XGBoost : Extreme Gradient Boosting

Chaymae Gasmi

1 Definition

XGBoost est une bibliothèque optimisée de renforcement de gradient distribuée conçue pour être très efficace, flexible et portable. Il implémente des algorithmes d'apprentissage automatique dans le cadre de Gradient Boosting. XGBoost fournit un boost d'arborescence parallèle (également connu sous le nom de GBDT, GBM) qui résout de nombreux problèmes de science des données de manière rapide et précise. XGBoost est une méthode d'apprentissage d'ensemble. Parfois, il ne suffit pas de se fier aux résultats d'un seul modèle d'apprentissage automatique. L'apprentissage d'ensemble offre une solution systématique pour combiner le pouvoir prédictif de plusieurs apprenants. Le résultat est un modèle unique qui donne la sortie agrégée de plusieurs modèles.

Avant d'exposer plus en détails le fonctionnement de l'algorithme, il est sans doute utile d'effectuer quelques rappels concernant l'apprentissage supervisé.

Modèle et paramètres

Un modèle en apprentissage supervisé fait souvent référence à la structure mathématique qui permet d'effectuer une prédiction $\hat{y}_i$ à partir d'un élément xi, pour i [0; n] et n le nombre d'individus de la base de données. Ainsi, un modèle courant est le modèle linéaire où la prédiction est donnée par :

$\hat{y}_i = \sum_j \hat{\theta}_j \times x_{i,j}$ une combinaison linéaire pondérée des j variables explicatives.Les paramètres sont les inconnus que l'on cherche à estimer à partir des données. Dans le cas d'une régression linéaire par exemple, les paramètres sont les coefficients .

La fonction objectif : fonction de perte et terme de régularisation

Quel que soit le problème que l'on souhaite résoudre à l'aide d'un modèle, on a toujours besoin d'un moyen de trouver une manière d'estimer les meilleurs paramètres pour le modèle. Pour parvenir à cela, il est nécessaire de faire appel à une fonction objectif. Cette fonction permet en effet de mesurer la performance d'un modèle en fonction d'un ensemble de paramètres.

Une fonction objectif peut être définie de la manière que l'on souhaite, pour autant, pour qu'elle soit pertinente, elle doit se composer d'une fonction de perte et d'un terme de régularisation : Obj(·) = L(·) + K(·) où Obj est la fonction objectif composée par L, une fonction de perte et K un terme de régularisation. La fonction de perte mesure la qualité de la prédiction du modèle sur les données fournies en apprentissage. Assez communément, pour un jeu de paramètres , la fonction de perte est l'erreur quadratique moyenne :

$\mathrm{L}(\theta) = \frac{1}{n} \sum_i (y_i - \hat{y}_i)^2$

Le terme de régularisation permet de contrôler la complexité du modèle et d'éviter le surapprentissage. Ce terme n'a pas de forme prédéfinie à priori et est souvent défini en fonction du problème. L'idée générale est qu'il doit permettre d'avoir un modèle simple et performant à la fois.

Le modèle sous-jacent à l'algorithme XGBoost est l'agrégation d'arbres (ou boosting d'arbres).

L'agrégation d'arbres est un modèle composé d'arbres de régression ou de classification.

2 Optimizing Tree structure

Pour $\hat{y} = \sum_{b=1}^{B} f_b$, une fonction coût géneral l, on veut minimiser la fonction objectif suivante :

$$\min \longleftarrow \sum_{i=1}^{n} l\left(y_i, \hat{y}_i\right) + \sum_{b=1}^{B} \Omega\left(f_b\right)$$

avec $\Omega\left(f_b\right) = \gamma|T| + \frac{1}{2}\lambda \sum_{j=1}^{|T|} w_j^2$, et $|T| = d + 1$ est le nombre de feuille de l'arbre de décision ,w_j sont les poids de régression d'une feuille (région) R_j

3 Additive training (Boosting)

On part d'une prédiction constante et on ajoute une nouvelle fonction à chaque fois :

$$\begin{aligned}
\hat{y}_i^{(0)} &= 0 \\
\hat{y}_i^{(1)} &= f_1\left(x_i\right) = \hat{y}^{(0)} + f_1\left(x_i\right) \\
\hat{y}_i^{(2)} &= f_1\left(x_i\right) + f_2\left(x_i\right) = \hat{y}^{(1)} + f_2\left(x_i\right) \\
&\cdots \\
\hat{y}_i^{(t)} &= \sum_{b=1}^{t} f_b\left(x_i\right) = \hat{y}^{(t-1)} + f_t\left(x_i\right)
\end{aligned}$$

4 XGBoost implementation

- Comment décidons-nous quels f_t (fractionnements) ajouter? On Optimise l'objectif! - La prédiction au rang t est $\hat{y}_i^{(t)} = \hat{y}_i^{(t-1)} + f_t\left(x_i\right)$, so that

$$\begin{aligned}
\mathrm{Obj}^{(t)} &= \sum_{i=1}^{n} l\left(y_i, \hat{y}_i^{(t)}\right) + \sum_{b=1}^{t} \Omega\left(f_b\right) \\
&= \sum_{i=1}^{n} l\left(y_i, \hat{y}_i^{(t-1)} + f_t\left(x_i\right)\right) + \Omega\left(f_t\right) + C
\end{aligned}$$

