Clasificación no balanceada: SMOTE & ADASYN

Tratamiento Inteligente de Datos

Pablo Valenzuela Álvarez (pvalenzuela@correo.ugr.es)

Índice

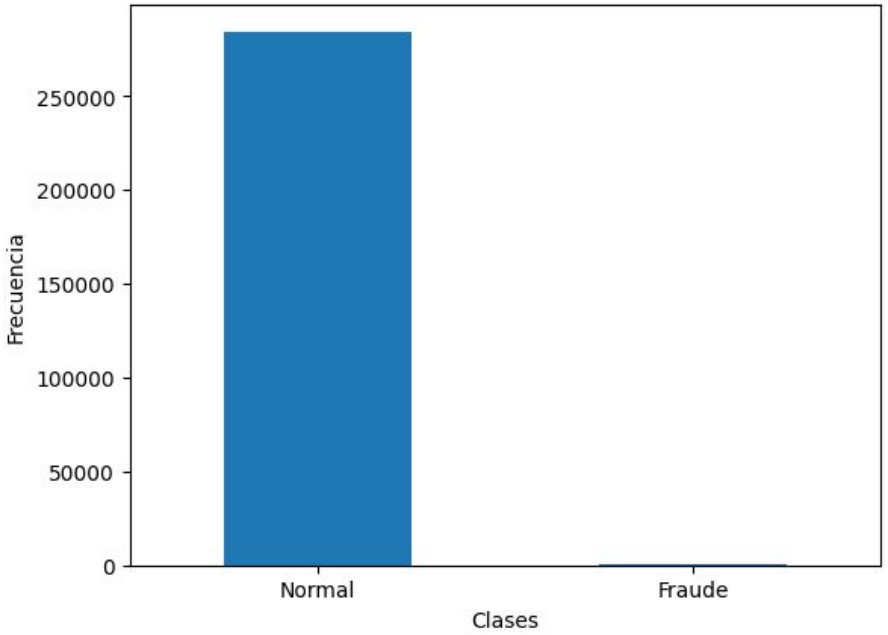
1. Introducción

El problema de las distribuciones de datos que no están balanceados es uno de los típicos que aparecen siempre en las disciplinas de **aprendizaje automático** y **minería de datos**. Se da cuando una de las clases está sensiblemente más representada que el resto, y por tanto, puede conducir a aprendizajes sesgados hacia la clase con mayor datos de interés.

A la hora de medir la efectividad del modelo, nos puede engañar la alta cantidad de aciertos que presenta. Solo tiene en cuenta a la clase mayoritaria por lo que genera la falsa sensación de que funciona bien.

1.1. Ejemplo práctico

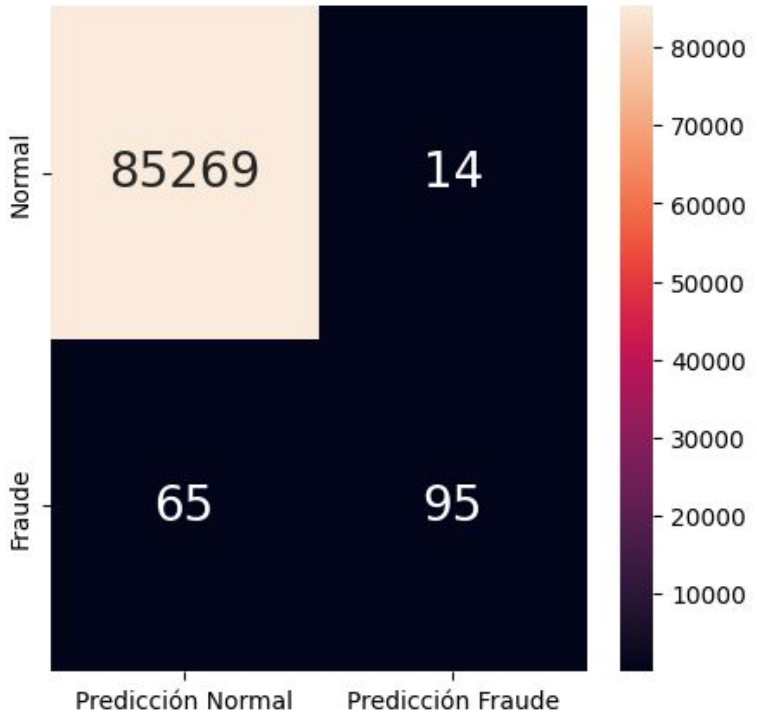
Vamos a usar un conjunto de datos usado para identificar **transacciones fraudulentas** hechas con tarjetas de crédito. El conjunto consta de cerca de **285000** casos de los cuales sólo 500 representan a la clase “fraude”. La distribución de esta clase no llega ni al **0,2%** del total.



https://www.kaggle.com/datasets/mlg-ulb/creditcardfraud/data.

