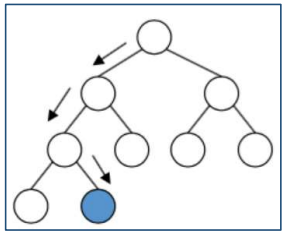
XGBoost :
eXtreme Gradient Boosting
(Potenciación del Gradiente Extrema)

Aprendizaje Automático

Luis Eduardo Falcón Morales

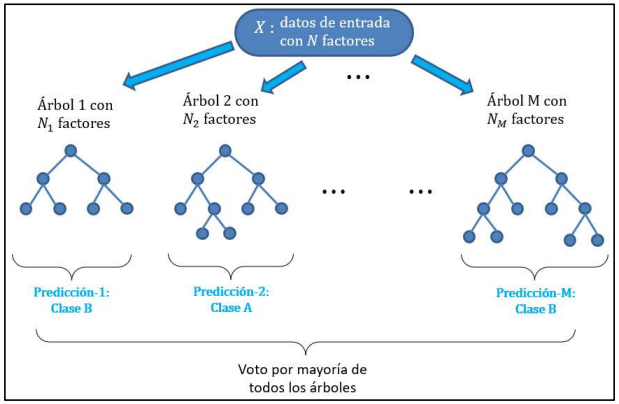
Métodos basados en Árboles

Árbol de Decisión
(Decision Tree)



Bosque Aleatorio
(Random Forest)

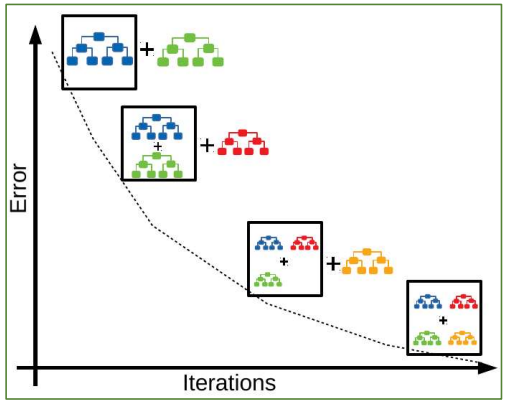
Muestreos con reemplazo



- **Bagging** (**B**ootstrap **Agg**rega**t**i**ng**) -

Voto por mayoría de
todos los árboles

Potenciación del Gradiente
(Gradient Boosting)



Entrenados de forma secuencial, cada nuevo árbol trata de mejorar los errores de los anteriores.

- **Boosting** -

Combinar algoritmos débiles para obtener como resultante uno fuerte: AdaBoost, Gradient Boosting, Stochastic Gradient Boosting, etc

