Aprendizaje Automático para la Detección de Bots en Repositorios Digitales

Rafael Bértoli[2],
César Estrebou[1] [0000-0001-5926-8827], and
Ariel Jorge Lira[2]

[1] III-LIDI, Facultad de Informática, UNLP – Centro CICPBA
[2] SEDICI – UNLP
bertoli.rafa@sedici.unlp.edu.ar
cesarest@lidi.info.unlp.edu.ar

Abstract. La detección de bots es un desafío crítico para los repositorios digitales académicos como SEDICI, con implicaciones para la seguridad cibernética, el análisis de tráfico y las estadísticas de acceso y uso.
Este estudio aborda la escasez de datos públicos y la necesidad de métodos eficaces para distinguir entre accesos humanos y automatizados en entornos web. Presentamos un nuevo dataset de logs web derivado de SEDICI y evaluamos diversos algoritmos de aprendizaje automático para la clasificación de accesos.
Nuestro análisis comparativo abarca desde métodos clásicos como Regresión Logística hasta técnicas avanzadas de ensemble como XGBoost y Random Forest. Los resultados muestran un rendimiento sobresaliente de los modelos basados en árboles con una efectividad superior al 97%.
Además, discutimos las implicaciones prácticas de implementar estos modelos en SEDICI para mejorar la precisión de las estadísticas de acceso y proporcionamos una base para futuras investigaciones en la detección de bots en repositorios digitales.

1 Introducción

El panorama digital actual enfrenta un desafío creciente: la proliferación de tráfico automatizado. Según informes recientes de Imperva, el 47,4% del tráfico de internet en 2022 fue generado por bots [11], un aumento significativo desde el 37,2% registrado en 2019 [10]. Este fenómeno plantea interrogantes sobre la integridad y eficiencia de los sistemas en línea.

Los bots, tanto benignos como maliciosos [12] tienen diferentes implicaciones para los sistemas. Esta distinción es crucial, especialmente para repositorios digitales académicos como SEDICI[1] de la UNLP, donde las estadísticas de uso son fundamentales para evaluar el impacto del acceso a sus recursos.

Aunque existen métodos tradicionales para identificar bots, como las listas de IPs y agentes de usuario, estos resultan insuficientes ante la evolución constante de las técnicas automatizadas. DSpace[2], el software utilizado por SEDICI,

[1] http://sedici.unlp.edu.ar/
[2] https://wiki.lyrasis.org/display/DSPACE/

emplea Apache SOLR[3] para registrar eventos, pero su enfoque basado en listas estáticas presenta limitaciones significativas.

La necesidad de una solución más robusta y adaptativa es evidente, considerando las implicaciones económicas y operativas de mantener y depurar grandes volúmenes de datos potencialmente contaminados [3]. El desafío actual radica en desarrollar mecanismos capaces de distinguir eficazmente entre el tráfico humano y el automatizado en tiempo real, antes de almacenar los datos.

En este contexto, se hace imperativo explorar nuevas estrategias que vayan más allá de las soluciones estáticas, adaptándose al cambiante ecosistema digital para garantizar la integridad y utilidad de los repositorios académicos en la era de la automatización [6].

Ante estos desafíos, el presente artículo propone un sistema de detección de bots para repositorios digitales basado en técnicas de aprendizaje automático. Adicionalmente, se fundamenta en la creación de un conjunto de datos derivado de los accesos al repositorio SEDICI, abordando así la necesidad de adaptabilidad frente al cambiante panorama del tráfico web. Los resultados preliminares son prometedores, demostrando la eficacia de varios clasificadores en la identificación precisa de tráfico automatizado. Este enfoque no solo promete mejorar significativamente la calidad de las estadísticas de uso en repositorios académicos, sino que también ofrece una vía para optimizar los recursos del sistema.

