CIMAT

Centro de Investigación en Matemáticas, A.C.

GRADIENT BOOSTING

Alondra Elizabeth Matos Mendoza
María Guadalupe López Salomón

12 de junio de 2023

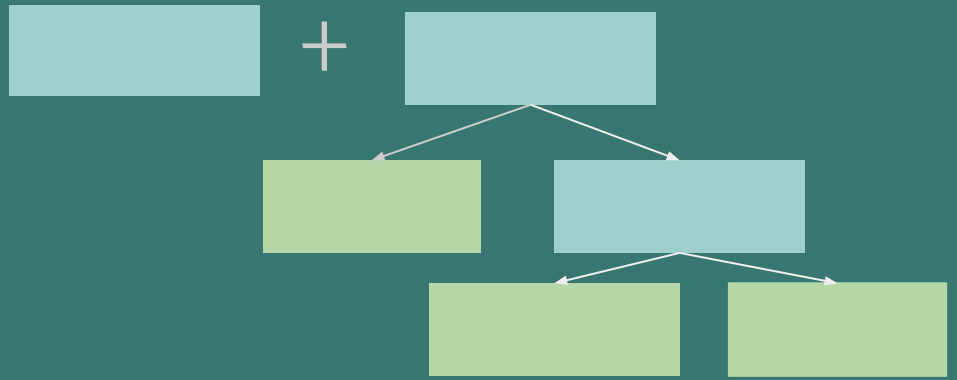


TABLA DE CONTENIDO

Introducción

- En aplicaciones de minería de datos industriales y comerciales, los conjuntos de datos suelen ser grandes y complejos, con diversas variables y observaciones. Estos datos pueden estar desordenados y tener valores faltantes, distribuciones asimétricas, errores de medición y valores atípicos.

  .

- Bajo esta problemática, se utilizan árboles de decisión que son rápidos, interpretables y robustos. Sin embargo, su precisión puede no ser óptima.

- Para abordar estos desafíos, se desarrollaron los modelos basados en Gradient Boosting.

Boosting trees

- Método utilizado en árboles de regresión y clasificación, en el que el espacio de valores de las variables predictoras se divide en regiones representadas por los nodos terminales del árbol.

- La regla predictiva es:

$$x \in R_j \rightarrow f(x) = \gamma_j$$

- Un árbol se representa formalmente como una suma de regiones y constantes

$$T(x;\Theta) = \sum_{j=1}^{J} \gamma_j I\left(x \in R_j\right)$$

- Los parámetros se determinan minimizando:

$$\hat{\Theta} = arg \min_{\Theta} \sum_{j=1}^{J} \sum_{x \in R_j} L(y_i, \gamma_j)$$

Gradient Boosting

- En Gradient Boosting, las deficiencias del modelo se identifican mediante el análisis de los gradientes que representan la **dirección** y la **magnitud del error** o **residual** para cada observación.

- El algoritmo comienza con un único nodo terminal en lugar de un árbol completo, y se inicializa con un valor constante que minimiza el costo total. Luego, se establece el número total de árboles a construir y para cada árbol, se realizan los siguientes pasos:

Gradient Boosting

Algorithm 1 Gradient Tree Boosting

1: **Entrada** Datos $\{(x_i, y_i)\}_{i=1}^{n}$ y una función de costo $L(y_i, f(x))$diferenciable.
2: Inicializar $f_0(x) = argmin_\gamma \sum_{i=1}^{N} L(y_i, \gamma)$
3: **for** $m = 1 \ : \ M$ **do**
4: **for** $i = 1, 2, ..., N$ **do**
5: Calcular $r_{im} = -\left[\frac{\partial L(y_i, f(x_i))}{\partial f(x_i)}\right]_{f=f_{m-1}}$
6: **end for**
7: Ajustar un árbol de regresión a los valores r_{im} y crear las regiones terminales R_{jm} $j = 1, 2, ..., J_m$
8: **for** $j = 1, 2, ..., J_m$ **do**
9: Calcular $\gamma_{jm} = arg \ \ \text{mín}_\gamma \sum_{x_i \in R_{jm}} L\left(y_i, f_{m-1}(x_i) + \gamma\right)$
10: **end for**
11: Actualizar $f_m(x) = f_{m-1}(x) + \nu \sum_{j=1}^{J_m} \gamma_{jm} I\left(x \in R_{jm}\right)$
12: **end for**
13: **Salida** $\hat{f}(x) = f_M(x)$

