados en minería de datos, los cuales son:

- Validación cruzada de K-Fold. Conjunto de datos *X* los cuales se dividen aleatoriamente en *K* partes iguales, $X_j, j = 1,...,K$ Para generar las parejas, se mantiene una parte de *K* para validación (*V*) y se combinan *K-1* partes para el entrenamiento (*E*).
- Validación cruzada de 5*2. Usa un conjunto (*X*) para entrenamiento y validación de igual tamaño, dividido aleatoriamente en dos partes: $X_{(1)}1$ y $X_{(2)}2$, estable el primer conjunto de entrenamiento y validación: T1 = $X_{(1)}1$ y V1= $X_{(2)}1$. Luego hay una función de intercambio para obtener la segunda pareja: T2 = $X_{(2)}1$ y V2 = $X_{(1)}1$. Esto corresponde al primer conjunto de validación (*fold*), se repite 5 veces, obteniendo diez conjuntos de entrenamiento y validación.

- Muestro "Bootstraping"**.** Permite generar nuevas muestras a partir de una original usando la técnica de reemplazo. Llevar a cabo una sola validación, usa todo el conjunto de datos originales, de lo contrario, se generan muchos de entrenamiento y validación. La probabilidad de tomar un ejemplo es *1/N* y la probabilidad de no escoger una instancia es *1-1/N*. La probabilidad de no tomar ninguna instancia después de *N* veces es:

$$\left(1-\frac{1}{N}\right)N \approx e^{-1} = 0.368$$

Ecuación 1. Probabilidad sin ninguna instancia

Generalmente el conjunto de datos de entrenamiento contenía el 63.2% de las instancias y para entrenamiento el 36.8% de las instancias.

2.2.2. Método de medición de error. Para analizar los errores generados a partir de un modelo de clasificación se empleó lo siguiente[22]:

2.2.2.1. Matriz de Confusión. Herramienta de visualización empleada en aprendizaje supervisado. Cada columna representa el número de predicciones de cada clase, y cada fila representa a las instancias en la clase real, como se muestra en la Tabla 1. Un beneficio es que permite ver si el sistema está confundiendo dos clases.

	Clase a predecir	
Clase Real	**SI**	**NO**
SI	TP: Verdaderos Positivos	FN: Falsos Negativos
NO	FP: Falsos Positivos	TN: Verdaderos Negativos

Tabla 1. Matriz de confusión

Las expresiones algebraicas se presentan en el Anexo A.

2.3. ARBOLES DE DECISION.

La técnica de aprendizaje usada con árboles de decisión, es fácil de utilizar y permite realizar inferencias inductivas y aproximación de funciones de valores discretos, son robustos frente a datos con ruido y capaces de aprender expresiones disjuntas. Es un conjunto de condiciones organizadas en una estructura jerárquica, de tal manera que la decisión final a tomar, se determina siguiendo las condiciones que se cumplen desde la raíz del árbol hasta sus hojas[23]. Los arboles permiten detectar reglas del negocio que pueden ser traducidas a lenguaje natural y en la

[22] E. Alpaydin, Introduction to Machine Learning. The MIT press Cambridge, MA, 2004.
[23] Hernández Orallo, José y Ramírez Quintana María José. Introducción a la minería de datos. Pearson. Prentice Hall. Madrid. 2004.

construcción de modelos de clasificación. El funcionamiento general de un árbol se basa en la aplicación de condiciones que pueden ser cumplidas, o no, por un registro; el registro pasa a través del árbol de premisa en premisa hasta que se evalúa totalmente o hasta que encuentra un nodo terminal. Esta técnica se aplica a la clasificación y a la predicción, a partir de la división de los datos en las hojas del árbol. Representan un conjunto de decisiones que generan reglas para la clasificación de los datos, además admiten atributos discretos y continuos, permitiendo un manejo adecuado a los atributos no significativos, los valores faltantes y los datos incongruentes. Su metodología se basa en el aprendizaje supervisado. Un árbol de decisión consiste en una representación de conocimiento relativamente simple, por lo tanto, los procedimientos utilizados en su aprendizaje son más sencillos con respecto a otros sistemas. A usar un árbol de decisión[24], se presentaron condiciones iniciales que se podían plantear a partir de una serie de preguntas, tomadas como variables de entrada al modelo, como son: ¿Presencia de guiones? ¿URL anormales?, etc., cada registro describe un camino dentro del árbol por donde pasa, hasta obtener una calificación o una clasificación según sea el caso. Los caminos que describe el árbol para llegar a los nodos terminales, representan el conocimiento adquirido y permiten la extracción de reglas de clasificación de la forma SI - ENTONCES. Algunos algoritmos basados en arboles de decisión, usan algunos indicadores como la razón de ganancia, la cual depende de la elección de atributos con valores, debido a que entre más clara sea la participación producida por los valores del atributo, la incertidumbre o entropía será menor en cada nodo, y menor la media de la entropía a ese nivel. Además trabajan algunos valores desconocidos, que los usan algoritmos como C4.5, el cual permite atributos desconocidos en el aprendizaje y en la validación. En el entrenamiento, los valores desconocidos se distribuyen con sus pesos de acuerdo a la frecuencia de aparición de cada posible valor del atributo. Al clasificar se alcanza un nodo con un atributo desconocido, el cual se divide en tantos casos como valores tenga el

[24] Morey, D., "Knowledge Management Architecture," Handbook of Data Management, Auerbach Publications, NY, 1998 (Ed: B. Thuraisingham).

atributo, dando un peso a cada resultado, con igual criterio que la frecuencia de aparición de cada posible valor del atributo en el entrenamiento. Esto es la técnica de clasificación con probabilidades, correspondientes a la distribución de ejemplos en cada nodo hoja. La poda del árbol de decisión, construido a partir de ejemplos, reflejó correctamente el grupo de casos, pero por ser diferentes entre sí, fueron bastante complejos, con trayectorias largas y desiguales. El algoritmo C4.5 efectúo la poda después de crear el árbol completo (post-poda), estimando la creación de un nodo sin carácter de hoja. Empezó en el nodo hoja y recursivamente continúa hasta el nodo raíz, en otros casos se realiza la construcción y poda simultáneamente (pre-poda). En la construcción del árbol de decisión se llevó a cabo las etapas de construcción del árbol, luego se generan reglas de parada y por último la poda el árbol, en la cual se identificaron y eliminaron las ramas que reflejaban ruido o valores atípicos.

2.4. PORQUE LOS ALGORITMOS DE ÁRBOLES DE DECISION

Existen algunas características, que permitieron decidir que algoritmos de clasificación se utilizarían en el presente análisis, entre las que tenemos: el número de ejemplos en el conjunto de entrenamiento, la dimensión de los atributos, su independencia, la presencia de sobreajuste (*overfitting*), requerimientos en términos de velocidad, rendimiento y memoria de procesamiento. Se empleó el principio la navaja de Occam[25], que consistió en utilizar el algoritmo menos complicado y solo se usaría algoritmos más complicados si fuera estrictamente necesario.

2.5. ALGORITMOS BASADOS EN ÁRBOLES DE DECISIÓN

El trabajo de grado se centra en arboles de decisión aplicados a tareas de clasificación, a partir de algoritmos propios de este método. Cualquier árbol de

[25] http://www.educadictos.com/la-navaja-de-ockham/. Consultado diciembre 2016.

decisión se puede convertir en reglas de clasificación, cuya estructura es del tipo Si <Condición> Entonces <Clase>. El algoritmo para generación de reglas consiste en que, por cada rama del árbol, las preguntas y valores se ubican en la parte izquierda de las reglas y la etiqueta del nodo hoja en la parte derecha (clasificación). Este genera un sistema de reglas complejo, por ello C4.5 realiza el podado de las reglas obtenidas. Para estimar el error, se realiza de igual forma que para el podado del árbol.

Existen algoritmos que generan arboles de decisión, como los que se explican a continuación.

2.5.1. Decisión Stump (Árbol de un solo nivel).

Este algoritmo es muy sencillo, y genera un árbol de decisión de un solo nivel, utilizando un único atributo para construir el árbol, basado en la ganancia de información, a pesar de su simplicidad, puede llegar a conseguir resultados interesantes. Tiene tres ramas: una para cuando el atributo es desconocido y las otras dos para cuando el valor del atributo en la evaluación sea igual a un valor concreto o distinto a este, cuando los atributos son simbólicos. El valor del ejemplo es mayor o menor a uno determinado en el caso de atributos numéricos. Se usa en problemas de dos clases, pero en problemas de más de dos clases no es posible encontrar tasas de error menores a 0.5. El fin de este algoritmo es construir un modelo, donde se toma solo un subconjunto de casos de entrenamiento, que se escoge basado en la distancia métrica entre pruebas del caso y de pruebas de otros casos en el espacio. Cada nodo representa una característica de un caso para ser clasificado, y cada rama un valor que el nodo toma. Los casos inician con el nodo raíz y se basan en sus valores característicos. En el peor de los casos un árbol de este tipo puede hacerse mejor si la característica seleccionada es informativa.

Para atributos simbólicos se considera cada valor posible y se calcula la ganancia de información con el atributo igual al valor, distinto a sus valores desconocidos.

Para los atributos simbólicos se selecciona el mejor punto de ruptura. Se debe tener en cuenta cuatro posibles casos para calcular la ganancia de información: que sea un atributo simbólico y la clase sea simbólica o que la clase sea numérica, o que sea un atributo numérico y la clase sea simbólica o que la clase sea numérica.

2.5.2. LMT- Logistic Model Tree[26].

La Regresión Logística es un algoritmo de clasificación que combina dos métodos de aprendizaje: un modelo de árbol de inducción con una regresión logística en las hojas del árbol, proporcionando una descripción adecuada de los datos, el cual consiste en una estructura de un árbol de decisión con funciones de regresión logística desde sus hojas. Para enumerar atributos con k valores, el nodo tiene k nodos hijos, y los casos son clasificados en las k ramas dependiendo del valor del atributo. En el caso de atributos de tipo numérico, el nodo tendrá dos nodos hijos y la prueba compara el valor del atributo con un umbral: se pueden clasificar los datos menores en la rama izquierda y los mayores en la rama derecha. Un LMT está compuesto por un juego N de nodos internos o no terminales y un juego de T hojas o nodos terminales. S representa el espacio, atravesado por todos los atributos presentes en los datos.

2.5.3. Algoritmo Random Forest[27].

Utiliza Bagging junto a una selección aleatoria de atributos. En cada nodo del árbol en el bosque, se selecciona aleatoriamente un subconjunto de los atributos disponibles en ese nodo y se elige el mejor de ellos de acuerdo al criterio de división empleado en el algoritmo base. El número de atributos seleccionado aleatoriamente

---

[26] Modelo basado en metaclasificadores para diagnostico en marcha patológica mediante análisis cinético Ana I. Aguilera, Luz D. Cala, Albero R. Subero. Centro de Análisis, Modelado y Tratamiento de Datos (CAMYTD), Facultad de Ciencias y Tecnología FACYT, Universidad de Carabobo, Valencia–Venezuela. http://servicio.bc.uc.edu.ve/ingenieria/revista/v17n2/art01.pdf

[27] Toma de decisiones en ensambles basados en árbol fuzzy. Piero Bonissone José M. Cadenas M. Carmen Garrido, R. Andrés Díaz-Valladares Antonio Muñoz. ESTYLF 2010, Huelva, 3 a 5 de febrero de 2010. http://www.uhu.es/estylf2010/trabajos/SS11-04.pdf

es un parámetro de entrada y diversos estudios avalan que el valor log_2 *(jMj ¡1),* donde jMj es el cardinal del conjunto de atributos disponibles en ese nodo en cuestión, produce un buen comportamiento del ensamble. Además, dado que el ensamble *Random Forest* hace uso de *Bagging*, el conjunto de entrenamiento de cada clasificador se forma seleccionando ejemplos con reemplazamiento a partir del conjunto de ejemplos originales. Por lo tanto, un ensamble *Random Forest* está basado en un doble nivel de aleatoriedad para conseguir diversidad. Para realizar la clasificación este algoritmo emplea un objeto desde un vector de entrada, el cual se ubica bajo cada uno de los árboles del bosque. Cada árbol genera una clasificación, la clasificación se escoge por el bosque teniendo en cuenta el árbol más votado sobre todos los del bosque. El árbol se implementa de la siguiente manera:

• Si se tiene un número de casos (N) en el conjunto de entrenamiento, se validan N casos de manera aleatoria sustituyendo datos originales. Este se toma como conjunto de entrenamiento.

• Si se tiene M variables de entrada, un número m menor a M, se especifica para cada nodo, m variables se seleccionan aleatoriamente de M y la mejor presencia de m se usa para dividir el nodo. El valor de m es constante durante el crecimiento del bosque.

• Cada árbol crece de la forma más extensa posible, sin llevar a cabo la poda".

Las características de *Random Forest*[28] son:

• Es insuperable en la precisión entre los algoritmos actuales.

• Trabaja de manera eficaz en bases de datos grandes

• Usa gran cantidad de variables de entrada sin necesidad de eliminar otras, dando estimaciones para determinar que variables son importantes en la clasificación.

