Máster Universitario en Ingeniería Informática

Universidad de Granada

Cyber Security Attacks

Estudio del dataset

Sistemas inteligentes para la Gestión en la Empresa (SIGE)

Autores

Mario Martínez Sánchez

Cristina del Águila Martín

Índice

1. Introducción **2**

2. Descripción de las técnicas empleadas **3**

2.1. Análisis exploratorio 3

2.1.1. Estructura del dataset 3

2.1.2. Evolución temporal de los ataques 4

2.1.3. Distribución geográfica 5

2.1.4. Correlación entre variables numéricas 5

2.2. Preprocesamiento 6

2.2.1. Tratamiento de valores nulos y binarización 6

2.2.2. Extracción de información del dispositivo 7

2.2.3. Generación de variables nuevas 9

2.3. Clasificación 9

2.3.1. Modelos tradicionales 9

2.3.2. Redes convolucionales 12

2.3.3. Ensemble 14

3. Análisis comparativo **16**

3.1. Comparativa de modelos 16

3.2. Discusión de resultados 16

4. Conclusión **18**

5. Bibliografía **19**

1. Introducción

En el presente trabajo se aborda un proceso completo de análisis y modelado aplicado al conjunto de datos Cyber Security Attacks (Incribo, 2023), el cual contiene información detallada sobre ataques cibernéticos, incluyendo atributos técnicos, datos de red, características del dispositivo y más. El objetivo principal ha sido desarrollar un sistema de clasificación capaz de identificar de forma precisa distintos tipos de ataques a partir de la información proporcionada.

Para alcanzar este objetivo, se ha seguido una metodología estructurada que comienza con un análisis exploratorio de los datos (EDA), en el que se identificaron patrones, valores atípicos y posibles relaciones entre variables. Posteriormente, se aplicaron técnicas de preprocesamiento como la normalización de datos, codificación de etiquetas y selección de características, con el fin de optimizar el rendimiento de los modelos.

En cuanto a la fase de clasificación, se han evaluado diferentes enfoques. Por un lado, se implementaron modelos clásicos de Machine Learning mediante el uso de algoritmos como Random Forest, XGBoost y versiones optimizadas mediante búsqueda de hiperparámetros. Por otro lado, se desarrolló una red neuronal profunda capaz de aprender representaciones más abstractas de los datos. Finalmente, se diseñó un modelo ensemble que combina las predicciones de todos los clasificadores anteriores, con el fin de mejorar la robustez y precisión del sistema.

Este enfoque integral permite comparar el rendimiento de distintas familias de modelos y valorar el impacto del aprendizaje profundo en contextos de ciberseguridad.

2. Descripción de las técnicas empleadas

2.1. Análisis exploratorio

Se ha realizado un análisis exploratorio sobre el conjunto de datos *Cyber Security Attacks*. Este dataset contiene información detallada sobre distintos ciberataques, como la geolocalización, el tipo de dispositivo, el año del ataque y otros atributos técnicos.

2.1.1. Estructura del dataset

El dataset original contiene las siguientes columnas:

```
RangeIndex: 40000 entries, 0 to 39999
Data columns (total 25 columns):
 #   Column                  Non-Null Count  Dtype
---  ------                  --------------  -----
 0   Timestamp               40000 non-null  object
 1   Source IP Address       40000 non-null  object
 2   Destination IP Address  40000 non-null  object
 3   Source Port             40000 non-null  int64
 4   Destination Port        40000 non-null  int64
 5   Protocol                40000 non-null  object
 6   Packet Length           40000 non-null  int64
 7   Packet Type             40000 non-null  object
 8   Traffic Type            40000 non-null  object
 9   Payload Data            40000 non-null  object
 10  Malware Indicators      20000 non-null  object
 11  Anomaly Scores          40000 non-null  float64
 12  Alerts/Warnings         19933 non-null  object
 13  Attack Type             40000 non-null  object
 14  Attack Signature        40000 non-null  object
 15  Action Taken            40000 non-null  object
 16  Severity Level          40000 non-null  object
 17  User Information        40000 non-null  object
 18  Device Information      40000 non-null  object
 19  Network Segment         40000 non-null  object
 20  Geo-location Data       40000 non-null  object
 21  Proxy Information       20149 non-null  object
 22  Firewall Logs           20039 non-null  object
 23  IDS/IPS Alerts          19950 non-null  object
 24  Log Source              40000 non-null  object
dtypes: float64(1), int64(3), object(21)
memory usage: 7.6+ MB
```

Figura 1: Columnas del dataset

Se verificó la existencia de valores nulos en varias columnas. La Figura 2 muestra las cinco variables con datos faltantes.