Cálculo de los pseudo residuales:

$$r_{im} = -\left[\frac{\partial L(y_i, f(x_i))}{\partial f(x_i)}\right]$$

Altura del padre	Altura de la madre	Sexo del hijo	Altura del hijo
160	140	M	160
210	160	M	200
140	160	F	160
180	150	M	180
200	160	F	190
160	150	M	170

$$L(y_i, f(x_i)) = \frac{1}{2}(y_i - f(x_i))^2$$

$$-\left[\frac{\partial L(y_i, f(x_i))}{\partial f(x_i)}\right] = -(y_i - f(x_i))$$

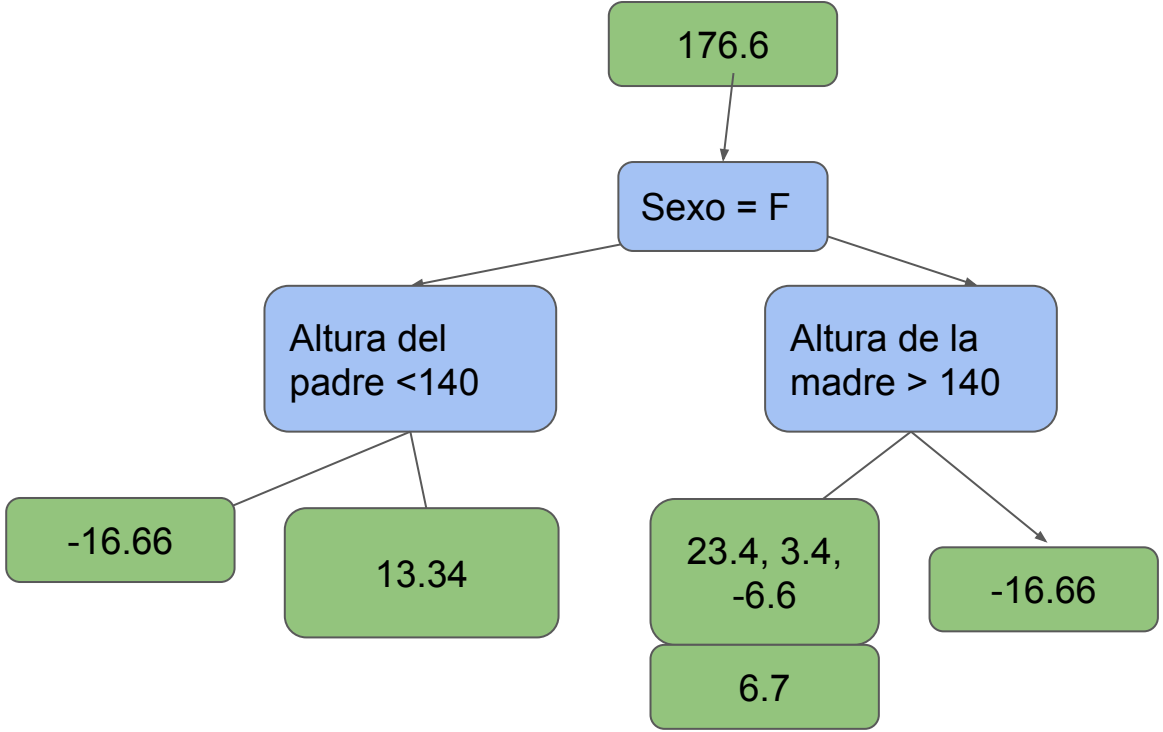
Predicción inicial = f(x) = 176.6

Residuales
-16.6
23.4
-16.6
3.4
13.4
-6.6

Ajustamos un árbol para predecir los residuales

Altura del padre, Altura de la madre y Sexo del hijo

Solo consideramos 4 hojas!!