1.1. Ejemplo práctico

Si entrenamos un modelo con este conjunto de datos, obtenemos resultados similares a los siguientes.

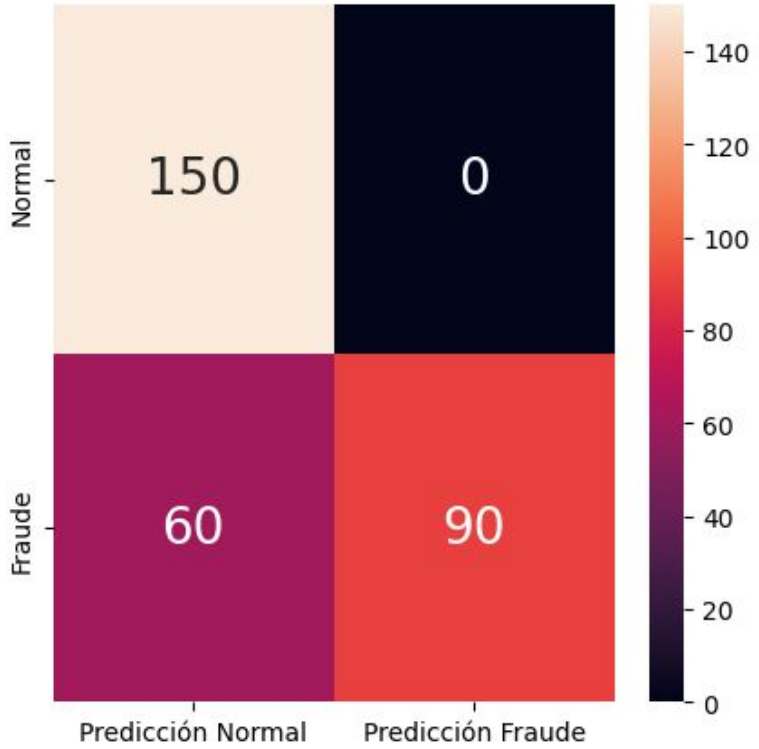


1.1. Ejemplo práctico

Clase	Precisión	Recall	F1-score	Soporte
Normal	0.9992	0.9998	0.9995	85283
Fraude	0.8716	0.5938	0.7063	160
		Accuracy	0.9991	85443

1.1. Ejemplo práctico

Si escogemos un conjunto de test balanceado con **150** ejemplos de cada clase, pasa lo siguiente.



1.1. Ejemplo práctico

Clase	Precisión	Recall	F1-score	Soporte
Normal	0.7143	1.000	0.8333	150
Fraude	1.000	0.6000	0.7500	150
		Accuracy	0.8000	300

1.2 Soluciones

1. Submuestreo: **Tomek**.
2. Sobremuestreo: **SMOTE** y **ADASYN**.
3. Algoritmos: **BalanceRandomForestClassifier** y **BalanceBaggingClassifier**.
4. Validación cruzada **estratificada**.

2. SMOTE

El significado de **SMOTE** es *Synthetic Minority Over-sampling Technique,* fue presentada en 2002 por **N. V. Chawla**, **K. W. Bowyer**, **L. O. Hall**, **W. P. Kegelmeyer**.