Modelos Conjuntos / Ensembles Models

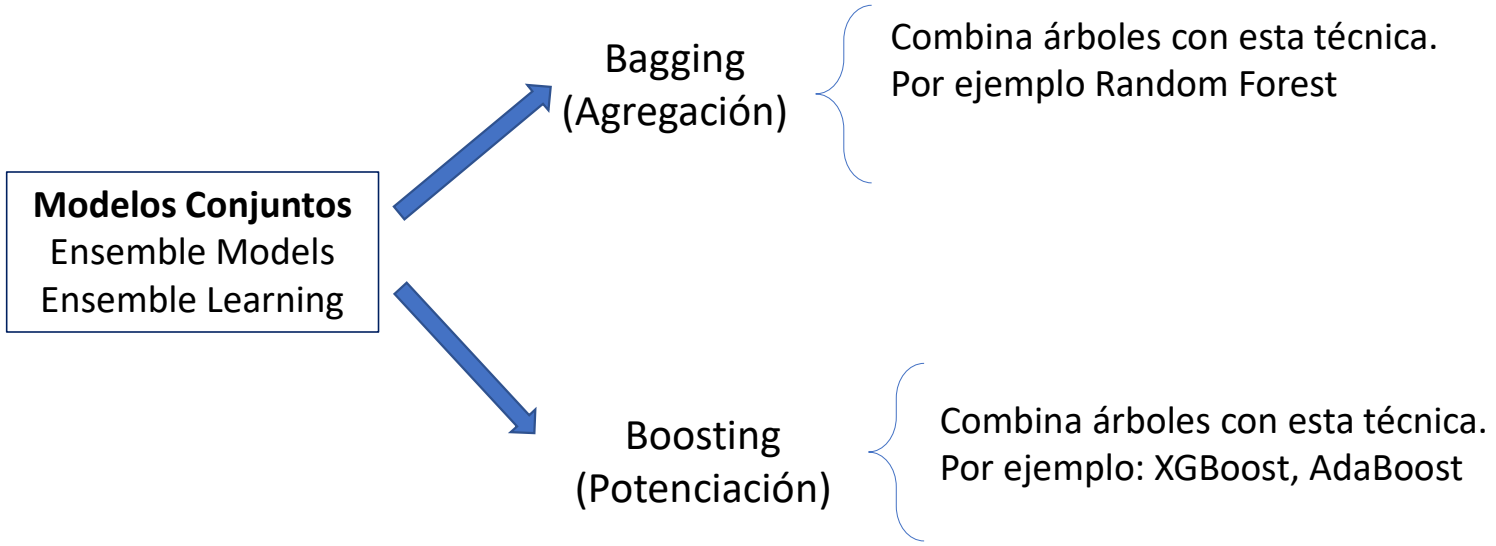
Estos métodos son menos propensos a errores ya que combinan los resultados de varios modelos. Reducen la varianza sobe todo en conjunto de datos muy ruidosos.

Modelos Conjuntos
Ensembles Models

- Se conjuntan direcctamente diferentes modelos por parte del analista y se someten a votación. Se puede usar por ejemplo VotingClassifier() de sklearn.

- Modelos que usan muchas versiones de sí mismos.

  Se basan en la técnica de muestreo llamada "**Bagging**: **Boostrap Aggregation**". En problemas de clasificación se puede decidir por votación y en problemas de regresión mediante el promedio de los modelos involucrados.


Modelos Conjuntos
Ensemble Models
Ensemble Learning
Bagging
(Agregación)
Combina árboles con esta técnica.
Por ejemplo Random Forest
Boosting
(Potenciación)
Combina árboles con esta técnica.
Por ejemplo: XGBoost, AdaBoost

De manera general:

- Los modelos Bagging aprenden en paralelo y los Boosting secuencialmente.
- Los modelos Bagging aplican bien para modelos que generan souluciones muy inestables de alta varianza y bajo sesgo, que usualmente nos llevan a problemas sobre-entrenados. Este es el caso de modelos basados en un solo árbol de decisión o el k-Vecinos Más Cercanos, kNN.
- Los modelos Boosting aplican bien para problemas con datos de baja varianza y alto sesgo, que usualmente nos llevan a problemas sub-entrenados. Este es el caso de árboles de muy poca profundidad o el modelo de regresión logística o regresión lineal.

Leo Breiman
UC Berkeley
[EEUU: 1928 – 2005]

Bagging
(1996)