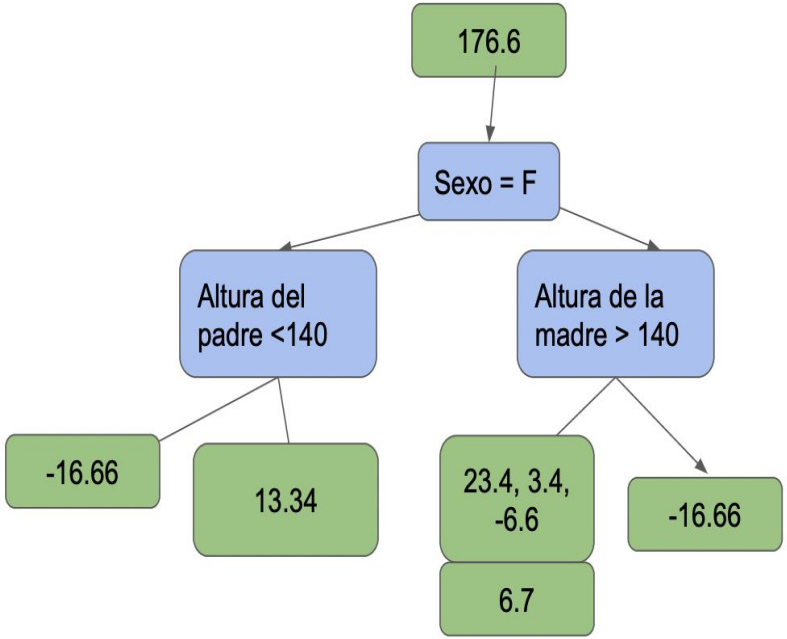
Altura del padre	Altura de la madre	Sexo del hijo	Altura del hijo	Residuales
160	140	M	160	-16.6
210	160	M	200	23.4
140	160	F	160	-16.6
180	150	M	180	3.4
200	160	F	190	13.4
160	150	M	170	-6.6

Para obtener los nuevos residuales:

Altura promedio

176.6 + Learning rate ✖

[0,1]



Y así sucesivamente...

GRADIENT BOOSTING
CLASIFICACIÓN

La función de costo para clasificación binaria es el negativo de la log-verosimilitud de la binomial (desviación):

$$L(y, p(x)) = -\sum_{i=1}^{N} y_i \log p(x_i) + (1 - y_i) \log(1 - p(x_i))$$

$$L(y, f(x)) = \sum_{i=1}^{N} -y_i f(x) + \log(1 + e^{f(x)})$$

log(odds): logaritmo de las razones de probabilidad

$$f(x_i) = \log(odds) = \log \frac{p(x_i)}{(1-p(x_i))} \longrightarrow p(x_i) = \frac{e^{log(odds)}}{1 + e^{log(odds)}}$$

GRADIENT BOOSTING
CLASIFICACIÓN

Se inicia con una hoja que representa la predicción inicial: log(odds)

log(4/2) = 0.693

Se obtiene la probabilidad mediante la función logística

Probabilidad de sufrir cardiopatía $= \frac{e^{log(odds)}}{1 + e^{log(odds)}} = \frac{e^{log(4/2)}}{1 + e^{log(4/2)}} = 0.666$

Edad	Anemia	Creatina f.	Cardiopatía
75	0	582	1
62	0	61	0
55	0	7861	1
65	0	146	1
50	0	196	0
50	1	111	1

GRADIENT BOOSTING
CLASIFICACIÓN

Para determinar la calidad de la predicción,

Se calculan los pseudo residuales:

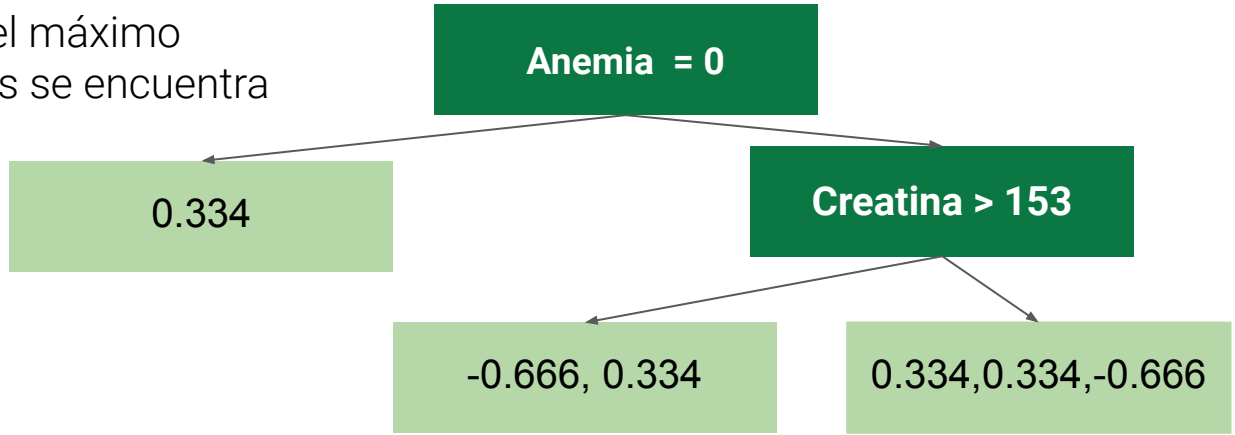
$$r_{im} = -\left[\frac{\partial L(y_i, f(x_i))}{\partial f(x_i)}\right]_{f=f_{m-1}}$$
$$= y_i - \frac{e^{f(x_i)}}{1 + e^{f(x_i)}}$$
$$= y_i - p(x_i)$$