Alerts/Warnings	20067
IDS/IPS Alerts	20050
Malware Indicators	20000
Firewall Logs	19961
Proxy Information	19851

dtype: int64

Figura 2: Proporción de valores nulos por variable

2.1.2. Evolución temporal de los ataques

Se extrajo el año a partir de la columna `Timestamp` para visualizar la evolución anual de los incidentes. En la Figura 3 se observa cierta estabilidad en los últimos años, exceptuano el año 2023, pues este dataset es de dicho año y todavía no se habían registrado los valores.

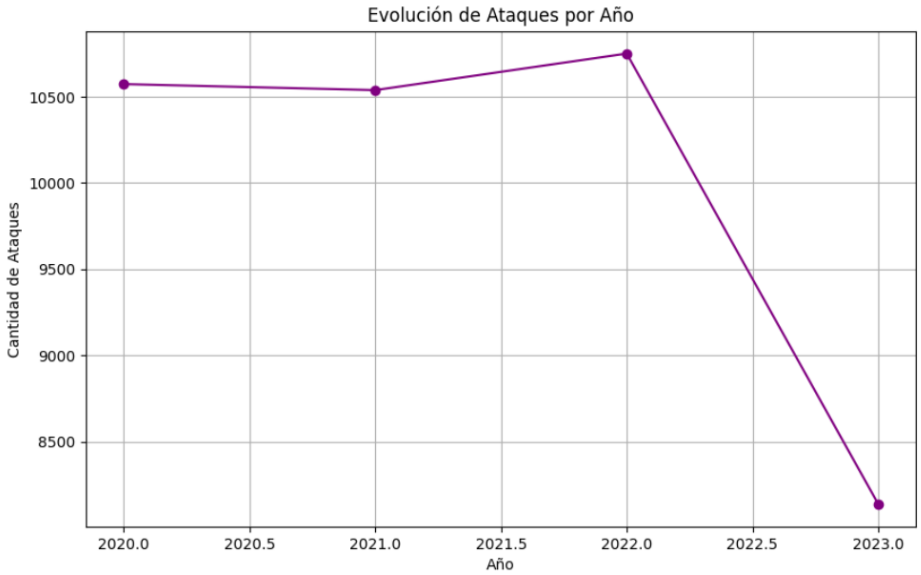

Figura 3: Evolución de ataques por año

2.1.3. Distribución geográfica

A partir de la columna `Geo-location Data` se extrajo el país asociado a cada incidente. La Figura 4 muestra los 10 países con mayor número de ciberataques reportados, siendo números bastante semejantes (entre 1450 y 1500 aproximadamente).

Figura 4: Top 10 países con más incidentes

2.1.4. Correlación entre variables numéricas

Finalmente, se analizó la correlación entre las variables numéricas presentes en el dataset. La Figura 5 muestra un mapa de calor donde se pueden identificar relaciones débiles entre características, por lo que se concluye que no hay necesidad de eliminar columnas por su correlación.