2 Estado del Arte en la Detección de Bots

2.1 Enfoques para Detección de Bots

La literatura especializada identifica cuatro enfoques principales para la detección de bots, cada uno basado en diferentes aspectos tecnológicos y metodológicos:

- Análisis de tráfico de bajo nivel: Este método examina los patrones y características del flujo de datos en la red. Incluye el estudio de parámetros como tiempos de respuesta, frecuencia de solicitudes, volumen de datos transferidos y patrones de conexión. Este enfoque puede revelar comportamientos atípicos que son característicos de la actividad automatizada [1, 9].
- Análisis de registros web: Se centra en el estudio detallado de las actividades y comportamientos registrados en el sistema. Este método implica el examen de logs de servidor, patrones de navegación, secuencias de clics y tiempos de sesión. Mediante técnicas de minería de datos y análisis estadístico, se pueden identificar patrones que diferencian el comportamiento humano del automatizado [7].
- Implementación de honeypots: Utiliza elementos señuelo cuidadosamente diseñados para atraer y detectar visitantes no autorizados o maliciosos. Estos honeypots pueden ser páginas web ocultas, enlaces invisibles o recursos falsos que, al ser accedidos, indican claramente la presencia de un bot. La sofisticación de los honeypots puede variar desde simples trampas hasta sistemas complejos que emulan entornos completos [2].

[3] https://solr.apache.org/

- Análisis de contenido generado por usuarios: Relevante en plataformas de redes sociales, donde los bots (denominados "bots sociales") son identificados por las características del contenido que publican [4, 16]. Este enfoque examina aspectos como la frecuencia de publicación, el uso del lenguaje, la variabilidad del contenido y los patrones de interacción. Se emplean técnicas de procesamiento de lenguaje natural y aprendizaje automático para distinguir entre contenido generado por humanos y por máquinas.

2.2 Disponibilidad Conjuntos de Datos

En el ámbito de la detección de bots, la escasez de conjuntos de datos públicos representa un obstáculo significativo para el desarrollo y evaluación de nuevos métodos. Existen conjuntos de datos disponibles, como Avazu, TalkingData y BuzzCity pero, no están específicamente diseñados para la identificación de bots. Esta limitación implica que las investigaciones que utilizan estos conjuntos de datos con sus etiquetas originales no abordan directamente el problema de la detección de bots.

Ante esta carencia, muchos investigadores optan por la creación de datos simulados para el entrenamiento de sus modelos. El proceso de etiquetado, fundamental en el aprendizaje supervisado, se vuelve particularmente desafiante cuando se trata de grandes volúmenes de datos heterogéneos. Para obtener etiquetas confiables de humanos/bots se suelen emplear diversas estrategias que incluyen: Generadores de tráfico programados, Programación manual de bots, Utilización de cuentas o direcciones IP de bots conocidos, Verificación de usuarios mediante CAPTCHA y Etiquetado basado en resultados de análisis previos.

Es importante señalar que el campo de la detección de bots carece de conjuntos de datos de referencia estandarizados. En la práctica, el etiquetado de bots es un proceso imperfecto. Mientras que es posible identificar actores con patrones de comportamiento claramente no humanos, existen bots altamente sofisticados que pueden pasar desapercibidos. Al mismo tiempo, usuarios humanos con patrones de navegación atípicos pueden ser erróneamente clasificados como bots.

Esta situación plantea un desafío significativo para la comparación objetiva de los avances en la identificación de bots. La naturaleza privada de muchos conjuntos de datos y el uso predominante de bots simulados limitan la validez externa de los resultados reportados. Las tasas de reconocimiento publicadas en diversos estudios, aunque indicativas, deben interpretarse con cautela en un ecosistema digital en constante evolución.

Por todo lo anterior, resulta de relevancia la creación de conjuntos de datos públicos más representativos y estandarizados. Esto no solo facilitaría la comparación de diferentes métodos, sino que también aceleraría el progreso en el campo de la detección de bots, permitiendo el desarrollo de soluciones más efectivas y generalizables.

3 Desarrollo del Conjunto de Datos Público

Para la creación de este conjunto de datos, se decidió utilizar todos los registros de SEDICI correspondientes a un día seleccionado de forma aleatoria, con un flujo de acceso regular por parte de los visitantes. En SEDICI, los logs de todos los accesos atendidos por el servidor APACHE se mantienen separados. Estos registros incluyen la IP de origen de la solicitud, la fecha y hora, el método HTTP utilizado, el código de respuesta, el tamaño de la respuesta, el agente de usuario, y el referer.

Una vez seleccionado el día en particular, se analizó el día completo, utilizando todos los datos mencionados anteriormente. Los registros se agruparon por IP, obteniendo así la cantidad de accesos por IP, la cantidad de accesos a cada tipo de recurso y la frecuencia de cada tipo de respuesta HTTP.