Considerando *p = 0.666,*

Edad	Anemia	Creatina f.	Cardiopatía	Residuo
75	0	582	1	0.334
62	0	61	0	-0.666
55	0	7861	1	0.334
65	0	146	1	0.334
50	0	196	0	-0.666
50	1	111	1	0.334

Se ajusta un árbol para predecir los pseudo residuales

Generalmente, el máximo número de hojas se encuentra entre 8 y 32



Edad	Anemia	Creatina f.	Cardiopatía	Residuo
75	0	582	1	0.334
62	0	61	0	-0.666
55	0	7861	1	0.334
65	0	146	1	0.334
50	0	196	0	-0.666
50	1	111	1	0.334

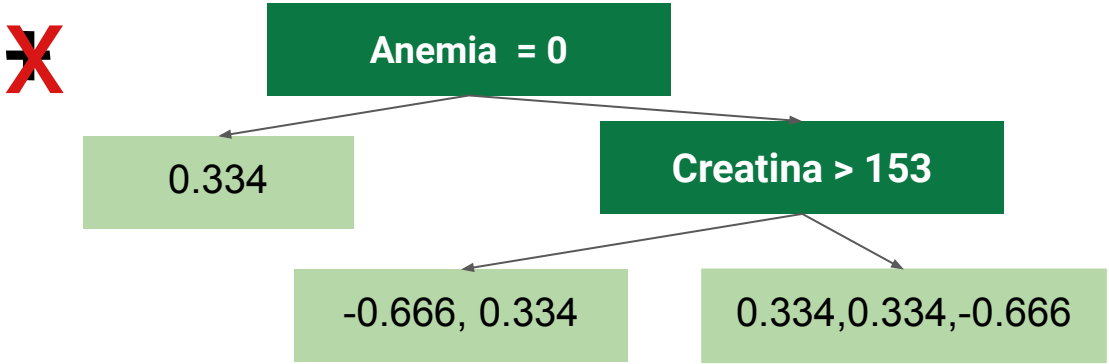
GRADIENT BOOSTING
CLASIFICACIÓN

GRADIENT BOOSTING

CLASIFICACIÓN

Se ajusta un árbol para predecir los pseudo residuales

log(4/2) = 0.693



¿Cuáles son los valores de salida?

$$\gamma_{jm} = arg \min_{\gamma} \sum_{x_i \in R_{jm}} L\left(y_i, f_{m-1}(x_i) + \gamma\right)$$

Se obtienen los valores de salida

La función de costo se puede aproximar mediante el polinomio de Taylor de segundo orden:

$$\sum_{x_i \in R_{jm}} L\left(y_i, f_{m-1}(x_i) + \gamma\right) \approx \sum_{x_i \in R_{jm}} L\left(y_i, f_{m-1}(x_i)\right) + \left[\sum_{x_i \in R_{jm}} \frac{\partial}{\partial f()} L\left(y_i, f_{m-1}(x_i)\right)\right] \gamma + \frac{1}{2}\left[\sum_{x_i \in R_{jm}} \frac{\partial^2}{\partial f()^2} L\left(y_i, f_{m-1}(x_i)\right)\right] \gamma^2$$

Igualando la derivada del costo y resolviendo para gamma:

$$\hat{\gamma}_{jm} = \frac{\sum_{x_i \in R_{jm}} y_i - p(x_i)}{\sum_{x_i \in R_{jm}} p(x_i)(1 - p(x_i))}$$

GRADIENT BOOSTING

CLASIFICACIÓN

Se calculan los valores de salida y se actualiza el valor de la predicción

log(4/2) = 0.693 + learning rate X **Anemia = 0**

- 0.334
- **Creatina > 153**
  - -0.666, 0.334
  - 0.334,0.334,-0.666

$$\frac{\sum residual}{\sum p_i(1-p_i)} = \frac{0.334}{0.666(0.334)} = 1.50$$

$$\frac{\sum residual}{\sum p_i(1-p_i)} = \frac{-0.666 + 0.334}{0.666(0.334) + 0.666(0.334)} = -0.746$$