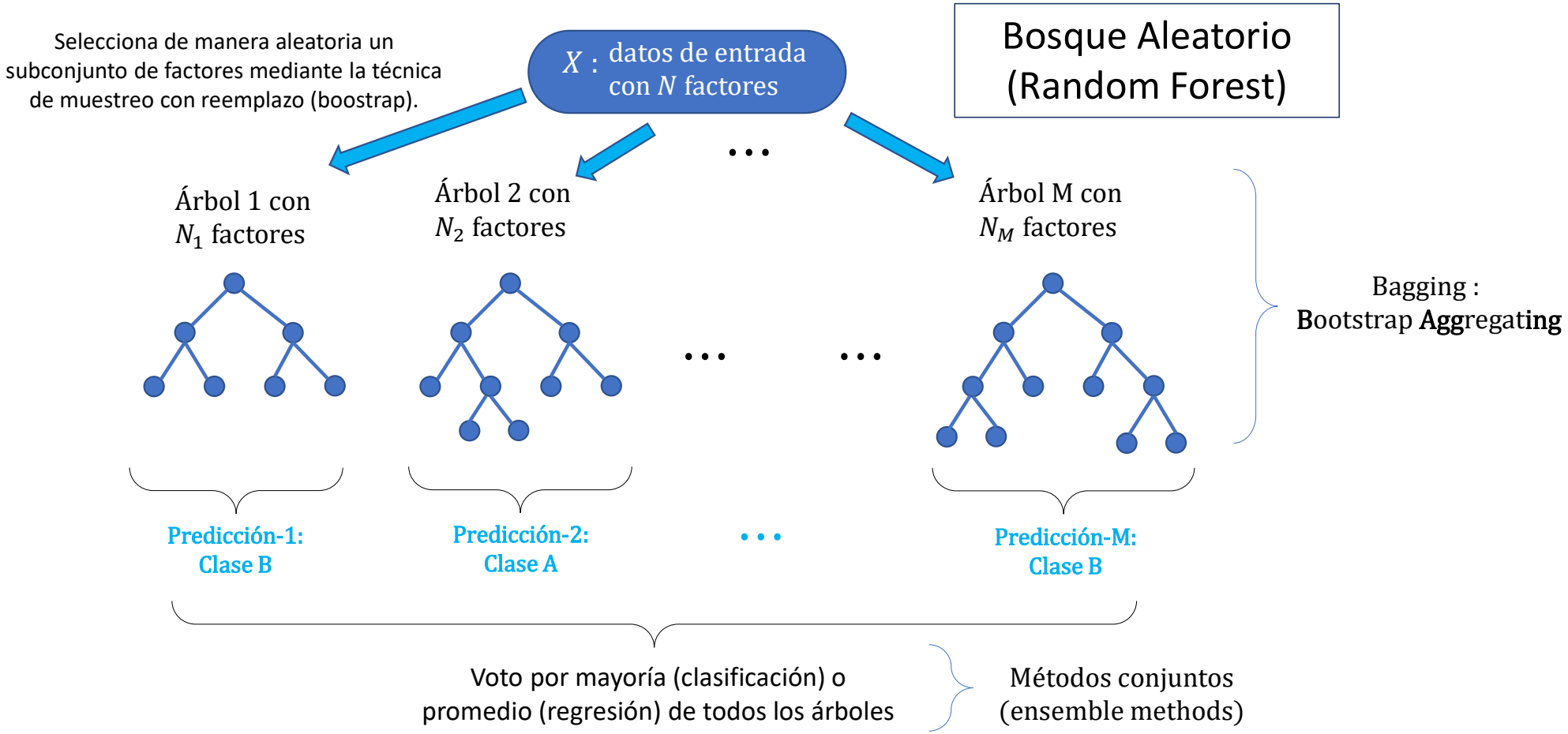
https://link.springer.com/content/pdf/10.1007/BF00058655.pdf

Boosting
(2016)

https://arxiv.org/abs/1603.02754

Tianqi Chen
Carnegie Mellon University
[China]

Bagging : Boostrap Aggregation (1996)



Boosting (Potenciación)

- Dentro de los modelos conjuntos, estos métodos aprenden con base a los errores de cada árbol: ajusta los nuevos árboles con base a los errores que cometieron los previos, a diferencia de la técnica Bagging que cada nuevo árbol se va construyendo de manera independiente.
- De manera general estas técnicas se describen como aquellas que transforman modelos de aprendizaje débil (**weak learners**) en modelos de aprendizaje fuerte (**strong learners**). Modelos de varianza grande generan un modelo más estable de menor varianza.
- El método **AdaBoost** es el más representativo de esta técnica.

