Machine Learning techniques for Behavioral Feature Selection in Network Intrusion Detection Systems

Vicente Martínez
Statistical Institute
Universidad de Valparaíso
Valparaíso, Chile
vicente.martinez@alumnos.uv.cl

Rodrigo Salas
Biomedical Engineering School
Universidad de Valparaíso
Valparaíso, Chile
rodrigo.salas@uv.cl

Oliver Tessini
Engineering Faculty
Universidad Andres Bello
Vina del Mar, Chile
oliver.tessini@uandresbello.edu

Romina Torres
Engineering Faculty
Universidad Andres Bello
Vina del Mar, Chile
romina.torres@unab.cl

***Abstract*—A Network Intrusion Detection System (NIDS) uses a user behavior model to detect anomalies and eventual cyberattacks. The most frequent types of attacks are: denial of service, brute force, penetration attacks based on browsers that exploit for example vulnerabilities such as "cross-site scripting´, attacks made by remote execution by command line, SSL attacks based on intercepting encrypted data to gain benefits, a back attack left by applications that expose remote access, among others. Through the use of the IXIA Perfect Storm tool, the data set UNSW-NB 15 was created, which can be obtained from the following link https://www.unsw.adfa.edu.au/unsw-canberra-cyber/cybersecurity/ADFA-NB15-Datasets/. The data set consists of 42 numerical attributes grouped in 6 categories. In this work the technique of Extremely Randomized Trees was applied to select the 15 most relevant characteristics, afterwards, the Random Forest technique was applied to determine, based on the selected characteristics, if the input corresponds to an attack pattern or not. The results show that the method obtains a 97% of performance with the F-measure applied to the Test set. In this work we conclude that it is possible to obtain a subset of relevant characteristics to the behavioral models based on network traffic used by NIDS, without affecting the predictive capacity of learning machines.**

***Index Terms*—Machine Learning, Feature Selection, Anomaly Detection**

I. Introduction

Los ciberataques a los Sistemas de Información podrían comprometer algunos de los tres aspectos esenciales de la seguridad de la información: la confidencialidad, la integridad y la disponibilidad de los datos [1]. Lo anterior podría afectar negativamente a una organización, colocando en riesgo la continuidad del negocio, porque muchas de esas amenazas tales como una denegación de servicio (DoS) afectan directamente la disponibilidad.

Los ciberataques se componen de siete fases [2]: 1) reconocimiento, 2) armamento, 3) entrega 4) explotación, 5) instalación, 6) comando y control Y 7) actuación sobre el objetivo. La primera fase es utilizada para obtener detalles del objetivo, por ejemplo: tipo de tecnología usada o la lista de correos electrónicos. Adicionalmente se busca generar una metodología suficiente de ataque para las siguientes fases. En la segunda fase se utiliza la información de la capa anterior para crear "payload" o códigos maliciosos para aprovechar alguna vulnerabilidad o debilidad del sistema. En la tercera fase se envía el payload generado a la víctima por diferentes medios. En la cuarta fase se ejecuta el "payload" cargando en la víctima y se logra la explotación de la vulnerabilidad. En la fase cinco se instala el malware en el sistema de la víctima. En la fase seis, se espera que ya se logre tomar el control del equipo de la víctima pudiendo robar información sensible. Finalmente en la séptima fase se toman acciones sobre los objetivos logrados (sobre las víctimas), ya sea robando información, dejándolos como "botnets", extraer contraseñas o imágenes, hasta activar cámaras o micrófonos.

Los tipos de ataques más frecuentes y recientes ocurridos durante el 2019 en Chile están relacionados a la explotación de vulnerabilidades ya conocidas en los sistemas, que pueden ser encontrados en el "catálogo de vulnerabilidades comunes (CVE). Estos CVEs identifican claramente la vulnerabilidad, en cuales versiones pueden ser encontradas, solución a la falla o acción de mitigación con referencias a múltiples fuentes de información tales como foros donde se explica su explotación. Ataques tales como recopilación de información, inyección de códigos maliciosos, fraudes, intrusión e intentos de intrusión, entre otros [3]. Adicionalmente, OWASP TOP-TEN 2017 señala que *las vulnerabilidades más frecuentes y de mayor impacto de código malicioso, por ejemplo "Inyección de comandos vía SQL", problemas de la gestión de sesión o quiebre del proceso de autenticación mediante ataques de fuerza bruta, exposición de información sensible, entre otros* [4].