500/166833

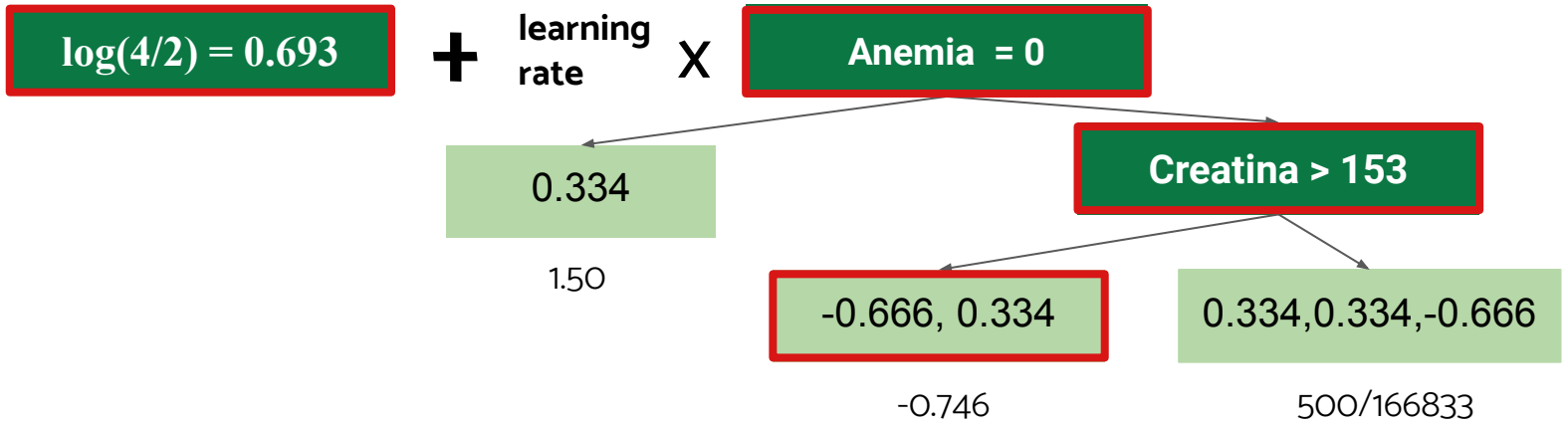
Inicialmente, las probabilidades para cada residuo son iguales, pero estas cambian conforme se añaden árboles.

GRADIENT BOOSTING
CLASIFICACIÓN

GRADIENT BOOSTING

CLASIFICACIÓN

Con las nuevas predicciones, se calculan los residuos y se repite el proceso.



Edad	Anemia	Creatina f.	Cardiopatía	Residuo
75	0	582	1	0.350
62	0	61	0	
55	0	7861	1	
65	0	146	1	
50	0	196	0	
50	1	111	1	

Sea learning rate = 0.1, la nueva predicción es:

$$f(x_1) = log(odds) = 0.693 + 0.1(-0.746) = 0.618$$

$$p(x_1) = \frac{e^{f(x)}}{1 + e^{f(x)}} = \frac{e^{0.618}}{1 + e^{0.618}} = 0.649$$

EXTREME GRADIENT BOOSTING

Sean:

- $T(x;\Theta_m)$: m-ésimo árbol de regresión con parámetros $\Theta_m = \{R_{jm}, \gamma_{jm}\}_1^J$
- $|T_m|$: Número de nodos terminales (hojas) en el m-ésimo árbol.
- γ_m: Valores de salida (pesos/scores) asociados a la hojas del m-ésimo árbol.

El objetivo del XGBoost es encontrar los valores de salida de las hojas que minimicen la siguiente función objetivo regularizada:

$$\mathbf{L}(f_M) = \left[\sum_{i=1}^{n} L(y_i, f(x))\right] + \sum_{m=1}^{M} \Omega(T(x;\Theta_m))$$

$$donde\ \Omega(T(x;\Theta_m)) = \alpha|T_m| + \frac{1}{2}\lambda||\gamma_m||^2$$

EXTREME GRADIENT BOOSTING

Algorithm 1 Gradient Tree Boosting

Entrada Datos $\{(x_i, y_i)\}_{i=1}^{n}$ y una función de costo $L(y_i, f(x))$diferenciable.
Inicializar $f_0(x) = argmin_\gamma \sum_{i=1}^{N} L(y_i, \gamma)$
for $m = 1 : M$ **do**
  for $i = 1, 2, ..., N$ **do**
    Calcular $r_{im} = -\left[\frac{\partial L(y_i, f(x_i))}{\partial f(x_i)}\right]_{f=f_{m-1}}$
  end for
  Ajustar un árbol de regresión a los valores r_{im} y crear las regiones terminales R_{jm} $j = 1, 2, ..., J_m$
  for $j = 1, 2, ..., J_m$ **do**
    Calcular $\gamma_{jm} = arg \ \min_\gamma \sum_{x_i \in R_{jm}} L\left(y_i, f_{m-1}(x_i) + \gamma\right)$
  end for
  Actualizar $f_m(x) = f_{m-1}(x) + \nu \sum_{j=1}^{J_m} \gamma_{jm} I\left(x \in R_{jm}\right)$
end for
Salida $\hat{f}(x) = f_M(x)$