Gradient Boosting (Potenciación del Gradiente)

- Al igual que los métodos Boosting, estas técnicas van ajustando los nuevos árboles generados con base a los errores de predicción de los árboles previos.
- La diferencia con Boosting, es que Gradient Boosting construye cada nuevo árbol basándose únicamente en las predicciones erróneas de los árboles previos, sin importar las predicciones correctas que ya se tengan.

XGBoost : eXtreme Gradient Boosting (2016)

https://arxiv.org/abs/1603.02754

Método basado en el modelo de potenciación del gradiente (gradient boosting), con las siguientes características adicionales:

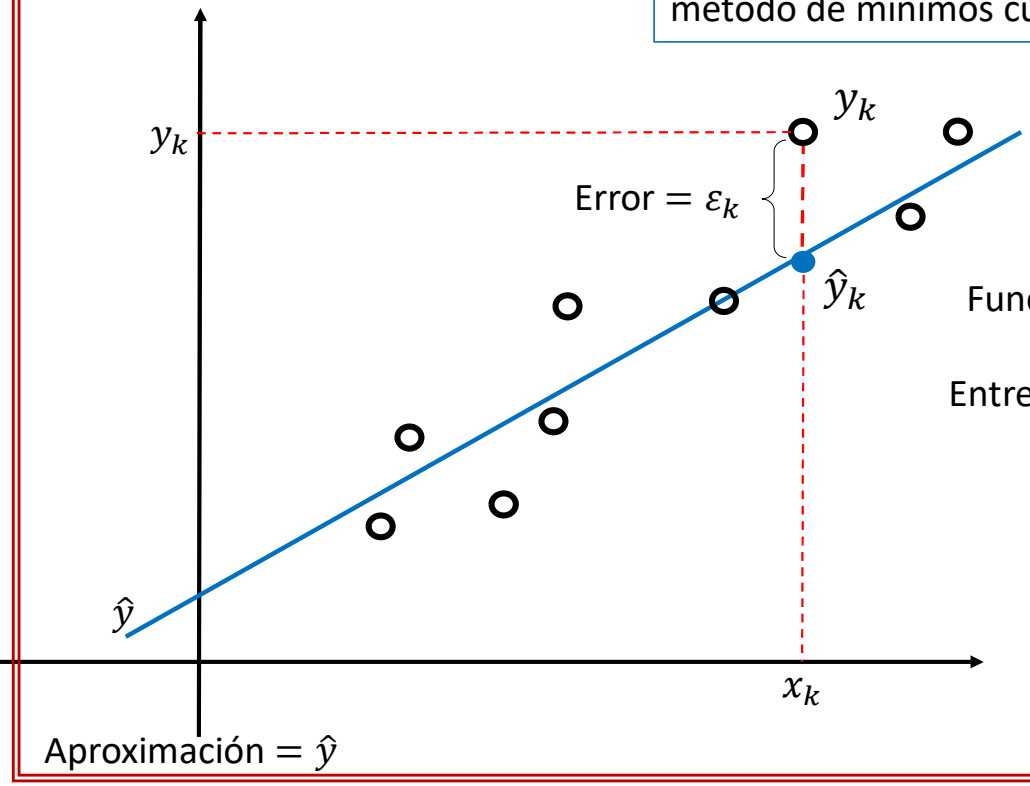
- Incluye un proceso de regularización interno, podemos decir que es un método gradient boosting regularizado.
- Los valores perdidos los maneja internamente: inicialmente les asigna un valor negativo muy grande y al ir particionando de formas diferentes, decide cuál valor asignarle de manera que se tenga la mejor partición.
- Incluye varias técnicas para acelerar el proceso de convergencia y predicción, como matrices dispersas (sparse matrices) y ramificación de árboles mediante el uso de cuantiles.
- Incluye técnicas que aprovechan mejor el uso de la memoria caché.
- Computacionalmente muy costoso, pero aplica algunas técnicas de paralelización cuando es posible.
- Tesis doctoral de Tianqi Chen en la Universidad de Washington.
- Este modelo se hizo muy popular por las competencias de Kaggle.

