UNIVERSIDAD POLITÉCNICA SALESIANA

SEDE QUITO

CARRERA DE COMPUTACIÓN

ANÁLISIS COMPARATIVO DEL RENDIMIENTO DE ALGORITMOS DE CLASIFICACIÓN BINARIA EN UN CONJUNTO DE DATOS DESBALANCEADOS

Trabajo de titulación previo a la obtención del
Título de Ingeniero en Ciencias de la Computación

AUTORES:
DIEGO ISMAEL BAHAMONDE MORALES
WILIAN STEVE TAPIA PISARRO

TUTORA:
PAULINA ADRIANA MORILLO ALCÍVAR

Quito - Ecuador
2022

CERTIFICADO DE RESPONSABILIDAD Y AUTORÍA DEL TRABAJO DE TITULACIÓN

Nosotros, Diego Ismael Bahamonde Morales con documento de identificación N° 1724355340 y Wilian Steve Tapia Pisarro con documento de identificación N° 1726528407; manifestamos que:

Somos los autores y responsables del presente trabajo; y, autorizamos a que sin fines de lucro la Universidad Politécnica Salesiana pueda usar, difundir, reproducir o publicar de manera total o parcial el presente trabajo de titulación.

Quito, 11 de marzo del año 2022

Atentamente,

.....
Diego Ismael Bahamonde Morales
1724355340

.....
Wilian Steve Tapia Pisarro
1726528407

CERTIFICADO DE CESIÓN DE DERECHOS DE AUTOR DEL TRABAJO DE TITULACIÓN A LA UNIVERSIDAD POLITÉCNICA SALESIANA

Nosotros, Diego Ismael Bahamonde Morales con documento de identificación No.1724355340 y Wilian Steve Tapia Pisarro con documento de identificación No. 1726528407, expresamos nuestra voluntad y por medio del presente documento cedemos a la Universidad Politécnica Salesiana la titularidad sobre los derechos patrimoniales en virtud de que somos autores del Articulo Académico: "Análisis comparativo del rendimiento de algoritmos de clasificación binaria en un conjunto de datos desbalanceados", el cual ha sido desarrollado para optar por el título de: Ingeniero en Ciencias de la Computación, en la Universidad Politécnica Salesiana, quedando la Universidad facultada para ejercer plenamente los derechos cedidos anteriormente.

En concordancia con lo manifestado, suscribimos este documento en el momento que hacemos la entrega del trabajo final en formato digital a la Biblioteca de la Universidad Politécnica Salesiana.

Quito, 11 de marzo del año 2022

Atentamente,

.....
Diego Ismael Bahamonde Morales
1724355340

.....
Wilian Steve Tapia Pisarro
1726528407

CERTIFICADO DE DIRECCIÓN DEL TRABAJO DE TITULACIÓN

Yo, Paulina Adriana Morillo Alcívar con documento de identificación N° 1715646574, docente de la Universidad Politécnica Salesiana, declaro que bajo mi tutoría fue desarrollado el trabajo de titulación: ANÁLISIS COMPARATIVO DEL RENDIMIENTO DE ALGORITMOS DE CLASIFICACIÓN BINARIA EN UN CONJUNTO DE DATOS DESBALANCEADOS, realizado por Diego Ismael Bahamonde Morales con documento de identificación N° 1724355340 y por Wilian Steve Tapia Pisarro con documento de identificación N° 1726528407, obteniendo como resultado final el trabajo de titulación bajo la opción Artículo Académico que cumple con todos los requisitos determinados por la Universidad Politécnica Salesiana.

Quito, 11 de marzo del año 2022

Atentamente,

.....
Ing. Paulina Adriana Morillo Alcívar, MsC
1715646574

Análisis comparativo del rendimiento de algoritmos de clasificación binaria en un conjunto de datos desbalanceados

1st Diego Ismael Bahamonde Morales
dbahamondem@est.ups.edu.ec

2st Wilian Steve Tapia Pisarro
wtapiap1@est.ups.edu.ec

3rd Paulina Adriana Morillo Alcivar
pmorillo@ups.edu.ec

***Resumen*—La identificación de la clase de un objeto es una tarea del aprendizaje de máquina supervisado cuyo rendimiento depende, casi exclusivamente, del conjunto de datos usado en el entrenamiento. Por lo tanto, uno de los retos que enfrentan los algoritmos de clasificación, específicamente de clasificación binaria, es aprender a distinguir claramente entre dos clases, cuando se tiene un número mucho mayor de instancias de una clase, que de otra. Para evitar el sesgo en la clasificación se suele recurrir a técnicas de balanceo de datos que buscan equilibrar el *dataset*, incrementando o reduciendo el número de instancias de la clase minoritaria y de la clase mayoritaria, respectivamente. Este trabajo propone un análisis comparativo del rendimiento de cuatro clasificadores como Regresión Logística, Random Forest, Redes Neuronales Artificiales y Nayve Bayes, combinados con cuatro técnicas diferentes de balanceo de datos Near Miss, SMOTE, SMOTEENN y SMOTETomek. Los resultados muestran que Near Miss logra un equilibrio adecuado entre las clases, de modo que, los algoritmos aumentaron su rendimiento general, alcanzando precisiones y exactitudes mayores al 95%. El resto de técnicas, por su parte, no aumentaron la capacidad de los clasificadores para identificar objetos de la clase minoritaria, a excepción de Random forest y Redes neuronales artificiales, que lograron una tasa de verdaderos negativos superior al 70%, manteniendo a su vez, una tasa de verdaderos positivos mayor al 80%. De igual forma, los tiempos de entrenamiento y prueba de los conjuntos de datos balanceados con técnicas de sobremuestreo o híbridos son muy superiores a los tiempos obtenidos por técnicas de submuestreo como Near Miss, ya que esta última reduce el número de instancias a ser procesadas por los modelos.**

***Palabras Clave*—Aprendizaje de Máquina, sobremuestreo, submuestreo, SMOTE, SMOTETomek, SMOTEENN, Near Miss, exactitud, precisión, sensibilidad, especificidad.**