[28] Leo Breiman and Adele Cutler. Random Forests. Consultado en Julio 2016. Disponible en http://www.stat.berkeley.edu/~breiman/RandomForests/cc_home.htm

• Permite estimar datos perdidos y mantener la exactitud del porcentaje de datos son fallidos.
• Se pueden generar otros árboles para ser usados posteriormente con otros datos, ya que dan información sobre la relación entre las variables y las clasificaciones.
• Permiten detectar interacciones entre variables.
•Posee un método eficaz para calcular la estimación de los datos que faltan y mantienen la precisión cuando falta una gran proporción de datos.
•Posee un método para calcular el error de equilibrio en los datos que pertenecen a poblaciones con clases desequilibradas.

Adicionalmente puede generar varios árboles, es muy eficiente y se ejecuta en un conjunto de datos con más de 100 variables. Para conjunto de datos muy grandes, se debe tener un espacio de memoria de almacenamiento adecuado, el cual debe ser proporcional al número de casos y de árboles a generar.

El uso de clasificación mediante el uso de este algoritmo ha sido usado por algunos autores para clasificar correos con Phishing. Esta clasificación se basa en asociarle a cada uno de los datos de prueba, una clase predefinidas.[29]

2.5.4. Algoritmo *RandomTree*[30].

Es un clasificador supervisado, donde en el árbol estándar, cada nodo se divide usando la mejor división entre todas las variables. En un bosque al azar, cada nodo se divide utilizando el mejor entre el subconjunto de predicadores elegidos al azar en ese nodo. Es un árbol representado aleatoriamente entre un juego de árboles

[29] Classification of Phishing Email Using Random Forest Machine Learning Technique Andronicus A. Akinyelu and Aderemi O. Adewumi. Private Bag Box X54001, Durban 4000, South Africa. Received 23 January 2014; Accepted 11 March 2014; Published 3 April 2014
[30] Analysis of WEKA Data Mining Algorithm REPTree, Simple Cart and RandomTree for Classification of Indian News. Sushilkumar Kalmegh. Associate Professor, Department of Computer Science, Sant Gadge Baba Amravati University. Amravati, Maharashtra- 444602, India. IJISET - International Journal of Innovative Science, Engineering & Technology, Vol. 2 Issue 2, February 2015.
http://ijiset.com/vol2/v2s2/IJISET_V2_I2_63.pdf

posibles[31], es decir que cada árbol tiene una posibilidad igual de ser probado con respecto a un conjunto de árboles, por lo cual se presenta una distribución "uniforme" de árboles. Al aplicar este algoritmo se generó un árbol como el mostrado en la Figura 1.

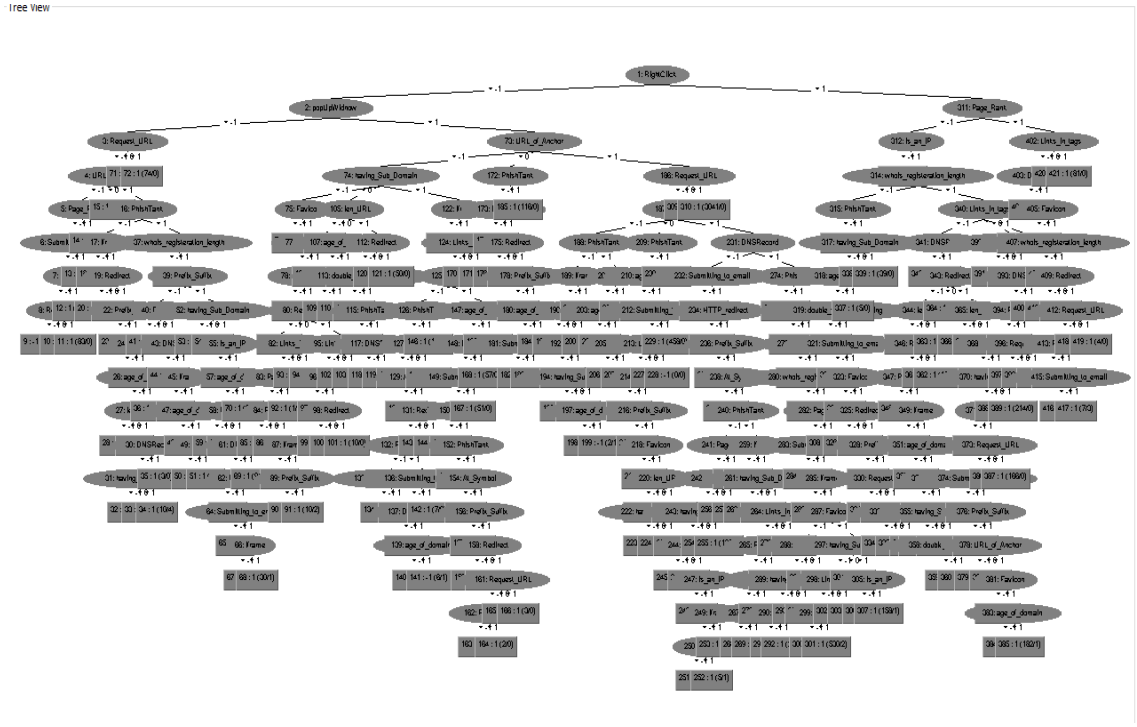


Figura 1. Árbol generado RandomTree

2.5.5. Algoritmo *REPTree*[32].

Utiliza la lógica del árbol de regresión y crea múltiples árboles en diferentes iteraciones. Selecciona el mejor de todos los árboles generados. Según Aurora

[31] Roderic D. M. Page. Tree Comparsion Software for Microsoft. Windows. User's Guide. Biogeography and Conservation Laboratory. http://taxonomy.zoology.gla.ac.uk/rod/cplite/ch6.pdf

[32] Mining Social Networking Data for Classification Using Reptree. Dr. B. Srinivasan. P.Mekala. ISSN: 2321-7782 (Online). Volume 2, Issue 10, October 2014.International Journal of Advance Research in Computer Science and Management Studies. http://s3.amazonaws.com/academia.edu.documents/35665379/V2I10-0056.pdf?AWSAccessKeyId=AKIAJ56TQJRTWSMTNPEA&Expires=1471486440&Signature=LjRoC9uI5vQDhOG%2BR6yh3%2FR7Beo%3D&response-content-disposition=inline%3B%20filename%3DMining_Social_Networking_Data_for_Classi.pdf

Agudo, Juan Carlos Alonso y Ruth Santana[33]. *REPTree* es un método de aprendizaje a partir de árboles de decisión, el cual construye el árbol a partir de la información de varianza y realiza la poda usando el criterio de reducción del error. Clasifica valores solo para atributos numéricos. Valores faltantes se logran partiendo las correspondientes instancias. Se fundamenta en modelos comprensible, es decir en reglas *IF THEN ELSE.* Se construye el árbol de decisión usando la ganancia de información y ejecuta la poda a partir del error reducido. Ordena una sola vez los valores de los atributos numéricos. Los valores faltantes se logran dividiendo las instancias correspondientes en segmentos. El árbol generado se observa en la Figura 2.

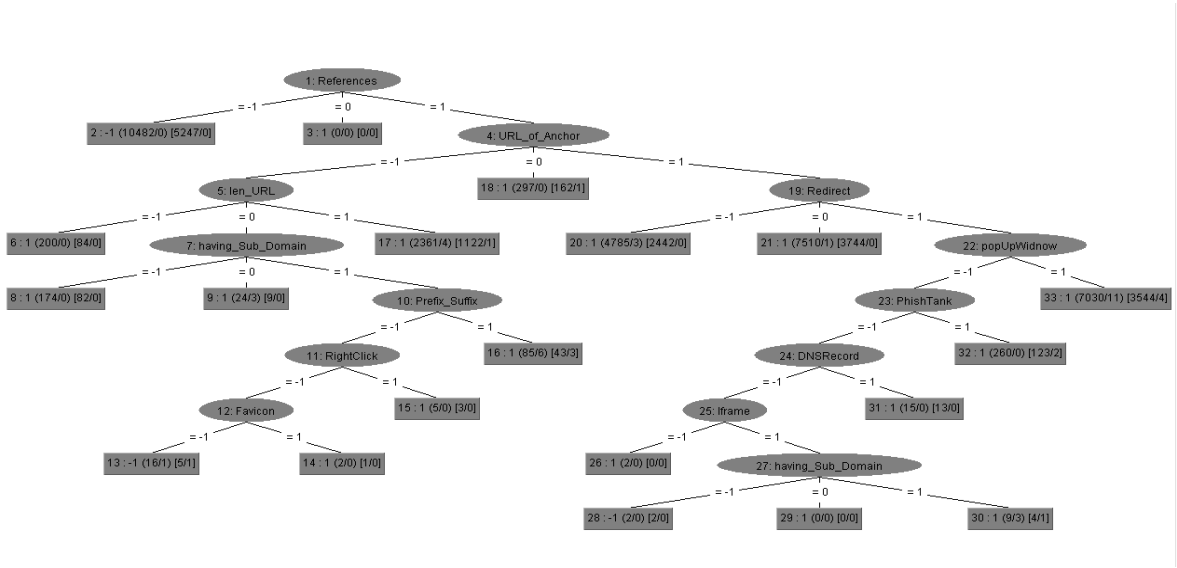


Figura 2. Árbol generado *REPTree*

2.5.6. Algoritmo PART[34].

Los sistemas de inducción de reglas, realizan dos fases, la primera genera un conjunto de reglas de clasificación y luego refinan estas reglas para mejorarlas,

[33] Evaluación de Modelos para Predicción Meteorológica. Universidad Carlos III de Madrid. 2004. http://www.it.uc3m.es/jvillena/irc/practicas/04-05/21mem.pdf

[34] Eibe Frank and Ian H. Witten (1998). "Generating Achúrate Rule Sets Without Global Optimization." In Shavlik, J., ed., Machine Learning: Proceedings of the Fifteenth International Conference, Morgan Kaufmann Publishers, San Francisco, CA.

realizando con esto su optimización global. Para el caso del algoritmo PART obtiene reglas sin realizar dicha optimización global, debido a que es complejo y costoso computacionalmente. Recibe el nombre por su modo de actuación (*obtaining rules from PARTial decision trees*), y fue desarrollado por los autores de WEKA. Se basa en dos estrategias básicas para la inducción de reglas: el covering y la generación de reglas a partir de árboles de decisión. En Covering se genera una regla y elimina los ejemplares que dicha regla cubre, generando otras que no queden ejemplos por clasificar. Para crear una regla, se genera un árbol podado, luego se obtiene la hoja que clasifique el mayor número de ejemplos, que se transforma en la regla, después se elimina el árbol. Uniendo estas dos estrategias, se logra mayor flexibilidad y velocidad. Se genera un árbol parcial [*PARTial decisión tree*], el cual contiene brazos con subárboles no definidos. Para su generación se integran los procesos de construcción y podado hasta obtener un subárbol estable que no puede simplificarse más, en cuyo caso termina el proceso y genera la regla a partir de dicho subárbol. El proceso de elección del mejor atributo se basa en la razón de ganancia, la expansión de los subconjuntos se realiza en orden, inicia con el de menos entropía y termina con el mayor. Si un subconjunto tiene menor entropía, aumenta la probabilidad que se genere un subárbol menor y genere una regla más general. El proceso continúa recursivamente expandiendo los subconjuntos hasta obtener hojas, momento en el que se realiza una vuelta atrás (*backtracking*), cuando esta se realiza y los hijos del nodo son hojas, comienza el podado como se hace en C4.5 (compara el error esperado del subárbol con el nodo), pero solo realiza la función de reemplazamiento del nodo por hoja (subtree replacement). Si hay un podado, se realiza otra vuelta atrás hacia el nodo padre y sigue explorando el resto de sus hijos; si no se puede realizar el podado el padre no continuará con la exploración del resto de nodos hijos. Al finalizar el proceso de expansión, se genera el árbol de decisión.

2.5.7. Algoritmo ID3.

El hecho de construir un árbol de decisión, implica encontrar aquellos que presenten las mejores características y un adecuado balanceado, usando el menor número de preguntas para lograr las respuestas en cada caso. Se construye de forma directa en el árbol de decisión de arriba abajo, sin usar backtracking. Con este se calcula su Ganancia, buscando seleccionar el atributo más útil, permitiendo separar mejor los casos con respecto a la clasificación planteada. Para la aplicación de dicho algoritmo, se deberá medir inicialmente la ganancia y el grado de entropía. Se pueden presentar muestras homogéneas o igualmente distribuidas para el caso de la clasificación: Es homogénea cuando todos clasifican igual, y presenta una incertidumbre mínima, es decir no se duda de la clasificación de alguno de sus elementos. Por otra parte, se tendrá una muestra igualmente distribuida, cuando tiene igual número de ejemplos para clasificar, presentando una incertidumbre máxima, considerada la peor situación para predecir su clasificación. Se tiene información cuando se reduce la incertidumbre, lo que indica que la ganancia de información es la reducción de la entropía. Por lo tanto, la pregunta ¿Qué característica crea las ramas más homogéneas y la mayor ganancia de información? Se responde a partir de:

- Calculo de la entropía total.
- División del conjunto de datos en sus diferentes atributos.
- Calcula de la entropía de cada rama y la suma de las ramas para calcular la entropía total.