Un sistema de detección de intrusiones en redes (NDIS) se utiliza para detectar actividades maliciosas o anómalas en la red [5], donde se destacan metodologías basadas en detectar anomalías en el comportamiento normal del usuario o del tráfico de red en general. En particular, las metodologías para detectar anomalías se realiza de dos modos; uno de ellos es mediante reglas que buscan patrones o firmas establecidas dentro del tráfico analizado; el otro es mediante análisis de comportamiento de los usuarios, es decir, cualquier desviación del perfil normal de comportamiento es considerado un potencial ataque [6]. El perfil de comportamiento de un usuario puede ser obtenido en base a diferentes indicadores estadísticos, tales como: ingreso, uso de CPU, uso de entradas y salidas, estilos de escritura de documentos, entre otros [6]. Usualmente se consideran dos fases: fase de aprendizaje durante la cual se construye el modelo y fase de detección durante la cual se

TABLE I
CLASIFICACIÓN DE LAS CARACTERÍSTICAS EXTRAÍDAS DEL TRÁFICO DE LA RED

Tipo	Características
Característica de Flujo	Srcip; Sport; Dstip; Dsport; Proto
Características Básicas	state; dur; sbytes; dbytes; sttl; dttl; sloss; dloss; slaad; dlaad; spkts; dpkts
Características de Contenido	swin; dwin; stcpb; dtcpb; smeansz; dmeansz; trans_depth; res_bdy_len;
Características de Tiempo	sjit; djit; stime; ltime; sinkpkt; dinpkt; tcprtt; synack; ackdat;
Otras	is_sm_ips_ports; ct_state_ttl; ct_flw_http_mthd; is_ftp_login; ct_ftp_cmd; ct_srv_src; ct_srv_dst; ct_dst_ltm; ct_src_ ltm;ct_src_dport_ltm;

compara el comportamiento actual con el modelo normal. Dado los problemas de privacidad, la mayoría de los modelos se basan en el comportamiento del sistema más que en el comportamiento del usuario. utilizando para ello el tráfico que considera aspectos tales como: protocolos utilizados, destino y duración de la conexión, frecuencia, entre otros.

En general, se puede evaluar un IDS con métricas que permitan medir la tasa de detección (la cual se define como la razón de vectores maliciosos detectados y la real cantidad de vectores maliciosos), la tasa de detección incorrecta (que mide el porcentaje de vectores de ataques mal clasificados sobre el total existente), el rendimiento (que mide la capacidad de que el sistema sea capaz de actuar sobre el tráfico de la red sin pérdida de paquete), su completitud (el que se define como la capacidad de detectar todos los ataques que comprometan la red), entre otros.

En este trabajo en desarrollo ("ongoing research"), se implementaron y evaluaron dos técnicas de aprendizaje de máquinas con el fin de seleccionar un conjunto mínimo de características que permita que los clasificadores de amenazas no pierdan generalidad. El resto de este trabajo se divide de la siguiente manera. Sección II describe el conjunto de datos con el que se trabajará y las técnicas a utilizar. Sección III discute los resultados de nuestra propuesta. Sección IV presenta la conclusión de esta primera etapa y discute el trabajo futuro.

II. MATERIALES Y MÉTODOS

Esta sección describe el conjunto de datos con el que se trabajará, las técnicas para seleccionar características y las técnicas clásicas utilizadas para construir modelos a partir del aprendizaje de los datos.