***Abstract*—Identifying the class of an object is a supervised machine learning task whose performance depends, almost exclusively, on the dataset used in training. Therefore, one of the challenges faced by classification algorithms, specifically binary classification, is learning to clearly distinguish between two classes, when you have a much larger number of instances of one class than another. To avoid bias in the classification, data balancing techniques are usually used that seek to balance the dataset, increasing or reducing the number of instances of the minority class and the majority class, respectively. This paper proposes a comparative analysis of the performance of four classifiers such as Logistic Regression, Random Forest, Artificial Neural Networks and Nayve Bayes combined with four different data balancing techniques Near Miss, SMOTE, SMOTEENN and SMOTETomek. The results show that Near Miss achieves a proper balance between the classes, so that the algorithms increased their overall performance, reaching precision and accuracy greater than 95%. The rest of the techniques, on the other hand, did not increase the ability of the classifiers to identify objects of the minority class, with the exception of Random forest and Artificial Neural Networks, which achieved a true negative rate greater than 70%, while maintaining a true positive rate greater than 80%. Similarly, the training and testing times of the balanced data sets with oversampling techniques or hybrids are far superior to the times obtained by undersampling techniques such as Near Miss, since the latter reduces the number of instances to be processed by the models.**

***Keywords*—Machine Learning, oversampling, undersampling, SMOTE, SMOTETomek, SMOTEENN, Near Miss, accuracy, precision, recall, specificity.**

I. Introducción

En la actualidad, el *Machine Learning* (ML) o Aprendizaje de Máquina esta presente en muchas aplicaciones de la vida diaria, desde predicciones del clima, recomendaciones de compra, detectores de spam, etc [1] [2] [3]. Estas aplicaciones, emplean mayormente algoritmos de aprendizaje supervisado como los clasificadores, usados para identificar la clase a la que pertenece un objeto. Estos algoritmos guardan una dependencia muy alta con el conjunto de datos del entrenamiento. Por lo tanto, cuando un conjunto de datos no contiene el mismo número de instancias por cada clase, el rendimiento general del algoritmo disminuye, al igual que su capacidad para identificar las instancias de las clases minoritarias, produciendo un sesgo en la clasificación [4] [2] [5]. Los conjuntos de datos desbalanceados (CDD) suelen estar presentes en diversos problemas como la detección de transacciones financieras fraudulentas, donde de cada 1000 transacciones apenas 5 son fraudulentas. De igual forma, la detección de tráfico de datos inusual en una red, donde el 88.5% del tráfico es normal, mientras que solo el 11.5% es inusual [6], ejemplos similares son muy comunes en la vida real, por lo que, en estos casos, la clasificación no resulta una tarea trivial, ya que, los algoritmos suelen favorecer a la clase mayoritaria cometiendo mayores errores en la predicción de la clase minoritaria.

Para mitigar este desequilibrio en las clases de los *datasets* se han desarrollado varias técnicas desde dos enfoques diferentes, por un lado están las técnicas a nivel de datos y por otro aquellas a nivel de algoritmo [2] [5] [3] [7]. En el primer caso, las técnicas a nivel de datos pretenden modificar el tamaño (aumentar) de las instancias del CDD, esto se realiza mediante técnicas de remuestreo [5] [8] en la fase de preprocesamiento

de datos, es decir, antes de entrenar el modelo de ML [3]. En el segundo caso, las técnicas a nivel de algoritmo, eliminan el desbalance modificando los algoritmos de aprendizaje [5] [9], por ejemplo los algoritmos sensibles al coste, en los que se da una mayor penalización a la clasificación errónea de la clase minoritaria en comparación a la clase mayoritaria [10] [2].

Uno de los algoritmos de remuestreo de clases es el sobremuestreo, que equilibra el CDD duplicando o generando muestras sintéticas en las clases minoritarias [5] [3] [8]. Sin embargo, este procedimiento puede provocar un sobreajuste en el modelo, además, de tener un alto coste computacional [10]. Otro método es el submuestreo, que actúa sobre la clase mayoritaria, eliminando muestras hasta equilibrar el *dataset*. Esto puede significar un riesgo, ya que se pueden eliminar demasiadas instancias, disminuyendo el rendimiento general del algoritmo [2] [5] [11].

En particular, este artículo propone un análisis comparativo de cuatro técnicas de balanceo a nivel de datos (Near Miss, SMOTE, SMOTETomek y SMOTEENN) combinadas con cuatro diferentes algoritmos de clasificación (Regresión Logística, Random Forest, Nayve Bayes y redes neuronales Perceptrón multicapa) y aplicadas sobre un conjunto de datos desbalanceados con dos clases. El objetivo de este trabajo es analizar si la aplicación de estas técnicas al conjunto de datos de entrenamiento, mejora el rendimiento de los algoritmos de clasificación binaria.

La estructura de este artículo se organiza de la siguiente manera. En la sección I-A se exponen los trabajos relacionados. La sección II se describe la metodología empleada en el procesamiento de los datos, la aplicación de las técnicas de balanceo de datos, el entrenamiento y la validación de los modelos de clasificación. La sección III presenta los resultados de los experimentos realizados. Finalmente, en la sección IV se muestran las conclusiones de este trabajo.

A. Trabajos Relacionados

El desbalance de clases en un conjunto de datos es un inconveniente que dificulta la tarea de clasificación. Sin embargo, la literatura ha propuesto distintas estrategias para aliviar el sesgo del conjunto de datos de entrenamiento [4] [12] [13]. Una de ellas, y la más utilizada, es la manipulación de la distribución de clases, que utiliza los métodos a nivel de datos. El Synthetic Minority Over-sampling Technique (SMOTE), propuesto por Chawla et al. [14], en el 2002, es la técnica más representativa del enfoque sobremuestreo . En [15] aplica SMOTE para equilibrar la clase sesgada y entrenar 28 clasificadores. Los resultados generados por la investigación concluyen en un aumento significante del rendimiento de los clasificadores al ser entrenados con *dataset* con y sin balanceo. Además, en [16] se han desarrollado diversas variantes, como: Borderline-SMOTE que identifica muestras ubicadas al límite entre clases para generar muestras sintéticas, generando un aumento del 10% a un máximo de 45.2% en los *datsets* Pima (UCI) y Haberman (UCI) respectivamente.