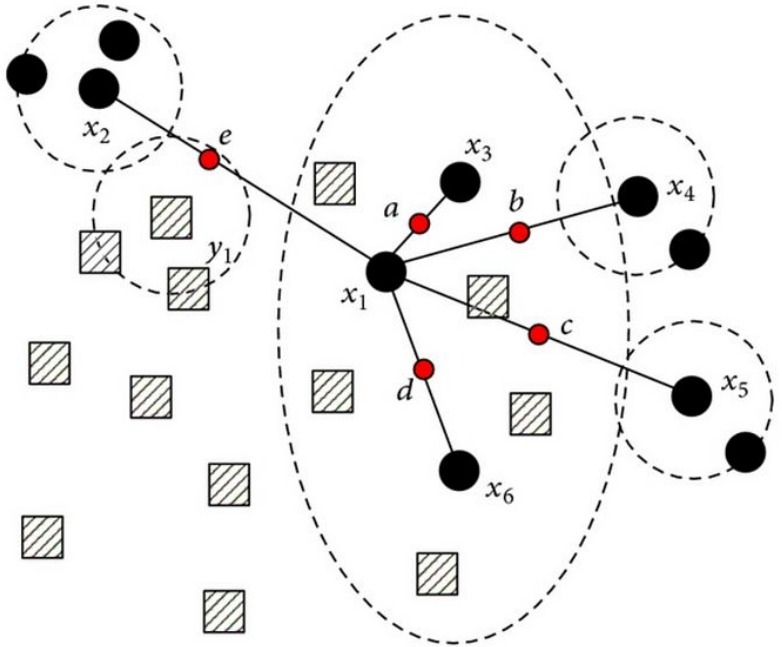
Como su nombre indica, es una técnica de sobre-muestreo. Esta técnica interpola nuevas instancias entre las ya existentes de la clase minoritaria.

Estas nuevas instancias se alinean entre los agrupamientos que se va encontrando de la clase minoritaria, de esta forma va añadiendo poco a poco nuevos casos en el espacio de características.

2.1 Funcionamiento

1. Selecciona la clase con menos instancias.
2. Encuentra un número "***k***" de agrupamientos de vecinos de esta clase.
3. Selecciona uno de estos agrupamientos al azar.
4. Genera nuevas instancias entre el agrupamiento escogido y los demás de la clase minoritaria. Se usa esta fórmula:

   *C = A + lambda * (B — A)*
5. Repetir desde el paso 1, hasta igualar el número de instancias de ambas clases



https://www.researchgate.net/publication/287601878/figure/fig1/AS:316826589384744@1452548753581/The-schematic-of-NRSBoundary-SMOTE-algorithm.png

2.2 Beneficios

- Nos ayuda a **balancear** los datos.
- **Reduce** el sesgo sobre la clase mayoritaria.
- **Fácil** de implementar.

2.3 Limitaciones

- No considera la **calidad** de las instancias. Al generar nuevas instancias linealmente entre las demás muestras de la clase minoritaria, no considera si esos casos son relevantes.
- No funciona bien con clases que están entremezcladas y puede generar **ruido**.
- Puede tener un **alto coste** computacional.
- Es sensible al número de **k** (vecinos cercanos).
- Solo funciona con **variables continuas**.

2.4 Implementación en Python

Si queremos aplicar esta técnica en Python necesitamos instalar la librería **imbalanced-learn**.

