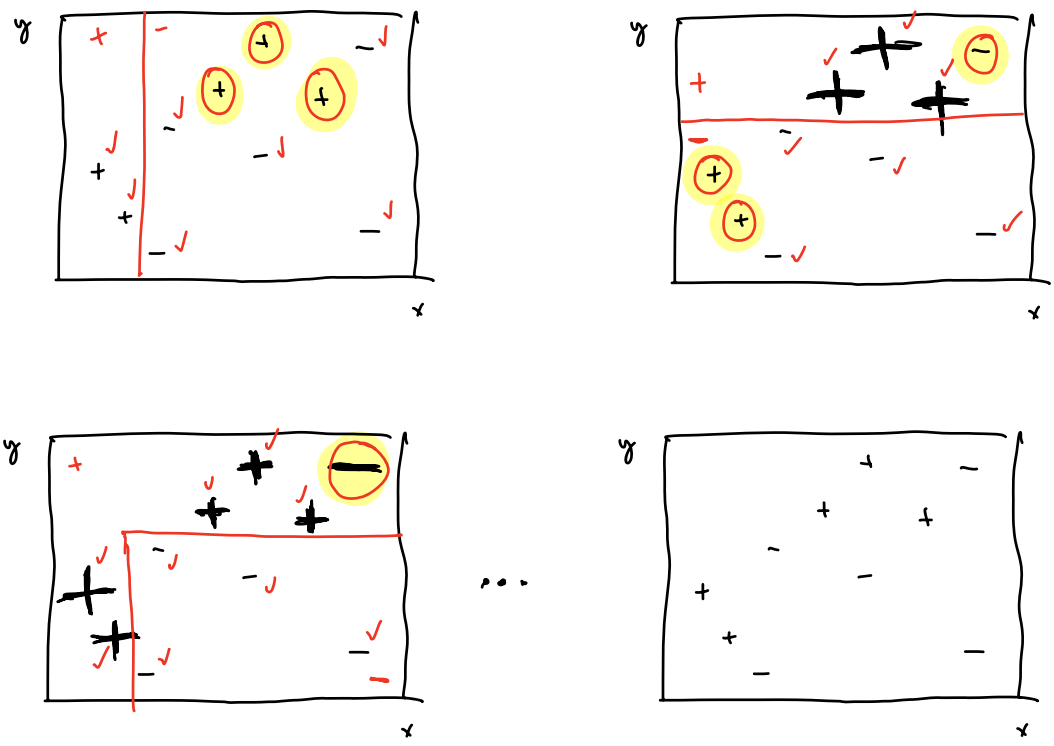
Adaboost: en cada iteración construimos un árbol de clasificación y le damos más peso a las observaciones en las que el algoritmo se equivocó en la iteración pasada:



Tenemos: $(x_1, y_1), (x_2, y_2) \dots (x_n, y_n)$, x_i vector de p dimensiones

↳ construir modelo $g^{[1]}(\cdot)$ para predecir y_i:

$(x_1, y_1), \dots (x_n, y_n) \longrightarrow g^{[1]}(\cdot)$ árbol

$\Downarrow$

actualizamos los pesos de cada observación

$(x_1, y_2), \dots (x_n, y_n) \longrightarrow g^{[2]}(\cdot)$

$\vdots$

$$\hat{f}_{adaboost}(x) = \sum_{m=1}^{M} \alpha_m g^{[m]}(x)$$

→ predicción final del adaboost es votación ponderada de cada uno de los árboles

α_m : peso de cada árbol

Inicializamos los pesos: $w_i^{[1]} = \frac{1}{n}$, $i = 1, \dots, n$.

