



Ciencia de
Datos

XGBoost

MARIANA PÉREZ PÉREZ
YORIEL NAVIER CARVAJALINO VIDAL
ADRIANA LUCIA CASTRO CARREÑO

XXX





El aprendizaje por Ensamble

En el campo de la Ciencia de Datos, uno de los desafíos principales es construir modelos de predicción que sean precisos, robustos y generalizables.

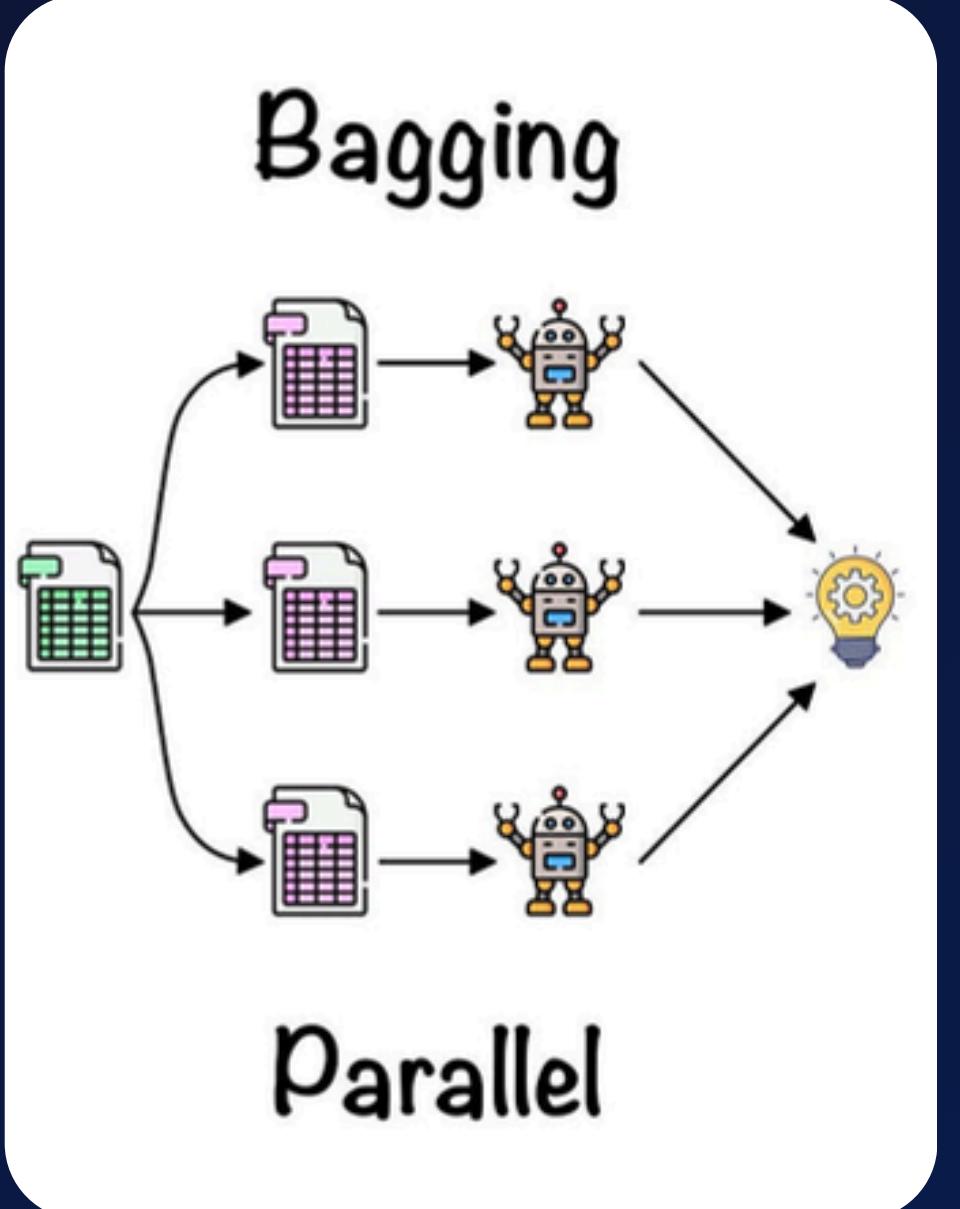
A menudo, un solo modelo (por ejemplo, un árbol de decisión simple) no logra captar la complejidad de los datos. Por eso, surge la idea de combinar varios modelos para mejorar el rendimiento: esta es la base del aprendizaje por ensamble (ensemble learning).