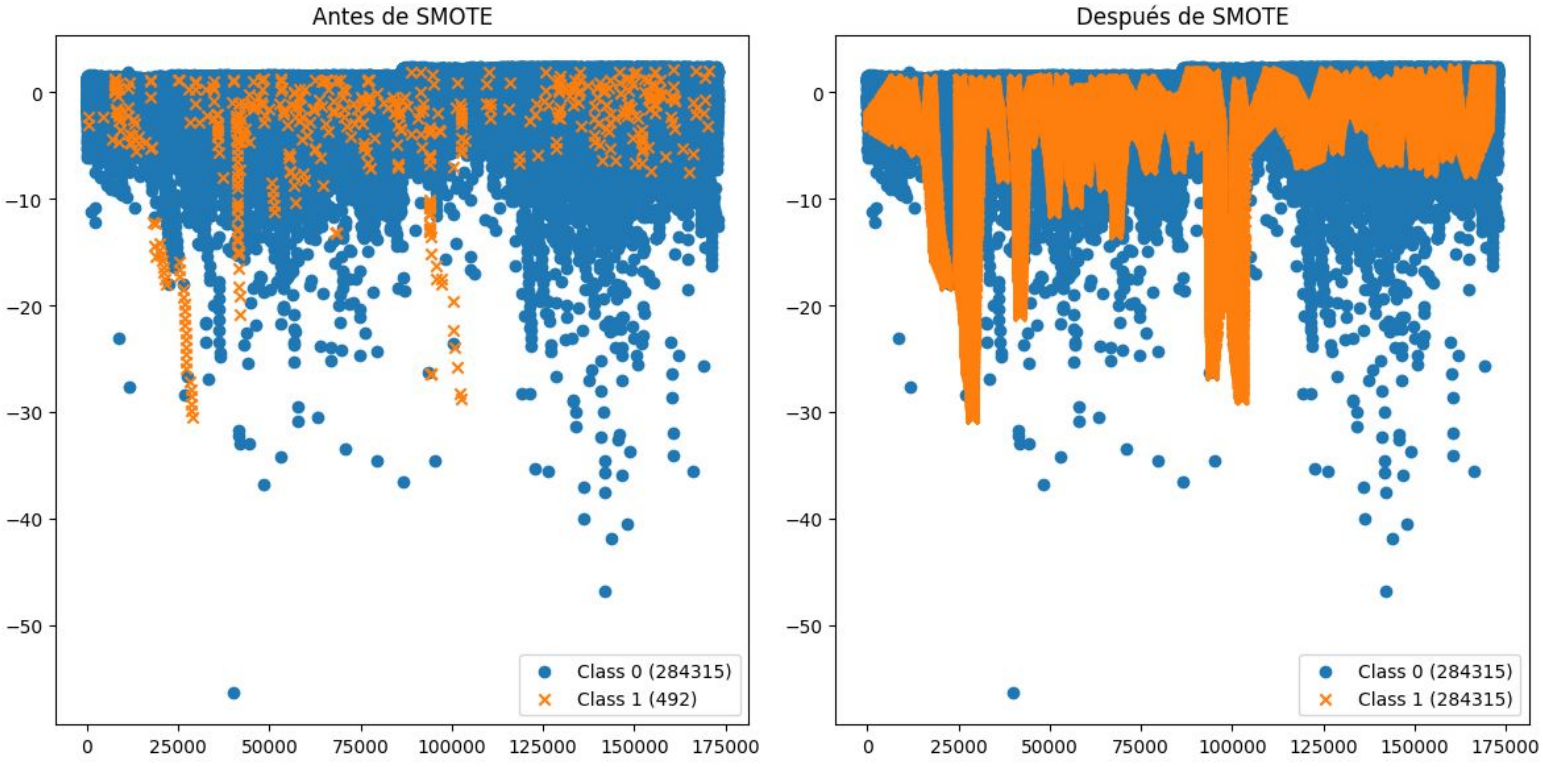
pip install imbalanced-learn

Y dentro del fichero Python:

```
from imblearn.over_sampling import SMOTE
```

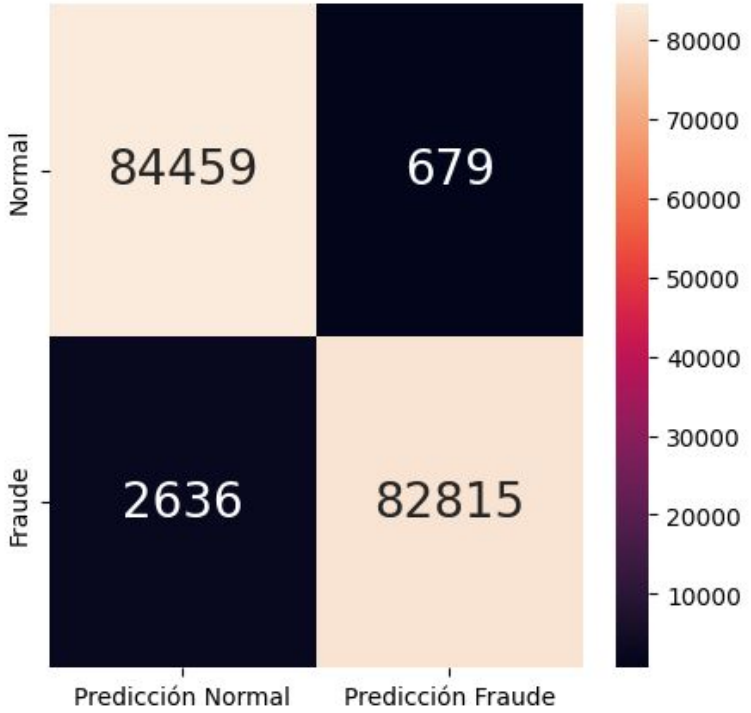
```
smote = SMOTE(k_neighbors=10, random_state=448)
X_smote, y_smote = smote.fit_resample(X,y)
```

2.4 Implementación en Python



2.4 Implementación en Python

El ratio de acierto para la clase positiva es mucho mayor.