A. Descripción del Conjunto de Datos

A través de la utilización de la herramienta IXIA Perfect Storm (https://www.ixiacom.com/es/products/perfectstorm), se creó el conjunto de datos `UNSW-NB 15` que contiene tráfico de red normal como ataques. A través de la ejecución de doce procedimientos Y utilizando Argus-IDS y Bro-IDS, se extrajeron 49 características de cada uno de los archivos pcap y estos fueron clasificados según el tipo de ataque o bien como tráfico normal de no haber ataque. Es importante mencionar que, UNSW-NB15 es un conjunto de datos de libre acceso [1] y puede obtenerse del siguiente enlace https://www.unsw.adfa.edu.au/unsw-canberra-cyber/cybersecurity/ADFA-NB15-Datasets/. En el conjunto de datos UNSW-NB 15 se detectaron $175,341$ registros, y estos fueron categorizados en 9 tipos: 1) Fuzzers, 2) Analysis, 3) Backdoor, 4) DoS, 5) Exploit, 6) Generic, 7) Reconnaissance, 8) Shellcode, 9) Worm, y 10) Normal.

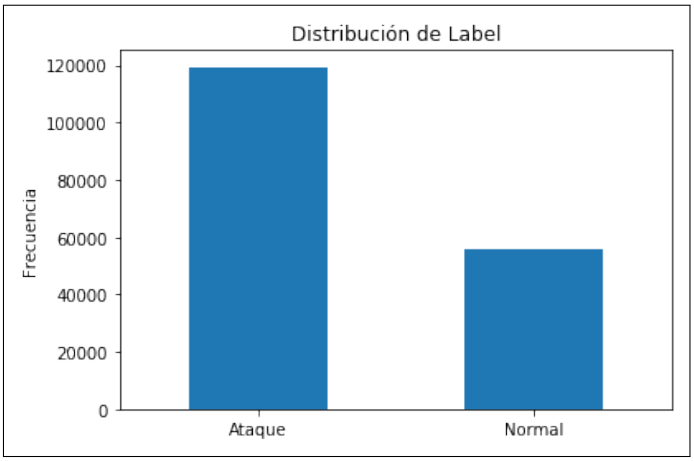


Fig. 1. Normal vs ataque

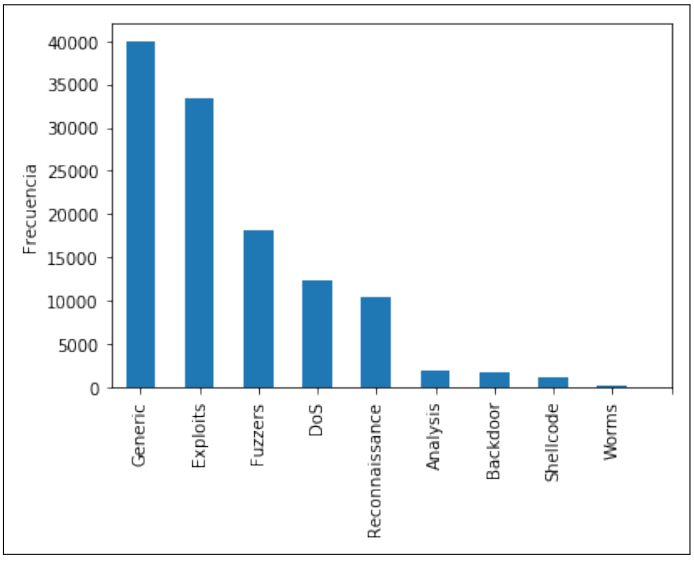


Fig. 2. Distribución de los ataques

Mustafa et al. [1] categorizaron las 49 características en 6 categorías según se muestra en la Tabla I. De las 49 variables, 2 corresponden a variables de salida y 5 variables son categóricas, por lo cual se estudiaron solo las 42 características correspondiente a valores numéricos y la variable de salida que indica si el patrón corresponde o no a un ataque.

B. Técnicas de Aprendizaje de Máquina

La técnicas de aprendizaje de máquina consisten en un conjunto de modelos altamente parametrizados y no lineales que poseen un ajuste o entrenamiento del tipo *data-driven*. Estas técnicas poseen una buena capacidad de generalización, es decir, que poseen un buen desempeño en la predicción utilizando datos nuevos que no fueron utilizados en la fase de ajuste o entrenamiento.