Principio:

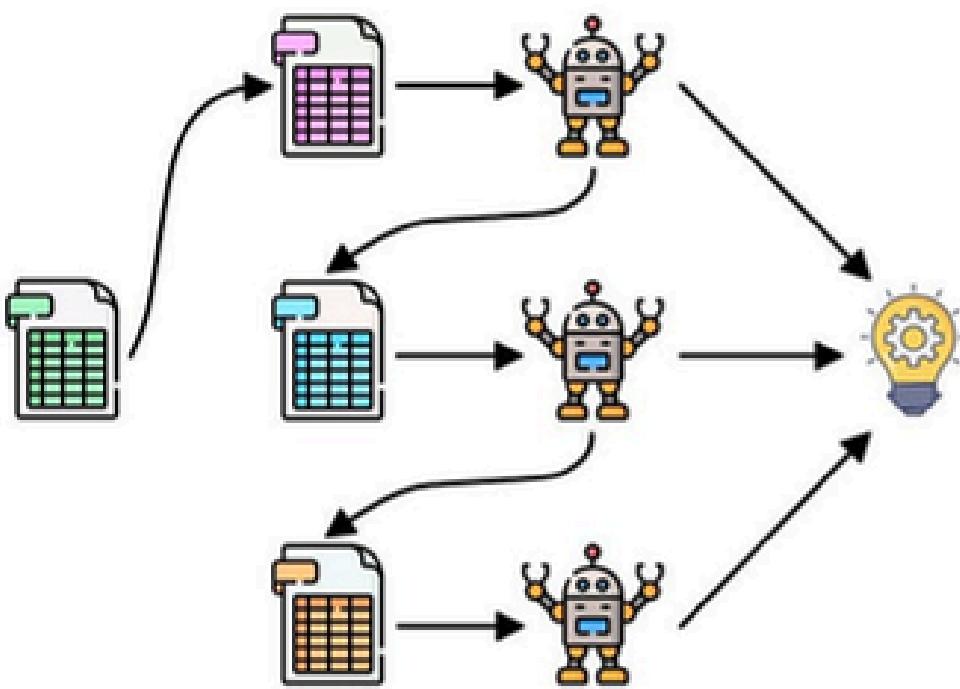
“Varios modelos débiles, si se combinan correctamente, pueden formar un modelo fuerte.”

Principales Enfoques

Bagging (Bootstrap Aggregating): entrena múltiples modelos en subconjuntos aleatorios de los datos y promedia sus resultados. Ejemplo: Random Forest.