La Tabla 1 muestra los atributos del conjunto de datos, su tipo de dato y una breve descripción de la información que contienen. El conjunto de datos inicial se compone de 9 atributos con un total de 457.245 entradas.

Atributo	Tipo	Descripción
dateRaw	Texto	Fecha y hora exacta en la que se recibió la solicitud en el servidor
ip	Texto	IP de origen de la solicitud
metodo	Texto	Método HTTP utilizado
recurso	Texto	Página o recurso al que se intenta acceder
codRespuesta	Entero	Código de respuesta HTTP
UserAgent	Texto	Agente de usuario utilizado en la solicitud
bytes	Entero	Tamaño de la respuesta en bytes
referer	Texto	Página desde la que se originó la solicitud
isBot	Entero	Indicador de si la solicitud fue realizada por un bot o un humano

Table 1: Definición de atributos seleccionados del log de accesos del SEDICI.

Una vez recopilados los datos del día completo, se compararon las IPs y los agentes de usuario con las listas estáticas de SEDICI para realizar un marcado preliminar del conjunto de datos. Posteriormente, se identificaron y marcaron automáticamente aquellos registros donde el agente de usuario contenía las palabras "bot" o "spider", generando así una nueva columna. En este punto, se tenía la certeza de que todos los registros marcados como bots eran efectivamente bots, ya que se utilizaron tanto listas de bots conocidos como aquellos que se autodenominan bots. Sin embargo, este marcado inicial resultó insuficiente para identificar a todos los bots, especialmente aquellos cuya identidad no era claramente conocida.

Un procedimiento adicional consistió en identificar todos los registros que accedieron al "robots.txt" o al sitemap, sitios que son muy poco visitados por usuarios humanos. Tras este análisis, se procedió a marcar manualmente otros accesos como bots. Para ello, las IPs se ordenaron por cantidad de accesos, y cada caso fue evaluado individualmente. Se marcó como bot únicamente cuando

había certeza, ya fuera por la procedencia de las solicitudes o por los patrones de acceso observados.

Atributo	Tipo	Descripción
ip	Texto	IP de origen de la solicitud
cant	Entero	Cantidad total de solicitudes realizadas en el día
primerAcceso	Texto	Hora del primer acceso del día
ultimoAcceso	Texto	Hora del ultimo acceso del día
sesionMasLarga	Texto	Duracion de la sesion mas larga
primerAccesosSesion	Texto	Hora del primer acceso de la session mas larga
ultimoAcessoSesion	Texto	Hora del ultimo acceso de la session mas larga
20X	Entero	Cantidad de respuestas dentro del grupo 20X
30X	Entero	Cantidad de respuestas dentro del grupo 30X
40X	Entero	Cantidad de respuestas dentro del grupo 40X
50X	Entero	Cantidad de respuestas dentro del grupo 50X
cantRaro	Entero	Cantidad de accesos a recursos inexistentes o intentos de hacking
cantBusqueda	Entero	Cantidad de búsquedas realizadas
cantVista	Entero	Cantidad de vistas de items
cantDescarga	Entero	Cantidad de descargas realizadas
cantOtro	Entero	Cantidad de accesos a paginas no relevantes
cantStatic	Entero	Cantidad de accesos a recursos estáticos
cantFeed	Entero	Cantidad de accesos a feeds RSS
cantRobot	Entero	Cantidad de accesos a "robots.txt" y sitemap
cantGet	Entero	Cantidad de solicitudes por el metodo HTTP GET
cantPost	Entero	Cantidad de solicitudes por el metodo HTTP POST
cantOther	Entero	Cantidad de solicitudes por un metodo HTTP que no sea ni GET ni POST
UA	Texto	Agente de usuario más utilizado por la IP en todas las solicitudes
cantBytes	Entero	Cantidad total de bytes transferidos
cantReferer	Entero	Cantidad de solicitudes con referer
isBot	Booleano	Indicador de si el comportamiento corresponde a un bot o a un humano

Table 2: Definición de atributos procesados y agrupados por IP del log crudo de accesos a páginas del SEDICI en un intervalo de 24 horas.