En este trabajo evaluaremos y compararemos el desempeño de las siguientes técnicas de aprendizaje de máquina:

1) *Random Forests*: Los bosques aleatorios consisten en un ensamblado de árboles de decisión, el cual se genera mediante la aplicación de técnicas de remuestreo conocido como Bootstrap. En la construcción de los árboles, cuando se dividen los nodos, la mejor separación es obtenida considerando un subconjunto aleatorio de características.
2) *Xgboost:* Es un modelo de ensamble de árboles de decision, que utiliza un algoritmo llamado boosting, para reducir los errores de cada árbol. Es un modelo secuencial, que utiliza la información obtenida por el árbol anterior, para reajustar los pesos en las peores predicciones para obtener un mejor resultado en el siguiente árbol.

El objetivo de introducir aleatoriedad en estas técnicas de ensamblado de aprendizaje de máquinas es poder reducir la varianza del estimador, lo que se obtiene al promediar las estimaciones de los árboles individuales. No obstante el sesgo es ligeramente incrementado. La implementación de estos modelos se realizó utilizando el toolbox de Scikit-Learn para python, cuya documentación puede encontrarse en https://scikit-learn.org/ [7]

C. Técnicas de Selección de Características

La calidad de las predicciones de los algoritmos de aprendizaje de máquina dependen principalmente de la cantidad de muestras y de la cantidad de información relevante contenida en cada variable para describir el fenómeno de interés [8].

Las principales etapas de un proceso de selección de características son: generación de subconjuntos, evaluación de los subconjuntos, medición del nivel de calidad de la solución en base a cada subconjunto, criterios de detención basado en una función de evaluación. Técnicas populares para reducir el conjunto de características son el análisis de componentes principales y/o componentes independientes. Según Veloz et al. [9], seleccionar las entradas relevantes reducirá el nivel de complejidad de los modelos generados de los datos.

Las técnicas de aprendizaje de máquinas de ensamblado de árboles de decisión tales como el *Random Forest* y *XGBoost* tienen de propiedad de ser útiles para la selección de características. En el Random Forest, cada nodo en el árbol de decisión consiste en una condición sobre una única característica, lo cual genera que el conjunto de datos sea dividido en dos partes tratando de separar la variable de respuesta. La medida utilizada para cuantificar el grado de condición "optimal local" es la impureza (por ejemplo, índice de impureza de Gini), la ganancia de información o la entropía. Por lo tanto, cuando se ajusta un árbol de decisión, se puede calcular en qué cantidad una característica puede decrecer la impureza en un árbol. De esta forma, se promedia la tasa de disminución de la impureza de cada característica en el ensamblado de árboles de decisión y luego se genera un ranking acorde a esta medida.

La ubicación relativa del nodo en el árbol, con respecto a su profundidad, es un indicador de la importancia relativa de la característica con respecto a que tan predictiva podría ser de la variable de salida. Los nodos ubicados en la parte alta del árbol contribuyen a separar una mayor fracción de datos, información que puede utilizarse en conjunto con la disminución de la impureza para medir la importancia relativa de la característica.

D. Validación

Existen distintos metodos para validar y evaluar los modelos de machine learning, el mecanismo más clásico es el de dividir la base de datos en dos subjontuntos llamados, entrenamiento y prueba. El modelo entrenará con el subconjunto de entrenamiento para luego hacer pruebas y predicciones utilizando el subconjunto de prueba. La deficiencia de este método es que solo utiliza un par de subconjuntos basados en una semilla. Esa deficiencia se puede mejorar evaluando los modelos con K-Fold Cross-Validation, que funciona dividiendo el algoritmo en K partes, para que cada uno de estos puedan ser evaluados como entrenamiento y prueba de forma independiente. Según [10], se debe utilizar las técnicas cuando:

- Kfold Cross validation: Es el estándar para evaluar el desempeño de un algoritmo con datos nuevos, con un K de 3, 5 o 10
- Stratified Cross-Validation: es esencial cuando existe una gran número de clases, o hay un desbalance entre las clases
- Train/Test: Cuando se busca rapidez, en algoritmos que trabajan de forma lenta

III. Resultados

En la tabla III las características más relevantes, utilizando los metodos de *XGBoost*, *Random Forest*, *K-best Chi*^{2}, *Regresión logistica* y *Lasso*.