Los valores de salida se encuentran minizando:

$$\mathbf{L}^{(m)} = \sum_{j=1}^{J_m} \sum_{x_i \in R_{jm}} L(y_i, f_{m-1}(x_i) + \gamma_{jm}) + \alpha |T_m| + \frac{1}{2}\lambda \sum_{j=1}^{J_m} \gamma_{jm}^2$$

EXTREME GRADIENT BOOSTING

En comparación al Gradient Boost, XGBoost usa la aproximación mediante el polinomio tanto para regresión como clasificación.

$$\mathbf{L}^{(m)} \approx \sum_{j=1}^{J_m} \sum_{x_i \in R_{jm}} L\left(y_i, f_{m-1}(x_i)\right) + \sum_{x_i \in R_{jm}} \frac{\partial}{\partial f(x_i)} L\left(y_i, f_{m-1}(x_i)\right) \gamma_{jm} + \sum_{x_i \in R_{jm}} \frac{1}{2} \frac{\partial^2}{\partial f(x_i)^2} L\left(y_i, f_{m-1}(x_i)\right) \gamma_{jm}^2 + \alpha |T_m| + \frac{1}{2} \lambda \sum_{j=1}^{J_m} \gamma_{jm}^2$$

Si se denota el gradiente como *g* y la matriz hessiana como *h*, y se eliminan los términos constantes que no tienen efecto en los valores de salida, la función objetivo se simplifica a:

$$\bar{\mathbf{L}}^{(m)} = \sum_{j=1}^{J_m} \left[\left(\sum_{x_i \in R_{jm}} g_i \right) \gamma_{jm} + \frac{1}{2} \left(\sum_{x_i \in R_{jm}} h_i + \lambda \right) \gamma_{jm}^2 \right] + \alpha |T_m|$$

EXTREME GRADIENT BOOSTING

Derivando el costo $\bar{\mathbf{L}}^{(m)} = \sum_{j=1}^{J_m} \left[\left(\sum_{x_i \in R_{jm}} g_i \right) \gamma_{jm} + \frac{1}{2} \left(\sum_{x_i \in R_{jm}} h_i + \lambda \right) \gamma_{jm}^2 \right] + \alpha |T_m|$

Igualando la derivada a 0 y resolviendo para gamma, se obtiene que la constante óptima para la j-ésima región es:

$$\frac{\partial}{\partial \gamma_{jm}} \bar{\mathbf{L}}^{(m)} = \sum_{x_i \in R_{jm}} g_i + \left(\sum_{x_i \in R_{jm}} h_i + \lambda \right) \gamma_{jm} \longrightarrow \gamma_{jm} = -\frac{\sum_{x_i \in R_{jm}} g_i}{\sum_{x_i \in R_{jm}} h_i + \lambda}$$

El valor del costo total del m-ésimo árbol con los valores de salida óptimos es:

$$\bar{\mathbf{L}}_{similaridad}^{(m)} = -\frac{1}{2} \sum_{j=1}^{J_m} \frac{\left(\sum_{x_i \in R_{jm}} g_i \right)^2}{\sum_{x_i \in R_{jm}} h_i + \lambda} + \alpha |T|$$

EXTREME GRADIENT BOOSTING

Para construir el árbol, se emplea un algoritmo voraz que parte de una sola hoja y añade ramas al árbol de manera iterativa.

Considerando *I_L* e L_R como los conjuntos de instancias correspondientes a los nodos izquierdo y derecho después de la partición, la ganancia (reducción de costo) después del split viene dado por:

$$
\begin{aligned}
Gain_{split} &= left_{similarity} + right_{similarity} - root_{similarity} \\
&= \frac{1}{2}\left[\frac{\left(\sum_{x_i \in I_L} g_i\right)^2}{\sum_{x_i \in I_L} h_i + \lambda} + \frac{\left(\sum_{x_i \in I_R} g_i\right)^2}{\sum_{x_i \in I_R} h_i + \lambda}\right] \\
&- \frac{1}{2}\frac{\left(\sum_{x_i \in I} g_i\right)^2}{\sum_{x_i \in I} h_i + \lambda} - \alpha
\end{aligned}
$$