2.4 Implementación en Python

Clase	Precisión	Recall	F1-score	Soporte
Normal	0.9697	0.9920	0.98084	85138
Fraude	0.9919	0.9692	0.9804	85451
		Accuracy	0.9806	170589

Tabla 3: Resultados tras aplicar SMOTE

3. ADASYN

ADASYN significa *Adaptative Synthetic Sampling*, y es una versión mejorada de SMOTE. Fue presentada en 2008 por **Haibo He, Yang Bai, Edwardo A. Garcia y Shutao Li.**

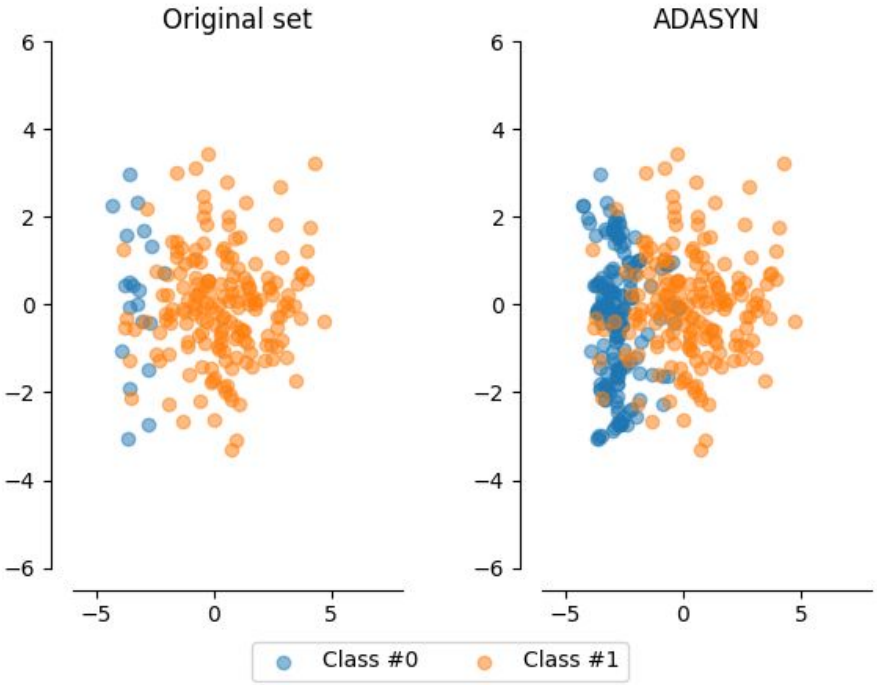
Esta técnica se ayuda de la distribución de densidad de la clase minoritaria a la hora de añadir nuevas instancias para aplicar ligeras variaciones en sus características, en vez de interpolar linealmente las instancias nuevas como hace SMOTE. Esto distribuye un poco más los datos y le da un toque más realista.

3.1 Funcionamiento

1. Selecciona la clase minoritaria.
2. Calcula el ratio de dificultad (DR) para un punto A, que representa el nivel de desbalanceamiento del grupo donde está ubicado el punto.

$$DR = \frac{Número\ de\ puntos\ de\ la\ clase\ mayoritaria}{Número\ de\ puntos\ de\ la\ clase\ minoritaria}$$

3. Generación de instancias aplicando DR, genera más donde el valor de este sea más alto ya que indica mayor desbalance en los datos.
4. Repetir desde el punto 1 hasta equilibrar ambas instancias.



https://www.researchgate.net/figure/Schematic-diagram-of-ADASYN-algorithm-application_fig5_344603283

3.2 Beneficios

- Nos ayuda a **balancear** los datos.
- Reduce el **sesgo** sobre la clase mayoritaria.
- Genera instancias de la clase minoritaria más **realistas**.
- Aplicabilidad a problemas del mundo real (detección de intrusos, investigación médica o detección de fraudes)

3.3 Limitaciones

- Coste computacional **alto**.
- Peligro de **sobreajuste**.
- No tiene en cuenta los **valores anómalos**, pueden afectar a la calidad de los datos.

