Caso práctico

Modelo de recuperación de préstamos

2 Caso práctico

Modelo de recuperación de préstamos

Trabajas en fondo de inversión que compra, entre otros, carteras de préstamos y acaban de adquirir una nueva cartera “Flamengo”. Se te pide analizarla y crear un modelo de propensión de impago. Para ello debes:

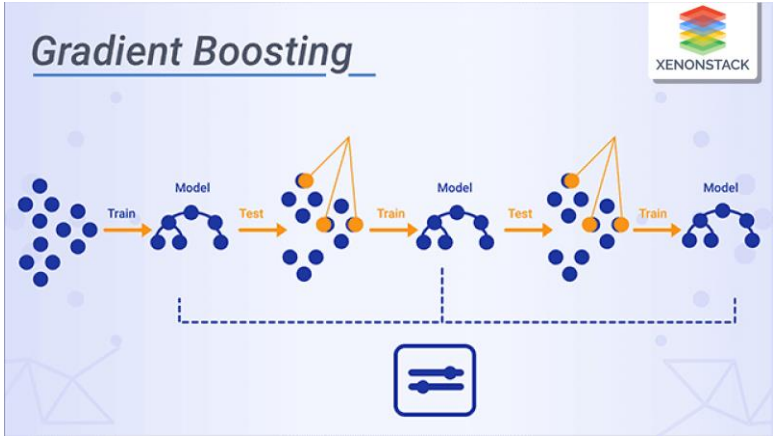
- Crear mediante Python un análisis descriptivo de los datos proporcionados
- Crear un modelo de clasificación binaria para detectar aquellos préstamos más propensos al impago
- Reportar la evolución de dicho modelo
- Predecir para un nuevo dataset su propensión al impago

B

Modelización

Modelización con python

Gradient Boosting es un método de ensamble que se enfoca en mejorar continuamente los errores del modelo, construyendo árboles de decisión secuenciales. A diferencia de Random Forest, que entrena árboles en paralelo, en Gradient Boosting cada árbol nuevo se entrena para corregir los errores cometidos por los árboles anteriores.



Proceso:
1.El primer árbol se entrena con los datos originales.
2.El segundo árbol se entrena con los errores (residuos) del primer árbol.
3.Este proceso se repite hasta que el modelo converge o alcanza un número de iteraciones determinado.
Este enfoque es útil en clasificación binaria, donde la tarea es predecir entre dos clases (por ejemplo, impago o no impago). Gradient Boosting crea modelos altamente precisos porque optimiza el error en cada etapa.

Algoritmos Populares de Gradient Boosting:
•XGBoost: Implementación optimizada de Gradient Boosting. Es muy rápido y ofrece muchas funcionalidades avanzadas.
•LightGBM: Variante que usa histogramas para optimizar el entrenamiento. Es eficiente en memoria y rápido en grandes datasets.
•CatBoost: Especialmente bueno para datos categóricos y desequilibrios en las clases.

XGBoost LightGBM CatBoost

B

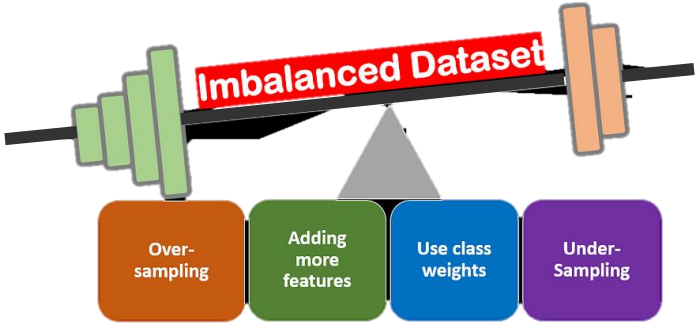
Modelización

Modelización con python

Problemas de Datos Desbalanceados

Un **problema de datos desbalanceados** ocurre cuando en un conjunto de datos**, las clases no están distribuidas de manera uniforme**. Un ejemplo común es la clasificación binaria en la que la mayoría de las etiquetas pertenecen a una clase (por ejemplo, 95% no impago y solo 5% impago). Este desequilibrio puede sesgar el modelo hacia la clase mayoritaria, afectando su rendimiento.