Contrariamente, reducir la clase con mayor proporción de datos se denomina submuestreo, aunque la eliminación de instancias en la clase mayoritaria puede representar la pérdida de información relevante para el modelo, especialmente si las clases están muy desbalanceadas [17]. Una de esta técnicas de submuestreo es la técnica Random Under Sampling (RUS), que elimina instancias del *dataset* de manera aleatoria [18]. Por ejemplo, en [19] se aplica está técnica sobre un conjunto de datos para la detección de fraudes en tarjetas de crédito y compara los resultados con tres clasificadores; Regresión Logística (RL), Nayve Bayes (NBC) y KNN, obteniendo un rendimiento significativo del algorirmo RL. De la misma manera, en [20] se busca mejorar el rendimiento en la identificación de transacciones fraudulentas realizadas con tarjetas de crédito, para ello combina la técnica Near Miss con la selección de características y se concluye que existe una mejora en la clasificación de ambas clases particularmente empleando Random Forest. De la misma manera en [20] busca mejorar la clasificación de tarjetas de crédito en transacciones fraudulentas, para ello combina la técnica Near Miss con la selección de características, concluyendo que el rendimiento mejoró de 0 a 100, es decir un incremento del 100% para la clase minoritaria. Near Miss es una técnica que tiene 3 versiones : versión 1 enfocada a la distancia más corta, versión 2 selecciona las distancias lejanas, versión 3 considera la distancia de los vecinos más cercanos [20] [21]. En [21] diseña un clasificador para identificar el nivel de gravedad causado por el COVID-19, emplea Near Miss versión 1, la experimentación demuestra mayor rendimiento al aplicar la técnica de submuestreo en comparación a un conjunto de datos desbalanceados. El clasificador con mayor efectividad luego del balance de clases fue Random Forest (RF) con 93.41%.

De acuerdo a la literatura, varios investigadores han utilizado simultáneamente métodos de sobremuestreo y submuestreo, a estos métodos se los conoce como métodos híbridos. Aplicar técnicas híbridas ayudan a reducir la pérdida de información del submuestreo y a reducir el coste computacional del sobremuestreo, además de aprovechar los beneficios de ambos métodos [22]. Varios artículos implementan SMOTE como técnica base de sobremuestreo y la combinan con técnicas de submuestreo. De esta manera en [23] y [18] aplican SMOTEENN, donde ENN es utilizado para eliminar el ruido del *dataset*, las dos investigaciones muestran un mejor desempeño, comparados con solamente el uso de SMOTE. Otra técnica seleccionada por los investigadores es SMOTE y Tomek Link (elimina instancias de la clase mayoritaria) denominada SMOTETomek. En [24], los resultados fueron prometedores, mejorando la medida F1-Score hasta un 59.73%.

Otras investigaciones [25] [21], implementa Redes Adversas Generativas (GAN), como un método de sobremuestreo. El método GAN usa dos redes neuronales una generativa y una discriminadora para crear nuevas muestras. La red neuronal generativa es la encargada de generar muestras a partir de la entrada proporcionada, la segunda red neuronal es un discriminador, su función es determinar cuán ajustados están los datos reales de los sintéticos, el proceso se repite hasta que la red discriminadora acepte el resultado. De esta manera

las investigaciones concluyeron que el método GAN evita el sobreajuste, debido a que crea nuevas muestras a partir del ruido generado por las redes generativas. Sin embargo, existe escasa literatura para el enfoque GAN en la generación de datos tabulares, lo que puede causar dificultades en la implementación [26].

II. MATERIALES Y MÉTODOS

Esta investigación busca aplicar las técnicas de balanceo de clases sobre el conjunto de datos, para analizar el rendimiento de los algoritmos de clasificación. Por lo tanto, previamente se realiza un preprocesamiento de datos para depurar el *dataset*. Luego se realiza la partición de los datos, usando el método de validación cruzada estratificada, para conservar el porcentaje de muestras de cada clase en cada partición, posteriormente se configuran los cuatro clasificadores seleccionados en el análisis que son: RL, RF, Redes Neuronales Artificiales (ANN) y NBC, y de igual forma, las cuatro técnicas de balanceo: Near Miss, SMOTETomek, SMOTEENN y SMOTE. De este modo, se entrena los modelos usando un conjunto de datos con y sin balanceo del clases. Finalmente, se evalúan los clasificadores calculando diferentes métricas de rendimiento.

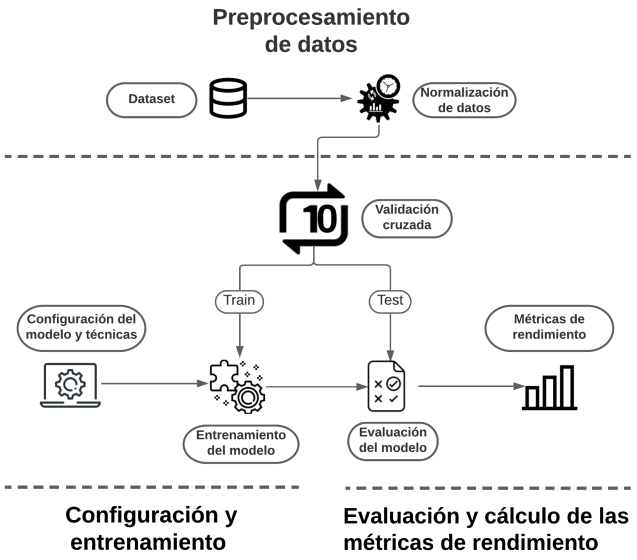

Fig. 1: Proceso experimental para el desarrollo metodológico.

A. Preprocesamiento de datos

La fase de preprocesamiento de datos consiste en la inspección del *dataset*, para determinar si existen valores nulos, duplicados o que se encuentren vacíos. Luego, se normalizan los datos para que los atributos se encuentren en la misma escala. De esta forma, se facilita el aprendizaje de los algoritmos y se minimiza el sesgo provocado por valores extremos. Una vez realizada la normalización de los datos, las instancias con sus respectivos atributos se almacenan en una variable X, mientras que las etiquetas (clases) en una variable Y.

B. Configuración y entrenamiento

En la etapa de configuración y entrenamiento de los modelos se realiza la validación cruzada, un mecanismo, que evita el sesgo en el entrenamiento causado por una partición particular del conjunto de datos, de este modo se realiza el proceso de entrenamiento y evaluación k veces con k particiones de entrenamiento y prueba diferentes. Finalmente, se promedian las métricas de evaluación de cada prueba para obtener el rendimiento final del modelo. En este caso, se utiliza un valor de $k = 10$ y se usa la validación cruzada estratificada para garantizar un número adecuado de instancias de ambas clases, ya que, considerando que es un CDD, si no se implementa esta técnica el conjunto de datos de entrenamiento podría resultar sin ningún ejemplo de la clase minoritaria, provocando un sesgo en la clasificación, beneficiando la clasificación de instancias de la clase mayoritaria en decremento del rendimiento, en la clasificación de instancias de la clase minoritaria.