3.4 Implementación en Python

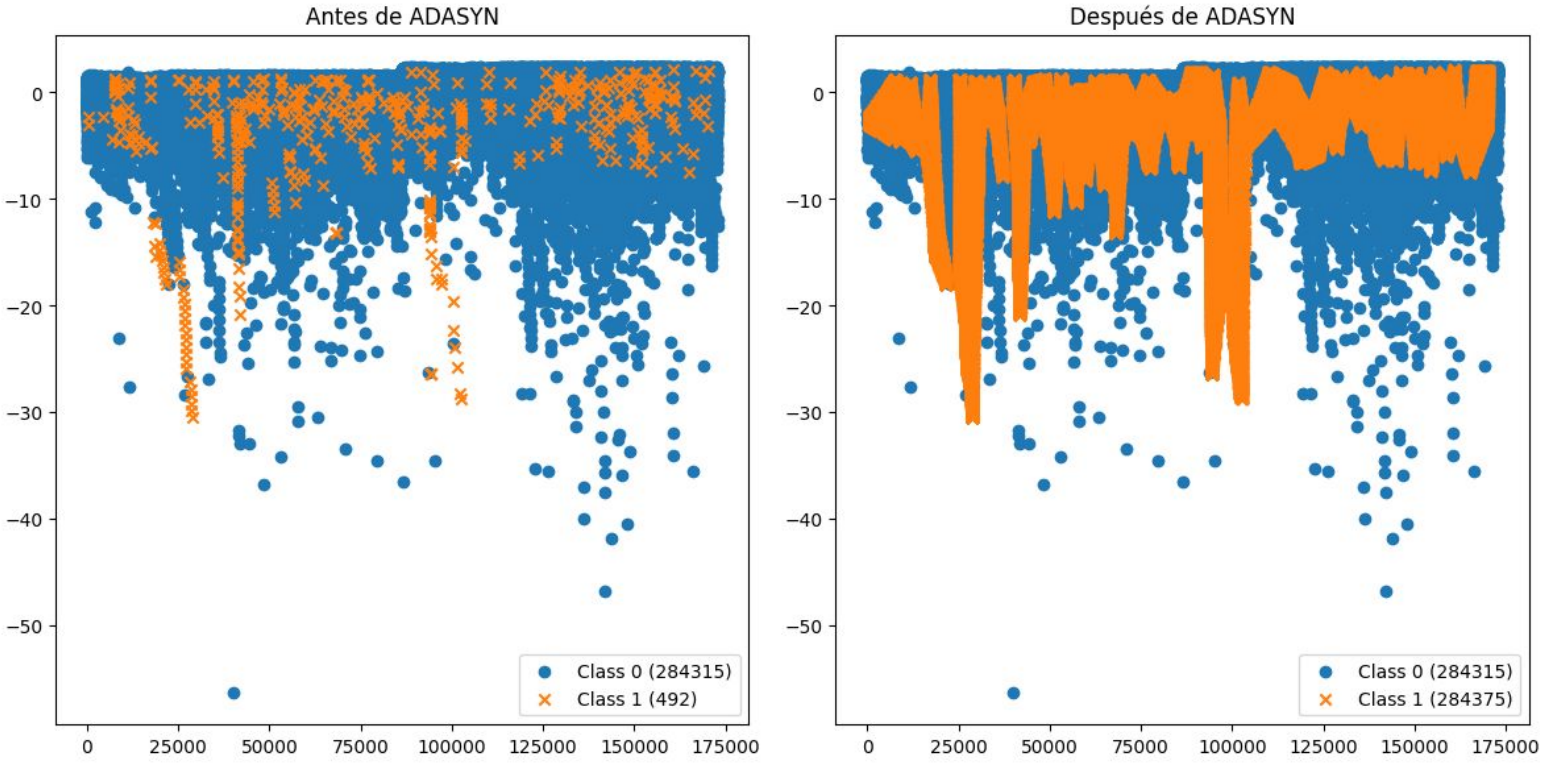
Esta técnica también está incluida en la librería **imbalanced-learn**.