La Tabla 2 presenta los atributos del conjunto de datos final, el tipo correspondiente y una breve descripción de la información que almacenan. Este conjunto de datos se compone de 26 atributos y un total de 28.842 ejemplos, de los cuales 22.000 (76,28%) fueron clasificados como accesos de humanos y 6.842 (23.72%) como accesos de bots.

Tanto la versión cruda del conjunto de datos descripta en la tabla 1 como la versión generada con información estadística descrita en la tabla 2 se encuentran disponibles en `https://github.com/GarrafaGalactica/BotDetectionTesis`.

4 Metodología

En esta sección, se presenta el enfoque metodológico utilizado para la detección de bots a partir de los logs proporcionados por el repositorio SEDICI. El proceso incluye desde el preprocesamiento de los datos, la generación de los modelos y evaluación de algoritmos de clasificación a través de diversas métricas.

4.1 Preprocesamiento de datos

El conjunto de datos procesado contiene atributos estadísticos del comportamiento de los accesos por IP, como se detalla en la Tabla 2. Se decidió eliminar los atributos categóricos en vez de numerizarlos. Los atributos numéricos, fueron normalizados utilizando la media y la desviación estándar de los valores.

4.2 Algoritmos para Clasificación

La selección de algoritmos para este estudio se fundamentó en una revisión de la literatura existente sobre detección de bots y clasificación de tráfico web. A continuación, se presentan brevemente los algoritmos de aprendizaje automático y redes neuronales empleados, acompañados de referencias a estudios previos en los que han demostrado su eficacia:

Gaussian Naive Bayes: Este clasificador probabilístico se basa en el teorema de Bayes. Se eligió por su simplicidad y eficacia en problemas con alta dimensionalidad. Stevanovic y Pedersen [14, 15] demostraron su efectividad en la detección de botnets.

Regresión Logística (LR): Un modelo lineal que estima la probabilidad de pertenencia a una clase. Se incluyó por su interpretabilidad y eficacia en problemas de clasificación binaria, como lo demuestra su uso en [8, 15].

Multiperceptrón (MLP): Modelo inspirado en redes neuronales biológicas, capaz de aprender patrones complejos. Se incluyó por su capacidad para capturar patrones no lineales y su efectividad demostrada en los estudios [8, 14, 15].

Máquinas de Vectores de Soporte (SVM, lineal y RBF): Este tipo de modelo, tanto en su versión lineal como con kernel RBF (Radial Basis Function), ha demostrado su eficacia en la detección de bots al alcanzar tasas de detección entre el 95% y el 97%, según estudios previos [5, 8, 13–15].

Análisis Discriminante Lineal (LDA): Este método busca una combinación lineal de características que separe mejor las clases. Es eficaz en problemas de clasificación binaria y su capacidad para manejar clases desbalanceadas.

Random Forest (RForest): Un conjunto de árboles de decisión que combina sus predicciones. Se eligió por su robustez y capacidad para manejar características irrelevantes, además de su excelente rendimiento en estudios previos [8, 15, 14].

Adaptive Boosting (AdaBoost): Este algoritmo combina clasificadores débiles para formar un clasificador fuerte, mejorando así la precisión del modelo. Se incluyó en este estudio debido a su efectividad en la detección de bots, demostrada en estudios previos con tasas de precisión del 95% [13].

XG Boost (XGB): Un algoritmo de Gradient Boosting conocido por su alto rendimiento. Se incluyó por su eficacia en una amplia gama de problemas de clasificación y los efectivos resultados reportados por [7].

K-Vecinos Cercanos (KNN): Un algoritmo no paramétrico que clasifica basándose en la proximidad a otros puntos de datos. Su inclusión se justifica por su uso efectivo con tasas de acierto del 93% en la detección de ataques web [5, 13, 14].

La selección de estos algoritmos se alinea con las tendencias observadas en la literatura sobre detección de bots y clasificación de tráfico web. Estudios previos han demostrado que enfoques basados en aprendizaje automático, pueden lograr tasas de efectividad que van del 95% al 98% en la detección de bots.

4.3 Configuración experimental

Los datos, previamente preprocesados y normalizados como se detalla en la sección anterior, se dividieron en conjuntos de entrenamiento (70%) y prueba (30%), asegurando la proporción original de clases en ambos conjuntos.