Los algoritmos que se seleccionaron para la clasificación binaria, por el respaldo que tienen en la literatura [15], fueron RL, RF, NBC y una red neuronal artificial de perceptrón multicapa (ANN). Las configuraciones de estos algoritmos se resumen en la Tabla I. En el caso de RL se utilizó una configuración muy similar para entrenar el modelo con los datos balanceados y con los datos desbalanceados, únicamente en el segundo caso se añadió el parámetro *class_weight* para indicar que los datos no estaban balanceados. Para RF se usó 100 árboles con muestras bootstrap y criterio Gini para la calidad de división. La estructura utilizada para ANN es perceptrón multicapa como clasificador, estableciendo como máximo 2000 iteraciones para considerar el tiempo de ejecución, la activación de la capa oculta es relu y la optimización de peso Adam debido a su funcionamiento con *datasets* grandes [27], [18]. NBC utiliza la configuración predeterminada, donde se establecen sus dos parámetros: priors el cual especifica la probabilidad previa para las clases y var_smothing, es decir, el valor de la varianza.

Para tratar el desbalance de clases binarias existen varias técnicas de las cuales se han seleccionado cuatro algoritmos con los siguientes enfoques: aumento de instancias de la clase minoritaria (SMOTE), disminución de las muestras de la clase mayoritaria (Near Miss) y por último, híbridos (SMOTETomek y SMOTEENN). La Tabla II muestra las configuraciones de estas técnicas.

Para la implementación de la técnica SMOTE es necesario indicar la tasa de aumento de las instancias de la clase minoritaria, que se calculó mediante la formulación descrita en [5] y cuyo valor fue de 0.9858. Las técnicas de muestreo híbrido SMOTETomek y SMOTEENN comparten el concepto de SMOTE para generar muestras basadas en la distancia entre cada dato y los vecinos más cercanos de la clase minoritaria. A su vez, ENN y Tomek Link se encargan de eliminar muestras superpuestas que se encuentran al límite entre las clases. Ambos enfoques se adaptan muy bien debido a que el uno genera muestras sintéticas para la clase minoritaria y el otro elimina muestras de la clase mayoritaria, hasta

Clasificadores									
RL balanceo		**RL sin balanceo**		**ANN**		**RF**		**NBC**	
Parámetro	**Valor**	**Parámetro**	**Valor**	**Parámetro**	**Valor**	**Parámetro**	**Valor**	**Parámetro**	**Valor**
penalty	l2	penalty	l2	hidden_layer_sizes	50,50	n_estimators	100	priors	None
dual	False	dual	False	activation	relu	bootstrap	True	var_smothing	1e-09
tol	0.0001	tol	0.0001	solver	adam	criterion	gini		
class_weight	balanced	solver	lbfgs	alpha	0,0001				
solver	lbfgs	multi_class	ovr						
multi_class	ovr								

Tabla I: Parámetros de entrada de los clasificadores

Técnicas de balanceo de datos							
Near Miss		**SMOTETomek**		**SMOTEENN**		**SMOTE**	
Parámetro	**Valor**	**Parámetro**	**Valor**	**Parámetro**	**Valor**	**Parámetro**	**Valor**
sampling_strategy	0.5	sampling_strategy	auto	sampling_strategy	auto	sampling_strategy	0.9858
version	2	random_state	None	random_state	None	random_state	None
n_neighbors	3	smote	None	smote	None	k_neighbors	5
n_neighbors_ver3	3	tomek	None	enn	None		

Tabla II: Parámetros de entrada para las técnicas de balanceo de datos

equilibrar las clases. Para implementar estos dos algoritmos se utiliza el parámetro del tamaño de la muestra, que se configuró con la opción automática 'auto', para remuestrear y eliminar instancias de forma más equitativa. En el caso de Near Miss, se seleccionó la versión 2, ya que al estar enfocada en las muestras lejanas suelen ser menos afectada por el ruido presente en los datos. La configuración de esta técnica incluye el parámetro de proporción de remuestreo que se establece en 0.5, para mantener las instancias de la clase minoritaria y eliminar las instancias de la clase mayoritaria hasta obtener el doble del tamaño de las muestras de la clase 0. La eliminación y selección de muestras se realiza considerando la distancia promedio de las instancias minoritarias y seleccionando el número de vecinos con parámetro 3-NN [28].

C. Evaluación y cálculo de las métricas de rendimiento

En esta sección, se presenta las métricas utilizadas para medir el rendimiento de los diferentes clasificadores [29] [2] [30], las cuales son: exactitud o *accuracy* (Acc) y *precisión* (Pr). En el caso de exactitud, se mide la capacidad de los modelos para predecir con éxito ambas clases (cero y uno), mientras que la precisión representa el ratio de instancias clase 1 que han podido ser detectadas.

Sin embargo, al tratar con CDD, estas métricas pueden resultar un poco engañosas, ya que en estos casos, suele haber más aciertos de la clase mayoritaria que de la clase minoritaria, lo que puede provocar que los errores cometidos en la predicción de la clase con menos instancias se desprecien.

Por lo tanto, también se usan métricas como la sensibilidad o *recall* (Rc) que mide la tasa de verdaderos positivos (TPR), la Especificidad (Ec) o la tasa de verdaderos negativos (TNR) y el *F1-Score* (F1) que es una métrica más equilibrada entre la precisión y la sensibilidad. Para el cálculo de estas métricas se utiliza una matriz de confusión, que muestra el número de verdaderos negativos o *true negatives* (TN), verdaderos positivos o *true positives* (TP), falsos positivos o *false positives* (FP) y falsos negativos o *false negatives* (FN). Las métricas utilizadas permiten analizar los aciertos y errores de cada clase y el rendimiento de los clasificadores de forma menos sesgada.

III. Resultados y Discusión

Los experimentos y la simulación se llevaron a cabo utilizando google colaboraty, por su prestación de servicio gratuito en la nube, a través de la conexión con una máquina virtual con 12 GB de RAM y 50 GB en disco duro, de los cuales, se utilizó 1.24 GB y 41.86 GB, respectivamente. El lenguaje de programación que se usó fue python V.3. Una de las ventajas de trabajar en la nube es que no se interrumpe el proceso, si el computador no tiene el hardware o software adecuado para la implementación. En caso de requerir mayores prestaciones de disco duro o memoria, se puede ampliar la capacidad de la máquina por un valor adicional, que solo se paga mientras se consumen los recursos. El código del desarrollo de este trabajo se encuentra disponible en el repositorio de github: https://github.com/Wilian21/Balanceo_Datos-Clasificadores_Binarios.git.