Para $m = 1, \dots, M$:

- construir un árbol de decisión utilizando los pesos $w_i^{[m-1]}$ para obtener un modelo $g^{[m]}$.

$$error^{[m]} = \frac{\sum_{i=1}^{n} w_i^{[m-1]} \mathbb{I}\{y_i \neq g_i^{[m]}(x_i)\}}{\sum_{i=1}^{n} w_i^{[m-1]}}$$

$$\alpha^{[m]} = \log\left(\frac{1 - error^{[m]}}{error^{[m]}}\right)$$

$$\tilde{w}_i^{[m]} = w_i^{[m-1]} \exp\left[\alpha^{[m]} \mathbb{I}\{y_i \neq g^{[m]}(x_i)\}\right]$$

$$w_i^{[m]} = \frac{\tilde{w}_i^{[m]}}{\sum_{j=1}^{n} \tilde{w}_j^{[m]}}$$

Xgboost: $(x_1, y_1) \dots (x_n, y_n)$ → construyo un modelo $f(x)$ para predecir.

$y_1 = 0.9$ $\quad f(x_1) = 0.8$ equivocó en 0.1

$y_2 = 1.4$ $\quad f(x_2) = 1.3$ equivocó en 0.1

$\vdots$ $\quad \vdots$

Quisiera construir un segundo modelo que me ayude a "corregir" esos errores:

$f(x_1) + h(x_1) = y_1$ "h" completara a "f"

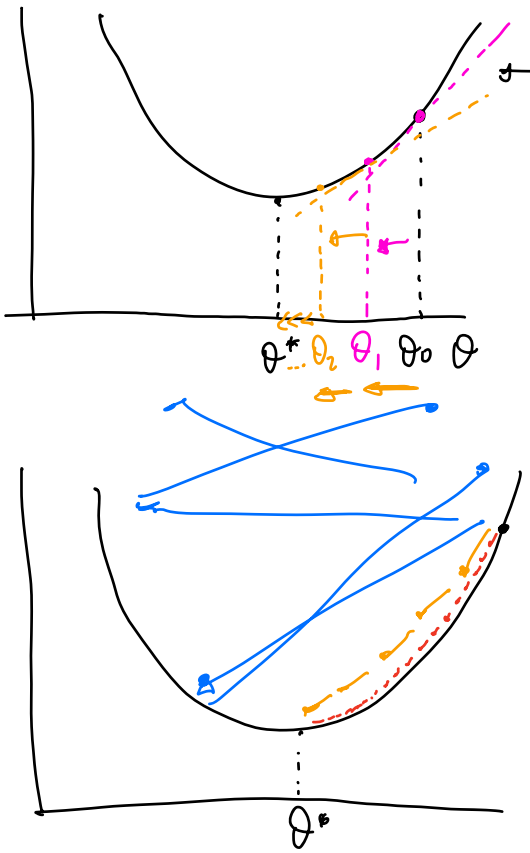
$f(x_2) + h(x_2) = y_2$

$\vdots$

$$h(x_1) = y_1 - f(x_1)$$
$$h(x_2) = y_2 - f(x_2)$$

→ quiero que h me ayude a predecir los errores del modelo f.

Algoritmos de gradient descent:



← función de pérdida.

learning rate.

$$\theta_{i+1} = \theta_i - \rho \frac{\partial J}{\partial \theta}$$

derivada de la función de pérdida con respecto al parámetro.

Si ρ es baja, necesito muchas iteraciones para que el algoritmo converja.

Si ρ es más alto, necesito menos iteraciones para converger

Si ρ es muy alto, puede que no converja.

Función de pérdida cuadrática:

$$L(y, f(x)) = \sum_{i=1}^{n} \frac{(y_i - f(x_i))^2}{2}$$

$$\frac{\partial L(y, f(x))}{\partial f(x_i)} = (y_i - f(x_i))$$ — error del modelo en la observación i.

$$f_2(x_i) = f(x_i) + h(x_i)$$ ~ h aproxima $(y_i - f(x_i))$ error del modelo f.

$$f_2(x_i) = f(x_i) + (y_i - f(x_i))$$

$$= f(x_i) - \underbrace{(f(x_i) - y_i)}_{\frac{\partial L(y, f(x))}{\partial f(x_i)}}$$

$$f_2(x_i) = f(x_i) - 1 \cdot \frac{\partial L(y, f(x))}{\partial f(x_i)}$$ learning rate = 1.