Boosting



Sequential

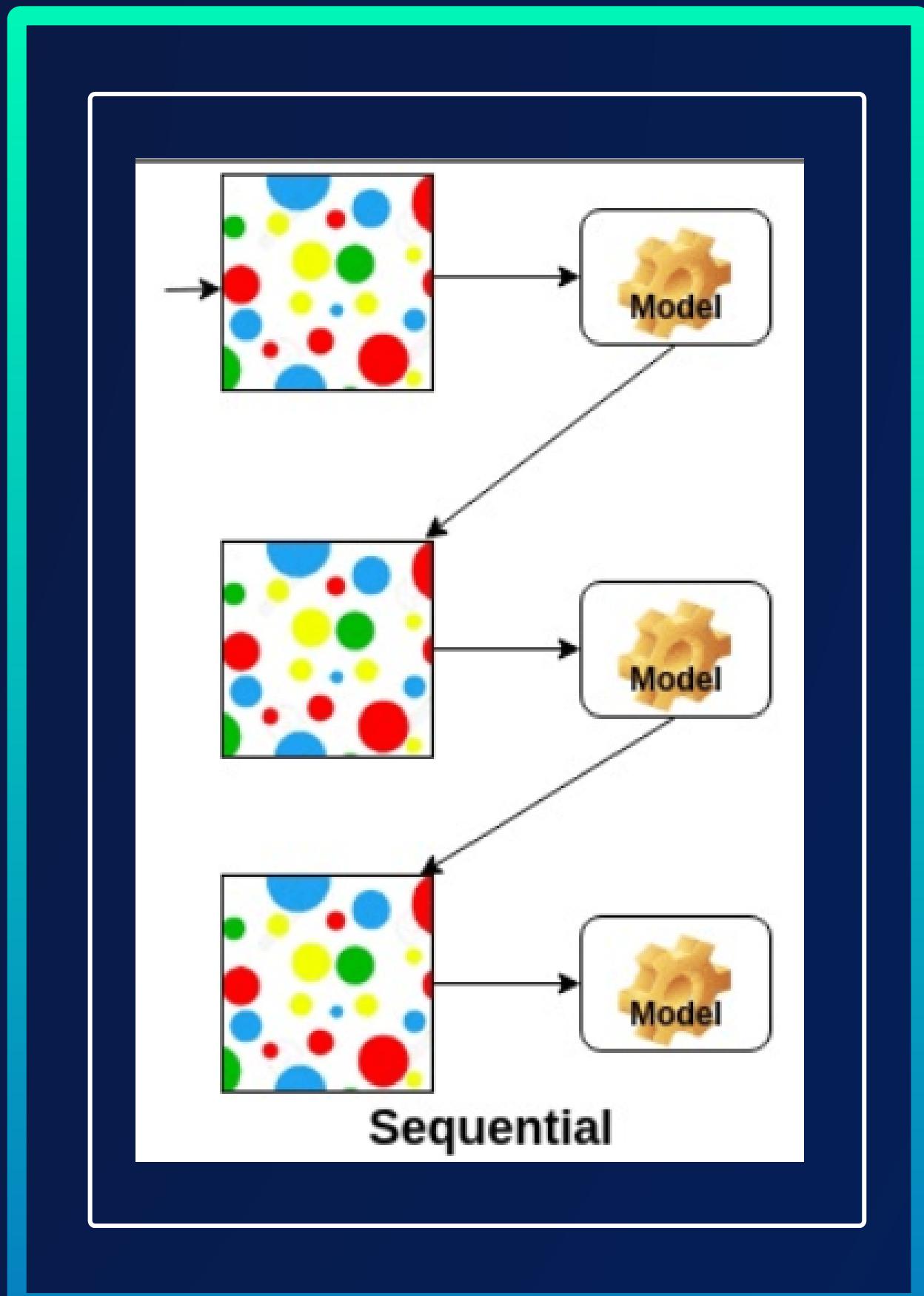
Boosting: entrena modelos secuencialmente, donde cada modelo nuevo aprende de los errores del anterior. Ejemplo: AdaBoost, Gradient Boosting, XGBoost.

XGBOOST

XGBoost (Extreme Gradient Boosting) es una versión optimizada del algoritmo Gradient Boosting Machine (GBM).

Fue desarrollada por Tianqi Chen en 2016 y rápidamente se convirtió en uno de los modelos más utilizados en la ciencia de datos moderna debido a su:

- Alta precisión.
- Velocidad de entrenamiento (usa optimizaciones en CPU y GPU).
- Capacidad de evitar el sobreajuste.
- Mecanismo interno de regularización.