Fichero Python:

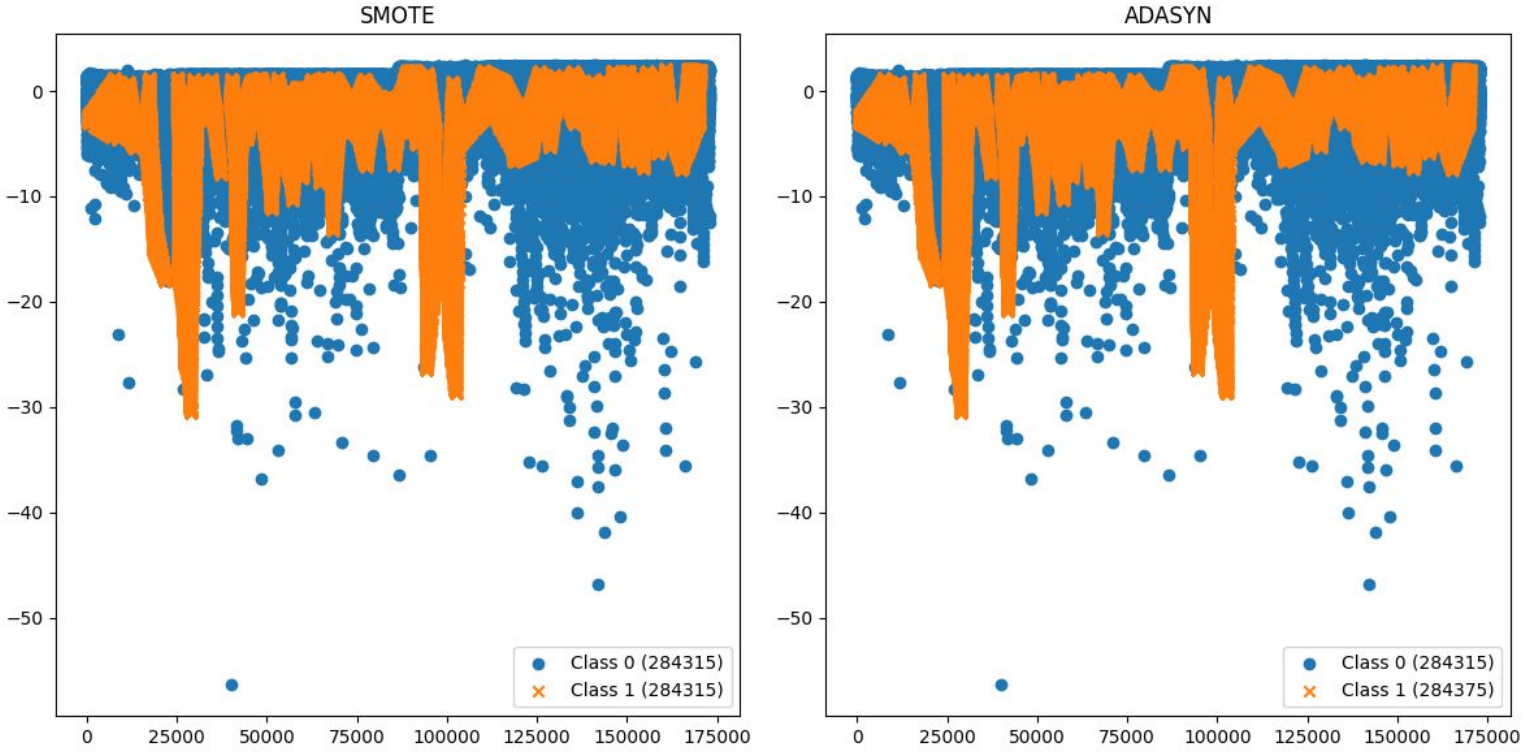
```
from imblearn.over_sampling import ADASYN
```

```
adasyn = ADASYN(n_neighbors=10, random_state=448)
X_adasyn, y_adasyn = adasyn.fit_resample(X,y)
```