Para la optimización de los hiperparámetros de los algoritmos descritos en la sección 4.2, se realizó una búsqueda exhaustiva en el espacio de hiperparámetros. Esta búsqueda se llevó a cabo mediante validación cruzada de 5 pliegues, utilizando el área bajo la curva ROC (AUC-ROC) como métrica principal de evaluación en los datos de entrenamiento. Este enfoque permitió identificar la configuración óptima para cada modelo.

Se llevaron a cabo 20 experimentos independientes con divisiones aleatorias de los datos en conjuntos de entrenamiento y prueba. Los modelos, generados con los parámetros óptimos, fueron evaluados utilizando diversas métricas, y el rendimiento final se obtuvo promediando los valores para cada una de estas métricas.

La implementación de los algoritmos se realizó principalmente utilizando la biblioteca scikit-learn en Python, incluyendo su implementación del Perceptrón Multicapa (MLP). Para el algoritmo XGBoost, se empleó la biblioteca xgboost.

4.4 Métricas de Evaluación

La evaluación del rendimiento de los clasificadores se realizó mediante diversas métricas, cada una aportando información sobre diferentes aspectos del desempeño de los modelos. Esto es fundamental para obtener una comprensión integral de la eficacia de los clasificadores, especialmente en contextos donde puede existir un cierto grado de desbalance en las clases.

Área bajo la curva Precision-Recall: La métrica principal utilizada en este estudio es el área bajo la curva Precision-Recall (PR-AUC). Es especialmente útil en escenarios con clases desbalanceadas, ya que es más sensible a los cambios en la clase minoritaria. Esta curva muestra la relación entre la *precision* y el *recall* a diferentes umbrales de clasificación.

Área bajo la curva ROC: Complementando la PR-AUC, se incluyó el área bajo la curva ROC (ROC-AUC). Esta medida tiene la capacidad para evaluar el rendimiento del modelo independientemente del umbral de clasificación elegido.

Accuracy (Exactitud): Proporción de todas las predicciones correctas sobre el total de instancias evaluadas.

Precision (Precisión): Proporción de predicciones positivas correctas, importante para minimizar falsos positivos.

Recall (Sensibilidad): Proporción de instancias positivas reales identificadas correctamente, crucial para asegurar que los bots no sean omitidos.

F1-Score: Media armónica de precisión y recall, proporcionando un equilibrio entre Precision y Recall.

Estas métricas se eligieron para proporcionar una evaluación integral del rendimiento de los clasificadores, teniendo en cuenta tanto la precisión como la sensibilidad en la detección de bots.

5 Resultados y Discusión

La tabla 3 y la figura 1 incluyen las métricas de Accuracy, Precision (bot y no bot), Recall (bot y no bot), F1-Score (bot y no bot), ROC AUC y PR AUC para los modelos evaluados con los datos de prueba. Al examinar los resultados, se observa una notable consistencia en el rendimiento relativo de los algoritmos a través de las diferentes métricas evaluadas. Los modelos que destacan en una métrica tienden a sobresalir en las demás. Esta estabilidad en las métricas es particularmente evidente en los modelos de mejor desempeño, como Random Forest, XGBoost y Gradient Boosting, que consistentemente ocupan las posiciones superiores en todas las medidas de evaluación. Por otro lado, si bien existe un desbalance moderado en los datos, este no parece influir negativamente en el rendimiento de los modelos de mejor desempeño, como lo demuestran los altos valores de Recall y F1-Score.

Los resultados obtenidos muestran que, en general, los modelos ofrecen un rendimiento notable, con métricas de efectividad (accuracy) superiores al 95% en la mayoría de los casos. Este nivel de desempeño sugiere que las características extraídas de los logs web son informativas y adecuadas para la tarea de clasificación. Sin embargo, al analizar las métricas por clase, se revelan matices importantes que deben considerarse en el contexto del desbalance de clases.