Una rama con entropía 0 se convierte en una hoja, representando una muestra homogénea, con igual clasificación; de lo contrario la rama se continúa dividiendo para clasificar mejor sus nodos.

Este algoritmo se ejecuta recursivamente en hojas, hasta llegar a presentar entropía nula, quedándonos con el atributo de mayor ganancia, logrando con esto el primer paso del árbol de decisión, es decir se identificará el primer nodo de decisión. Luego

nos ubicaremos en cada uno de los subconjuntos que definen cada valor del atributo seleccionado y se repite el proceso, buscando crear el árbol completo definitivo.

El nodo con entropía nula es una respuesta, por representar una muestra homogénea, donde la clasificación es igual para todos.

Una ventaja de los árboles de decisión como máquinas de predicción, es que permiten explicar por qué un ejemplo clasifica de una forma u otra, además extrae un procedimiento a partir del uso de reglas condicionales.

El algoritmo ID3 permitirá realizar un árbol de decisión con las siguientes características[35]:

- Cada nodo pertenece a un atributo y cada rama a su posible valor. Una hoja del árbol especifica el valor de la decisión. La decisión viene dada por la trayectoria desde la raíz a la hoja representativa de esa decisión.
- Se asocia a cada nodo el atributo más representativo, que no haya sido considerado en la trayectoria desde la raíz.
- Su usa la entropía para determinar el impacto informativo del atributo. El atributo será más útil si la entropía y la incertidumbre son mínimas.

Inconvenientes presentados con el algoritmo ID3.

- Es bien definido para aquellos atributos categóricos, es decir con un rango finito de valores.
- No presenta el mejor árbol, debido a que usa un método voraz, optimizando cada paso de manera individual, pero donde se pueden presentar un resultado más óptimo a partir de otro ejemplo.
- No se comporta bien para casos donde no se conoce el valor de algún atributo.

[35] Ciberconta. ”Sistemas de inducción de árboles de decisión: utilidad en el análisis de crisis bancarias”. Disponible: http://ciberconta.unizar.es/Biblioteca/0007/arboles.html [citado en 4 de Marzo de 2008]

Para dar solución a los anteriores problemas Quinlan[36] propuso el algoritmo C5.0, el cual es más completo y óptimo, por lo cual:

- Cuando existe falta de información en los atributos, se debe usar ejemplos que posean información respecto al atributo analizado, para calcular su entropía y probabilidad asociadas a este.
- El problema de la optimalidad global en los árboles de decisión, está dentro del conjunto que presenta una solución eficiente o los que no la tienen. Aplica algunas heurísticas, pero no garantiza una solución óptima.
- Es mejor el tratamiento del problema de los atributos con infinitos valores.
- Si el atributo es de tipo general se pueden aplicar métodos de clusterización, para saber en qué nodo se puede dividir y utilizar como pregunta en la pertenencia de una característica o no. Con esto se puede fijar el número de nodos a priori. Este método se usa cuando el atributo toma valores en un conjunto numérico ordenado.

2.5.8. Algoritmo J48.

Algoritmo de inducción que origina una estructura de reglas o árbol a partir de subconjuntos (ventanas) de casos extraídos de un conjunto de datos de "entrenamiento". Genera reglas y mide su "bondad" usando criterios, que miden la precisión en su clasificación. Los criterios principales de este algoritmo son:

- Calcula el valor de la información generada por una regla candidata (rama, con una rutina que se llama "info").
- Calcula la mejora global que genera una regla/rama a partir de una rutina llamada Gain (beneficio).

Con estos criterios se calcula una relación coste/beneficio de cada ciclo que permite decidir si crea, dos nuevas reglas, o una sola. Realiza el proceso de los datos en sucesivos ciclos. En cada uno de estos se incrementa el tamaño de la "ventana" de

[36] J. R. Quinlan, \Induction of decisión trees," Machine learning, vol. 1, no. 1, pp. 81 {106, 1986}

proceso en un porcentaje determinado respecto al conjunto de datos. Las reglas buscan clasificar correctamente un número cada vez mayor de casos en el conjunto total. Cada ciclo emplea los resultados conseguidos por el ciclo anterior. La variable de salida debe ser categórica.

Los resultados obtenidos fueron: Aciertos del 99.994 %, la clase phishing y no phishing lograron un 100% de aciertos. Se ajustó el parámetro denominado *Confidence Factor*, para controlar el tamaño del árbol mediante el proceso de poda, el cual influyo notoriamente en el tamaño y capacidad de predicción del árbol construido, permitiendo la operación de poda y la probabilidad de error muy baja, lo cual exigió que la diferencia en los errores de predicción antes y después de podar fuera mayor. Un parámetro importante es el sobreaprendizaje denominado *overfitting*, permitió obtener un árbol de 5 hojas (*leaves*) y 8 nodos (*size of the tree*). Cuando el número de nodos es pequeño, el clasificador es simple. Se eligió un factor de 0,001 para árboles simples, logrando simplificarlo a 5 hojas y 8 nodos, y se mantuvo el porcentaje de aciertos en 99,994. Para árboles complejos se evaluaron con un valor factor de 1.0, pero se mantuvo el mismo porcentaje de aciertos. Con este algoritmo no se hace sobreaprendizaje. Con sobreaprendizaje, el árbol obtenido presentaría un porcentaje de aciertos mejor, pero se presentaría *underfitting*, que indica que el modelo no es complejo para llevar a cabo el aprendizaje correctamente. Con el factor de 0,25 se obtuvo un árbol más sencillo. Ver Figura 3. Se utilizó el parámetro denominado *Confidence Factor*, para controlar el tamaño del árbol mediante el proceso de poda, influyendo notoriamente en el tamaño y capacidad de predicción del árbol construido, con una baja probabilidad de error, lo cual exigió la diferencia en los errores de predicción antes y después de podar. Se presentó sobreaprendizaje, obteniendo un porcentaje de aciertos mejor característico de *underfitting,* es decir que un modelo no es complejo para llevar a cabo el aprendizaje correctamente.

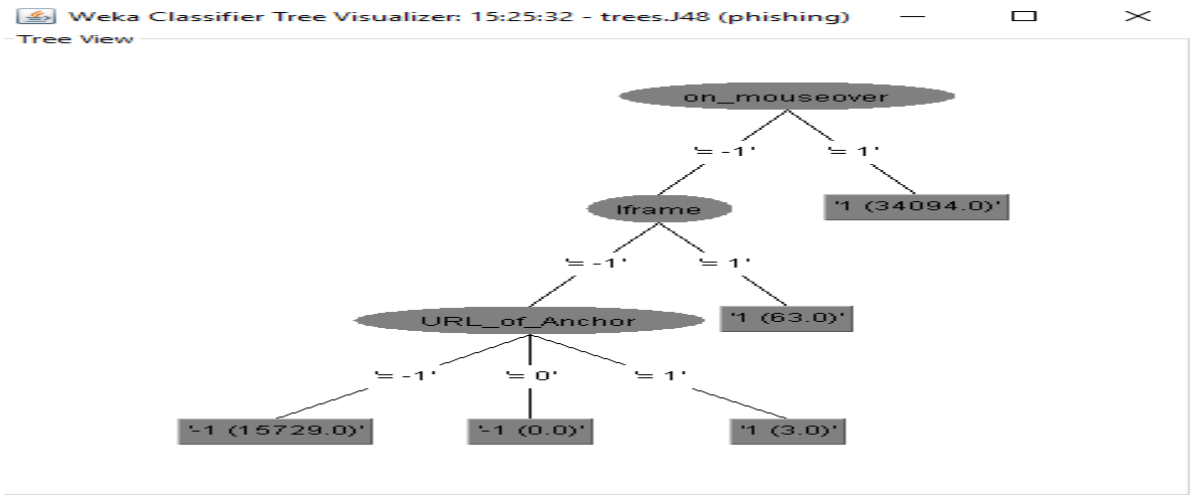


Figura 3. Árbol con J48 CF 0.25

Lo anterior permitió concluir lo siguiente: Con las características *If on_mouseover* y *URL_of_Anchor* detectadas como phishing, se obtiene un nivel de aciertos de 15729 URL´s afectadas, 3 aciertan como Legítimos. Con if *on_mouseover* es phishing and *Iframe* es phishing aciertan 63 como phishing. Al elegir el otro nodo que clasifica, se obtuvo que *on_mouseover* obtiene un nivel de aciertos de 34094 URL´s legítimas. Con un Confidence Factor de 0.001, se obtuvo lo siguiente: Si *on_mouseover, Iframe* y *URL_of_Anchor* clasifican como phishing; se logró un nivel de aciertos de 15729 URL con presencia de Phishing, y 3 acertaron como Legítimos. Si *on_mouseover e i*frame se clasifican como phishing, aciertan 63 como legítimos. Al elegir el otro nodo que clasifica, se obtuvo que on_mouseover obtiene un nivel de aciertos de 34094 URL legítimas. Con el *Confidence Factor* de 1.0, se concluyó: Si *on_mouseover*, iframe y *URL_of_Anchor* son clasificados como phishing se obtuvo un nivel de aciertos de 15729 URL con presencia de phishing, acertaron 3 URL como legitimas. Si *on_mouseover e i*frame son phishing, aciertan 63 legitimas. Al elegir el otro nodo que clasifica, se obtuvo que *on_mouseover* obtiene un nivel de aciertos de 34094 URL legítimas.

2.6. RESUMEN DEL CAPITULO

Este capítulo dio a conocer las bases teóricas sobre las cuales se desarrolló el trabajo de investigación, se presentan los diferentes algoritmos a usar en el análisis. La herramienta que se seleccionó para el proceso de minería, genera algunos reportes en cada uno de los procesos desarrollados, cada uno de ellos depende de la técnica usada para el análisis y procesamiento de los datos. En algunos de estos se presenta el árbol generado. Adicionalmente se realiza la clasificación sobre el conjunto de datos, para luego realizar la validación del conjunto de datos.

El capítulo inicia planteando el problema que se tiene en la identificación de Phishing y la forma como el mundo informático, debe usar técnicas tradicionales (procesos estadísticos, sistemas de detección de Intrusos) y técnicas de minería de datos para encontrar características que identifiquen Phishing en base a la información almacenada. Se realizó una recopilación de la literatura más avanzada en el tema. Adicionalmente se presentó el marco general para identificación de ataques de Phishing mediante alertas tempranas, sobre el cual se valida el conjunto de datos obtenidos en las primeras fases.

3. ESCENARIO A ANALIZAR

Este proceso inició con la etapa de limpieza y preprocesamiento para definir los datos a usar en la validación. Se generó una etapa de extracción de reglas buscando valorar el rendimiento del modelo de árbol, mediante el análisis de matrices de confusión.

3.1. DESCRIPCIÓN DEL CONJUNTO DE DATOS

Los tipos de datos que se analizaron inicialmente para usar en el proyecto, se extrajeron de www.phishme.com y http://www.archive.ics.uci.edu/ml/. Después de un análisis exhaustivo se eligieron dos tipos de *dataset* o conjuntos de datos definitivos realizando una inscripción a www.phishtank.com, obteniendo URL´s legítimas. Esta fase de preparación fue la más dispendiosa[37], representó el mayor esfuerzo y es un paso importante para diseñar el modelo[38], la calidad de los datos es un elemento clave. Para lograr el otro *dataset* afectadas por phishing, se llevó a cabo un proceso de *crawling*, usando una API obtenida de www.diffbot.com, orientada a la extracción de datos web a partir de Inteligencia Artificial; implementado un código en Java, el cual se puede apreciar en el Anexo B. Se generaron 25000 registros, que se analizaron con técnicas de conteo e histogramas. Luego se construyó el *dataset* total usado en el análisis, compuesto de 50000 registros mezclando las URL´s, del cual se tomó un porcentaje para aprendizaje y otro para evaluación. La información que dio origen a la tabla general, se compuso de las variables derivadas, logradas a partir de observaciones y características dentro de las URL´s. Se usó el término "atributo" para referenciar las variables (columnas) que están en ella. En la

[37] Giudici, Paolo. (2003). Applied Data Mining Stadistical Methods for Busyness and Industry. Chichester. Jhon Wiley & Sons, Inc. 364p.

[38] Parr Rud, Olivia. Data Mining Cookbook. Modeling data for Marketing, Risk and Customer Relationship Management. Wiley Computer Publishing. New York. 2001

preparación se tuvo en cuenta la calidad de los datos[39], desde el acceso a las fuentes. La identificación de las características, involucro un proceso de limpieza, para manejar error de datos, outliers y datos desconocidos.

3.1.1. Selección de datos.

Los datos obtenidos se analizaron con Microsoft Excel, que permitió manejar algunos atributos con histogramas.

