Detección de Ataques

Sara Zavala, Juan Fernando de León, Alexa Bravo[1]

[1]*Facultad de Ingenieria,*
Ingenieria en Ciencias de la Computacion y TI, Universidad del Valle de Guatemala,
Ciudad de Guatemala, Guatemala.

Malware attacks may result in theft, encryption, removal of personal data, or even the risk of being spied on without consent. For this reason, people have sought ways to counteract them, proposing to implement and validate innovative machine learning methods to detect them. This article describes the use of three models, specialized in classification, to test their precision in malware detection.

I. INTRODUCCIÓN

El malware está diseñado para interferir con el funcionamiento normal de una computadora, es un término general para llamar a los virus, troyanos y otros programas que se utilizan para infectar sistemas y redes. Existen diferentes tipos de ataques de malware que se utilizan para propagarse a los sistemas informáticos. Entre esos ataques se encuentra el SPAM, el intercambio de archivos por medios extraíbles y el intercambio de archivos punto a punto en el que el malware se introduce en archivos como música o imágenes. A continuación se describe el uso de tres modelos de Machine Learning, árbol de decisión, maquina de soporte vectorial y Naive Bayes, para la clasificación de intrusiones.

II. MARCO TEÓRICO

A. Árbol de Decisión

Es un modelo de aprendizaje supervisado, que se utiliza para resolver problemas de clasificación y regresión. Para los problemas de clasificación se emplea, cuando se quiere predecir el valor de una variable por medio de la clasificación de información en función de otras variables.

Se le conoce como "árbol de decisión"porque comienza con un nodo raíz, que se expande en ramas construyendo una arquitectura parecida a un árbol. Es una representación gráfica para obtener todas las posibles soluciones de un problema y/o decisión basado en las condiciones dadas. Son fáciles de construir, visualizar e interpretar.

Su estructura se basa en; nodos internos que son las caracteristicas a considerar para tomar una decisión, las ramas que representan el rumbo de la decisión y los nodos finales que son el resultado de la decisión tomada.

Crear un árbol de decisión para un problema de clasificación se lleva a cabo aplicando el algoritmo de Hunt que se basa en dividir sub-conjuntos que buscan una separación óptima, para obtenerla se puede considerar el Error de Clasificación, el índice Gini o la Entropía.

B. Máquina de Soporte Vectorial

Conocido como SVM por sus siglas en inglés (Support Vector Machine) es uno de los algoritmos supervisados más populares para resolver problemas de clasificación en aprendizaje automático.

Tienen como objetivo encontrar la forma óptima de clasificar entre varias clases. Esta clasificación se realiza maximizando el margen n-dimensional de separación entre clases, al que se conoce como hiperplano y los vectores que definen el borde de la separación son los vectores de soporte.

Cuando las clases no son linealmente separables se utiliza el kernel que es inventar una dimensión nueva en la cual se pueda encontrar un hiperplano y separar las clases.

Las maquinas de soporte vectorial tienen la ventaja de ser eficaz en espacios de altas dimensiones y son muy versátiles. Se pueden utilizar en el

reconocimiento de imágenes, en filtros de spam, reconocimiento óptico de caracteres, entre otros.

C. Naive Bayes

Es una técnica de clasificación basada en el Teorema de Bayes con la suposición ïngenua"de independencia condicional entre cada par de características dado el valor de la variable de clase.

El modelo Naive Bayes es fácil de construir y muy útil para conjuntos de datos muy grandes, ya que funciona bien en la predicción de clases múltiples.

Una de las ventajas de Naive Bayes es que utiliza un enfoque probabilístico; todos los cálculos se realizan en tiempo real y los resultados se generan instantáneamente.

El algoritmo de Bayes se puede utilizar en predicción en tiempo real, predicción multiclase, clasificación de texto, Filtrado de spam, análisis de sentimientos , sistemas de recomendación, entre otros.

III. METODOLOGÍA

A. Análisis Exploratorio

Se cargaron los cuatro datasets y se realizó una breve visualización de las 49 variables que contiene cada uno, incluyendo el nombre de cada una y el tipo.

B. Pre-prosesamiento

Se juntaron los datasets para formar uno solo con una cantidad de 40263811 datos. De este nuevo dataset se tomó una muestra representativa con un total de 201319 datos. Se revisó la columna "LABEL" para observar los tipos de ataques con los que cuenta y se realizo un conteo de cada uno.

Se eliminaron:
UDP Scan, NULL Scan, XMAS Scan y FIN Scan que no eran necesarios porque cuentan con muy pocos datos y se guardo este nuevo dataset creado.

C. Selección de Características

El data set con el que se trabajó constaba de 40 columnas en las cuales se albergan alrededor de cuatro mil millones de datos. Cada una de las columnas tenian caractersisticas distintivas de las variables que el data set manejaba, sin embargo, debido a la magnitud de los datos y las necesidades que se tenia, se tuvo que realizar una depuracion de de las mismas para poder quedarnos con menos cantidad de datos pero siempre sin perder las partes fundamentales del dataset, o al menos perder el menor porcentaje posible.