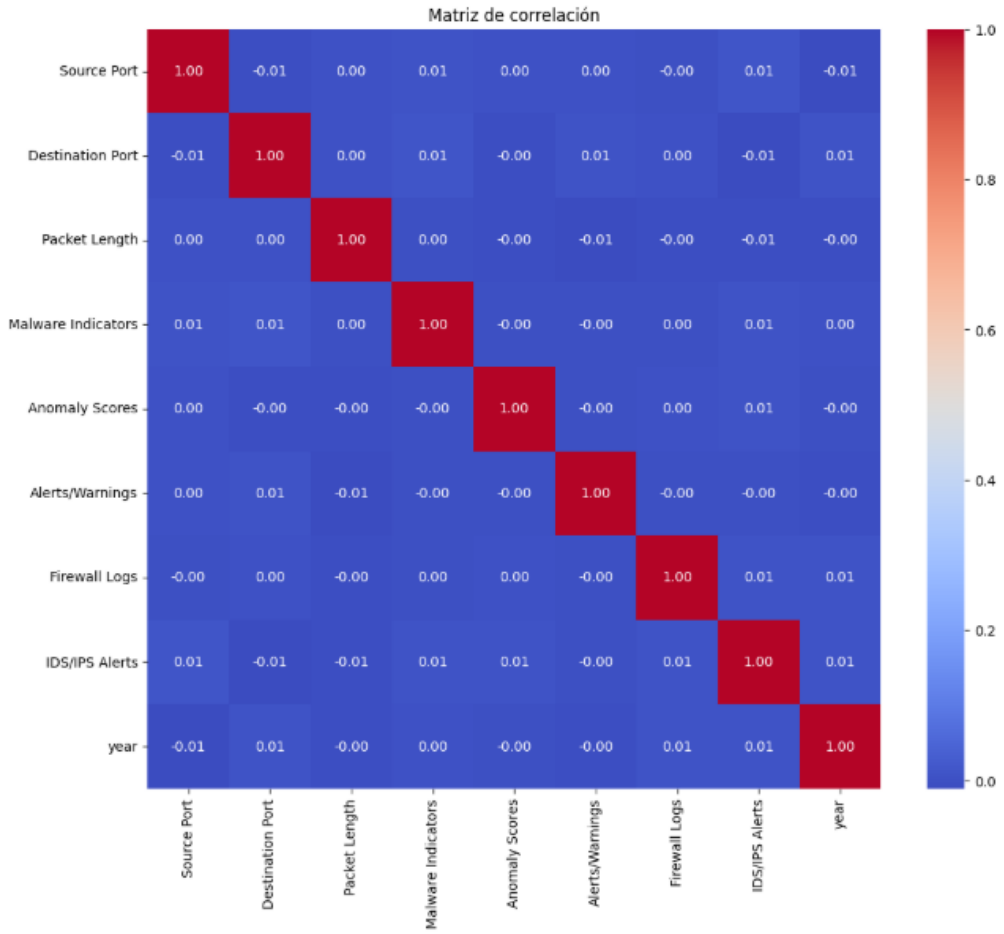

Figura 5: Matriz de correlación entre variables numéricas

2.2. Preprocesamiento

Antes de aplicar modelos de clasificación, nos centramos en preparar y limpiar los datos para mejorar la calidad del análisis. Durante esta etapa, hemos transformado las columnas, tratado valores nulos y extraido nuevas variables relevantes.

2.2.1. Tratamiento de valores nulos y binarización

Aunque algunas columnas no tienen valores nulos, otras sí presentan una gran cantidad de datos faltantes, como `Alerts/Warnings`, `IDS/IPS Alerts`, `Malware Indicators`, `Firewall Logs` y `Proxy Information`. En lugar de eliminar estas columnas, hemos decidido transformarlas en variables binarias, codificando la presencia o ausencia de alerta, actividad maliciosa o registro como 1 o 0. De esta manera, mantenemos la información útil sin descartar registros.

```
[nan 'Alert Triggered']
[nan 'Alert Data']
['IoC Detected' nan]
['Log Data' nan]
['150.9.97.135' nan '114.133.48.179' ... '60.51.30.46' '137.76.130.8'
 '112.169.115.139']
['Known Pattern B' 'Known Pattern A']
```

Figura 6: Transformación a binario de columnas de alertas

Para la columna `Proxy Information`, que tiene direcciones IP, hemos reemplazado los valores nulos con la categoría `Sin proxy`, ya que hemos interpretado la ausencia de proxy como una característica en sí misma.

Además, en la columna `Attack Signature`, observábamos que aparecían patrones conocidos como `A` y `B`. Reasignábamos todos los valores distintos a `Known Pattern B` como `A`, simplificando así esta variable categórica para facilitar su uso en los modelos.

2.2.2. Extracción de información del dispositivo

A partir de la columna `Device Information`, hemos extraido el nombre del navegador, que estaba antes de la barra inclinada (`/`). Así hemos creado una nueva columna llamada `Browser`, la cual hemos usado para analizar la distribución de dispositivos según el navegador.

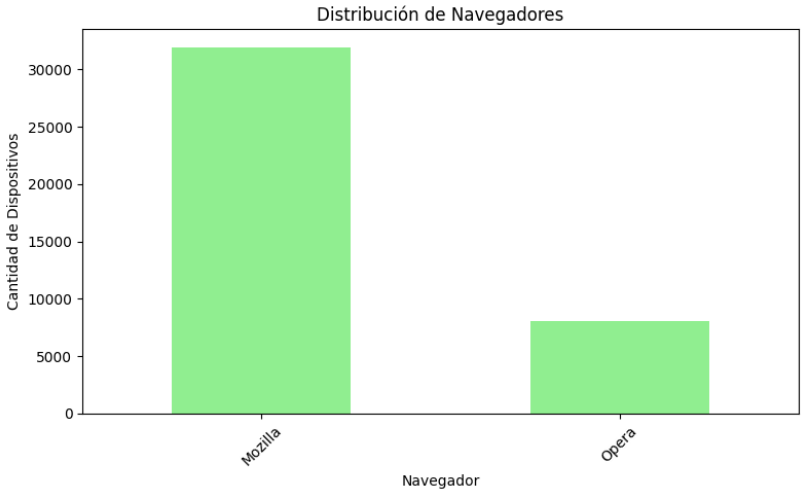

Figura 7: Distribución de navegadores utilizados

Además, hemos extraido el SO del dispositivo aplicando expresiones regulares. Hemos clasificado los sistemas operativos en cinco grandes grupos: `Windows`, `Linux`, `Android`, `Mac OS` y `Otro`. Esta simplificación nos va a permitir identificar patrones por plataforma.