3.1.2. Muestreo.

Por la cantidad de registros disponibles y a lo exigente del procesamiento, fue necesario realizar un muestreo estratificado para determinar si la URL estaba o no afectada por phishing. Se trabajó con el 68.5% de los datos para aprendizaje y 31.5% para evaluación. La distribución queda como se observa en la Tabla 2. Esto lo distribuyo automáticamente WEKA.

Total Población	100%	50000
Aprendizaje	66 %	33000
Evaluación	34 %	17000

Tabla 2. Distribución de los datos

3.1.3. Integración de datos.

Para facilidad y manejo de la información hallada, se consolidó en una sola tabla denominada *dataset* o vista minable, mejorando la eficiencia, el tiempo de preparación y la aplicación de las técnicas de minería de datos. Con la cantidad de URL´s obtenidas, se obtuvo información pertinente sobre algunas alertas tempranas para identificar ataques de phishing, usando minería de datos e identificando

[39] Parr Rud, Olivia. Data Mining Cookbook. Modeling data for Marketing, Risk and Customer Relationship Management. Wiley Computer Publishing. New York. 2001

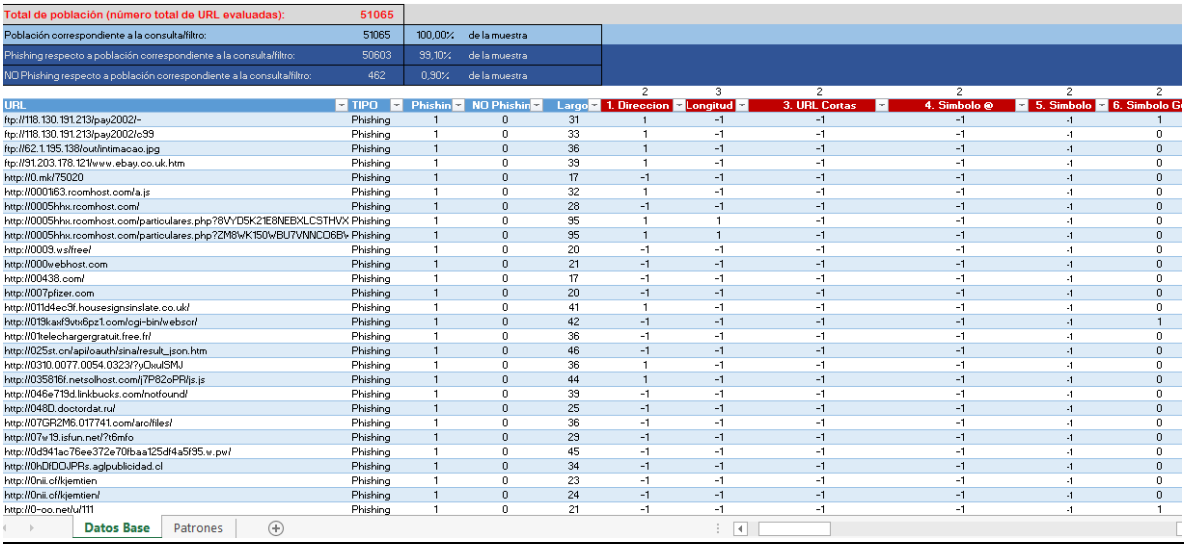
características en las URL´s afectadas por phishing, con el uso de mecanismos de clasificación a partir de árboles de decisión.

3.1.4. Preparación de datos para Reglas de Clasificación.

Para aplicar las técnicas de clasificación, se procedió a categorizar los atributos en “1” indicando URL´s legítimas y “-1” aquellas con características de Phishing.

3.1.5. Manipulación de los datos.

El conjunto de datos de URL’s no afectados por phishing, se hallaba en un archivo con extensión *JSON* siendo difícil su manipulación y relación entre los datos, por lo cual se convierte en formato CVS, donde se logró extraer valores nulos, datos en formato bruto, incoherencias, etc. Se usaron delimitadores, a partir de descomposiciones o “chunks” a partir de comas, slashes o espacios. Con el uso de expresiones de relación y lógicas, se obtuvo una mejor estructura en los datos, mediante ordenamientos y filtros, a partir variables como protocolo, host, enlaces, subdominio, nombre superior, extensión de país. Este proceso nos permitió identificar incoherencias y campos vacíos, además se filtraron y eliminar registros. Posteriormente se seleccionaron los atributos o variables derivadas para crear la matriz transaccional. Ver Figura 4.



Total de población (número total de URL evaluadas):	51065		
Población correspondiente a la consulta/filtro:	51065	100,00%	de la muestra
Phishing respecto a población correspondiente a la consulta/filtro:	50603	99,10%	de la muestra
NO Phishing respecto a población correspondiente a la consulta/filtro:	462	0,90%	de la muestra

					2	3	2	2	2	2
URL	TIPO	Phishin	NO Phishin	Largo	1. Direccion	Longitud	3. URL Cortas	4. Simbolo @	5. Simbolo	6. Simbolo G
ftp://118.130.191.213/pay2002/-	Phishing	1	0	31	1	-1	-1	-1	-1	1
ftp://118.130.191.213/pay2002/c99	Phishing	1	0	33	1	-1	-1	-1	-1	0
ftp://62.1.195.138/out/intimacao.jpg	Phishing	1	0	36	1	-1	-1	-1	-1	0
ftp://91.203.178.121/www.ebay.co.uk.htm	Phishing	1	0	39	1	-1	-1	-1	-1	0
http://0.mk/75020	Phishing	1	0	17	-1	-1	-1	-1	-1	0
http://0001163.roomhost.com/a.js	Phishing	1	0	32	1	-1	-1	-1	-1	0
http://0005hhx.roomhost.com/	Phishing	1	0	28	-1	-1	-1	-1	-1	0
http://0005hhx.roomhost.com/particulares.php?8VYD5K21E8NEBXLCSTHVX	Phishing	1	0	95	1	1	-1	-1	-1	0
http://0005hhx.roomhost.com/particulares.php?ZM8WK150WBU7VNNCO6BI	Phishing	1	0	95	1	1	-1	-1	-1	0
http://0009.ws/free/	Phishing	1	0	20	-1	-1	-1	-1	-1	0
http://000webhost.com	Phishing	1	0	21	-1	-1	-1	-1	-1	0
http://00438.com/	Phishing	1	0	17	-1	-1	-1	-1	-1	0
http://007pfizer.com	Phishing	1	0	20	-1	-1	-1	-1	-1	0
http://011d4ec9f.housesignsinslate.co.uk/	Phishing	1	0	41	1	-1	-1	-1	-1	0
http://013kaxf9vtx6pz1.com/cgi-bin/webscr/	Phishing	1	0	42	-1	-1	-1	-1	-1	1
http://01telechargergratuit.free.fr/	Phishing	1	0	36	-1	-1	-1	-1	-1	0
http://025st.cn/api/oauth/sina/result_json.htm	Phishing	1	0	46	-1	-1	-1	-1	-1	0
http://0310.0077.0054.0323/?yOxulSMJ	Phishing	1	0	36	1	-1	-1	-1	-1	0
http://0358161.netsolhost.com/j7P62oPR/js.js	Phishing	1	0	44	1	-1	-1	-1	-1	0
http://046e719d.linkbucks.com/notfound/	Phishing	1	0	39	-1	-1	-1	-1	-1	0
http://048D.doctordat.ru/	Phishing	1	0	25	-1	-1	-1	-1	-1	0
http://07GR2M6.017741.com/arc/files/	Phishing	1	0	36	-1	-1	-1	-1	-1	0
http://07w19.isfun.net/?t6mfo	Phishing	1	0	29	-1	-1	-1	-1	-1	0
http://0d941ac76ee372e70fbaa125df4a5f95.v.pw/	Phishing	1	0	45	-1	-1	-1	-1	-1	0
http://0hDfDCJPRs.aglpublicidad.cl	Phishing	1	0	34	-1	-1	-1	-1	-1	0
http://0nii.cf/kjemtien	Phishing	1	0	23	-1	-1	-1	-1	-1	0
http://0nii.cf/kjemtien/	Phishing	1	0	24	-1	-1	-1	-1	-1	0
http://0-oo.net/u/111	Phishing	1	0	21	-1	-1	-1	-1	-1	1

Datos Base | Patrones

Figura 4. Datos procesados en Excel

3.1.6. Validación y Limpieza de Datos

Este proceso permitió una presentación adecuada de los datos analizarlos y facilito el uso de las técnicas de clasificación usando minería de datos. Algunas etapas fueron:

3.1.6.1. Datos Perdidos. Se presentó al realizar el análisis en Excel, una cantidad algunas URL´s con valores en blanco, por lo cual se eliminaron, ya que no eran representativas. Tabla 3.

Registro	Valor presentado
3208	<<vacio>>
10629	<<vacio>>
16148	<<vacio>>
16972	<<vacio>>
16974	<<vacio>>

Tabla 3. Datos vacíos

3.1.6.2. Datos Inconsistentes. Con la compilación del código realizado en Python, se obtuvo el archivo de entrada *ARFF*, con algunos errores en URL´s que ya no existían. La cantidad hallada fue de 46 de 50000 registros evaluados, por lo cual se trabajó con un total de 49954 registros. Los registros inconsistentes se eliminaron.

La limpieza de datos, permitió identificar y eliminar ciertos registros que generarán inconsistencias e imprecisiones. Un 1% del total de los registros fueron eliminados por ser valores inconsistentes y nulos. La existencia de atributos irrelevantes podría afectar la precisión del algoritmo de aprendizaje, pero al trabajar con árboles de decisión[40], estos poseen mecanismo de aprendizaje donde se realiza la selección de atributos por su relevancia.

[40] Abdelmalik Moujahid e Iñaki Inza. University of the basque country. Department of Computer Science and Artificial Intelligence. Manual de prácticas de minería de datos. usando el software WEKA. https://addi.ehu.es/bitstream/10810/4627/1/tr10-1.pdf. Consultado Enero 2016

3.1.7. Creación de Variables Derivadas.

Esta fase permitió mejorar el rendimiento en el procesamiento y mejor interpretación de los resultados generados. Algunas variables derivadas, identificadas a partir del análisis de las características halladas en las URL´s, se usaron en estudios anteriores [41][42]. Algunas se hallaron a partir de la observación y análisis (Tabla 4).

Atributo	**Nombre**
Uso de dirección IP	is_an_IP
Longitud de la URL	len_URL
TinyURL	HTTP_redirect
URL que tiene el símbolo @	At_Symbol
Uso de "//"para redireccionar	double_slash_redirecting
Añadir prefijo o sufijo separados por (-) en el dominio.	Prefix_Suffix
Uso de uno o varios subdominios	having_Sub_Domain
HTTPS (Protocolo de transferencia de hipertexto con Secure Sockets Layer).	SSL_confidence
Duración del dominio registrado	whois_registeration_length
Favicon.	Favicon
Uso de Puertos no estándares	Port
Inclusión en la URL del símbolo "HTTPS" como parte del dominio	HTTPS_URL
Requerimiento en URL	Request_URL
Anclaje en URL	URL_of_Anchor
Vínculos en etiquetas <Meta>, <Script>, <Link>	Links_in_tags
Envío de información a correo electrónico	Submitting_to_email
URL anormal	Abnormal_URL
Desvío de Sitio Web.	Redirect
Personalización de la barra de estado	on_mouseover
Deshabilitar el botón derecho	RightClick
Uso de la ventana pop-up.	popUpWindow
Redireccionamiento a través de IFRAME	Iframe
Vigencia del dominio	age_of_domain
Registro DNS	DNSRecord
Rango de página	Page_Rank
Es Phishing o es Phishing.	PhishTank
Presencia de Índices	index_url
Presencia de Referencias	References

Tabla 4. Variables derivadas

[41] Aburrous, Maher, Hossain, M. A. Dahal Keshav and Thabtah, Fadi. Experimental Case Studies for Investigating E-Banking Phishing Techniques and Attack Strategies. Published online: 30 April 2010

[42] Mohammad, Rami, McCluskey, T.L. and Thabtah, Fadi Abdeljaber (2014) Intelligent Rule based Phishing Websites Classification. IET Information Security, 8 (3). pp. 153-160. ISSN 1751-8709

Realizado la comparación de los algoritmos basados en árboles de decisión para identificar alertas tempranas ante ataques de Phishing, se observó que el algoritmo J48, agrega un nodo cada vez que se ingresa una característica, mejorando el rendimiento y la calidad del análisis.

3.1.8. Procesamiento.

El *dataset* se almacenó como un archivo CSV (Valores Separados por Comas), para trabajar con WEKA[43], que usa el formato de datos ARFF (Attribute Relation File Format), empleado en el proceso de aprendizaje. El ARFF fue generado a partir del prototipo de software implementado en Python y se usó como la entrada al análisis de los distintos algoritmos de árboles de decisión. El código en Python se encuentra en el Anexo C.

3.1.9. Selección final de atributos.

De los 57 atributos seleccionados inicialmente, se dejaron 28 atributos, descartando 29 debido a su carácter informativo y cuyo porcentaje de poblamiento del 10% no fue representativo, debido a que la condición para uso debe oscilar entre el 60 y 70%.