Las columnas que se consideraron poco relevantes para este analisis fueron las siguientes:

BIFLOWDIRECTION, DIRECTION, DSTTOSRCSECONDBYTES, FIREWALLEVENT, FLOWACTIVETIMEOUT, FLOWID, FLOWINACTIVETIMEOUT, FRAMELENGTH, IPV4DSTADD, IPV4SRCADDR, MAXIPPKTLEN,MINIPPKTLEN, OOORDERINPKTS, OOORDEROUTPKTS, SAMPLINGINTERVAL y SRCTODSTSECONDBYTES

```
#Selección de características

# Eliminar columnas que no aportan

df_sample_1.drop(columns=['BIFLOW_DIRECTION', 'DIRECTION', 'DST_TO_SRC_SECOND_BYTES', 'FIREWALL_EVENT', 'FLOW_ACTIVE_TIMEOUT',
```

Figura 1: Dropeo de columnas no utiles

Ya que se depuraron las columnas, luego fue necesario realizar un cambio en el tipo de variables que se tenian. Las variables cualitativas fueron cambiadas de manera que estas pudieran ser cuatificadas y ser reconocidas de esta manera. Esto se realizó con el fin de poder comparar datos de manera mas sencilla. Estos cambios se pueden notar en las columnas de las etiquetas y de los tipos de protocolos existentes.

D. Implementación de Modelos

Para la implementación de los modelos de Machine Learning se separaron los 197718 datos en un 55 % de entrenamiento, 15 % para validación y el 30 % para pruebas. Se realizo un balanceo

de datos utilizando prácticas vistas en clase. La matriz de correlación con las variables del dataset. Se cálculo la asimetria y curtosis de las columnas LABEL y L7PROTONAME. Y un diagrama de dispersión con las variables LABEL, PROTOCOLMAP y L7PROTONAME.

Se empezó con la implementación del modelo de árbol de decision, luego de separar los datos, como se menciona anteriormente se toman los de la columna LABEL que son los que tienen la información necesaria para determinar si es un ataque de malware.

El siguiente algoritmo que se implementó fue un SVM, debido a que este modelo es bastante robusto, se amplio el tiempo de ejecución y se tuvo que fraccionar el dataset.

Y por ultimo se implementó el modelo de Naive Bayes, utilizando el método multinomial, asi se obtuvo el accuracy de los datos separados en test, train y validación.

IV. RESULTADOS

A. Árbol de Decisión

	precision	recall	f1-score	support
Normal flow	1.00	1.00	1.00	149937
SYN Scan - aggressive	1.00	1.00	1.00	149842
Denial of Service R-U-Dead-Yet	1.00	1.00	1.00	150004
Denial of Service Slowloris	1.00	1.00	1.00	150217
accuracy			1.00	600000
macro avg	1.00	1.00	1.00	600000
weighted avg	1.00	1.00	1.00	600000

Figura 2: Matriz de Confusión test

	precision	recall	f1-score	support
Normal flow	1.00	1.00	1.00	75212
SYN Scan - aggressive	1.00	1.00	1.00	74387
Denial of Service R-U-Dead-Yet	1.00	1.00	1.00	75056
Denial of Service Slowloris	1.00	1.00	1.00	75345
accuracy			1.00	300000
macro avg	1.00	1.00	1.00	300000
weighted avg	1.00	1.00	1.00	300000

Figura 3: Matriz de Confusión validación

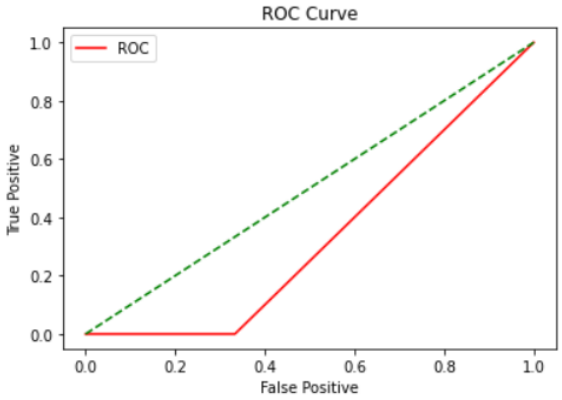

Figura 4: Gráfica de ROC

Se obtuvieron resultados muy buenos en las matrices de confusión obtenidas, ya que ambas cuentan con un accuracy igual a 1, lo que indica que el modelo funciona muy bien. Sin embargo la curva de ROC nos indica resultados negativos ya que se encuentra abajo del clasificados al azar.