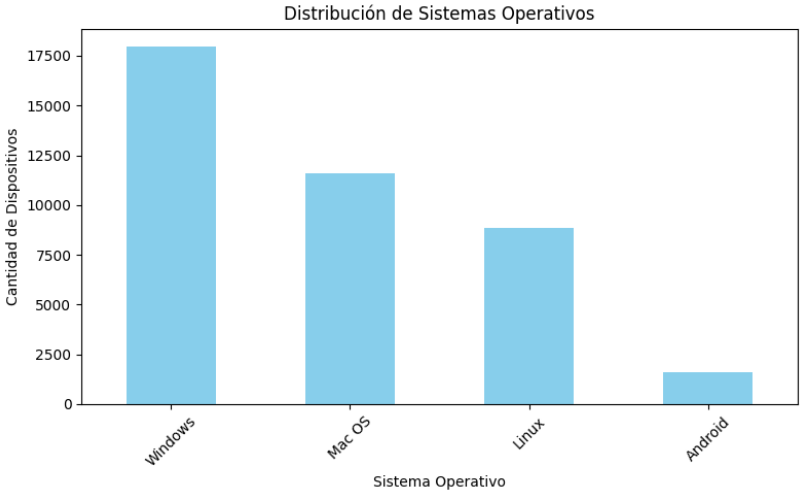

Figura 8: Distribución de sistemas operativos

Dentro de `Mac OS`, también hemso contado los dispositivos específicos como `iPad`, `iPod`, `iPhone` y `Macintosh`, ya que aportan detalles útiles sobre el tipo de terminal comprometido.

```
Número de iPads: 1551
Número de iPods: 2656
Número de iPhones: 4223
Número de Macintosh: 5813
```

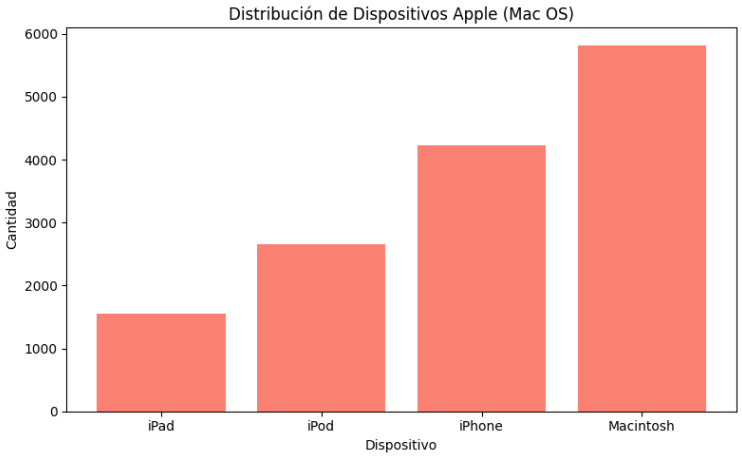

Figura 9: Distribución de dispositivos Apple (Mac OS)

2.2.3. Generación de variables nuevas

Durante el preprocesamiento también hemos generado nuevas variables a partir de otras ya existentes. Por ejemplo, hemos extraido el año del ataque desde la columna `Timestamp`, lo que nos ha facilitado el análisis temporal. También hemos extraido el país desde la columna `Geo-location Data`, que contenía datos geográficos en formato de texto. Nos hemos quedado con el último elemento tras separar por comas, asumiendo que representaba el país de origen del incidente.

2.3. Clasificación

En esta sección abordamos distintas técnicas de clasificación con el objetivo de predecir el tipo de ataque informático al que pertenece una muestra. Para ello, comparamos modelos tradicionales de machine learning, modelos basados en redes neuronales y una técnica de ensamblado (ensemble) que combina las predicciones de todos ellos.

La variable objetivo es `Attack Type`, con tres clases: **DDoS**, **Intrusion** y **Malware**. A continuación, se detallan los resultados obtenidos por cada enfoque.

2.3.1. Modelos tradicionales

Los modelos tradicionales que hemos usado han sido Random Forest y XGBoost. Estos algoritmos son muy conocidos por su eficacia en tareas de clasificación multiclase. Hemos aplicado preprocesamiento con codificación one-hot para variables categóricas y normalización para las variables numéricas. Además, hemos empleado `SelectKBest` para la selección de características relevantes.

Random Forest sin optimización El primer modelo entrenado ha sido un Random Forest con 50 árboles y una profundidad máxima de 30. La matriz de confusión obtenida se muestra en la Figura 10. Como puede observarse, el modelo presenta un fuerte sesgo hacia la clase **DDoS**, con una alta *recall* en esta clase pero bajo rendimiento en las demás.

	precision	recall	f1-score	support
DDoS	0.34	0.98	0.51	672
Intrusion	0.59	0.04	0.07	663
Malware	0.46	0.03	0.05	665
accuracy			0.35	2000
macro avg	0.46	0.35	0.21	2000
weighted avg	0.46	0.35	0.21	2000