En las figuras 3 y 4 se muestran las matrices de confusión evaluadas en el conjunto de test para los modelos *Random Forest* y *XGBoost* utilizando validación cruzada stratified. Asi mismo en las 5 y 6 se muestran las matrices de confusion de los modelos *Random Forest* y *XGBoost*, utilizando las variables 15 variables relevantes. Los resultados muestran que a pesar que hemos disminuido considerablemente las características utilizados en el modelo de aprendizaje de máquinas, el desempeño de mantiene muy similar.

En la tabla III se muestran los resultados del desempeño evaluado en el conjunto de test al utilizar Random Forest y XGBoost como clasificadores, donde se utilizó el conjunto de

TABLE II
SELECCIÓN DE CARACTERÍSTICAS

	XGBoost	Random Forest	K-Best Chi2	R. Logistic	Lasso
proto				X	X
state					X
dur					
sbytes	X	X	X	X	
dbytes			X	X	
sttl		X	X	X	X
dttl		X		X	X
sloss					X
dloss					
service					X
sload	X	X	X		X
dload	X	X	X	X	
spkts				X	
dpkts				X	
swin			X	X	X
dwin			X	X	
stcpb	X				
dtcpb	X				
smean	X	X			X
dmean		X	X		X
trans_depth					
res_bdy_len			X		
sjit	X		X	X	
djit	X		X	X	
dur	X	X			X
rate		X	X	X	X
sintpkt	X		X	X	X
dintpkt		X	X	X	
tcprtt	X	X			
synack	X	X			X
ackdat	X	X			
is_sm_ips_ports					
ct_state_ttl		X			
ct_flw_http_mthd					
is_ftp_login					
ct_ftp_cmd					
ct_srv_src	X				X
ct_srv_dst	X	X			
ct_dst_ltm					
ct_src_ltm					
ct_src_dport_ltm					
ct_dst_sport_ltm					
ct_dst_src_ltm					

Fig. 3. Random Forest 10Fold

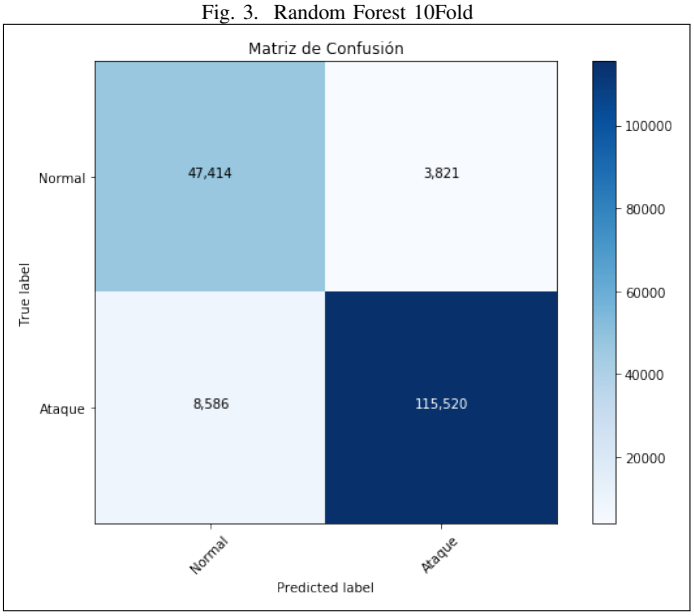
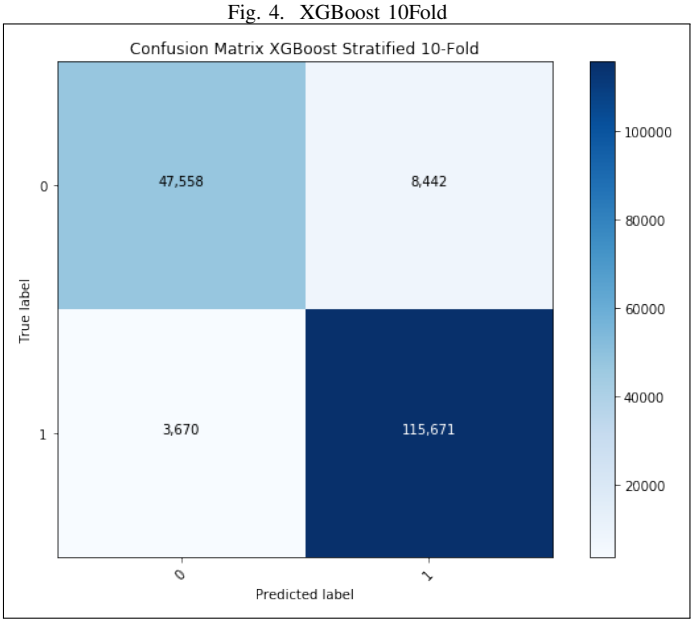


Fig. 4. XGBoost 10Fold



características completo y también considerando sólo las 15 más relevantes.