El conjunto de datos seleccionado para la parte experimental de este trabajo se denomina Malware Analysis *dataset*: Pe Section Headers, creado por Oliveira, es de dominio público y se encuentra alojado en IEEEDataPort [31]. Este *dataset* fue parte de su investigación sobre detección y clasificación

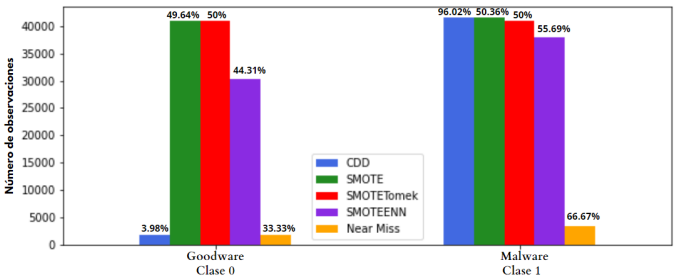

Fig. 2: Distribución de clases con y sin técnicas de balanceo

	Clase predicha																			
	Regresión logística (RL)										Random forest (RF)									
	Sin balanceo		Near Miss		SMOTETomek		SMOTEENN		SMOTE		Sin balanceo		Near Miss		SMOTETomek		SMOTEENN		SMOTE	
	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1
Clase 0	181	164	314	31	4661	3541	338	5707	3786	4408	156	189	335	10	8047	155	6004	41	7991	204
Real 1	2159	6155	0	690	2395	5807	589	7007	1479	6835	81	8233	8	682	332	7870	88	7508	410	7904
Acc	0.7317		0.9700		0.6381		0.5384		0.6434		0.9688		0.9830		0.9702		0.9905		0.9628	
F1	0.8408		0.9780		0.6616		0.6900		0.6989		0.9838		0.9873		0.9699		0.9915		0.9626	
Pr	0.9740		0.9570		0.6211		0.5511		0.6079		0.9775		0.9855		0.9807		0.9946		0.9748	
Rc (TPR)	0.7402		1		0.7080		0.9224		0.8221		0.9902		0.9891		0.9594		0.9884		0.9507	
Ec (TNR)	0.5246		0.9101		0.5682		0.055		0.4620		0.4521		0.9710		0.9811		0.9932		0.9751	
$t_{train}(s)$	2.2585		0.4181		6.2867		5.5805		6.4753		32.9080		2.9628		88.2141		68.0232		89.3971	
$t_{test}(s)$	0.2102		0.0369		0.4210		0.3287		0.3976		3.5659		0.3234		9.6207		7.4831		9.7331	
	Nayve Bayes (NBC)										Perceptrón multicapa (ANN)									
	Sin balanceo		Near Miss		SMOTETomek		SMOTEENN		SMOTE		Sin balanceo		Near Miss		SMOTETomek		SMOTEENN		SMOTE	
	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1
Clase 0	24	321	313	32	7962	240	5816	229	7910	285	2	343	331	14	6765	1437	4612	1433	6832	1363
Real 1	813	7501	10	680	6956	1246	6329	1267	7004	1310	2	8312	6	684	2052	6150	1020	6576	2094	6220
Acc	0.8690		0.9600		0.5613		0.5192		0.5584		0.9602		0.9812		0.7872		0.8201		0.7905	
F1	0.9297		0.9705		0.2571		0.2786		0.2643		0.9797		0.9860		0.7788		0.8426		0.7823	
Pr	0.9589		0.9554		0.8385		0.8469		0.8216		0.9604		0.9801		0.8109		0.8216		0.8211	
Rc (TPR)	0.9022		0.9860		0.1519		0.1668		0.1575		0.9997		0.9920		0.7498		0.8656		0.7480	
Ec (TNR)	0.0695		0.9072		0.9707		0.9621		0.9652		0.0057		0.9594		0.8247		0.7629		0.8336	
$t_{train}(s)$	0.5203		0.1435		1.1031		0.9063		1.0931		187.3296		48.7669		1552.5179		1740.0144		1957.6577	
$t_{test}(s)$	0.0372		0.0115		0.0808		0.0680		0.0810		16.2642		5.7357		147.7276		225.8952		204.5734	

Tabla III: Resultado de los clasificadores con y sin técnicas de balanceo mediante las medidas de rendimiento

de malware usando Deep Learning, donde determinaron el entorno dinámico Cuckoo Sandbox para la detección de código malicioso, los ejecutables de virusshare para descargar las muestras de malware y portableapps para los ejemplos de goodware junto con los directorios de Windows 7 x86. Obteniendo un total de 43.293 instancias, con seis características o atributos de las cuales cuatro son numéricas (tamaños de los datos en el disco, dirección de memoria del primer byte, entropía y el tamaño de la sección cuando carga en memoria), una es categórica o la variable a predecir y el último atributo es de tipo texto (hash) que contiene una cadena encriptada.

La distribución de clases está dividida con el 96.02% de la clase uno o clase malware (clase mayoritaria) versus 3.98% de la clase cero que corresponde a la clase goodware (clase minoritaria). Este *dataset* fue seleccionado, principalmente por que aun no se ha abordado en otros estudios sobre clasificación con *datasets* desbalanceados, por ser un conjunto que sirve para clasificación binaria, dado que tiene únicamente dos clases y además, porque involucra datos de aplicación en el mundo real.

Después de la selección del conjunto de datos, se realizó la normalización de los atributos. En la fase exploratoria del *dataset* no se encontraron datos faltantes, ni atípicos, por lo que no se realizó ninguna corrección adicional. Solo se eliminó la columna hash de tipo texto, puesto que es un atributo que no aporta información, para la predicción de la variable a predecir. Con el conjunto de datos normalizado, se procedió a aplicar las técnicas de balanceo de datos, obteniendo cuatro *datasets* adicionales, cuyos tamaños y número de instancias se visualizan en la Tabla IV. La técnica Near Miss disminuyó las muestras de la clase uno, hasta mantener el doble de la clase cero. SMOTETomek redujo 559 instancias de la clase uno, mientras que aumentó 39.284 en la clase cero, obteniendo un *dataset* relativamente equilibrado. SMOTEENN generó 28.499 muestras para la clase cero, mientras que para la clase uno, eliminó 3.589 muestras de los vecinos más cercanos. Por último, SMOTE generó 39.252 muestras sintéticas para la clase cero. La distribución de las clases de cada conjunto se muestra en la Figura 2.