Model	Accuracy	Precision		Recall		F1-Score		AUC	
		No Bot	Bot	No Bot	Bot	No Bot	Bot	ROC	PR
Random Forest	0.9712	0.9696	0.9780	0.9947	0.8836	0.9820	0.9284	0.9854	0.9690
XGBoost	0.9709	0.9691	0.9783	0.9947	0.8818	0.9818	0.9275	0.9854	0.9689
Gradient Boosting	0.9703	0.9694	0.9740	0.9937	0.8830	0.9814	0.9263	0.9851	0.9671
AdaBoost	0.9622	0.9649	0.9509	0.9880	0.8661	0.9763	0.9065	0.9778	0.9557
Multiperceptron	0.9687	0.9679	0.9722	0.9933	0.8773	0.9804	0.9223	0.9771	0.9536
K-NN	0.9577	0.9692	0.9129	0.9774	0.8843	0.9733	0.8984	0.9701	0.9433
Regresión Logística	0.9623	0.9578	0.9831	0.9961	0.8362	0.9766	0.9037	0.9660	0.9330
LDA	0.8323	0.8294	0.8817	0.9942	0.2294	0.9039	0.3390	0.9572	0.8653
SVM (RBF)	0.8823	0.9030	0.7964	0.9616	0.5867	0.9293	0.6199	0.9494	0.8505
SVM (Linear)	0.8461	0.8560	0.7840	0.9749	0.3654	0.9101	0.4494	0.9396	0.8005
SVM (Poly)	0.7709	0.8043	0.8133	0.9477	0.1108	0.8440	0.1312	0.8012	0.5160
Gaussian N Bayes	0.7869	0.7935	0.4603	0.9864	0.0430	0.8795	0.0786	0.7532	0.3514

Table 3: Comparación de rendimiento entre modelos de clasificación para detección de bots en logs web.

Los modelos basados en árboles y ensemble (XGBoost, Random Forest y Gradient Boosting) mantienen su posición como los de mejor rendimiento global. Random Forest lidera ligeramente con una accuracy de 97.12% y un balance destacable entre precisión y recall para ambas clases. Estos modelos muestran una capacidad superior para manejar el desbalance de clases, con valores de F1-score para la clase minoritaria (bots) superiores a 0.92.

El modelo Multiperceptron (MLP) y AdaBoost siguen mostrando un rendimiento competitivo, aunque se observa una ligera disminución en el recall de la clase minoritaria en comparación con los modelos de ensemble.

K-Nearest Neighbors (KNN) y Regresión Logística muestran un rendimiento sólido, pero se evidencia una mayor dificultad para manejar el desbalance de clases. KNN muestra un mejor equilibrio entre precisión y recall para la clase minoritaria, mientras que la Regresión Logística tiene una alta precisión pero un recall más bajo para los bots.

Las Máquinas de Vectores de Soporte (SVM), independientemente del kernel utilizado, junto con el Análisis Discriminante Lineal (LDA) y Gaussian Naive Bayes (GNB), muestran un rendimiento considerablemente inferior, especialmente en la detección de la clase minoritaria. Esto se refleja en valores muy bajos de recall para los bots, lo que indica una fuerte tendencia a clasificar erróneamente los bots como no bots.

Si bien el tiempo de entrenamiento y de inferencia no se consideran críticos para la aplicación de detección de bots en el SEDICI, se pueden realizar algunas consideraciones sobre estos tiempos, ya que estos impactan en la aplicación final.

Los modelos de ensemble (XGBoost, Gradient Boosting, Random Forest) suelen requerir tiempos de entrenamiento más largos debido a su complejidad, pero ofrecen inferencias rápidas una vez entrenados.

KNN tiene un tiempo de entrenamiento prácticamente nulo, pero puede ser lento en la inferencia, especialmente con grandes conjuntos de datos.

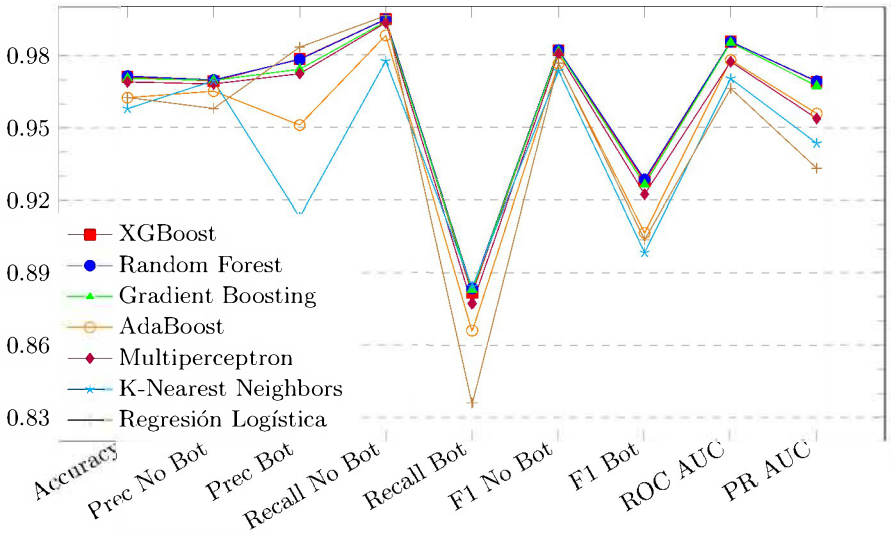

Fig. 1: Rendimiento de mejores modelos con diferentes métricas.