Algorithm 2 Algoritmo voraz exacto para la búsqueda de splits

Entrada: I, conjunto de instancias del nodo actual,
Entrada: d, número de variables (características/columnas)
$gain \leftarrow 0$
$G \leftarrow \sum_{i \in I} g_i$, $H \leftarrow \sum_{i \in I} h_i$
for $k = 1 \ : \ K$ **do**
  $G_L \leftarrow 0$, $H_L \leftarrow 0$
  for $j \in (I, ordenado\ por\ x_{jk})$ **do**
    $G_L \leftarrow G_L + g_j$, $H_L \leftarrow H_L + h_j$
    $G_R \leftarrow G - G_L$, $H_R \leftarrow H - H_L$
    $score \leftarrow \max\left(score, \frac{G_L^2}{H_L+\lambda} + \frac{G_R^2}{H_R+\lambda} - \frac{G^2}{H+\lambda}\right)$
  end for
end for
Salida: Split con el máximo score de ganancia

Para determinar la división óptima:

Weighted Quantile Sketch

Algorithm 3 Algoritmo aproximado para la búsqueda de splits

for $k = 1 \ : \ K$ **do**
  Proponer $S_k = \{s_{k1}, s_{k2}, ..., s_{kl}\}$, donde S_k es el conjunto de percentiles de la variable k
  El conjunto S_k puede proponerse por árbol (global) o por split (local)
end for
for $k = 1 \ : \ K$ **do**
  $G_{kv} \leftarrow \sum_{j \in \{j | s_{k,v} \geq x_{jk} > s_{k,v-1}\}} g_i$
  $H_{kv} \leftarrow \sum_{j \in \{j | s_{k,v} \geq x_{jk} > s_{k,v-1}\}} h_i$
end for
Seguir el mismo paso que en la sección anterior para encontrar max puntuar solo entre los splits propuestas

Sparsity-aware Split Finding

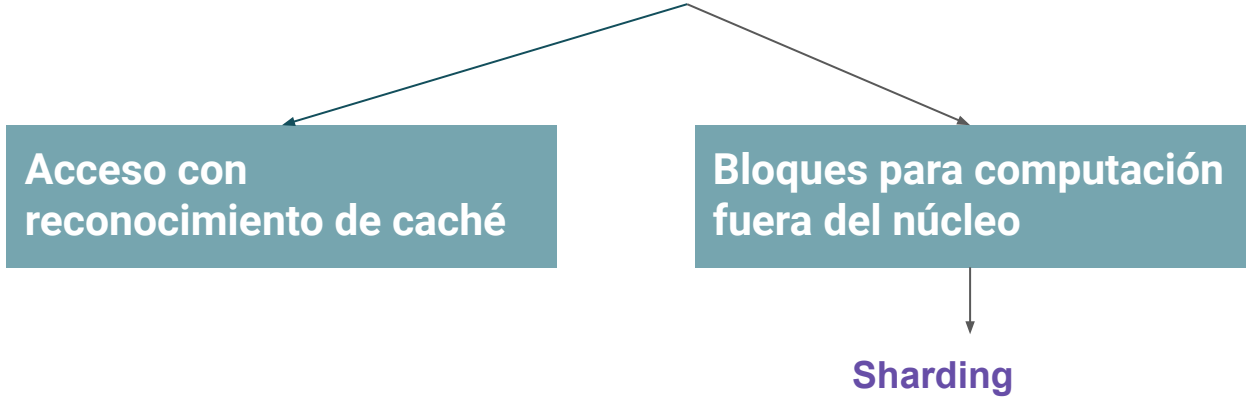
Cuando hay datos faltantes…

Algorithm 4 Sparsity-aware Split Finding

Entrada: I, conjunto de instancias del nodo actual
Entrada: $I_k = \{i \in I | x_{ik} \neq missing\}$
Entrada: d, número de variables (características/columnas)
$gain \leftarrow 0$
$G \leftarrow \sum_{i \in I} g_i$, $H \leftarrow \sum_{i \in I} h_i$
for $k = 1 : K$ **do**
  Los valores faltantes se clasifican a la derecha.
  $G_L \leftarrow 0$, $H_L \leftarrow 0$
  for $j \in (I_k, ordenado\ ascendentemente\ por\ x_{jk})$ **do**
    $G_L \leftarrow G_L + g_j$, $H_L \leftarrow H_L + h_j$
    $G_R \leftarrow G - G_L$, $H_R \leftarrow H - H_L$
    $score \leftarrow \max\left(score, \frac{G_L^2}{H_L+\lambda} + \frac{G_R^2}{H_R+\lambda} - \frac{G^2}{H+\lambda}\right)$
  end for
  Los valores faltantes se clasifican a la izquierda.
  $G_R \leftarrow 0$, $H_R \leftarrow 0$
  for $j \in (I_k, ordenado\ descendentemente\ por\ x_{jk})$ **do**
    $G_R \leftarrow G_R + g_j$, $H_R \leftarrow H_R + h_j$
    $G_L \leftarrow G - G_R$, $H_L \leftarrow H - H_R$
    $score \leftarrow \max\left(score, \frac{G_L^2}{H_L+\lambda} + \frac{G_R^2}{H_R+\lambda} - \frac{G^2}{H+\lambda}\right)$
  end for
end for
Salida: Split y dirección predeterminada con el máximo score de ganancia