IV. Conclusión y Trabajo Futuro

Con el fin de poder entender un patrón de ciberataque, el disponer de una gran cantidad de variables hace difícil su entendimiento, más aún si se considera que las variables poseen diferentes niveles de importancia. Por lo tanto, identificar las características relevantes podrían ayudar para obtener entendimiento humano del modelo del comportamiento de la red. En este trabajo encontramos un subconjunto de características relevantes para los modelos de comportamiento basados en tráficos de red utilizados por los NIDSs, sin perjudicar la capacidad predictiva del modelo.

Como trabajo futuro, se espera cruzar estos resultados con los resultados obtenidos con métodos de aprendizaje profundo los cuales pueden seleccionar de manera automática las características sin que estas sean indicadas previamente de manera arbitraria. Además, se espera construir un IDS de prueba, que no sólo considere modelos de comportamiento basados en el tráfico de red, sino que también aquellos basados en el comportamiento normal de los usuarios. Se espera además dotar a estos sistemas con ciclos de auto-adaptación que permitan monitorear la obsolescencia de los modelos de manera de

TABLE III
Desempeño de los clasificadores

Técnica	Metodo	No. Caract.	Exactitud	Precisión	Sensibilidad	F1
XGBoost	10Fold	42	0.9309	0.9319	0.9692	0.9502
XGBoost	10Fold	15	0.9289	0.9291	0.9694	0.9488
XGBoost	Train Test	42	0.9607	0.9636	0.9792	0.9713
XGBoost	Train Test	15	0.9533	0.9548	0.9778	0.9662
Random Forest	10Fold	42	0.9292	0.9308	0.0.9678	0.9490
Random Forest	10Fold	15	0.9279	0.9293	0.9677	0.9481
Random Forest	Train Test	42	0.9596	0.9621	0.9793	0.9706
Random Forest	Train Test	15	0.9526	0.9557	0.9759	0.9657

Fig. 5. Random Forest 10Fold 15 Variables mas relevantes

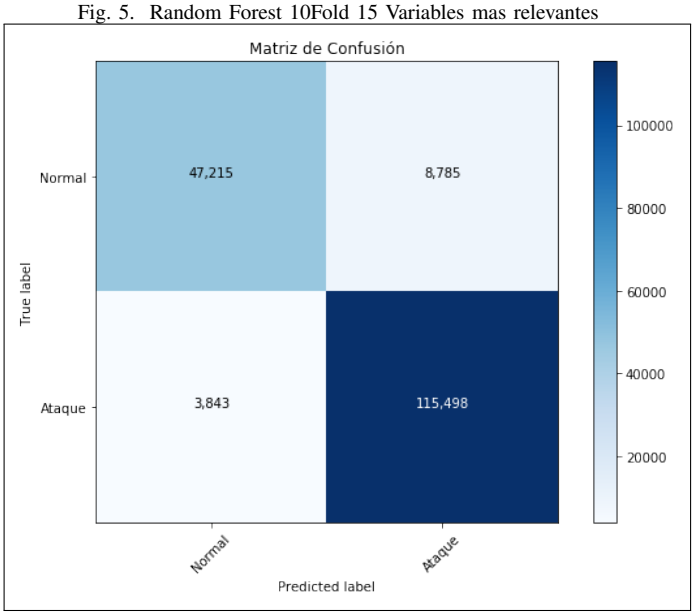
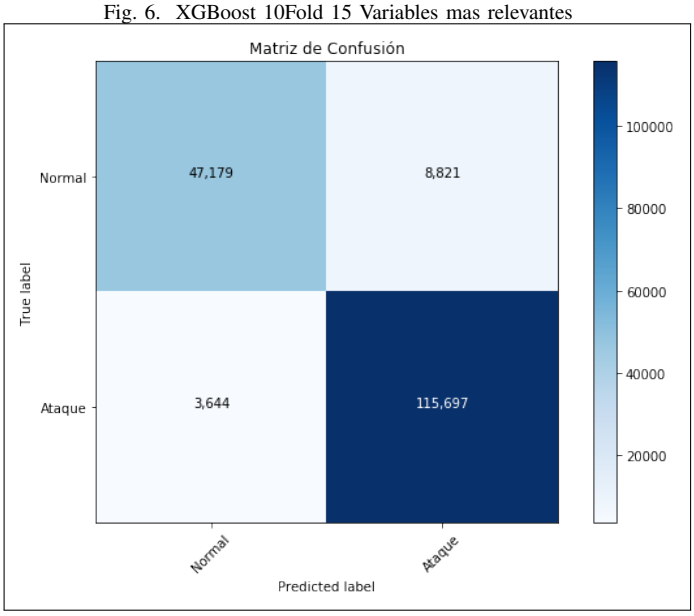