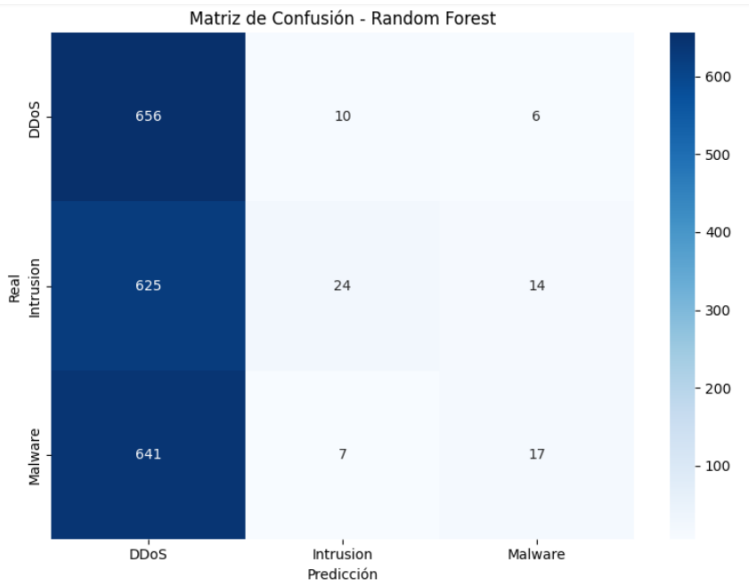

Figura 10: Matriz de confusión - Random Forest sin optimización

El modelo clasificó correctamente 656 de 672 instancias reales de tipo DDoS, lo que indica un buen desempeño para esta clase. Sin embargo, una gran proporción de las muestras reales de tipo Intrusion y Malware fueron clasificadas erróneamente como DDoS (625 y 641 respectivamente). Esto evidencia una fuerte inclinación del modelo hacia la clase DDoS, obteniendo un porcentaje de precisión mucho inferior al deseado.

Random Forest con optimización (GridSearchCV) Posteriormente, hemos hecho una búsqueda de hiperparámetros usando `GridSearchCV`, evaluando distintas combinaciones de profundidad del árbol (`max_depth`), número de árboles (`n_estimators`) y número mínimo de muestras para dividir un nodo (`min_samples_split`). Los mejores hiperparámetros encontrados fueron:

- `max_depth = 20`
- `n_estimators = 100`
- `min_samples_split = 2`

La matriz de confusión del modelo optimizado se muestra en la Figura 11. En esta figura puede observarse que el modelo sigue teniendo un fuerte sesgo hacia la clase **DDoS**, clasificando erróneamente la mayoría de las instancias de **Intrusion** y **Malware** como DDoS.

```
Mejores hiperparámetros encontrados:
{'max_depth': 20, 'min_samples_split': 2, 'n_estimators': 100}
              precision    recall  f1-score   support

        DDoS       0.35      0.98      0.51       672
   Intrusion       0.66      0.04      0.08       663
     Malware       0.51      0.04      0.07       665

    accuracy                           0.36      2000
   macro avg       0.50      0.35      0.22      2000
weighted avg       0.50      0.36      0.22      2000
```

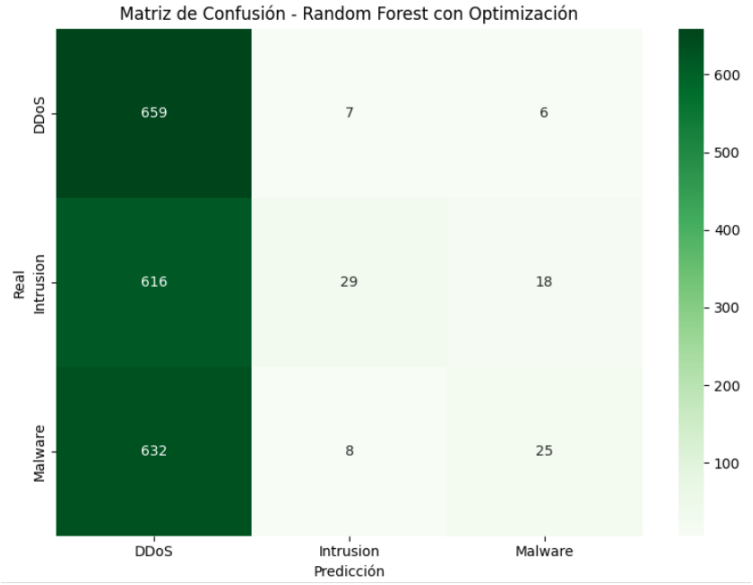

Figura 11: Matriz de confusión - Random Forest con optimización

En cuanto a las métricas, hemos obtenido una **precisión** del **35 y 36 %**. El **recall** de la clase **DDoS** alcanzó el **98 %**, mientras que para las clases **Intrusion** y **Malware** se redujo considerablemente (entre el 3 y el 4 %). A pesar de la optimización, el modelo sigue sin ser efectivo para detectar clases, lo que evidencia la necesidad de técnicas adicionales.