La siguiente fase consistió en el entrenamiento de los modelos, donde se utilizó la validación cruzada estratificada con el 80% (de cada clase) para el entrenamiento y 20% (restante) para las pruebas. En la Tabla III se puede observar las diferentes métricas de rendimiento de cada clasificador antes y después de aplicar las técnicas de balanceo. Por lo tanto, se comparan los rendimientos de los algoritmos de clasificación luego de aplicar el balanceo de datos, en contraste con el desempeño de los clasificadores entrenados con el *dataset* original (sin balanceo). Así, la exactitud para los datos desbalanceados es mayor al 70% en todos los modelos. Sin embargo, la tasa de verdaderos negativos para NBC y ANN es muy baja, menor al 7%, lo que indica que la mayoría de aciertos corresponden a la clase mayoritaria. En el caso de los modelos RL y RF la TNR es cercana al 50%.

Al entrenar los clasificadores usando el *dataset* balanceado con Near Miss se observa un equilibrio en el número de aciertos de ambas clases. Por el contrario, las técnicas de remuestreo e híbridas (SMOTETomek, SMOTEENN y SMOTE), en general, aumentan el número de TN, pero a su vez, reducen el número de TP. Esto provoca un decremento significativo de los valores de exactitud en los modelos RL, NBC y ANN. Aunque, no sucede lo mismo con RF, donde el número de aciertos de ambas clases es equilibrado y por lo tanto, los valores de Acc se mantuvieron cercanos para todos los conjuntos de datos. Todas las métricas de rendimiento de todos los algoritmos entrenados con el conjunto de datos balanceado con Near Miss fueron superiores al 90%, lo que muestra que al aplicar el balance de datos con esta técnica se mejora el aprendizaje de los modelos y se reduce el sesgo en la clasificación.

Por otro lado, al analizar los resultados de los clasificadores entrenados con el *dataset* balanceado con la técnica SMOTE-

	Técnicas de balanceo							
	Near Miss		SMOTETomek		SMOTEENN		SMOTE	
	Antes	Después	Antes	Después	Antes	Después	Antes	Después
0	1725	1725	1725	41009	1725	30224	1725	40977
1	41568	3450	41568	41009	41568	37979	41568	41568

Tabla IV: Instancias antes y después del balanceo de datos

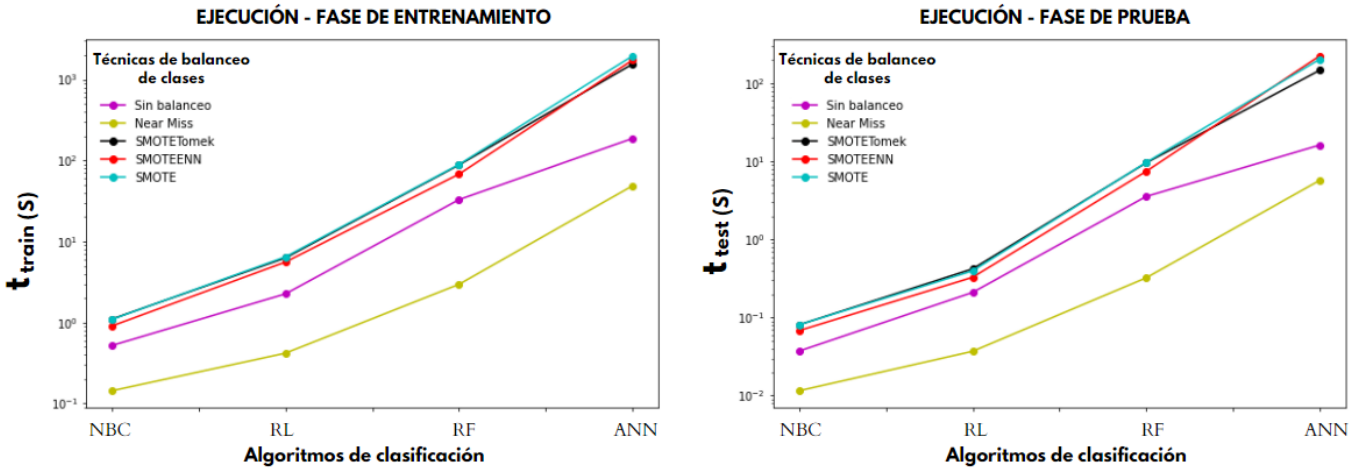

Fig. 3: Tiempos de ejecución train y test (escala logarítmica)

Tomek, se observa que para el modelo RL, la precisión y por ende el F1-score se reducen entre un 20 y 30% en comparación con las métricas obtenidas con el *dataset* original. En cambio el recall y la especificidad se mantuvieron sin mayores variaciones, lo que implica que el balance de los datos no contribuyó mayormente al aprendizaje. En el caso de NBC y ANN, la reducción de la precisión es menor al 15% y para RF es prácticamente imperceptible (menor al 2%). Respecto al recall, en el modelo NBC, este disminuye significativamente, aunque aumenta casi en la misma proporción la especificidad. Para ANN, en cambio, el Rc y la Ec están relativamente balanceados y son superiores al 70%. Por último, para Random forest, el Rc no presentó cambios significativos, pero la Ec mostró un incremento de más del 50%. Estos resultados evidencian, que ANN si aumentó su capacidad para detectar la clase cero, sin perder su capacidad de detección de la clase 1, De igual manera, RF obtiene los valores más altos de rendimiento en la detección de ambas clases, pero no sucede lo mismo con NBC que perdió su capacidad para identificar instancias de la clase mayoritaria.

Continuando el análisis, la técnica de balanceo SMOTEENN combinada con RL no obtuvo resultados favorables en la mayoría de la métricas de rendimiento, excepto en el recall que aumenta alrededor de un 20%, empero, la especificidad cae a menos de 6%. Estos resultados muestran, que para este caso, el balance de los datos no contribuyó al aprendizaje de la clase minoritaria. En los clasificadores RF y ANN, en cambio, hubo incremento en la TNR del 54% y 70 %, respectivamente. La precisión, el F1-score y el recall se mantuvieron sin variaciones importantes en el RF, pero para ANN disminuyeron alrededor de diez puntos porcentuales. En general, para RF y ANN el balance de clases favorece a reducir el sesgo de la clasificación. Respecto al rendimiento del NBC, se muestra un aumento del 90% en la tasa de verdaderos negativos, en decremento de la tasa de verdaderos positivos que se reduce a menos del 20%. Estos valores muestran, que el algoritmo Nayve Bayes ha aumentado su capacidad para detectar la clase cero (minoritaria), pero a su vez, ha disminuido su capacidad para detectar la clase uno (mayoritaria).