Fig. 6. XGBoost 10Fold 15 Variables mas relevantes



determinar en tiempo de ejecución cuando es necesario re-entrenar y reemplazar los modelos sin la necesidad de reiniciar el IDS.

AGRADECIMIENTOS

El trabajo de Romina Torres fue parcialmente apoyado por el proyecto interno DI-02-19/R de la Universidad Andrés Bello.

REFERENCES

[1] N. Moustafa and J. Slay, "The significant features of the unsw-nb15 and the kdd99 data sets for network intrusion detection systems," in *2015 4th International Workshop on Building Analysis Datasets and Gathering Experience Returns for Security (BADGERS)*. Los Alamitos, CA, USA: IEEE Computer Society, nov 2015, pp. 25–31. [Online]. Available: https://doi.ieeecomputersociety.org/10.1109/BADGERS.2015.014

[2] F. Garba, "The anatomy of a cyber attack: Dissecting the cyber kill chain," 03 2019. [Online]. Available: https://www.researchgate.net/publication/330658097_The_Anatomy_of_a_Cyber_Attack_Dissecting_the_Cyber_Kill_Chain

[3] C. G. de Chile, "Informe de gestión de seguridad cibernética csirt," Gobierno de Chile, Tech. Rep., 06 2020. [Online]. Available: https://www.csirt.gob.cl/media/2020/07/14IMT20-00024-02.pdf

[4] OWASP, "Los diez riesgos más críticos en aplicaciones web," OWASP, Tech. Rep., 2017. [Online]. Available: https://wiki.owasp.org/images/5/5e/OWASP-Top-10-2017-es.pdf

[5] N. Moustafa and J. Slay, "The evaluation of network anomaly detection systems: Statistical analysis of the unsw-nb15 data set and the comparison with the kdd99 data set," *Information security journal: A global perspective*, vol. 25, no. 1-3, pp. 18–31, 2016.

[6] J. Peng, K.-K. R. Choo, and H. Ashman, "User profiling in intrusion detection," *J. Netw. Comput. Appl.*, vol. 72, no. C, pp. 14–27, Sep. 2016. [Online]. Available: https://doi.org/10.1016/j.jnca.2016.06.012

[7] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of*

Machine Learning Research, vol. 12, pp. 2825–2830, 2011.

[8] S. Chabert, T. Mardones, R. Riveros, M. Godoy, A. Veloz, R. Salas, and P. Cox, "Applying machine learning and image feature extraction techniques to the problem of cerebral aneurysm rupture," *Research Ideas and Outcomes*, vol. 3, p. e11731, 2017. [Online]. Available: https://doi.org/10.3897/rio.3.e11731

[9] A. Veloz, R. Salas, H. Allende-Cid, H. Allende, and C. Moraga, "Identification of lags in nonlinear autoregressive time series using a flexible fuzzy model," *Neural Process. Lett.*, vol. 43, no. 3, pp. 641–666, Jun. 2016. [Online]. Available: http://dx.doi.org/10.1007/s11063-015-9438-1

[10] J. Brownlee, *XGBoost With Python*. Machine Learning Mastery, 2016.