- On considére la fonction cout l_2 :

$$\begin{aligned}
\mathrm{Obj}^{(t)} &= \sum_{i=1}^{n} \left(y_i - \left(\hat{y}_i^{(t-1)} + f_t\left(x_i\right)\right)\right)^2 + \Omega\left(f_t\right) + C_1 \\
&= \sum_{i=1}^{n} \left[2\left(\hat{y}_i^{(t-1)} - y_i\right) f_t\left(x_i\right) + f_t\left(x_i\right)^2\right] + \Omega\left(f_t\right) + C_2
\end{aligned}$$

5 Developpement de Taylor

On rapelle le developpement de Taylor : - $f(x+\Delta x) \approx f(x) + f'(x)\Delta x + \frac{1}{2}f''(x)\Delta x^2$ On définit $g_i = \partial_{\hat{y}_i^{(t-1)}} l\left(y_i, \hat{y}_i^{(t-1)}\right), \quad h_i = \partial^2_{\hat{y}_i^{(t-1)}} l\left(y_i, \hat{y}_i^{(t-1)}\right)$ Donc :

$$\text{Obj}^{(t)} \approx \sum_{i=1}^{n}\left[l\left(y_i, \hat{y}_i^{(t-1)}\right) + g_i f_t(x_i) + \frac{1}{2}h_i f_t(x_i)^2\right] + \Omega(f_t) + C$$

- On considére le cas de la fonction cout l_2 :

$$g_i = \partial_{\hat{y}_i^{(t-1)}}\left(\hat{y}_i^{(t-1)} - y_i\right)^2 = 2\left(\hat{y}_i^{(t-1)} - y_i\right)$$
$$h_i = \partial^2_{\hat{y}_i^{(t-1)}}\left(\hat{y}_i^{(t-1)} - y_i\right)^2 = 2$$

6 Retour sur les objectifs

- On définit $J : \mathbb{R}^p \longrightarrow T, x \mapsto R_j$ for $j : x \in R_j$, On Supprime les constantes et regroupe par feuilles :

$$\begin{aligned}\text{Obj}^{(t)} &\approx \sum_{i=1}^{n}\left[g_i f_t(x_i) + \frac{1}{2}h_i f_t^2(x_i)\right] + \Omega(f_t)\\ &= \sum_{i=1}^{n}\left[g_i w_{J(x_i)} + \frac{1}{2}h_i w^2_{J(x_i)}\right] + \gamma|T| + \frac{1}{2}\lambda\sum_{j=1}^{|T|}w_j^2\\ &= \sum_{j=1}^{|T|}\left[\left(\sum_{i:x_i\in R_j} g_i\right)w_j + \frac{1}{2}\left(\sum_{i:x_i\in R_j} h_i + \lambda\right)w_j^2\right] + \gamma|T|\end{aligned}$$

- C'est la somme de $|T|$ fonctions quadratiques indépendantes , on reconnait deux faits sur la fonction quadratique à variable unique

$$\arg\min_x Gx + \frac{1}{2}Hx^2 = -\frac{G}{H}, H > 0, \quad \min_x Gx + \frac{1}{2}Hx^2 = -\frac{1}{2}\frac{G^2}{H}$$

- On définit $G_j = \sum_{i:x_i\in R_j} g_i$ and $H_j = \sum_{i:x_i\in R_j} h_i$, donc :

$$\text{Obj}^{(t)} \approx \sum_{j=1}^{|T|}\left[G_j w_j + \frac{1}{2}\left(H_j + \lambda\right)w_j^2\right] + \gamma|T|$$

- Supposons la structure de l'arbre $\left(R_j\right)_{j=1}^{|T|}$ est fixée , le poids optimal de chaque feuille et la valeur objective résultante sont :

$$w_j^{\cdot} = -\frac{G_j}{H_j + \lambda}, \quad \text{Obj}^{(t)} \approx -\frac{1}{2}\sum_{j=1}^{|T|}\frac{G_j^2}{H_j + \lambda} + \gamma \mid T$$

7 XGBoost parameters

- ETA [par défaut = 0,3, alias: taux d'apprentissage] Réduction de la taille des pas utilisée dans la mise à jour pour éviter le surajustement. Après à chaque étape de boost, nous pouvons

directement obtenir le poids des nouvelles fonctionnalités, et eta réduit les poids des fonctionnalités pour rendre le processus de renforcement plus conservateur.
- GAMMA [par défaut = 0, alias: minsplitloss] Réduction minimale des pertes requise pour effectuer une nouvelle partition sur un nœud feuille de l'arbre. Plus le gamma est grand, plus il est conservateur l'algorithme sera.
-Maxdepth [par défaut = 6] Profondeur maximale d'un arbre. Augmenter cette valeur rendra le modèle plus complexe et plus susceptible de dépasser t. 0 indique non limite.
-Colsamplebytree [par défaut = 1]
-Rapport de sous-échantillon des colonnes lors de la construction de chaque arbre.
-Le sous-échantillonnage se produira une fois dans chaque itération d'amplification.

8 Bibliographie

https://www.institutdesactuaires.com/global/gene/link.php?news$_{l}ink = mem$