3.4 Implementación en Python

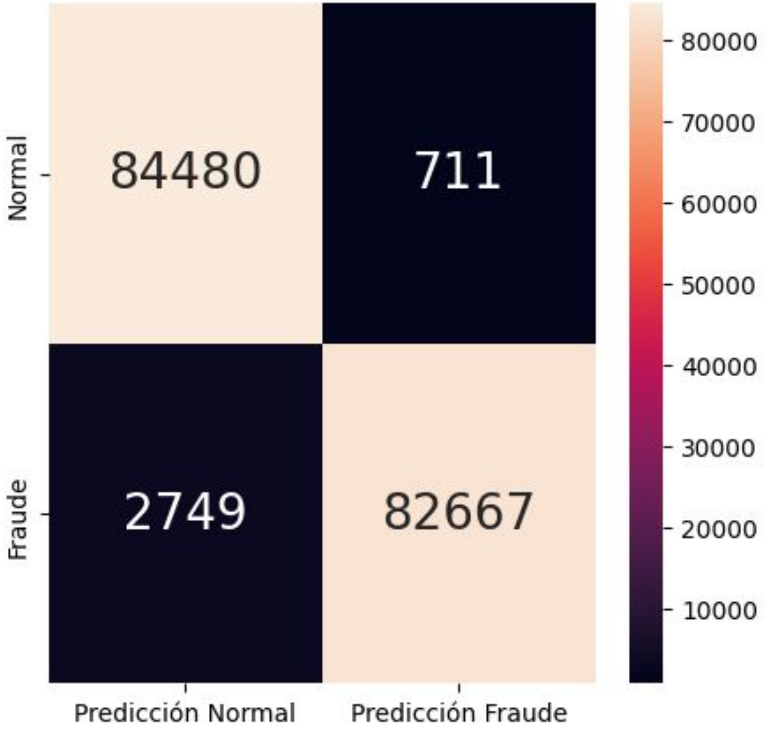


3.4 Implementación en Python



3.4 Implementación en Python

Obtenemos datos muy similares a los de la técnica anterior.



3.4 Implementación en Python

Técnica	Clase	Precisión	Recall	F1-score	Soporte
SMOTE	Normal	0.9697	0.9920	0.98084	85138
	Fraude	0.9919	0.9692	0.9804	85451
			Accuracy	**0.9806**	170589
ADASYN	Normal	0.9685	0.9917	0.9799	85191
	Fraude	0.9915	0.9678	0.9795	85416
			Accuracy	**0.9797**	170607

Tabla 4. Comparación de resultados de SMOTE y ADASYN.

4. Conclusión

1. **La generación de instancias**: SMOTE interpola entre los conjuntos de la clase minoritaria y ADASYN tiene en cuenta el ratio de dificultad para generar instancias más realistas.
2. **El conjunto de datos**: Puede aumentar el coste de computación ante datos muy desbalanceados, o contener outliers con influyan en el resultado final.

Sabiendo estos dos factores, ambas técnicas han demostrado su eficacia al identificar las clases en el ejemplo práctico y queda determinada su elección a las **preferencias del usuario**.

5. Bibliografía

“Clasificación con datos desbalanceados.” *Aprende Machine Learning*, https://www.aprendemachinelearning.com/clasificacion-con-datos-desbalanceados/.

“Credit Card Fraud Detection.” *Kaggle*, https://www.kaggle.com/datasets/mlg-ulb/creditcardfraud/data.

“Overcoming Class Imbalance with SMOTE: How to Tackle Imbalanced Datasets in Machine Learning.” *Train in Data Blog*, https://www.blog.trainindata.com/overcoming-class-imbalance-with-smote/.

“SMOTE and ADASYN (Handling Imbalanced Data Set) | by Indresh Bhattacharyya | Coinmonks.” *Medium*, https://medium.com/coinmonks/smote-and-adasyn-handling-imbalanced-data-set-34f5223e167.

“Adaptive Synthetic Sampling (ADASYN) | Adaptive Synthetic Sampling (ADASYN) Definition, Adaptive Synthetic Sampling (ADASYN) Use in ML.” *Activeloop*, https://www.activeloop.ai/resources/glossary/adaptive-synthetic-sampling-adasyn/.

“Data Imbalance: How is ADASYN different from SMOTE?” *Medium*, https://medium.com/@penpencil.blr/data-imbalance-how-is-adasyn-different-from-smote-f4eba54867ab.

“SMOTE — Version 0.11.0.” *Imbalanced-Learn*, https://imbalanced-learn.org/stable/references/generated/imblearn.over_sampling.SMOTE.html.

“ADASYN — Version 0.11.0.” *Imbalanced-Learn*, https://imbalanced-learn.org/stable/references/generated/imblearn.over_sampling.ADASYN.html.

“SMOTE: Synthetic Minority Over-sampling Technique.” *Journal of Artificial Intelligence Research*, https://www.jair.org/index.php/jair/article/view/10302.

“ADASYN: Adaptive Synthetic Sampling Approach for Imbalanced Learning.” *Soft Computing and Intelligent Information Systems*, https://sci2s.ugr.es/keel/pdf/algorithm/congreso/2008-He-ieee.pdf.