3.2. DESARROLLO DEL ESCENARIO.

Para esta fase en empleo WEKA[44], herramienta usada para analizar y aplicar técnicas de análisis de datos, a través del aprendizaje automático.[45]

3.2.1. Preparación de los datos en WEKA.

La entrada de datos fue en formato ARFF (Attribute-Relation File Format), como se informó en el apartado 3.1.8

---

[43] Hall M, Frank E, Holmes G, Pfahringer B, Reutemann P, Witten IH. Waikato Environment for Knowledge Analysis. [Online].Available from: ttp://www.cs.waikato.ac.nz/ml/weka/.
[44] http://www.cs.waikato.ac.nz/ml/weka/. Consultada Enero 2016.
[45] https://weka.waikato.ac.nz/explorer. Consultada Enero 2016.

3.3. FASES DEL ANALISIS.

3.3.1. Objetivo del análisis a implementar en WEKA.

Este se orientó a comparar algunos métodos de clasificación basado en arboles de decision para detectar características dentro de URL´s, e identificar aquellas usadas por los atacantes para materializar phishing. Las preguntas planteadas fueron: ¿Qué características dentro de la URL, presenta anomalías que permitan identificar amenazas de phishing? ¿Existen características y diferencias entre las URL´s legítimas y las afectadas por Phishing? ¿Cuántas URL´s están afectadas por Phishing y cuantas no? ¿Se pueden identificar alertas tempranas para identificar ataques de Phishing?. Para dar respuesta a las preguntas, y después del análisis se inició la identificación de las alertas tempranas de ataques de phishing.

3.3.2. Preprocesamiento de los datos.

En esta fase se precisaron los datos a analizar en el archivo ARFF, las instancias y atributos respectivos, detalles y propiedades, como se explica a continuación:

3.3.2.1. Características de los atributos. Los atributos fueron simbólicos, y sus valores oscilaron entre (1, 0,-1). Los valores desconocidos (*Missing*) y únicos (*Unique*) fueron cero. El histograma con la distribución de frecuencia de cada uno de los atributos, permitió su visualización, buscando determinar el mejor clasificador o frontera que dividió los datos en diversas clases. Con el atributo *Request_URL*, se presentó una separación en la clase phishing con respecto a la legítima, con valores diferentes, concluyendo que se tenía un problema desbalanceado.

3.3.2.2. Selección de atributos. Se presentaron atributos redundantes e irrelevantes, los cuales se eliminaron, para evitar complejidad con *overfitting*. Se evaluaron métodos de búsqueda y evaluación, seleccionando un filtro con el método de búsqueda *Greedy Stepwise* y de evaluación *CfsSubsetEval*, con una *Crossvalidation* de 5 hojas, buscando más precisión en el modelo. Se concluyó que

las características *favicon, https, url-anchor, on_mouseover, popup, Iframe y dns_record* fueron seleccionadas en cada uno de los 5 Fold de validación cruzada, a diferencia de *age_rank*, seleccionado solo en 4 Fold.

También se realizó un análisis con un metaclasificador denominado *atribute selected classifier* y un evaluador denominado *BestFirst*, logrando igual porcentaje de aciertos, pero seleccionando 8 atributos. Aplicando estos factores, se observó que el porcentaje de aciertos aumento de 99.994% a 99.996% y las instancias incorrectas disminuyo a 0.004% y el error de la media absoluta fue cero y se evidenció en la matriz de confusión.

3.4. RESUMEN DEL CAPITULO

En este capítulo se detalló el escenario al cual se aplicaría el proceso de aprendizaje mediante arboles de decisión para dar solución al problema. Se describieron las siguientes etapas:

- Descripción del conjunto de datos (Selección, Integración, Manipulación, validación, limpieza de datos)
- Creación de Variables derivadas a partir de observación y análisis.
- Desarrollo del escenario
- Fases de análisis iniciales de valores presentados.

Se hace énfasis en la extracción de conocimiento, como elemento clave del modelo de identificación de ataques de Phishing planteado, definiendo lo siguiente:
En el proceso de clasificación, se emplearon los algoritmos basados en árboles de decisión, a partir del número de atributos y clases, con los datos de entrenamiento, para luego validarlos con todo el conjunto de datos.

En la descripción de los algoritmos para cada una de las diferentes etapas, se presenta el funcionamiento y los resultados que se piden obtener en cada árbol de decisión.

4. ANALISIS DE RESULTADOS

Para el proceso de optimización de las características, se usaron algunos algoritmos basados en arboles de decisión y validación cruzada; donde se evaluaron indicadores como: True Positive Rate o Recall, False Positive Rate, Precisión, F-Measure y Class, además se calculó el porcentaje de aciertos para 10 hojas con datos ordenados.

4.1. RESULTADOS OBTENIDOS.

Al aplicar WEKA a cada uno de los algoritmos seleccionados, se obtuvieron los resultados que se consolidaron en la Tabla 5.

		DecisionStump	LMT	J48	RandomForest	RandomTree	REPTree
Precisión (%)		92,872	99,902	99.994	99,926	99,904	99,904.
Aciertos Phishing %		81.6	99	100	99.8	99.8	99.7
Aciertos Legítimos %		100	100	100	100	99.9	100
Matriz de confusión	Clase Phishing	15729	15755	15729	15765	15768	15755
	Valores fallidos	3556	40	3	30	27	40
	Clase legítimos	30604	34085	34094	34083	34073	34086
	Valores fallidos	0	9	0	11	21	8

Tabla 5. Resultados obtenidos del análisis

En la tabla se aprecia que al usar el algoritmo *DecisionStump* se obtuvo una precisión inferior al 99 %, con relación a los otros algoritmos, debido a ser árboles de un solo nivel, con el algoritmo *J48* se logró una precisión de 99.994%, *RandomForest* 99.9258%, y los algoritmos *RandomTree* y *REPTree* lograron un

99,90%. La validación cruzada usada fue del 10%. Se concluye que los tres atributos más relevantes fueron *Iframe*, *URL_anchor* y *on_mouseover.*

Con respectos a los tiempos empleados, ejecutando los diversos algoritmos, se obtuvieron los resultados relacionados en la Tabla 6.

Algoritmo	**Tiempo en segundos**
DecisionStump	0,26
J48	0,45
LMT	23,38
RandomForest	63,4
RandomTree	0,57
REPTree	2,31

Tabla 6. Duración del proceso

4.2. ANÁLISIS DE VALORES CON CRECIMIENTO EXPONENCIAL.

Según los resultados anteriores, se determina que el porcentaje de instancias incorrectamente clasificadas de una población de 50,000 URL, oscilan entre un 0,074% y 0,098% lo que indica que puede llegar a ser representativo si se aplica este análisis a una cantidad significativa que puede llegar a millones de URL´s analizadas. Para predecir el crecimiento de las instancias clasificadas incorrectamente, se promedian los valores obtenidos en los diferentes métodos, obtenido los valores de la Tabla 7.

Modelo	**Duración (sg)**	**Precisión (%)**
DecisionStump	0,26	92,872%
J48	0,45	99,994%
LMT	23,38	99,902%
RandomForest	63,4	99,926%
RandomTree	0,57	99,904%
REPTree	2,31	99,904%
Promedio obtenido	**15,062**	**98,750 %**

Tabla 7. Promedios Duración y Precisión

Se plantea una función de crecimiento exponencial, la cual se expresa de la siguiente manera:

$$Y_t = Y_0 e^{kt}$$

Donde *Yt* corresponde al valor de la población en el instante t > 0; *Yo* corresponde al tamaño de la población inicial, el cual corresponde a 50000, valor en *t* = 0, si se procede a analizar otras poblaciones; *k* corresponde a la tasa de instancias clasificadas incorrectamente cuyo valor promedio es de 0,296% que ocurre en el transcurso entre *t = 0 y t > 0*, cuyo valor promedio es de 15,062 segundos. El valor de *e = 2,718281828459*

El crecimiento exponencial o geométrico transcurre si el índice de crecimiento propio de una función es correspondiente al presente valor de dicha función, por esta razón se llama formalmente, ley exponencial. La relación entre el tamaño de la variable dependiente con el tamaño del índice de crecimiento es establecido por razón de la ley de proporción directa. Teniendo en cuenta lo anterior, se puede sacar la conclusión de que si en una población *Yo* posee la variación en el tiempo de forma proporcional a su valor, estará implicando un crecimiento vertiginoso en el tiempo

Se hace referencia entonces al crecimiento de una función exponencial (La función exponencial, es lo que conocemos por función real *e* elevado a la potencia de *kt*, *e* es respondiente al número de Euler). Despegando k tenemos:

$$k = \frac{\ln\left(\frac{Y_t}{Y_0}\right)}{t}$$

$$k = \frac{\ln\left(\frac{10.000.000}{50.000}\right)}{15,062} = 0,351767\%$$

Lo anterior nos indica que al aumentar la cantidad de URL´s, se puede lograr un porcentaje de instancias clasificadas incorrectamente de 0,35%, es decir disminuiría el porcentaje de instancias clasificadas correctamente a 99,65%. Ver Figuras 5 y 6 que muestra el comportamiento que se presentó con la función planteada.

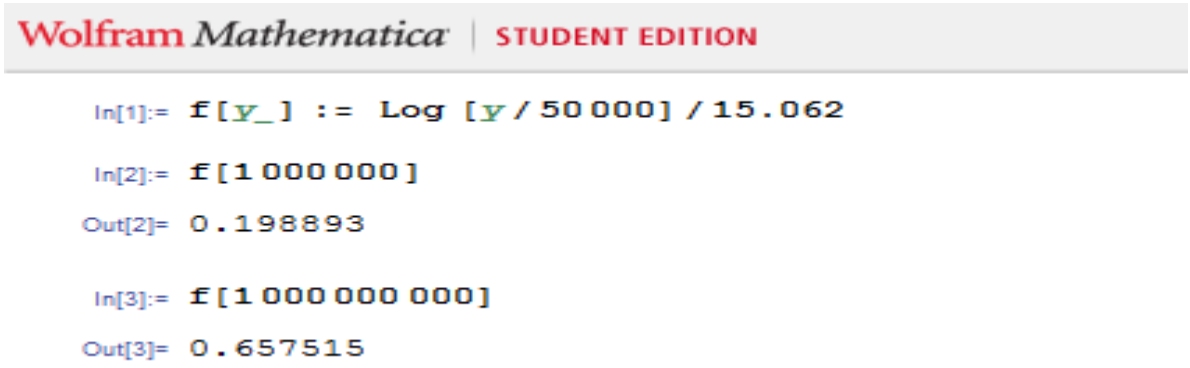


Figura 5. Función de crecimiento exponencial

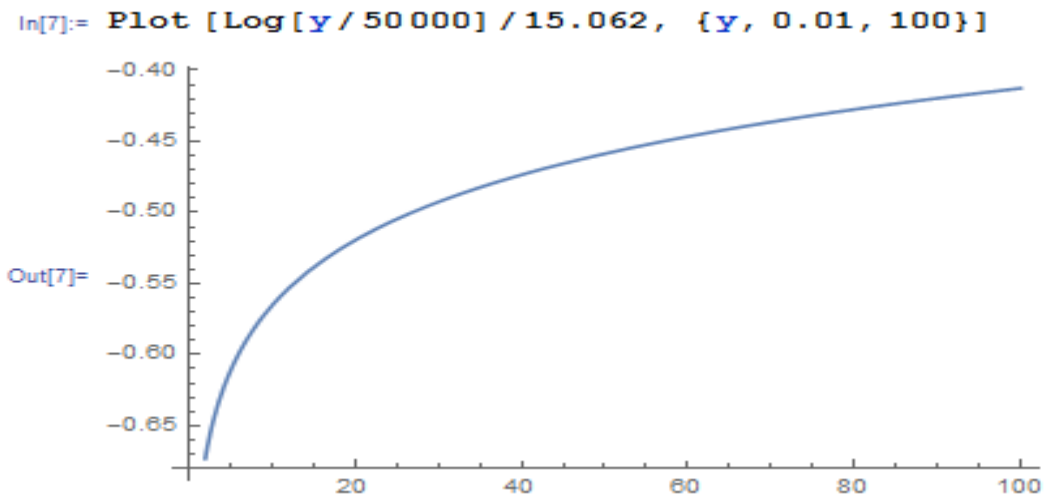


Figura 6. Curva de crecimiento exponencial

Si tenemos una población de 1000.000.000 de URL´s para evaluar con el método de *RandomForest*, usando el tiempo promedio obtenido, se lograría lo siguiente:

$$k = \frac{\ln\left(\frac{1.000.000.000}{50.000}\right)}{15,062} = 2,3088\%$$

Por lo tanto al aumentar la cantidad de URL´s a 1000.000.000, se logrará un porcentaje de instancias clasificadas incorrectamente de 2,3088%, es decir disminuiría el porcentaje de instancias clasificadas correctamente a 97,6912%. Esto indica que la relación existente en identificar Phishing en URL´s es directamente proporcional a la cantidad que se evalúa.