Diseño computacional

El XGBoost utiliza las siguientes técnicas de programación para poder optimizar tomando en cuenta el hardware del ordenador:



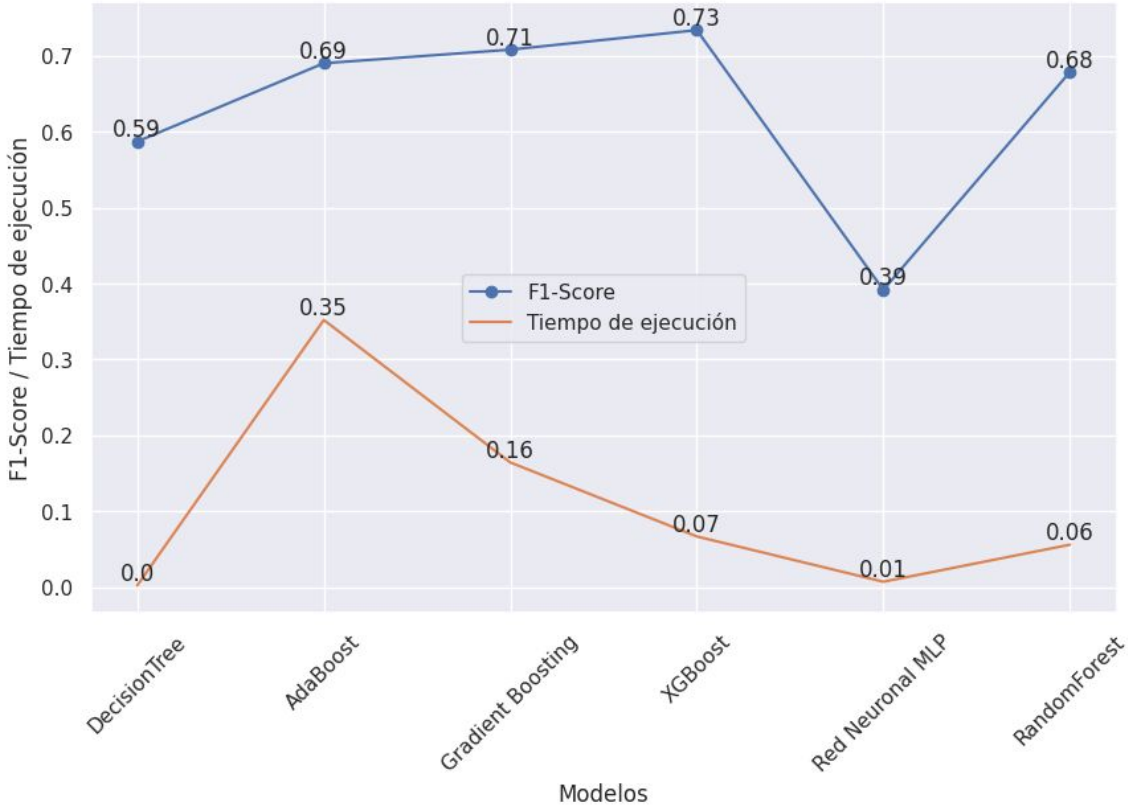
Aplicación: Heart failure clinical records

APLICACIÓN

Se consideró la base de datos correspondiente a registros médicos de 299 pacientes que sufrieron falla cardiaca, recolectados durante su seguimiento médico, en donde cada perfil de paciente tiene 13 características clínicas.

- Edad
- Anemia
- Creatina cinasa
- Diabetes
- Fracción de eyección
- Plaquetas
- Sexo
- Creatinina sérica
- Fumador
- Periodo de seguimiento en días
- Fallecimiento

Rendimiento y tiempo de ejecución de GBoosting y XGBoost vs otros métodos de clasificación



Modelos de clasificación implementados:

- Decision Tree Classifier
- AdaBoost Classifier
- Gradient Boosting Classifier
- XGBoost
- Random Forest
- MLP Classifier

Thank you!