XGBoost XGBoost es un algoritmo de *boosting* basado en árboles, ampliamente reconocido por su eficiencia y rendimiento en tareas de clasificación. Para esta evaluación, hemos entrenado un clasificador `XGBClassifier` con los siguientes hiperparámetros:

- `n_estimators = 100`
- `max_depth = 10`
- `learning_rate = 0.1`

La Figura 12 muestra la matriz de confusión obtenida. A diferencia de los modelos anteriores, XGBoost ha conseguido una clasificación más equilibrada entre las tres clases: **DDoS**, **Intrusion** y **Malware**. Sin embargo, la confusión entre las clases sigue siendo considerable, y el modelo no logra una separación clara entre las categorías.

	precision	recall	f1-score	support
DDoS	0.35	0.23	0.28	672
Intrusion	0.33	0.33	0.33	663
Malware	0.34	0.46	0.39	665
accuracy			0.34	2000
macro avg	0.34	0.34	0.33	2000
weighted avg	0.34	0.34	0.33	2000

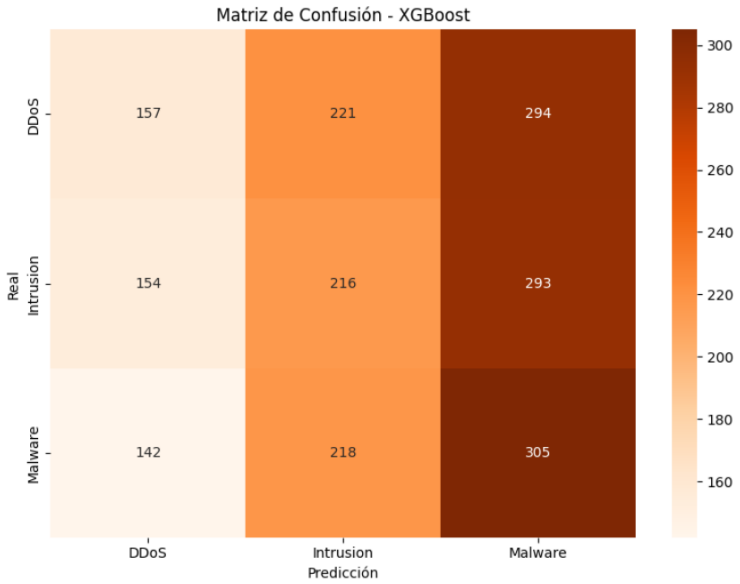

Figura 12: Matriz de confusión - XGBoost

En términos generales, XGBoost ha ofrecido un rendimiento ligeramente superior en cuanto a equilibrio entre clases, aunque la exactitud global sigue siendo baja (con una precisión del 34 %). Esto podría indicar la necesidad de mayor ingeniería de características o el uso de técnicas avanzadas de ajuste.

2.3.2. Redes convolucionales

Hemos implementado una red neuronal profunda usando *Keras* y *TensorFlow*, con el objetivo de explorar el desempeño de modelos basados en aprendizaje profundo. La arquitectura del modelo se describe a continuación:

- Capa densa de 512 neuronas con activación ReLU, seguida de normalización por lotes y *dropout* del 40 %.
- Capa densa de 256 neuronas con activación ReLU, normalización por lotes y *dropout* del 30 %.
- Capa densa de 128 neuronas, activación ReLU y *dropout* del 30 %.
- Capa densa de 64 neuronas, activación ReLU y *dropout* del 20 %.
- Capa de salida con activación `softmax` para clasificación multiclase.

La función de pérdida que hemos usado ha sido `sparse_categorical_crossentropy`, y el optimizador ha sido `adam`. Hemos empleado `EarlyStopping` con paciencia de 5 épocas para evitar el sobreajuste.

El modelo ha conseguido una precisión muy superior a la de los modelos anteriores: 73 %. Como se puede observar en la matriz de confusión, sigue fallando bastante en la predicción de **Malware**, indicando en gran parte de los casos que se trata de **Intrusion Predicción**.

	precision	recall	f1-score	support
DDoS	1.00	0.97	0.99	672
Intrusion	0.56	1.00	0.72	663
Malware	0.89	0.21	0.34	665
accuracy			0.73	2000
macro avg	0.82	0.73	0.68	2000
weighted avg	0.82	0.73	0.68	2000