4.3. METODOS DE ENSAMBLE.

Los árboles de decisión tienen diferentes ventajas respecto a los demás métodos de clasificación existentes, usados en minería de datos. Una ventaja es que no requieren de relaciones o dependencias lineales en los atributos. Además debido a la forma en que se construyen estos métodos, se comportan muy bien en espacios de alta dimensión, así como con un gran número de ejemplos de entrenamiento, siendo fáciles de ajustar. El uso de los métodos de ensamble siempre mejora los resultados a pesar que requieren mantener todos los métodos independientes trabajando en paralelo.

4.3.1. Bagging.

Según la técnica de minería de datos[46] el algoritmo de Bagging es un método simple y efectivo para generar un conjunto de clasificadores, a partir de una evidencia. Este

[46] Hernández Orallo, José y Ramírez Quintana María José. Introducción a la Minería de datos. Pearson Prentice Hall. Madrid. 2004. Pág. 488.

se deriva del mecanismo denominado *bootstrap aggregation,* el cual genera subconjuntos de entrenamiento, usando una selección aleatoria con reemplazamiento, sobre una muestra de *m* ejemplos de entrenamiento a partir del conjunto original formado por *m* ejemplos. Si se utiliza siempre el mismo algoritmo, los modelos aprendidos en cada iteración son siempre similares, porque existe un porcentaje de ejemplos comunes en todos los subconjuntos de entrenamiento.

Los arboles de decisión son muy sensibles al conjunto de entrenamiento, es decir pequeños cambios en el conjunto de entrenamiento provocan que el algoritmo de aprendizaje construya un árbol diferente. Esto se presenta cuando el conjunto de entrenamiento es limitado, lo cual se conoce como debilidad o inestabilidad, y se denomina “aprendizaje débil” o *weak learner*. También se usa en métodos de regresión, donde se calcula la media de los valores predichos por cada uno de los modelos del conjunto. Aplicando este método al conjunto de datos de proyecto, se observa que las instancias clasificadas correctamente obtienen un valor de 99.994% y un porcentaje de 0.006% de atributos clasificados incorrectamente. Ver Figura 7.

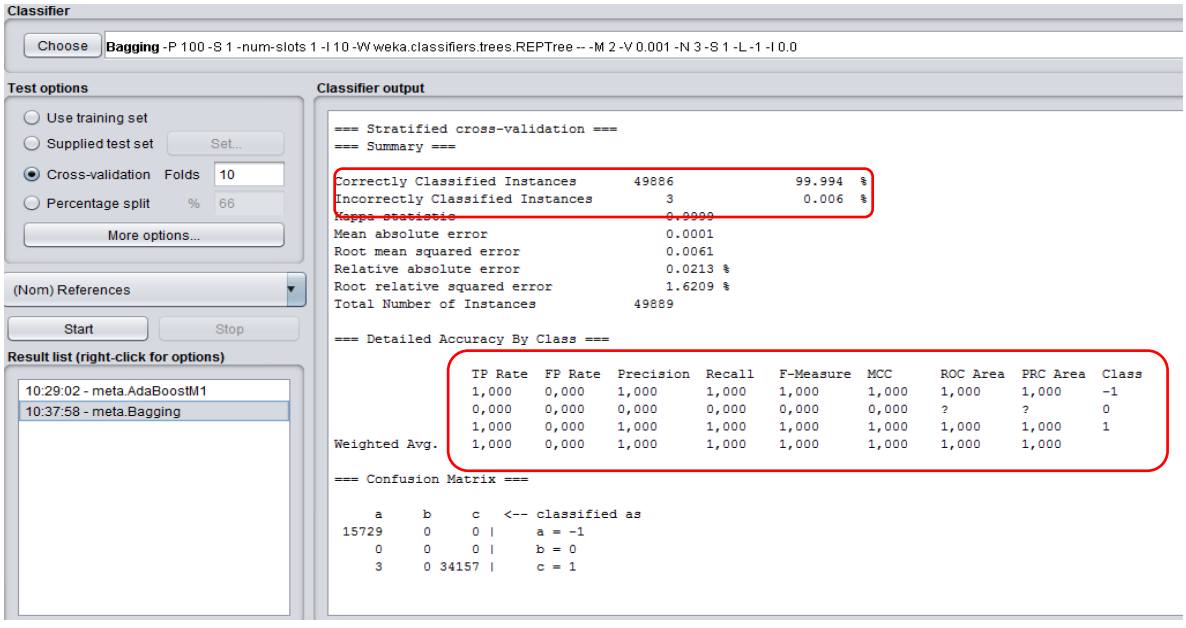


Figura 7. Resultado del método Bagging

4.3.2. Boosting.

Este mecanismo se basa en la asignación de un peso para cada ejemplo del conjunto de entrenamiento. En cada iteración, este realiza un aprendizaje en un modelo que minimiza la suma de los pesos de los ejemplos clasificados erróneamente. Es una estrategia bastante astuta, porque construye los nuevos modelos tratando de corregir los errores cometidos previamente. Da más relevancia para predecir los ejemplos nuevos a los modelos que tiene un mejor comportamiento, a cambio de situarlos en el nivel como lo hace el *Bagging*. Aplicando la técnica de *AdaBoostM1* representativo de la técnica de *Boosting* al conjunto de datos de proyecto, se observa que las instancias clasificadas correctamente, obtienen un valor de 99.978% y un porcentaje de 0.022 % de atributos clasificados incorrectamente. Ver Figura 8.

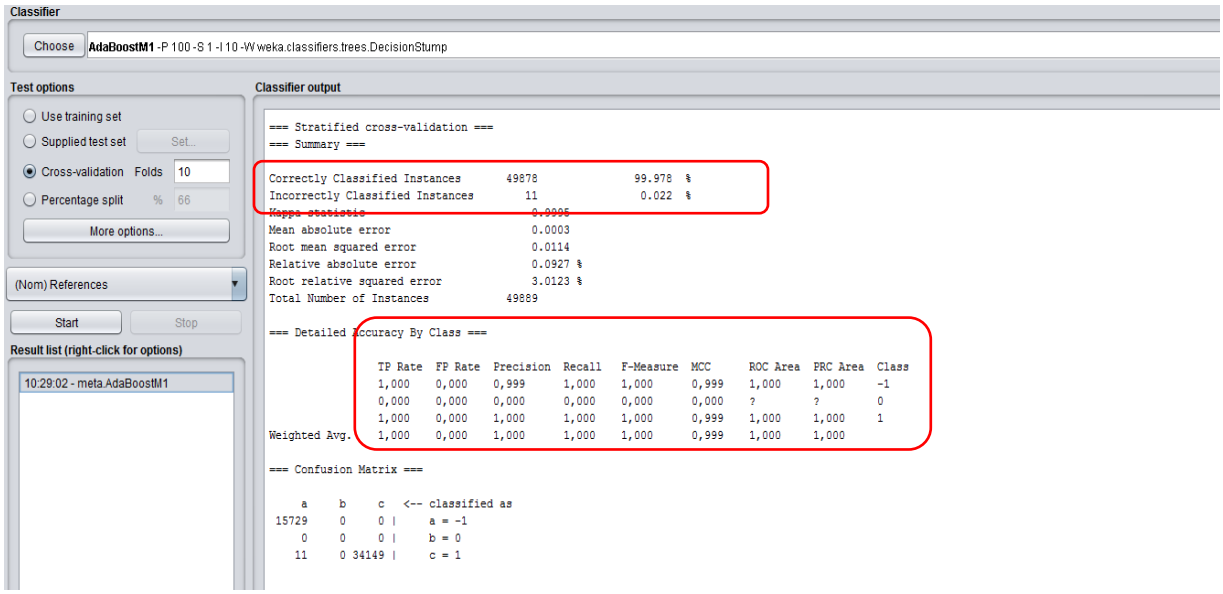


Figura 8. Resultado del método AdaBoostM1

4.3.3. Stacking.

Es un método hibrido basado en la multiclasificación, combina modelos generados por diferentes algoritmos de aprendizaje. Aplicando la técnica de Stacking al

conjunto de datos del proyecto, se observa que las instancias clasificadas correctamente, obtienen un valor de 68.472% y un porcentaje de 31.528 % de atributos clasificados incorrectamente. Ver Figura 9.

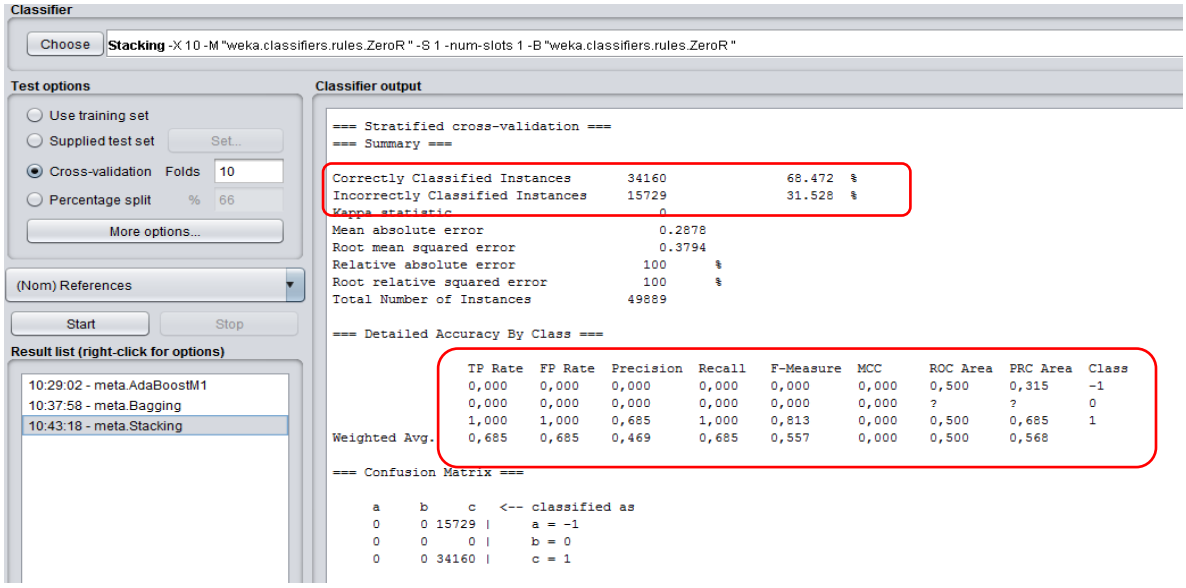


Figura 9. Resultado del método Stacking

Se prueba además con *Voting* generando los mismos resultados de *Stacking*.

A continuación, se muestran los resultados de la precisión y los tiempos empleados en las técnicas de ensamble analizadas. Ver Tablas 8 y 9.

Bagging	*AdaBoostM1*	*Stacking*
1.0	0.99	0.0

Tabla 8. Precisión en métodos de ensamble

Bagging	*AdaBoostM1*	*Stacking*
11 segs	3.7segs	0.66 segs

Tabla 9. Duración en métodos de ensamble

Se observa que al usar los métodos de ensamble *Bagging y AdaBoostM1* se logra mejorar la precisión. Se concluye que *AdaBoostM1* obtiene mejor precisión, pero como los problemas presentan ruido, *Bagging* resulta ser más robusto.

4.4. RESUMEN DEL CAPITULO

Durante el desarrollo del capítulo, se realizaron una serie de experimentos aplicados al conjunto total de datos. Se evaluaron varios algoritmos para presentar el comportamiento de algunas características ubicadas en las URL's y poder identificar alertas tempranas de ataques de phishing, dentro de un proceso de clasificación y validación de un árbol de decisión. El propósito del experimento, fue orientado a evaluar la precisión de los algoritmos: Decision Stump, J48, LMT, Random Forest, RandomTree y REPTree. Se usó el análisis de matrices de confusión y se realizó un análisis a partir de la medida "precisión", encontrando que los arboles de decisión basados en el algoritmo J48, obtuvieron los mejores resultados para el análisis propuesto de identificación de alertas tempranas de ataques de Phishing, logrando una precisión de 99.994%. Como resultado obtenido, se concluye que los atributos más relevantes para identificar phishing en las URL's son: Iframe, URL_anchor y on_mouseover.

CONCLUSIONES Y TRABAJOS FUTUROS

CONCLUSIONES

1. Los arboles de decisión son utilizados con mayor frecuencia para realizar aprendizaje supervisado, presentando un buen comportamiento predictivo en algunos análisis como el referente a la identificación de alertas tempranas de ataques de Phishing. Así mismo se usan técnicas para la extracción de reglas, las cuales permiten representar el comportamiento de un modelo a través de expresiones simbólicas, que facilita una mejor comprensión. El enfoque de este trabajo de grado ha sido utilizar el concepto de árbol de decisión como parte de un análisis realizado a distintos algoritmos basados en árboles de decision para identificar características dentro de las URL´s que pueden producir Phishing, para generar alertas tempranas a los usuarios y evitar que sean víctimas de actos fraudulentos.

2. El conocer e interpretar claramente la función del algoritmo J48, es significativo para identificar el proceso de poda de un árbol de decisión sin necesidad de extraer reglas, evitando realizar varios entrenamientos y reduciendo tiempos de procesamiento.

