Argentina programa 4.0

Universidad Nacional de San Martín

Módulo 3

Aprendizaje automático

Extreme Gradient Boosting | XGBoost

Gradient Boosting

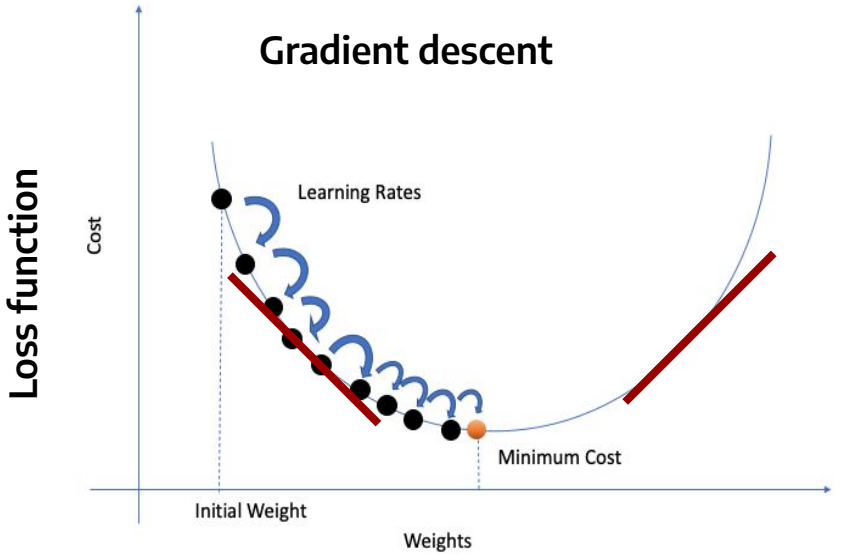
Gradient Boosting = Gradient descent + Boosting

En cada paso se mejora un “weak learner”

En AdaBoost se identifican problemas por su mayor peso

En GradientBoosting se identifican estos problemas por gradientes

- Residual error = diferencia entre el valor real y la predicción
- Gradient = residual error + signo (de la derivada o pendiente)



Gradient Boosting

El uso de gradientes permite:

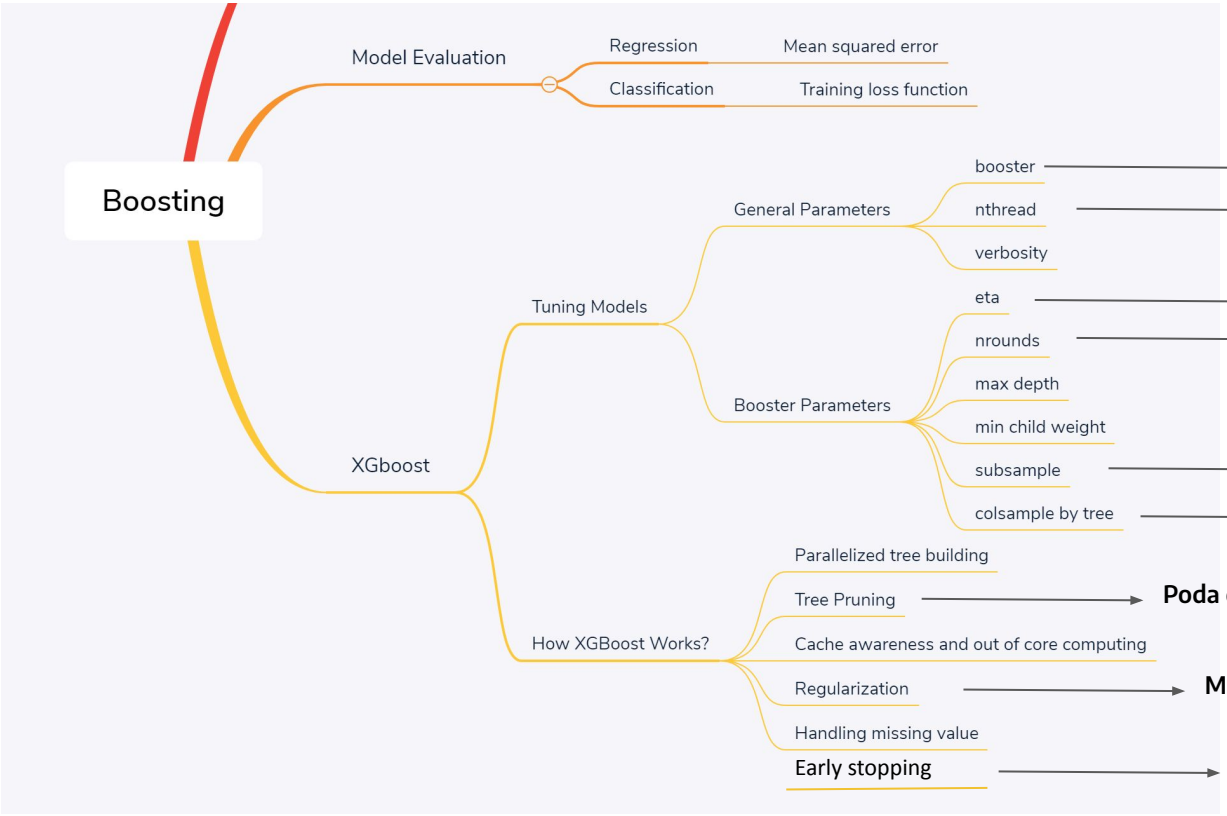
- **Intercambiar distintas funciones de pérdida (loss)**
  - Classifier: **loss*{'log_loss', 'exponential'}, default='log_loss'***
  - Regressor: **loss*{'squared_error', 'absolute_error', 'huber', 'quantile'}, default='squared_error'***

XGBoost

XGBoost = *Extreme Gradient Boosting*

- **Ensamble de modelos**
- **Empieza con un modelo base (base learner)**
  - Generalmente son decision trees (gbtree = gradient boosted tree)
  - Pero también pueden ser modelos lineales (gblinear = gradient boosted linear)
- **Funciona en forma similar a Gradient Boosting**
  - Agrega: uso de la segunda derivada de la función de pérdida
  - Optimizaciones computacionales, regularización para balancear sesgo vs desviación
    - `Estas optimizaciones corren en paralelo!`

XGBoost: afinación, parámetros



Tipo de modelo (learner): tree, categorical, dart, linear

Numero de hilos de procesamiento (CPU)

Velocidad de aprendizaje (learning rate)

n_estimators

muestreo aleatorio (filas)

muestreo aleatorio (columnas)

Poda de arboles (pueden quedar mas cortos que max_depth)

Modelos más simples (menor desvio en predicciones futuras)

Si el entrenamiento empeora, se detiene!

XGboost: objetivo y balance de sesgo vs desvio

Por qué queremos dos componentes en el objetivo?

- **Optimizar para reducir la pérdida (Training Loss) nos garantiza obtener modelos *predictivos* que ajustan bien a la distribución de base**
- **Optimizar regularización garantiza obtener modelos *más simples*, que en general tienden a tener menos desvío en predicciones futuras**
  - Modelos más estables!

$$Obj(\Theta) = L(\Theta) + \Omega(\Theta)$$

Training Loss measures how well model fit on training data

Regularization, measures complexity of model

XGboost: sub-muestreo

XGBoost puede sub-muestrear, x filas o x columnas