Por último, se observa los resultados de la clasificación, usando el conjunto de datos balanceado con la técnica SMOTE. En el caso de la Regresión logística, hay una reducción en la especificidad, la precisión, el F1-Score y un ligero aumento en el recall. Estas estimaciones muestran, que el aprendizaje no se benefició del balance de clases. En cambio, para Random forest, si aumentó la especificidad, mientras que el resto de las métricas se mantuvieron superiores al 95%, mostrando que el algoritmo mantuvo su capacidad para detectar instancias de la clase uno, y a su vez, aumentó su capacidad para detectar la clase cero. En el caso de ANN, la precisión presentó una ligera reducción. Sin embargo, los valores de TPR y TNR estuvieron más equilibrados, denotando un aprendizaje balanceado de ambas clases. Nayve Bayes, por otra parte, a pesar de que aumentó su TNR ($> 90\%$), disminuyó casi en la misma proporción su TPR, es decir, perdió la facultad de detección de la clase mayoritaria.

Adicionalmente, se midió el tiempo en segundos empleado en el entrenamiento (t_{train}) y en la fase de prueba (t_{test}) de cada algoritmo. Estos tiempos se muestran después de las métricas de rendimiento en la Tabla III. Del mismo modo, en la ordenada (eje y) de las gráficas de figura 3 se muestra el tiempo en escala logarítmica, para apreciar las diferencias entre las diferentes técnicas y modelos de ML. Como se puede observar, el clasificador con menores tiempos de entrenamiento y prueba fue el NBC con un valor promedio de 0.7532 y 0.0557 segundos, respectivamente, Por el contrario, el algoritmo ANN obtuvo los tiempos más altos, alcanzando valores superiores a los 1500s en el entrenamiento y a 230s en la ejecución de las pruebas. RL y RF obtuvieron tiempos promedios de entrenamiento de 4.2038s y 56.3011s, mientras que para la fase de pruebas obtuvieron tiempos promedios de 0.2789s y 6.1452s, respectivamente.

En general, las gráficas de la Figura 3 muestran que NearMiss reduce considerablemente los tiempos de entrenamiento

y prueba en todos los clasificadores, esto se debe a que el número de instancias procesadas es menor. En contraste, las técnicas de sobremuestreo e híbridas, obtienen valores similares en los tiempos de entrenamiento y prueba, que son mayores a los tiempos conseguidos con el CDD.

IV. Conclusiones, limitaciones y prospectivas

Este trabajo realizó un análisis del rendimiento de cuatro algoritmos de clasificación (Regresión logística, Nayve bayes, Random forest y Redes neuronales artificiales), después de usar diferentes técnicas de balanceo de clases (Near Miss, SMOTETomek, SMOTEENN, SMOTE) sobre el conjunto de datos original. Del análisis se desprende, que cuando los *datasets* no tienen un número similar de instancias de cada clase, se produce un sesgo en la clasificación que beneficia, generalmente, a la clase mayoritaria. Por esta razón, aunque la exactitud de todos los modelos entrenados con el *dataset* original, sea alta (mayor al 70%), no significa que sean capaces de identificar ambas clases con la misma precisión. De ahí los valores de especificidad bajos (menores al 50%).

Por lo tanto, el uso de técnicas de balanceo de datos puede ayudar a aumentar la capacidad de los modelos para detectar ambas clases con una precisión aceptable. Así, la técnica de submuestreo de clases, Near Miss, destacó entre las otras técnicas de sobremuestreo e híbridas, ya que logró un rendimiento superior al 90% en todos los modelos, es decir, aumentó la capacidad de los modelos para detectar instancias de la clase minoritaria, sin perjudicar la detección de las instancias de la clase mayoritaria. SMOTETomek, SMOTENN y SMOTE por su parte, no obtuvieron resultados tan favorables, para los modelos de Regresión Logística y Nayve Bayes, puesto que en el primer caso no aumentó la tasa de verdaderos negativos y en el segundo, aumentó la TNR, pero en detrimento de la tasa de verdaderos positivos. Por otra parte, en los modelos RF y ANN los conjuntos de datos balanceados con SMOTETomek, SMOTEENN y SMOTE si aumentaron el número de instancias de la clase minoritaria detectadas correctamente, sin afectar mayormente la sensibilidad de los modelos.

Respecto al rendimiento general de los clasificadores, se concluye que la exactitud de RL y NBC no aumenta al utilizar el conjunto de datos balanceado con técnicas de sobremuestreo o híbridas, como las utilizadas en este artículo, mientras que si lo hace, al aplicar técnicas de submuestreo como Near Miss. ANN por su parte, si aumenta la capacidad de detectar las instancias de la clase negativa al tener un conjunto balanceado, pero disminuye un poco su desempeño en la detección de la clase positiva. Random forest, en cambio, aumenta significativamente su desempeño en la clasificación de instancias de la clase cero, manteniendo su capacidad de detección de las instancias de la clase uno.

En cuanto a los tiempos de entrenamiento y prueba, Near Miss logró los menores valores, pues al reducir el número de muestras de la clase mayoritaria también disminuye significativamente el t_{train} y el t_{test} de todos los clasificadores, mientras que en las otras técnicas de balanceo sucede lo contrario. Es importante considerar que los tiempos medidos, dependen también de la infraestructura computacional usada en la implementación de los modelos. Por lo tanto, se propone realizar ensayos en máquinas con diferentes capacidades de cálculo y memoria, para contrastar los resultados.

A pesar de que los resultados muestran un bajo rendimiento en los clasificadores RL y NBC, se debe tener en cuenta que estos algoritmos son probabilísticos y trabajan bajo la premisa de independencia entre las variables, por lo que es posible que algunos de los atributos del conjunto de datos no cumplan este requerimiento. Ergo, como trabajo futuro se plantea incluir un análisis de dependencia de las variables (atributos) del *dataset*. Adicionalmente, se podrían incorporar otras técnicas de balanceo de clases aplicadas a los modelos de clasificación, además de las aplicadas sobre el conjunto de datos, que se analizaron en este artículo.

References

[1] A. Deshpande, C. Kamath, and M. Joglekar, "A comparison study of classification methods and effects of sampling on unbalanced data," in *2019 International Conference on Smart Systems and Inventive Technology (ICSSIT)*. IEEE, 2019, pp. 1056–1063.