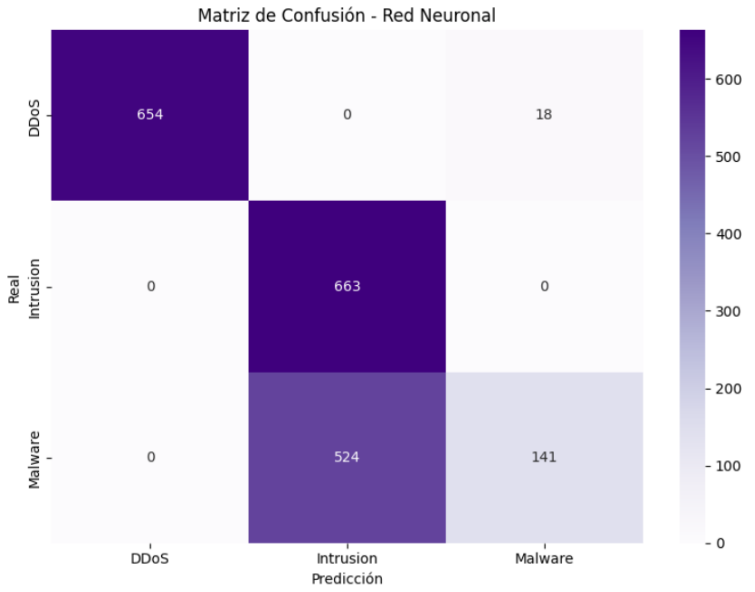

Figura 13: Matriz de confusión - Red Neuronal Multicapa

De este modo, el modelo ha mostrado un fuerte aprendizaje en entrenamiento, aunque podría mejorar en la predicción de la clase **Malware**, lo que se intentará resolver con la implementación de un ensemble.

2.3.3. Ensemble

Para mejorar la precisión del sistema de predicción y aprovechar las fortalezas de distintos algoritmos, hemos implementado un modelo de *ensemble learning* mediante la técnica de **votación**. Este enfoque consiste en combinar múltiples clasificadores base para tomar decisiones más robustas.

En este caso, hemos integrado los siguientes clasificadores:

- **Random Forest**
- **Random Forest con hiperparámetros optimizados**
- **XGBoost**
- **Res Neuronal Convolucional (CNN)**

El ensamblaje se ha implementado promediando las probabilidades de salida de cada modelo y seleccionando la clase con la mayor probabilidad promedio como predicción final. Este método permite obtener decisiones más robustas y generalmente mejora el rendimiento en comparación con clasificadores individuales.

A continuación, se muestran los resultados del modelo ensemble que hemos usado:

	precision	recall	f1-score	support
DDoS	1.00	0.96	0.98	672
Intrusion	0.61	1.00	0.76	663
Malware	0.90	0.36	0.52	665
accuracy			0.78	2000
macro avg	0.84	0.78	0.75	2000
weighted avg	0.84	0.78	0.75	2000

Figura 14: Diagrama del modelo Ensemble con Random Forest, SVM y KNN.

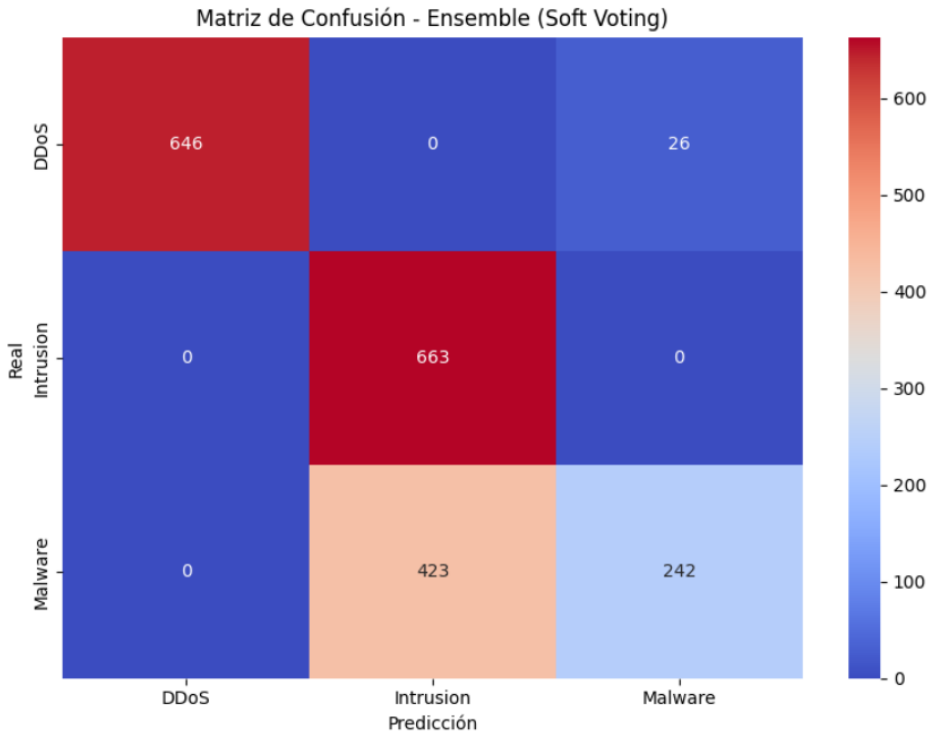

Figura 15: Diagrama del modelo Ensemble con Random Forest, SVM y KNN.