Principales Aplicaciones

01

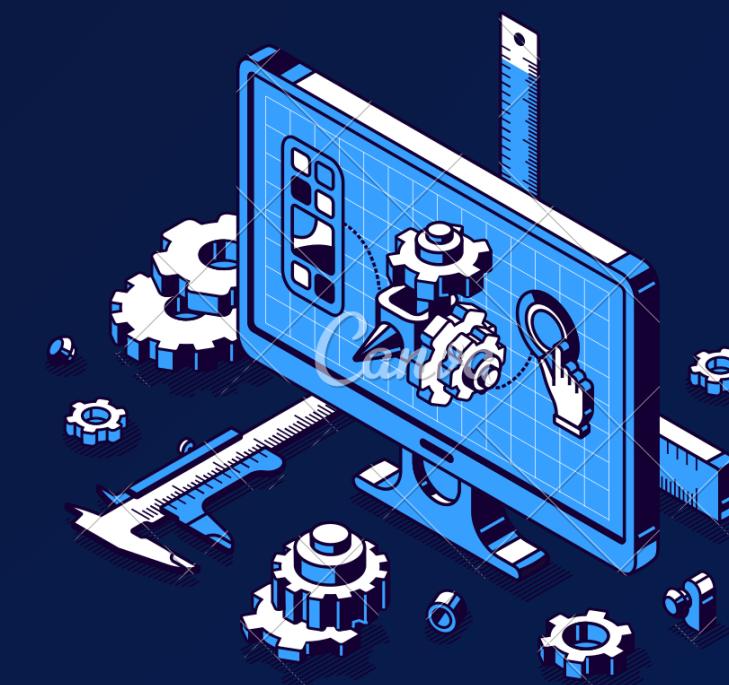
Clasificación

- Cuando la variable objetivo es categórica (por ejemplo, “fraude” o “no fraude”, “aprobado” o “rechazado”).
- El modelo predice la probabilidad de pertenecer a cada clase.

02

Regresión

- Cuando la variable objetivo es numérica continua (por ejemplo, el precio de una vivienda o el nivel de ventas).
- El modelo intenta predecir un valor real aproximado.



× × ×

Variables en el modelo: objetivo y predictoras



Variable objetivo (Target)

Es la variable que queremos predecir.

Ejemplo:

- En un modelo de detección de fraude → fraude = Sí/No.
- En un modelo de riesgo crediticio → riesgo = Alto/Bajo.
- En un modelo de predicción de ventas → monto_ventas (numérico).



Variables predictoras (Features)

Son las características que se usan para hacer la predicción.

- Ejemplo: edad, ingresos, historial de crédito, tipo de cliente, región, etc.

El objetivo del modelo es encontrar patrones en las variables predictoras que expliquen la variable objetivo. Este proceso se apoya en los principios de la estadística inferencial, que permiten evaluar qué variables realmente aportan información significativa.

Fundamentos estadísticos

01

Hipótesis nula y alternativa

En estadística, se plantea una hipótesis nula (H_0) que asume que no hay relación entre una variable y el resultado, y una hipótesis alternativa (H_1) que indica que sí la hay.

Por ejemplo:

- **H_0 :** La edad no influye en la probabilidad de fraude.
- **H_1 :** La edad sí influye en la probabilidad de fraude.

El objetivo es recolectar evidencia suficiente para rechazar o no rechazar la hipótesis nula.

02

El valor p (p-value)

El p-value indica la probabilidad de obtener los resultados observados si la hipótesis nula fuera verdadera.

- Si $p < 0.05$, se rechaza $H_0 \rightarrow$ la variable es estadísticamente significativa.
- Si $p > 0.05$, no se rechaza $H_0 \rightarrow$ la variable probablemente no tiene efecto.

Aunque XGBoost no calcula directamente el p-value, este concepto es importante en la fase de análisis exploratorio para decidir qué variables pueden ser útiles antes del modelado.

Fundamentos estadísticos

03

Variables significativas

Una variable significativa es aquella que aporta información relevante para predecir la variable objetivo.

En modelos estadísticos tradicionales, esto se determina mediante pruebas de hipótesis; en XGBoost, se mide a través de la importancia de características (feature importance).

Selección y evaluación de variables

🎯 ¿Cómo selecciona XGBoost las variables?

XGBoost entrena una secuencia de árboles que aprenden de los errores del modelo anterior. En cada árbol, analiza todas las variables y busca cuál logra dividir mejor los datos para reducir el error.

Split y Umbral (Threshold): cómo el modelo decide

Para encontrar la mejor división, XGBoost prueba muchos puntos de corte dentro de cada variable. Cada prueba crea una posible división, conocida como split.

Por ejemplo, si el modelo está analizando la variable ingresos, podría probar:

- ¿Ingreso > 2 millones?
- ¿Ingreso > 3 millones?
- ¿Ingreso > 4 millones?

