Proyecto del segundo parcial

David Paez [1] Paúl Cabadiana [2]
[1]Escuela de Ingeniería Mecatrónica, Universidad Internacional del Ecuador, Quito, Ecuador
*Autores Principales/Corresponding Authors, e-mail: dapaezgu@uide.edu.ec, pacabadianaca@uide.edu.ec

***Abstract*— Este estudio analiza la efectividad de dos técnicas de ensamblaje, Bagging y Boosting, en la predicción de clientes de tarjetas de crédito que incurren en mora. Los servicios de tarjetas de crédito operan como préstamos de dinero, donde los bancos asumen pérdidas por falta de pago, mitigadas en parte por el interés por mora. Bagging, y Boosting, fueron evaluados en términos de su capacidad para mejorar la precisión y estabilidad de los modelos predictivos. Se detallaran el procedimiento realizado para la implementación de los dos modelos y se compararan directamente por medio de los valores de sus métricas como lo es el f1-Score.**

***Keywords*—**

***Resumen*— This study analyzes the effectiveness of two assembly techniques, Bagging and Boosting, in predicting credit card customers who incur delinquencies. Credit card services operate as money lending, where banks assume losses from non-payment, mitigated in part by default interest. Bagging, specifically Random Forest, and Boosting, including AdaBoost and Gradient Boosting, were evaluated in terms of their ability to improve the accuracy and stability of predictive models. The procedure carried out for the implementation of the two models will be detailed and they will be compared directly through the values of their metrics such as the f1-Score.**

***Palabras Clave*—Bagging, Boosting, Mora, f1-Score, Python**

Bagging, Boosting, Mora, f1-Score, Python

I. INTRODUCCION

Los servicios de tarjetas de crédito funcionan a forma de prestamos de dinero por una entidad bancaria que es luego cobrado en plazos establecidos en un contrato. Es decir que por cada falta de pago de en tarjeta, es el banco el que asume la perdida de ese dinero por lo que se creó el interés por mora, el cual es una penalización que se aplica a los pagos retrasados de deudas[1], y representa un costo adicional que suele ser un porcentaje del total de la deuda, que los deudores deben pagar adicionalmente por incumplir los plazos de pago acordados. Por lo que un banco detecte un aumento de clientes que entran en mora, significa que el banco está sufriendo perdidas.

Bagging, abreviatura de "Bootstrap Aggregating", es una técnica de ensamblaje que mejora la precisión y estabilidad de los modelos de aprendizaje automático mediante la combinación de predicciones de múltiples modelos. En el caso de Random Forest, este método implica la creación de numerosos árboles de decisión, cada uno entrenado con una muestra diferente del conjunto de datos original obtenida mediante muestreo con reemplazo (bootstrap). Cada árbol realiza una predicción independiente y, en el caso de problemas de clasificación, la predicción final se determina por voto mayoritario, mientras que para problemas de regresión se promedian las predicciones. Esta combinación reduce la varianza y ayuda a prevenir el sobreajuste, mejorando la precisión general del modelo.

Boosting es otra técnica de ensamblaje que construye un modelo fuerte a partir de una serie de modelos débiles, generalmente árboles de decisión simples. A diferencia del bagging, boosting construye modelos secuenciales, donde cada nuevo modelo se entrena para corregir los errores cometidos por los modelos anteriores. Los datos mal clasificados en etapas anteriores reciben mayor peso en las etapas posteriores, obligando al nuevo modelo a centrarse en estos ejemplos difíciles. Modelos de boosting populares incluyen AdaBoost y Gradient Boosting. AdaBoost ajusta las ponderaciones de las instancias de entrenamiento en cada iteración, mientras que Gradient Boosting optimiza una función de pérdida aditiva. Esta técnica tiende a reducir tanto el sesgo como la varianza, produciendo modelos altamente precisos y robustos.

II. METODOLOGIA

A. *Análisis de datos*

El análisis de los datos del dataset se realizó mediante las funciones en el código observados durante las clases, empezando por la visualización del DataFrame, aplicamos el código data.info para dectectar que hay en sun mayoria datos Int y Float, con unas pocas excepciones de datos object, en las columnas objeto se comprobaron valores discretos y se encontraron datos nan y 0, despues se aplico una descripción de la data por medio de data.decribe, donde no se encontraron datos nan pero si anomalias como números sin función aparente en la data para el caso de las columnas M1-M6, finalmente en estas columnas se aplico una busqueda de los

valores -2 y 0 q eran los datos anómalos a eliminar.