La Regresión Logística es rápida tanto en entrenamiento como en inferencia. Los modelos SVM pueden tener tiempos de entrenamiento largos, especialmente con kernels no lineales y grandes conjuntos de datos.

El Multiperceptron puede requerir tiempos de entrenamiento moderados a largos, dependiendo de su arquitectura, pero suele ser rápido en inferencia.

Los modelos basados en árboles y ensemble, se perfilan como las opciones más efectivas para la tarea de detección de bots. Sin embargo, la elección final del modelo debe considerar no solo el rendimiento, sino también factores como la interpretabilidad, los tiempos de entrenamiento e inferencia, y los recursos computacionales disponibles. Random Forest emerge como una opción particularmente atractiva, ofreciendo el mejor F1-Score y PR-AUC para bots junto con una alta interpretabilidad.

6 Conclusiones y Trabajo Futuro

Este trabajo contribuye al campo de la detección de bots mediante logs web. Se destaca la creación de un dataset preliminar de logs web compartido públicamente, abordando la escasez de datos en este ámbito. Esta contribución facilitará futuras investigaciones y permitirá la validación y expansión de los hallazgos presentados.

En el ámbito de la clasificación de accesos, se implementaron y evaluaron diversos algoritmos basados en técnicas actuales de detección de bots. Los resultados son prometedores, con varios modelos logrando una accuracy superior al 97%. Los algoritmos basados en árboles y ensemble, como XGBoost, Gradient Boosting y Random Forest, demostraron ser particularmente efectivos para esta

tarea. Estos hallazgos contribuyen al estado del arte en la detección de bots y ofrecen potencial para aplicaciones prácticas en entornos reales.

La implementación de estos modelos en SEDICI podría mejorar notablemente la precisión de las estadísticas de acceso a publicaciones. Al distinguir con mayor exactitud entre accesos humanos y de bots, se mejoraría la calidad de los datos de uso del repositorio. Esto permitiría una comprensión más precisa del impacto y alcance real de las publicaciones académicas en la plataforma.

Hay varias líneas de investigación futuras que podrían expandir este trabajo. Por un lado, está la recolección de más información, junto con la ampliación del período cubierto por los logs. Por otro lado, está la incorporación de datos de accesos históricos para enriquecer el contexto de cada entrada, lo que permitiría análisis más profundos y precisos.

El refinamiento de técnicas de preprocesamiento ofrece oportunidades interesantes, como experimentar con diferentes ventanas de tiempo con las que se procesan datos crudos y se calculan estadísticas. Además, explorar técnicas de procesamiento en tiempo real (online) podría complementar el enfoque offline actual, permitiendo la detección de bots en escenarios de alta velocidad y volumen de datos.

La investigación futura en modelos de clasificación podría explorar arquitecturas de aprendizaje profundo como Autoencoders, Redes Convolucionales y Redes Recurrentes. Comparar el rendimiento de estos modelos avanzados con algoritmos clásicos podría ofrecer nuevas perspectivas sobre los patrones de acceso de bots y humanos.

También se puede avanzar hacia un enfoque de seguridad agregando la posibilidad de no solo determinar si se trata de un humano o un bot sino también, de determinar en caso de que sea bot, si es un bot "bueno" o un bot que este intentando vulnerar el sitio, o hasta de que manera lo este intentando.

La validación y aplicación de estos modelos en entornos de producción es un objetivo futuro con potencial. Pruebas exhaustivas en condiciones reales validarían su eficacia y revelarían desafíos prácticos. Desarrollar un sistema integrado para detectar bots en SEDICI sería un avance hacia la aplicación práctica de esta investigación.