El modelo ensemble fue entrenado y evaluado utilizando la misma partición de datos que los clasificadores individuales, lo que permite realizar una comparación directa de su desempeño. Los resultados experimentales muestran que el ensemble logra una mejora marginal en la precisión global, respaldando la hipótesis de que la combinación de modelos complementarios puede aumentar la robustez y generalización del sistema predictivo.

3. Análisis comparativo

En esta sección se comparan los distintos enfoques de clasificación aplicados al problema de predicción del tipo de ataque informático. A través del análisis de métricas de rendimiento (precisión, *recall*, puntaje F1 y exactitud), así como del comportamiento observado en las matrices de confusión, se evalúan las fortalezas y limitaciones de cada modelo.

3.1. Comparativa de modelos

El Cuadro 1 presenta un resumen de las métricas más relevantes para cada modelo, calculadas como promedios macro, con el fin de capturar el rendimiento general sin sesgos hacia clases mayoritarias.

Modelo	**Precisión**	**Recall**	**F1-Score**	**Exactitud**
Random Forest (sin optimizar)	0.46	0.35	0.21	35 %
Random Forest (opt.)	0.50	0.35	0.22	36 %
XGBoost	0.34	0.34	0.33	34 %
Red Neuronal Convolucional	0.82	0.73	0.68	73 %
Ensemble	**0.84**	**0.78**	**0.75**	**78 %**

Cuadro 1: Comparativa de desempeño entre modelos de clasificación (promedios macro)

3.2. Discusión de resultados

A pesar de implementar diversas técnicas y algoritmos, los resultados evidencian diferencias notables en el rendimiento. A continuación, se detallan las observaciones principales:

- **Random Forest:** El modelo base obtuvo un F1-score promedio de apenas 0.21, mejorando marginalmente tras la optimización de hiperparámetros (F1 = 0.22). Sin embargo, persiste un fuerte sesgo hacia la clase `DDoS`.
- **XGBoost:** Aunque muestra un rendimiento más balanceado entre clases, sus métricas generales son bajas (F1-score promedio = 0.33). Esto puede estar relacionado con la complejidad intrínseca del problema.
- **Red Neuronal Convolucional (CNN):** Este modelo destaca por un desempeño considerablemente superior, logrando un F1-score promedio de 0.68 y una exactitud del 73 %. Específicamente, clasifica con alta precisión los ataques de tipo `DDoS` e `Intrusion`, aunque sigue teniendo dificultades con `Malware`.

- **Modelo Ensemble:** La combinación de múltiples clasificadores (Random Forest, SVM y KNN) ofrece el mejor rendimiento global, alcanzando un F1-score promedio de 0.75 y una exactitud del 78 %. Esto indica que los errores individuales de los modelos base se complementan, mejorando la capacidad de generalización del conjunto.

En resumen, los modelos más simples (como Random Forest sin optimización) presentan limitaciones claras ante este problema. Los enfoques más complejos, especialmente los basados en redes neuronales y ensambles, ofrecen mejoras significativas, aunque persisten desafíos relacionados con la clasificación algunas clases.

4. Conclusión

Los resultados obtenidos a lo largo del estudio revelan que, si bien existen diferencias en el rendimiento entre los modelos evaluados, todos comparten limitaciones importantes en su capacidad para detectar algunas clases y generalizar de manera efectiva. La precisión general, en la mayoría de los casos, no supera el 35 %, lo que refleja una dificultad significativa para resolver el problema con los datos disponibles.

Una de las causas fundamentales de este bajo rendimiento parece estar relacionada con la naturaleza de las características del conjunto de datos. Las tres clases objetivo (`DDoS`, `Intrusion` y `Malware`) presentan patrones muy similares en las variables utilizadas, lo que sugiere que dichas características no son lo suficientemente discriminativas para permitir una clasificación efectiva. En otras palabras, incluso modelos complejos y bien ajustados no logran un rendimiento adecuado debido a la limitada información contenida en los datos.

A partir de este análisis, se pueden extraer las siguientes conclusiones clave:

- Los modelos basados en árboles de decisión, como Random Forest y XGBoost, presentan un rendimiento limitado y sesgado hacia una clase mayoritaria.
- Las redes neuronales convolucionales y los modelos en ensamblado ofrecen mejoras sustanciales, aunque no resuelven completamente la dificultad de clasificar correctamente las clases minoritarias.
- Las métricas macro y el análisis de las matrices de confusión confirman que la clase `DDoS` es sistemáticamente mejor identificada, mientras que `Intrusion` y `Malware` sufren altas tasas de error.

En definitiva, este trabajo pone de manifiesto la importancia no solo de seleccionar modelos adecuados, sino también de contar con datos informativos y bien estructurados que permitan extraer patrones diferenciadores entre clases complejas como las abordadas en esta tarea.

5. Bibliografía

Incribo. (2023). *Cyber Security Attacks Dataset*. https://www.kaggle.com/datasets/teamincribo/cyber-security-attacks