Relación de XGBoost con el modelo de Regresión Lineal mediante el método de mínimos cuadrados

Modelo propuesto:

$$\hat{y} = a + bx = \sum_j \omega_j x_j$$

Se desan encontrar los valores de los "pesos" o "parámetros" ω_j del modelo durante el proceso "entrenamiento", i.e., mediante el método de mínimos cuadrados.



y_k : valor observado real, dado x_k.

$\hat{y}_k$: predicción del valor observado y_k.

$\hat{y}_k = \hat{a} + \hat{b}x_k$

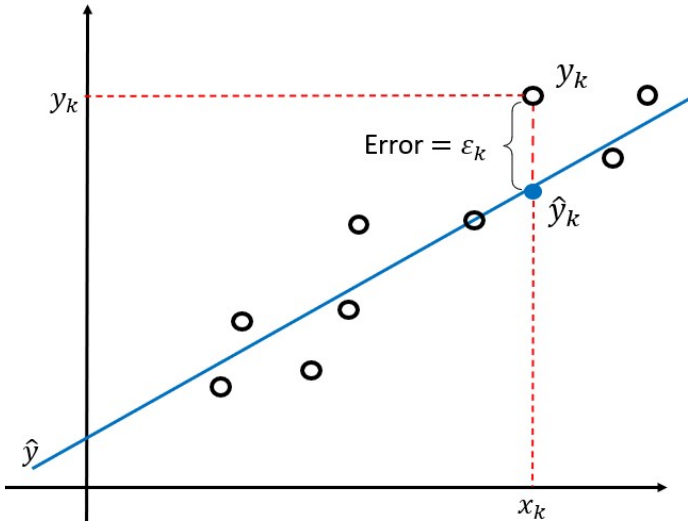
Error o Residuo de cada dato x_k:

$$\varepsilon_k = y_k - \hat{y}_k$$

Función de Costo del proceso de Entrenamiento, $L(\omega)$:

$$SSE = \sum_{k=1}^{n} \varepsilon_k^2 = \sum_{k=1}^{n} (y_k - \hat{y})^2$$

El método de **XGBoost** se basa en seguir construyendo nuevos árboles de decision con base a los residuos o errores de los árboles previos.



El método XGBoost se caracteriza por incluir en la función objetivo del modelo, tanto la **función de costo $L(\omega)$ del proceso de entrenamiento**, como el **término de regularización** $\Omega(\omega)$:

$$obj(\omega) = L(\omega) + \Omega(\omega)$$

Como ya lo comentamos, la función de costo del proceso de entrenamiento para el caso de regression generalmente es la que mide los errores o residuos mediante la suma de sus cuadrados:

$$L(\omega) = \sum_{k=1}^{n} \varepsilon_k^2 = \sum_{k=1}^{n} (y_k - \hat{y})^2$$

donde $\hat{y} = \sum_j \omega_j x_j$. Para el caso de clasificación se utiliza la función de costo de logaritmos.

Y el término de regularización se propone como sigue:

$$\Omega(\omega) = \alpha T + \beta \sum_{j=1}^{T} \omega_{qj}^2$$

donde T es el total de hojas del árbol; ω_{qj} el peso asignado a cada hoja del árbol de acuerdo a la cantidad de datos q que se le hayan asignado.

Como un proceso de minimizació de la función de costo $L(\omega)$, se aplica la técnica estándar de minimización de un curso de Cálculo para obtener los parámetros ω_j del modelo. XGBoost.

Puedes consultar los detalles técnicos en el artículo original donde se publicó el modelo XGBoost:

https://arxiv.org/abs/1603.02754

Igualmente, puedes encontrar más información en la página de la documentación oficial de la librería que implmenta el método XGBoost:

https://xgboost.readthedocs.io/en/latest/tutorials/model.html