References

1. Mohammad Alauthman, Nauman Aslam, Mouhammd Al-kasassbeh, Suleman Khan, Ahmad Al-Qerem, and Kim-Kwang Raymond Choo. An efficient reinforcement learning-based botnet detection approach. *Journal of Network and Computer Applications*, 150:102479, 2020.
2. Sawsan Almahmoud, Bassam Hammo, Bashar Al-Shboul, and Nadim Obeid. A hybrid approach for identifying non-human traffic in online digital advertising. *Multimedia Tools and Applications*, 81(2):1685–1718, 2022.
3. Omar A Alshikhi, Bandar M Abdullah, et al. Information quality: definitions, measurement, dimensions, and relationship with decision making. *European Journal of Business and Innovation Research*, 6(5):36–42, 2018.

4. D. A. Belokurov, E. S. Shamakova, and V.S. Kolomoitcev. Using machine learning techniques to identify bot accounts on a social network. In *2021 Wave Electronics and its Application in Information and Telecommunication Systems (WECONF)*, pages 1–5, May 2021.
5. Alberto Cabri, Grażyna Suchacka, Stefano Rovetta, and Francesco Masulli. Online web bot detection using a sequential classification approach. In *2018 IEEE 20th International Conference on High Performance Computing and Communications; IEEE 16th International Conference on Smart City; IEEE 4th International Conference on Data Science and Systems (HPCC/SmartCity/DSS)*, pages 1536–1540, June 2018.
6. Hanlin Chen, Hongmei He, and Andrew Starr. An overview of web robots detection techniques. In *2020 International Conference on Cyber Security and Protection of Digital Services (Cyber Security)*, pages 1–6, 2020.
7. Marek Gajewski, Olgierd Hryniewicz, Agnieszka Jastrzebska, Mariusz Kozakiewicz, Karol Opara, Jan Wojciech Owsiński, Sławomir Zadrożny, and Tomasz Zwierzchowski. Data-driven human and bot recognition from web activity logs based on hybrid learning techniques. *Digital Communications and Networks*, 2023.
8. Shivani Gaonkar, Nandini Fal Dessai, Jenny Costa, Ashlesha Borkar, Shailendra Aswale, and Pratiksha Shetgaonkar. A survey on botnet detection techniques. In *2020 International Conference on Emerging Trends in Information Technology and Engineering (ic-ETITE)*, pages 1–6, Feb 2020.
9. Wan Nur Hidayah Ibrahim, Syahid Anuar, Ali Selamat, Ondrej Krejcar, Rubén González Crespo, Enrique Enrique Herrera-Viedma, and Hamido Fujita. Multilayer framework for botnet detection using machine learning algorithms. *IEEE Access*, 9:48753–48768, 2021.
10. Imperva. Bad bot report 2020: Bad bots strike back. https://www.imperva.com/resources/resource-library/reports/2020-bad-bot-report/, 2020. Último acceso: 11-03-2024.
11. Imperva. 2023 imperva bad bot report: Key learnings. https://www.imperva.com/blog/2023-imperva-bad-bot-report-key-learnings/, 2023. Último acceso: 04-12-2023.
12. Xigao Li, Babak Amin Azad, Amir Rahmati, and Nick Nikiforakis. Good bot, bad bot: Characterizing automated browsing activity. In *2021 IEEE Symposium on Security and Privacy (SP)*, pages 1589–1605, 2021.
13. Saima Saleem, Muhammad Sheeraz, Muhammad Hanif, and Umar Farooq. Web server attack detection using machine learning. In *2020 International Conference on Cyber Warfare and Security (ICCWS)*, pages 1–7, Oct 2020.
14. Khlood Shinan, Khalid Alsubhi, Ahmed Alzahrani, and Muhammad Usman Ashraf. Machine learning-based botnet detection in software-defined network: A systematic review. *Symmetry*, 13(5), 2021.
15. Matija Stevanovic and Jens Myrup Pedersen. An efficient flow-based botnet detection using supervised machine learning. In *2014 International Conference on Computing, Networking and Communications (ICNC)*, pages 797–801, Feb 2014.
16. T. Velayutham and Pradeep Kumar Tiwari. Bot identification: Helping analysts for right data in twitter. In *2017 3rd International Conference on Advances in Computing,Communication Automation (ICACCA) (Fall)*, pages 1–5, Sep. 2017.