B. Imputación de datos

Tras la realización del análisis se llego a la conclusión que para los valores de M1-M6, se debe realizar el cambio de los datos -2 y 0, pues no se encontraron especificados en la información brindada por el proyecto, donde se indica que -1 es pago a tiempo, 1 es pago con un mes de retraso, 2 pago con dos meses de retraso y continua de forma sucesiva. Para las columnas D1-D6 y P1-P6 no se encontraron necesidades de imputar. Entrando a los datos objetos se tomara en cuenta que dirante la creación del modelo, ID y G serán eliminados, pues ninguno toma verdadera relevancia para la predicción y exactitud del modelo, en el caso de la columa EC se buscara imputar los datos nan y 0 con los valores más frecuentes y una vez hecho ese remplazo, se realizo el cambio de sus valores a datos númericos con el fin de facilitar y/o posibilitar el análisis en el modelo, la variable 'Casado' se remplazo con 1, 'Soltero' con 2, 'Otros' con 3. La última columna SP igualmente se reemplazaron los valores 'Si' con 1 y 'No' con 0.

III. RESULTADOS

Al ser el objetivo de este proyecto el obtener el mejor modelo basado en el dato de "f1-Score" ya que este dato es la proporción adecuada entre precisión y la sensibilidad (falsos positivos y falsos negativos), que son críticos en la aprobación de créditos. Los resultados que ofrece el modelo entrenado y los resultados esperados se basan precisamente en el valor f1-Score, en el caso de este trabajo se presenta dentro del rango de 0 a 1. Basado en esta premisa se obtuvieron los siguientes resultados con la imputación de datos realizada y el ajuste de las métricas.

A. Bagging

El modelo obtuvo un valor f1-Score de 0.89 la evidencia se observa en la Fig.1.

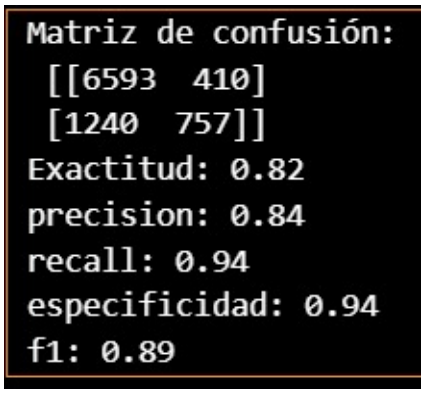

Fig. 1. f1-Score del primer caso

B. Boosting

El modelo obtuvo un valor f1-Score de 0.47 la evidencia se observa en la Fig.2.

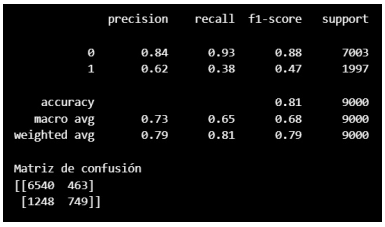

Fig. 2. f1-Score del segundo caso

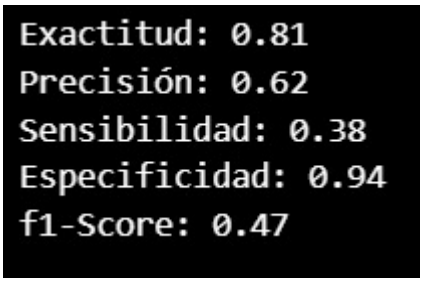

Fig. 3. f1-Score del segundo caso

IV. CONCLUSIONES

Tras comparar los resultados del valor de f1-Score, el cual es el valor de guía para la definición del mejor modelo, se concluye que el modelo de Bagging (RandomForest) es el mejor modelo para el análisis de este caso con un f1-Score de 0.89 tras la imputación de sus datos y las mejoras en sus métricas, en comparación con el modelo Boosting de 0.47 el cual consideramos como extremadamente bajo. No obstante, Bagging también lo supera en las demás métricas a exepción de especifidad donde comparten un valor de 0.94, el este resultado demuestra que Bagging posee una mayor estabilidad a nivel general.

REFERENCIAS

[1] Cuatro claves para entender cómo funciona el interés por mora en las deudas por créditos. [Online]. Available: https://www.elcomercio.com/actualidad/negocios/funcionamiento-interes-mora-deudas-creditos.html