3. Como resultado de los diferentes experimentos efectuados sobre el modelo de árbol de decisión propuesto para la clasificación del conjunto total de datos, se logró durante la validación un desempeño adecuado de clasificación, por lo cual se empleó un experimento que se enfocó a las URL´s con Phishing, usando el total de los datos.

Los resultados indicaron que el árbol de decisión J48 obtuvo la mejor precisión con un 99,994 % y una validación cruzada de 10 veces; con respecto a DecisionStump, LMT, RandomForest, RandomTree y REPTree, se concluyó que

estas generar mayores tiempos de proceso, mientras que los nodos obtenidos con J48, son mínimas, garantizando mayor precisión en los resultados.

4. Se concluyó que al encontrar simultáneamente características en las URL´s como presencia de Iframe, URL_of_Anchor y on_mouseover pueden indicar que están afectadas por Phishing.

5. Se evaluaron otros conjunto de datos extraídos de diferentes fuentes, con fechas de generación más recientes pero los indicadores arrojados fueron similares en cuanto a precisión y tiempos de ejecución

RECOMENDACION DE FUTUROS DESARROLLOS

Los resultados logrados, indican que el método de clasificación basado en árboles de decisión permite identificar las características más relevantes a tener en cuenta para identificar ataques de Phishing en URL´s. Sin embargo, a pesar de las limitaciones presentadas y los resultados obtenidos, el autor recomienda como futuros desarrollos:

1. Implementar diferentes técnicas de clasificación para generar conocimientos a partir de los algoritmos existentes, con el fin de optimizar las reglas generadas en cada uno de ellos.

2. Inspeccionar otras fuentes de datos referentes a Phishing, para poder ampliar el análisis de las características más representativas y derivadas, de manera sincrónica.

3. Desarrollar nuevas investigaciones y estudios que permitan combinar diferentes técnicas que permitan la extracción de reglas.

4. Diseñar nuevas alterativas de presentación de las reglas extraídas a partir de algoritmos como *DecisionStump, J48, LMT, RandomForest, RandomTree y REPTree.*

Se deben analizar mejores estrategias que permitan optimizar la aplicación de técnicas de Minera de Datos, analizando lo siguiente:

1. El volumen de información que presentan los sitios Web orientados a identificar y analizar ataques informáticos, deben ser potencializados mediante técnicas de extracción en tiempo real.

2. Tener en cuenta la presencia de datos sesgados, y aplicar modelos de minería de datos a la información para la identificación de ataques de Phishing, de manera oportuna y en tiempo real.

3. Manejar información a pesar de la diversidad y dispersión de fuentes, producto de la complejidad al acceso y al cambio continúo de las URL´s analizadas.

Para establecer predicción de los datos, se debe contemplar lo expuesto por[47], quienes calcularon el índice de Jaccard, llegando a lograr el ranking de la URL. Con estos rangos, que puede disuadir a los intentos de presencia o no de Phishing en URL. El clasificador *Random Forest* lo utilizaron para predicción.

[47] PHISHSTORM: DETECTING PHISHING WITH STREAMING ANALYTICS. Pratik Patil*1, Prof. P.R. Devale2. *[1]Information Technology, BVUCOE, Pune, Maharashtra, India. patilpratiknagarale@gmail.com[1, 2] Information Technology, BVUCOE, Pune,

BIBLIOGRAFIA

ABURROUS, Maher, Hossain, M. A. Dahal Keshav and Thabtah, Fadi. Experimental Case Studies for Investigating E-Banking Phishing Techniques and Attack Strategies. Published online: 30 April 2010

AKINYELU and ADEREMI O. Adewumi. . Classification of Phishing Email Using Random Forest Machine Learning Technique Andronicus A. Private Bag Box X54001, Durban 4000, South Africa. Received 23 January 2014; Accepted 11 March 2014; Published 3 April 2014.

AREITO, Javier, Seguridad de la información. Redes, Informática y Sistemas de Información, pág. 34 a 56, 2008, ed. Paraninfo.

CANAVOS, George. Probabilidad y Estadística. Aplicaciones y métodos. Mc Graw-Hill. México. 1998.

CLARK, P. and Boswell, r. Datamining. Practical Machine Learning tolos and Technicals with Java Implementations. Morgan Kaufmann Publishers, 2000.

DIAZ-GOMEZ, Pedro, Valencia Rodriguez José Alfonso and Gomez Luis E. H. Management - An Achilles Heel of Information Assurance Security: A Case Study of Verizon's Data Breach Reports. Cameron University, Lawton, Oklahoma, USA. Universidad Piloto de Colombia. RRP. http://search.proquest.com/openview/82859422c7e73b4b71ec287f2a8622d2/1?pq-origsite=gscholar. June 2011.

DELEN, Dursun. Fast, Andrew. Practical Text Mining and Statistical Analysis for Non-Structured text Data Applications. Elsevier, USA. 2012.

GARCÍA Richard D. Rondón. Gestionar la inseguridad para mejorar la seguridad de la información disponible en Word Wide Web: http://www.acis.org.co/fileadmin /Revista_ 10 5/RGarcia.pdf

GÓMEZ VIEITES, Álvaro. Enciclopedia de la Seguridad Informática. 2ª Edición Actualizada. ISBN978-84-9964-036-5. Ed. Alfa Omega. México DF. 2011. Pág. 148.

HASTIE, Trevor, Tibshirani, Robert, Friedman, Jerome. The Elements of Statistical Learning. Data Mining, Inference, and Prediction, Second Edition. Springer. ISBN 978-0-387-84858-7. USA. 2013

ICONTEC, Instituto Colombiano de normas técnicas y Certificación, Documentación de tesis, trabajos de grado y otros trabajos de investigación, NTC 1486, 2008.

KHADIR, Ripsa P.; Sony P. International Journal of Advanced Research in Computer Science, Mar/Apr2015, Vol. 6 Issue 2, p23-27, 5p, Base de datos: Applied Science & Technology Source.

MANNING C, Raghavan , Schütze H. Introduction to Information Retrieval: Cambridge University Press; 2008.

MOHAMMAD, Rami, McCluskey, T.L. and Thabtah, Fadi Abdeljaber (2012) An Assessment of Features Related to Phishing Websites using an Automated Technique. In: International Conferece For Internet Technology And Secured Transactions. ICITST 2012. IEEE, London, UK, pp. 492-497. ISBN 978-1-4673-5325-0.

MOHAMMAD, Rami, McCluskey, T.L. and Thabtah, Fadi Abdeljaber (2014) Intelligent Rule based Phishing Websites Classification. IET Information Security, 8 (3). pp. 153-160. ISSN 1751-8709

NUTTAPONG SANGLERDSINLAPACHAI and Arnon Rungsawang, “Using Domain Toppage Similarity Feature in Machine Learning-based Web,” in Third International Conference on Knowledge Discovery and Data Mining, 2010.

O. SALEM, H. Alamgir and K. M, “Awareness Program and AI based Tool to Reduce Risk of Phishing Attacks,” in Computer and Information Technology (CIT),2010 IEEE 10th International Conference., June 29 2010-July 1 2010.

PRATIK Patil*1, Prof. P.R. Devale2. *1Information Technology, BVUCOE, Pune, Maharashtra, India. patilpratiknagarale@gmail.com1. 2 Information Technology, Bvucoe, Pune. International Journal of Scientific & Engineering Research, Volume 7, Issue 5, May-2016. ISSN 2229-5518

RAMI M. MOHAMMAD, Fadi Thabtah and Lee McCluskey . Predicting phishing websites based on self-structuring neural network. Received: 17 April 2013 / Accepted: 10 September 2013. Springer-Verlag London 2013

SADEH N, Tomasic A, Fette I. Learning to detect phishing emails. Proceedings of the 16th international conference on World Wide Web. 2007: p. 649-656.

WEBBER Carine, Do Prado Lima Maria de Fátima and Hepp Felipe. Testing Phishing Detection Criteria and Methods. S. Sambath and E. Zhu (Eds.): Frontiers in Computer Education, AISC 133, pp. 853–858. Springer-Verlag Berlin Heidelberg 2012.

WITTEN IH, Frank E. Data mining practical machine learning tools and techniques with Java implementations. New York, NY, USA: March 2002

XINDONG Wu and Vipin Kumar. The Top Ten Algorithms in Data Mining. Chapman & Hall/CRC Data Mining and Knowledge Discovery Series. CRC Press. USA. April 2009.

ZHAO Q. and S. S. Bhowmick, \Association rule mining: A survey," Nanyang Technological University, Singapore, 2006.

ANEXOS

ANEXO A. Expresiones Algebraicas.

a) Tasa de Error:

$$Error = \#errores / \#ejemplo = \frac{|FN| + |FP|}{N}$$

Donde

N= |FN|+|FP|+|TN|+|FN| representa el total de ejemplos del conjunto de validación

b) Exactitud (Accuracy). Es la proporción del número total de predicciones que son Correctas.

$$Accuracy = \frac{|TP| + |TN|}{|FP| + |FN| + |TP| + |TN|}$$

c) Recall. Proporción de casos positivos que fueron identificados correctamente.

$$\mathrm{Re}call = \frac{|TP|}{|TP| + |FN|}$$

d) Precisión (precisión). Es la predicción de casos positivos que fueron clasificados correctamente.

$$\mathrm{Pr}ecision = \frac{|TP|}{|TP| + |FP|}$$

e) Tasa de Falsos Negativos (FN Rate). Es la proporción de casos positivos que son incorrectamente clasificados como negativos. Mensajes suplantados clasificados como mensajes legítimos

$$FN = \frac{|FN|}{|TP| + |FN|}$$

f) Tasa de Falsos Positivos (FP Rate). Es la proporción de casos negativos que son incorrectamente clasificados como positivos. Mensajes legítimos clasificados como suplantados

$$FP = \frac{|FP|}{|TN| + |FP|}$$

ANEXO B. Código para realizar Crawling

```
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package phishingvalid;
import java.io.BufferedReader;
import java.io.File;
import java.io.FileWriter;
import java.io.IOException;
import java.io.InputStreamReader;
import java.io.PrintStream;
import java.net.HttpURLConnection;
import java.net.MalformedURLException;
import java.net.URL;
import java.util.Iterator;
import org.json.simple.JSONArray;
import org.json.simple.JSONObject;
import org.json.simple.parser.JSONParser;
import org.json.simple.parser.ParseException;

/**
 *
 * @author AV
 */
public class PhishingValid {

    /**
     * @param args the command line arguments
     * @throws org.json.simple.parser.ParseException
     */
    @SuppressWarnings("empty-statement")
    public static void main(String[] args) throws ParseException {
        // TODO code application logic here
        try {
            try (PrintStream fileStream = new PrintStream(new File("phishingInvalid.txt"))) {
                for(int i = 677; i<7000; i++){
                    URL url = new
URL("http://api.diffbot.com/v3/discussion?token=29ba97adb5d0353d5fbf70aa4476465f&url=https%3A%2F%2Fwww.phishta
nk.com%2Fphish_search.php%3Fpage%3D"+i+"%26valid%3Dn%26Search%3DSearch");
                    HttpURLConnection conn = (HttpURLConnection) url.openConnection();
                    conn.setRequestMethod("GET");
                    conn.setRequestProperty("Accept", "application/json");
                    if (conn.getResponseCode() != 200) {
                        throw new RuntimeException("Failed : HTTP error code : "
                                + conn.getResponseCode());
                    }
                    BufferedReader br = new BufferedReader(new InputStreamReader(
                            (conn.getInputStream())));
                    StringBuilder sb = new StringBuilder();
                    String line;

                    System.out.println(i+" Output from Server .... \n");
                    while ((line = br.readLine()) != null) {
                        sb.append(line);
                    }
```

```
                JSONParser parse = new JSONParser();
                Object obj = parse.parse(sb.toString());
                JSONObject json = new JSONObject((JSONObject) obj);
                JSONArray jsonObjects = (JSONArray) json.get("objects");
                try{
                    json = (JSONObject) jsonObjects.get(0);
                    jsonObjects = (JSONArray) json.get("posts");
                    Iterator it = jsonObjects.iterator();
                    it.next();
                    while (it.hasNext()) {
                        json = (JSONObject) it.next();
                        if(!json.get("text").toString().endsWith("...")){
                            fileStream.println(json.get("text").toString());
                        }
                    }
                    conn.disconnect();
                } catch (Exception e) {
                    e.printStackTrace();
                }
            }
        }
        } catch (MalformedURLException e) {
                        e.printStackTrace();
        } catch (IOException e) {
                        e.printStackTrace();
            }
    }

}
```