Cada número que prueba (2, 3 o 4 millones) es un umbral (threshold).

Selección y evaluación de variables

⚖️ ¿Cómo decide cuál split es el mejor?

XGBoost no elige al azar: calcula una métrica llamada “Gain” (ganancia) para cada división. El Gain indica cuánto mejora el modelo si usa ese split.

Entonces, el modelo:

Split y Umbral (Threshold): cómo el modelo decide

Entonces, el modelo:

- 1.Calcula el Gain de “¿Ingreso > 2 millones?”
- 2.Calcula el Gain de “¿Ingreso > 3 millones?”
- 3.Calcula el Gain de “¿Ingreso > 4 millones?”
- 4.Compara los resultados y se queda con el split que tenga la mayor ganancia.

Ese split y ese umbral son los que más ayudan a separar correctamente los datos y a reducir el error.

Selección y evaluación de variables

Selección y evaluación de variables

Gain (Ganancia)

Mide cuánto mejora el modelo cada vez que usa una variable

Weight (Peso)

Número de veces que una variable aparece en las divisiones de los árboles.

Cover (Cobertura)

Mide la cantidad de muestras afectadas por una división con esa variable.

XGBoost ofrece medidas internas para identificar qué variables tienen mayor influencia:

Proceso de aprendizaje XGBoost

Paso 1:

Entrenamiento inicial

XGBoost comienza con un modelo simple, por ejemplo, una predicción promedio (en regresión) o la clase mayoritaria (en clasificación).

Paso 2 :

Cálculo del error

Luego calcula los errores (residuos) entre las predicciones y los valores reales. Estos errores representan lo que el modelo aún no ha aprendido.

Paso 3 :

Ajuste por gradiente

Se construye un nuevo árbol para corregir los errores anteriores.

Aquí entra el concepto de gradiente, que indica la dirección y magnitud del cambio necesario para reducir el error.

Matemáticamente, el modelo minimiza una función de costo:

$$Obj(\theta) = \sum_{i=1}^n l(y_i, \hat{y}_i) + \Omega(f_k)$$

- $l(y_i, \hat{y}_i)$ = función de pérdida (mide el error entre real y predicho).
- $\Omega(f_k)$ = término de regularización (controla la complejidad del modelo).

Proceso de aprendizaje XGBoost

Paso 4:

Actualización del modelo

El nuevo árbol se añade al conjunto anterior, con un peso de aprendizaje (learning rate) que regula cuánto corrige.

Paso 5 :

Iteraciones

Este proceso se repite muchas veces hasta que los errores dejan de mejorar significativamente o se alcanza un número máximo de árboles.



Regularización

El sobreajuste (overfitting) ocurre cuando el modelo se ajusta demasiado a los datos de entrenamiento, perdiendo capacidad de generalizar. XGBoost incorpora dos mecanismos de regularización que lo hacen más robusto que otros modelos de boosting:

Regularización L1 (Lasso):

Elimina características irrelevantes al forzar que algunos coeficientes sean exactamente cero.

Regularización L2 (Ridge):

Penaliza los valores grandes de los parámetros para evitar que dominen el modelo.

Evaluación del rendimiento del modelo

Una vez entrenado, XGBoost se evalúa con métricas de desempeño que indican qué tan bien generaliza:

- **Accuracy**: proporción de predicciones correctas.
- **Precision**: exactitud entre las predicciones positivas y las verdaderas.
- **Recall** (Sensibilidad): capacidad del modelo para detectar los casos positivos.
- **F1-Score**: equilibrio entre precisión y recall.
- **AUC-ROC**: mide la capacidad del modelo para distinguir entre clases.



Conclusiones

XGBoost es un algoritmo de Machine Learning altamente eficiente y preciso que mejora sus predicciones aprendiendo de los errores mediante boosting. Gracias a su regularización evita el sobreajuste, selecciona las variables más relevantes y generaliza bien en datos reales. Por su rendimiento y estabilidad, es uno de los modelos más utilizados en la Ciencia de Datos moderna para resolver problemas predictivos complejos.

XXX



XXX



Muchas Gracias