B. Máquina de Soporte Vectorial

```
[[1 0 0]
 [0 1 0]
 [0 2 0]]
              precision    recall  f1-score   support

           2       1.00      1.00      1.00         1
           3       0.33      1.00      0.50         1
           4       0.00      0.00      0.00         2

    accuracy                           0.50         4
   macro avg       0.44      0.67      0.50         4
weighted avg       0.33      0.50      0.38         4
```

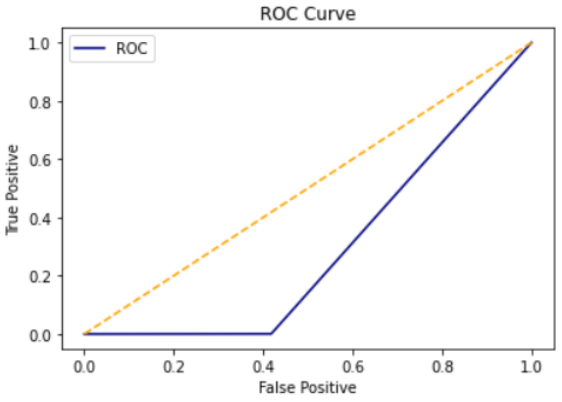

Figura 5: Gráfica de ROC

C. Naive Bayes

	precision	recall	f1-score	support
1	0.74	0.72	0.73	149937
2	0.79	1.00	0.88	149842
3	0.99	0.70	0.82	150004
4	0.91	0.97	0.94	150217
accuracy			0.85	600000
macro avg	0.86	0.85	0.84	600000
weighted avg	0.86	0.85	0.84	600000

Figura 6: Matriz de Confusión

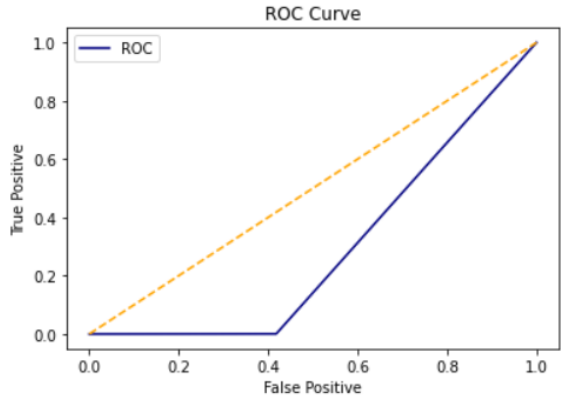

Figura 7: Gráfica de ROC

Se obtuvo un resultado aceptable en la matriz de confusión, ya que tiene un accuracy igual a 0.85. Sin embargo la curva de ROC nos indica resultados negativos ya que se encuentra abajo del clasificados al azar.

V. DISCUSIÓN

Al analizar los resultados de los tres algoritmos implementados, se pudo observar el desempeños de estos. El algoritmo de Arboles de Decisión nos devolvió una certeza del 100 %, sin embargo, esto no es del todo confiable. Esto se pudo deber a que el modelo calló en sobreajuste. Por lo que no es recomendable utilizar este modelo para la predicción.

Seguido de esto, se trabajó con la implementacion de SVM. Derivado del tamaño del dataset, trabajar con este algoritmo resultó ser complicado. Por lo tanto se redujo mas el dataset, esto causó que la muestra no fuera representativa. Los resultados no fueron certeros, se puede apreciar, en la matriz de confución que se obtuvo una precisión del 50 %.

Finalmente, el algoritmo Naive Bayes resultó ser el mas preciso de los tres . Teniendo una certeza de 85 % podemos decir que el modelo generado es un modelo acptable para predecir los distintos tipos de ataques.

VI. CONCLUSIONES

- El modelo de SVM es el menos efectivo ya que por la cantidad de datos este fue el que más tiempo tardó para realizar procedimientos
- Las matrices de confusión indican que, el modelo que tuvo la mayor exactitud para predecir lo solicitado fue el del árbol de decisión. Se puede observar que tiene un porcentaje bastante alto en comparación a los otros dos.
- Como parte del preprocesamiento, fue necesario descartar todas las columnas que tuvieran datos fuera de los parámetros establecidos.

VII. BIBLIOGRAFIA

[1] Palo Alto Network. (2020). Malware. https://www.paloaltonetworks.com/cyberpedia/what-is-malware

[2]Under-sampling — Version 0.9.0. (2022). Retrieved 26 February 2022, from https://imbalanced-learn.org/stable/under$_s ampling.html$

[3]How to balance a dataset in Python. (2021). Retrieved 26 February 2022, from https://towardsdatascience.com/how-to-balance-a-dataset-in-python-36dff9d12704

[4]curves?, D. (2016). Did I just invent a Bayesian method for analysis of ROC curves?. Retrieved 26 February 2022, from https://stats.stackexchange.com/questions/189411/did-i-just-invent-a-bayesian-method-for-analysis-of-roc-curves: :text=ROC

[5]scikit-learn.(2017-2020) Support Vector Machine. https://scikit-learn.org/stable/modules/svm

[6]Brownlee, J. (2018). How to Use ROC Curves and Precision-Recall Curves for Classification in Python. Retrieved 26 February 2022, from https://machinelearningmastery.com/roc-curves-and-precision-recall-curves-for-classification-in-python/

[7](2022). Retrieved 26 February 2022, from material/2017/06/dataan-port/NaiveBayesPractical.pdf https://www.archer.ac.uk/training/course-