ANEXO C. Prototipo de Software en Python.

```
import xlrd
from time import time
import requests
import urllib3
from bs4 import BeautifulSoup
class Phishingtest:
                ##Si es una IP
    def regla1(cuerpo):
        ips = []
        pos_inicial = -1
        num1 = False
        num2 = False
        num3 = False
        hexad = False
        indicador = 7
        boolindic = cuerpo.startswith("https", 0)
        if(boolindic == True):
                indicador = 8
        boolhexa = cuerpo.startswith("0x",
indicador)
        try:
            while True:
                pos_inicial = cuerpo.index('.',
pos_inicial+1)
                ips.append(pos_inicial)
        except ValueError:
            pass
        apariciones = cuerpo.count(".")
        for i in range (0,256):
            if(apariciones == 1):
                num1 = num1 or
cuerpo.startswith(str(i), ips[0]+1)
            if(apariciones == 2):
                num1 = num1 or
cuerpo.startswith(str(i), ips[0]+1)
                num2 = num2 or
cuerpo.startswith(str(i), ips[1]+1)
            if(apariciones >= 3):
                num1 = num1 or
cuerpo.startswith(str(i), ips[0]+1)
                num2 = num2 or
cuerpo.startswith(str(i), ips[1]+1)
                num3 = num3 or
cuerpo.startswith(str(i), ips[2]+1)
        if(num1 == True or num2 == True or num3
== True):
                return -1
        elif (boolhexa == True):
                return -1
        else:
                return 1
    ##Largo de la URL
    def regla2(cuerpo):
                tamanio = len(cuerpo)
                if(tamanio <= 54 ):
                    return 1
                elif(tamanio > 54 and tamanio <=
75  ):
                    return 0
                else:
                            return -1
    ##Redireccionamiento HTTP Redirect

    def regla3(cuerpo):
                apariciones =
cuerpo.count("bit.ly")
                if (apariciones >= 1):
                            return 1
                else:
                            return -1
    ##Contiene @
    def regla4(cuerpo):
                apariciones = cuerpo.count("@")
                if (apariciones >= 1):
                            return 1
                else:
                            return -1
    ##Contiene redireccionamiento //
    def regla5(cuerpo):
                apariciones = cuerpo.count("//")
                if (apariciones > 1):
                            return -1
                else:
                            return 1
    ##analiza si tiene un - en el cuerpo de
la url
    def regla6(cuerpo):
                ips = []
                pos_inicial = -1
                indicador = 7
                boolindic =
cuerpo.startswith("https", 0)
                if(boolindic == True):
                            indicador = 8
                try:
                            while True:

                pos_inicial = cuerpo.index('/',
pos_inicial+1)

                ips.append(pos_inicial)
                except ValueError:
                            pass
                slash = cuerpo.count("/")
                if(slash == 0):
                            return 1

                if(slash == 2):
                            guion =
cuerpo.find("-",  indicador, ips[1]-1)
                else:
                            guion =
cuerpo.find("-",  indicador, ips[2]-1)
                if(guion != -1):
                            return -1
                else:
                            return 1
    def regla7 (cuerpo):
        try:
            lista = open("lista.txt","r")
            listapro = []
            limite = 0
            m = (cuerpo.split("/")[2])
            if m.startswith("www."):
                limite = 1
            for c in lista:
                c = c.split(" ")[0]
                listapro.append("." + c.lower())
            lista.close()
            for c in listapro:
                if m.count(c) == 1:
                    limite = limite + 1
                    break
            if m.count(".") == limite + 1:
                return -1
            elif m.count(".") == limite + 2:
                return 0
            else:
                return 1
        except:
            return -1
    def regla10(soup, cuerpo):
        try:
            favi = ""
            respuesta = 1
            for url in soup.find_all(
rel="shortcut icon"):
                favi = str(url['href'])
                n , m = None
                if (favi ==None):
                    favi ="html"
                    noes = True
                else:
                    noes = False
                    m = cuerpo.split(".",1)
                    n = favi.split(".",1)

                if (n[0] != m[0] ) or (noes ==
False):
                    respuesta = 1
                else:
                    respuesta = -1
            return respuesta
        except:
            return "N"
    def regla12 (cuerpo):
        try:
            m = (cuerpo.split("/")[2])
            if m.count("https") > 1:
                return -1
            else:
                return 1
        except:
            return 1
    def regla13(soup,cuerpo):
        try:
            ctdor = 0
            apr = 0
            for url in soup.find_all(href=True):
                ctdor = ctdor + 1
                atr = str(url['href'])
                m = cuerpo.split(".",1)
                n = atr.split(".",1)
                if (n[0] != m[0]):
                    apr = apr + 1
            if (ctdor == 0):
                ctdor = percent = 0
            else:
                percent = (apr*100)/ctdor
            if (percent < 22):
                return -1
            elif (percent >= 22 and percent <=
61):
                return 0
            else:
                return 1
        except:
            return "N"
    def regla14(soup):
        try:
            ctdor = 0
            apr = 0
            for url in soup.find_all('a'):
                ctdor = ctdor + 1
                atr = str(url['href'])
                if (atr==None):
                    atr = "html"
                if (atr == " " or
atr.startswith("#")):
                    apr = apr + 1
            if (ctdor == 0):
                ctdor = percent = 0
            else:
                percent = (apr*100)/ctdor
            if (percent < 31):
                return -1
            elif (percent >= 31 and percent <=
67):
                return 0
            else:
                return 1
        except:
```

```
        return "N"
def regla15(soup,cuerpo):
    try:
        ctdor = 0
        apr = 0
        for url in soup.find_all('meta'):
            ctdor = ctdor + 1
            try:
                atr = str(url['href'])
            except(KeyError):
                atr = " "
            m = cuerpo.split(".",1)
            n = atr.split(".",1)
            if (n[0] != m[0]):
                apr = apr + 1

        for url in soup.find_all('script'):
            ctdor = ctdor + 1
            try:
                atr = str(url['href'])

            except(KeyError):
                atr = " "
            m = cuerpo.split(".",1)
            n = atr.split(".",1)
            if (n[0] != m[0]):
                apr = apr + 1
        for url in soup.find_all('link'):
            ctdor = ctdor + 1
            try:
                atr = str(url['href'])

            except(KeyError):
                atr = " "
            m = cuerpo.split(".",1)
            n = atr.split(".",1)
            if (n[0] != m[0]):
                apr = apr + 1
        if (ctdor == 0):
            ctdor = percent = 0
        else:
            percent = (apr*100)/ctdor
        if (percent < 22):
            return 1
        elif (percent >= 22 and percent <= 61):
            return -1
        else:
            return 0
    except:
        return "N"

    ##Lugar del archivo, abre el archhivo, obtiene la hoja por el indice
    start_time = time()
    print ("Se procedera a leer el archivo..... ")
    file_location = "phishing.xlsx"
    workbook = xlrd.open_workbook(file_location)
    sheet = workbook.sheet_by_index(0)

    ##abrir el archivo donde se almacenaran los resultados
    ## w para escritura
    # a para escribir desde el final
    print ("Se abrira archivo para insertar los resultados..... ")
    man_plant = open("plantilla.txt","r")
    man_resul = open("resultados.arff","w")
    ##html_resul = open("html.txt","w")
    for linea in man_plant:
        man_resul.write(linea)
    man_plant.close()
    print ("Comenzara el proceso.....")
    print ("Dependiendo del rendimiento de de computador puede tardar unos cuantos segundos o minutos.....")
    octavo = int (sheet.nrows / 8)
    dohtml = True
    for i in range (1,sheet.nrows):
        cuerpo = str(sheet.cell_value(i,1))
        ##print(cuerpo)
        if(i == octavo):
            print("Vamos en una octava parte del calculo")
        if(i == octavo*2):
            print("Vamos en una cuarta parte del calculo")
        if(i == octavo*4):
            print("Vamos en la mitad del calculo")
        if(i == octavo*6):
            print("Vamos en tres cuartos del calculo")
        man_resul.write("\n")
        ##print (regla1(cuerpo))
        r1 = regla1(cuerpo)
        man_resul.write(str(r1) + ",")
        ##print (regla2(cuerpo))
        r2=regla2(cuerpo)
        man_resul.write(str(r2) + ",")
        ##print (regla3(cuerpo))
        r3=regla3(cuerpo)
        man_resul.write(str(r3) + ",")
        ##print (regla4(cuerpo))
        r4=regla4(cuerpo)
        man_resul.write(str(r4) + ",")
        ##print (regla5(cuerpo))
        r5=regla5(cuerpo)
        man_resul.write(str(r5)+ ",")
        ##print (regla6(cuerpo))
        r6=regla6(cuerpo)
        man_resul.write(str(r6)+ ",")
        ##print (regla7(cuerpo))
        r7=regla7(cuerpo)
        man_resul.write(str(r7)+ ",")
        ##print (regla12(cuerpo))
        r12=regla12(cuerpo)
        man_resul.write(str(r12)+ ",")
        try:
            if i<13:

                ##man_resul.write("\n")
                tr = urllib3.PoolManager()
                r = tr.urlopen('GET',cuerpo)
                soup = BeautifulSoup(r.data, "html.parser")
                r10=regla10(soup,cuerpo)
                man_resul.write(str(r10) + ",")
                r13=regla13(soup,cuerpo)
                man_resul.write(str(r13) + ",")
                r14=regla14(soup)
                man_resul.write(str(r14) + ",")
                r15=regla15(soup,cuerpo)
                man_resul.write(str(r15) + " ")
            else:
                pass
        except:
            man_resul.write("X,X,X,X")
    print ("Proceso terminado con exitoÂ¡Â¡Â¡Â¡")
    man_resul.close()
    ##html_resul.close()
    elapsed_time = time()- start_time

    print("Tiempo de ejecucion total: %.10f segundos." % elapsed_time)
```

ANEXO D. Buenas prácticas técnicas.

Las buenas prácticas[48] se refieren a toda experiencia guiada por principios, objetivos y procedimientos adecuados a una determinada perspectiva y a la experiencia, demostrando su eficacia y utilidad en un contexto concreto[49]. La información es el activo más importante de la empresa, por lo tanto, necesita ser protegida. Las políticas y buenas prácticas a implantar deben cumplir lo establecido por la ISO/IEC 27002:2013, la cual se compone de 14 dominios, 35 objetivos de control y 114 controles, buscando garantizar autenticidad, disponibilidad e integridad de la Información. A continuación, se definen algunas buenas prácticas técnicas generales, referente al uso adecuado de procesos que son vulnerables ante ataques de phishing.

A. Empleo de un servidor de nombre de dominio (DNS) personalizado.

La configuración de un servidor de DNS traduce los nombres de dominio que usa el usuario y los traduce a direcciones IP. Se debe solicitar al proveedor de ISP, el servicio que verifique y filtre los sitios de acuerdo a listas de Phishing.

B. Identificar cuáles son servidores DNS gratuitos.

Existen varios servidores DNS gratuitos entre los que tenemos: Google, Dnsadvantage, OpenDNS, Norton y ScrubIt.

C. Actualización de las versiones más recientes del navegador.

Las versiones más recientes de los navegadores utilizan listas de Phishing, alertando al usuario que el sitio donde accede no es seguro.

[48] http://www.planandino.org/bancoBP/node/3 . Consultado enero 2016

[49] www.ops.org.bo/textocompelto/prensa/concurso-buenaspracticas/conceptos.pdf. Consultado Octubre 2016 .

D. Uso de servicios para comprobar la autenticidad de los sitios donde se va a acceder.

Es importante consultar los servicios que verifican si el acceso a un sitio es seguro o no.

www.PhishTank.com

www.onlinelinkscan.com

www.safeweb.norton.com

www.siteadvisor.com

Se debe siempre copiar la URL y pegarla en el cuadro de texto del sitio para verificar su autenticidad.

E. Comprobación de los siguientes factores técnicos.

Debido a que algunos sitio de Phishing no se filtren o no estén en las listas negras, se deben comprobar algunos aspectos entre los que tenemos:

- Verificar que la URL inicie con HTTPS, para garantizar una conexión fiable.
- Que la URL este identificada en verde en la barra de direcciones.
- Verificar el nombre correcto del dominio donde se está ingresando, ya que se pueden estar utilizando nombres similares.
- Comprobar el contenido del sitio donde se está ingresando, porque puede estar ingresando a un sitio falsificado.
- No se deben enviar correos con contenidos mayores a 5MB.
- Se debe mantener actualizado el software antivirus
- No se debe desinstalar el antivirus en los equipos de cómputo.
- No navegar en sitios de URL's con símbolos sospechosos (-, #, &, etc.)
- Se debe restringir el intercambio de archivos por protocolos FTP.
- Queda prohibido el acceso a los sitios o páginas Web desconocidas o que tengan en sus URL características que generen sospecha.
- Las direcciones deben estar sin símbolos o caracteres que aumenten la su longitud, o ser muy cortas.

- Nunca se debe acceder a enlaces que sean solicitados a través de SMS/MMS/Email ya que pueden ser víctimas de suplantaciones o de código malicioso.
- Se deben usar contraseña que tengan más de 12 posiciones entre caracteres, números y símbolos, letras mayúsculas A, B, C, letras minúsculas a, b, c; números 0, 1, 2, 3, 4, 5, 6, 7, 8, 9; símbolos del teclado (todos los caracteres del teclado que no se definen como letras o números) y espacios ` ~! @ # $ % ^ & * ( ) _ - + = { } [ ] \ | : ; " ' < > , . ? /