[2] J. Gao, L. Gong, J. Wang, and Z. Mo, "Study on unbalanced binary classification with unknown misclassification costs," in *2018 IEEE International Conference on Industrial Engineering and Engineering Management (IEEM)*. IEEE, 2018, pp. 1538–1542.

[3] W.-C. Lin, C.-F. Tsai, Y.-H. Hu, and J.-S. Jhang, "Clustering-based undersampling in class-imbalanced data," *Information Sciences*, vol. 409, pp. 17–26, 2017.

[4] R. Blake and P. Mangiameli, "The effects and interactions of data quality and problem complexity on classification," *Journal of Data and Information Quality (JDIQ)*, vol. 2, no. 2, pp. 1–28, 2011.

[5] D. Lee and K. Kim, "An efficient method to determine sample size in oversampling based on classification complexity for imbalanced data," *Expert Systems with Applications*, vol. 184, p. 115442, 2021.

[6] S. Bagui and K. Li, "Resampling imbalanced data for network intrusion detection datasets," *Journal of Big Data*, vol. 8, no. 1, pp. 1–41, 2021.

[7] J. Zhai, J. Qi, and C. Shen, "Binary imbalanced data classification based on diversity oversampling by generative models," *Information Sciences*, vol. 585, pp. 313–343, 2022.

[8] L. Cai, H. Wang, F. Jiang, Y. Zhang, and Y. Peng, "A new clustering mining algorithm for multi-source imbalanced location data," *Information Sciences*, vol. 584, pp. 50–64, 2022.

[9] H. Ali, M. N. M. Salleh, R. Saedudin, K. Hussain, and M. F. Mushtaq, "Imbalance class problems in data mining: A review," *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 14, no. 3, pp. 1560–1571, 2019.

[10] J. Kim and M. Comuzzi, "A diagnostic framework for imbalanced classification in business process predictive monitoring," *Expert Systems with Applications*, vol. 184, p. 115536, 2021.

[11] X. Li and L. Zhang, "Unbalanced data processing using deep sparse learning technique," *Future Generation Computer Systems*, vol. 125, pp. 480–484, 2021.

[12] F. Mohd, M. A. Jalil, N. M. M. Noora, S. Ismail, W. F. F. Yahya, and M. Mohamad, "Improving accuracy of imbalanced clinical data classification using synthetic minority over-sampling technique," in *International Conference on Computing*. Springer, 2019, pp. 99–110.

[13] F. A. Naim, U. H. Hannan, and M. Humayun Kabir, "Effective rate of minority class over-sampling for maximizing the imbalanced dataset model performance," in *Proceedings of Data Analytics and Management*. Springer, 2022, pp. 9–20.

[14] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "Smote: synthetic minority over-sampling technique," *Journal of artificial intelligence research*, vol. 16, pp. 321–357, 2002.

[15] K. Alkharabsheh, S. Alawadi, V. R. Kebande, Y. Crespo, M. Fernández-Delgado, and J. A. Taboada, "A comparison of machine learning algorithms on design smell detection using balanced and imbalanced dataset: A study of god class," *Information and Software Technology*, vol. 143, p. 106736, 2022.

[16] H. Han, W. Wang, and B. Mao, "Borderline-smote: A new over-sampling method in imbalanced data sets learning. advances in intelligent computing. icic 2005," *Lecture Notes in Computer Science*, vol. 3644.

[17] M. Y. Arafat, S. Hoque, and D. M. Farid, "Cluster-based under-sampling with random forest for multi-class imbalanced classification," in *2017 11th International Conference on Software, Knowledge, Information Management and Applications (SKIMA)*, 2017, pp. 1–6.

[18] A. Muaz, M. Jayabalan, and V. Thiruchelvam, "A comparison of data sampling techniques for credit card fraud detection," *International Journal of Advanced Computer Science and Applications*, vol. 11, no. 6, 2020. [Online]. Available: http://dx.doi.org/10.14569/IJACSA.2020.0110660

[19] F. Itoo, S. Singh *et al.*, "Comparison and analysis of logistic regression, naïve bayes and knn machine learning algorithms for credit card fraud detection," *International Journal of Information Technology*, vol. 13, no. 4, pp. 1503–1511, 2021.

[20] N. M. Mqadi, N. Naicker, and T. Adeliyi, "Solving misclassification of the credit card imbalance problem using near miss," *Mathematical Problems in Engineering*, vol. 2021, 2021.

[21] A. Jabbar, X. Li, and B. Omar, "A survey on generative adversarial networks: Variants, applications, and training," *ACM Computing Surveys (CSUR)*, vol. 54, no. 8, pp. 1–49, 2021.

[22] Q. Wang, "A hybrid sampling svm approach to imbalanced data classification," in *Abstract and Applied Analysis*, vol. 2014. Hindawi, 2014.

[23] H.-C. Chen, E. Prasetyo, S. S. Kusumawardani, S.-S. Tseng, T.-L. Kung, K.-Y. Wang *et al.*, "Learning performance prediction with imbalanced virtual learning environment students' interactions data," in *International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing*. Springer, 2021, pp. 330–340.

[24] A. Hanskunatai, "A new hybrid sampling approach for classification of imbalanced datasets," in *2018 3rd International Conference on Computer and Communication Systems (ICCCS)*. IEEE, 2018, pp. 67–71.

[25] J. Kim and H. Park, "Reduced cnn model for face image detection with gan oversampling," in *International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing*. Springer, 2021, pp. 232–241.

[26] J. Engelmann and S. Lessmann, "Conditional wasserstein gan-based oversampling of tabular data for imbalanced learning," *Expert Systems with Applications*, vol. 174, p. 114582, 2021.

[27] 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.

[28] G. Lemaître, F. Nogueira, and C. K. Aridas, "Imbalanced-learn: A python toolbox to tackle the curse of imbalanced datasets in machine learning," *Journal of Machine Learning Research*, vol. 18, no. 17, pp. 1–5, 2017. [Online]. Available: http://jmlr.org/papers/v18/16-365.html

[29] P. Shukla and K. Bhowmick, "To improve classification of imbalanced datasets," in *2017 International Conference on Innovations in Information, Embedded and Communication Systems (ICIIECS)*. IEEE, 2017, pp. 1–5.

[30] S. Goyal, "Handling class-imbalance with knn (neighbourhood) undersampling for software defect prediction," *Artificial Intelligence Review*, pp. 1–42, 2021.

[31] A. Oliveira, "Malware analysis datasets: Pe section headers," 2019. [Online]. Available: https://dx.doi.org/10.21227/2czh-es14