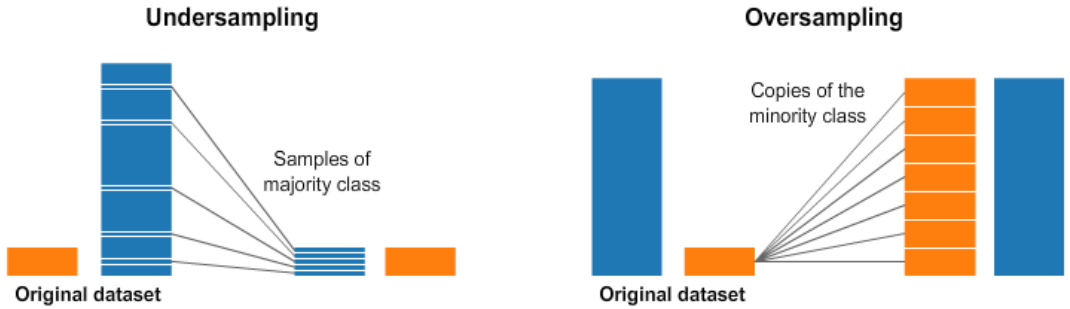
Problemas comunes con datos desbalanceados:
1.Modelos sesgados: Los modelos tienden a predecir la clase mayoritaria con mayor frecuencia, ignorando la clase minoritaria.
2.Métricas de evaluación inapropiadas: Métricas como la exactitud (accuracy) pueden ser engañosas, ya que un modelo que predice siempre la clase mayoritaria puede tener una alta precisión pero bajo rendimiento en la clase minoritaria.



B

Modelización

Modelización con python



Métodos para manejar datos desbalanceados:

1. Recolección de más datos:
Obtener más ejemplos de la clase minoritaria puede ser una solución ideal, aunque no siempre es posible en la práctica.

2. Ajuste de clases mediante muestreo:
•Submuestreo (undersampling): Se eliminan ejemplos de la clase mayoritaria para igualar el número de ejemplos en ambas clases.
- Ventaja: Reduce el sesgo hacia la clase mayoritaria.
- Desventaja: Se pierden datos valiosos de la clase mayoritaria.

•Sobremuestreo (oversampling): Se generan ejemplos adicionales de la clase minoritaria (duplicando ejemplos o usando técnicas sintéticas).
- Ventaja: Proporciona más ejemplos de la clase minoritaria.
- Desventaja: Puede generar sobreajuste si se usan demasiados datos duplicados.

•SMOTE (Synthetic Minority Over-sampling Technique): Genera nuevos ejemplos sintéticos de la clase minoritaria basados en la interpolación entre los datos existentes.
- Ventaja: Crea ejemplos más diversos sin simplemente duplicar datos.
- Desventaja: Puede generar ruido si se crean ejemplos irrelevantes.

B Modelización

Modelización con python

Evaluación de Performance

Para evaluar la performance de un modelo de **Gradient Boosting para clasificación binaria**, los indicadores más utilizados son:

1. **Accuracy**: Proporción de predicciones correctas. No siempre es útil en datasets desbalanceados.
2. **Precision**: Proporción de verdaderos positivos entre los elementos que fueron clasificados como positivos. Útil cuando los falsos positivos son costosos.
3. **Recall (Sensibilidad)**: Proporción de verdaderos positivos entre todos los elementos positivos. Es útil cuando los falsos negativos son más costosos.
4. **F1 Score**: La media armónica de precision y recall. Es ideal para evaluar modelos en datasets desbalanceados.
5. **AUC-ROC**: Área bajo la curva ROC. Mide la capacidad del modelo para distinguir entre clases, especialmente en problemas de clasificación binaria.

Accuracy	Predictions/ Classifications	$\frac{\text{Correct}}{\text{Correct + Incorrect}}$
Precision	Predictions/ Classifications	$\frac{\text{True Positive}}{\text{True Positive + False Positive}}$
Recall	Predictions/ Classifications	$\frac{\text{True Positive}}{\text{True Positive + False Negative}}$
F1	Predictions/ Classifications	$\frac{\text{2 * True Positive}}{\text{True Positive + 0.5 (False Positive + False Negative)}}$