5

Modèles AdaBoost

5. Modèles – Boosting - AdaBoost

AdaBoost :

Le boosting adaptatif met à jour les poids attachés à chacune des observations de l'ensemble de données d'apprentissage.

Le modèle d'ensemble est défini comme une somme pondérée de L apprenants faibles (stumps c'est à dire des arbres qui ne font qu'une seule séparation, avec un seul noeud) construits séquentiellement et une agrégation en une simple combinaison linéaire pondérée par des coefficients exprimant la performance de chaque apprenant.

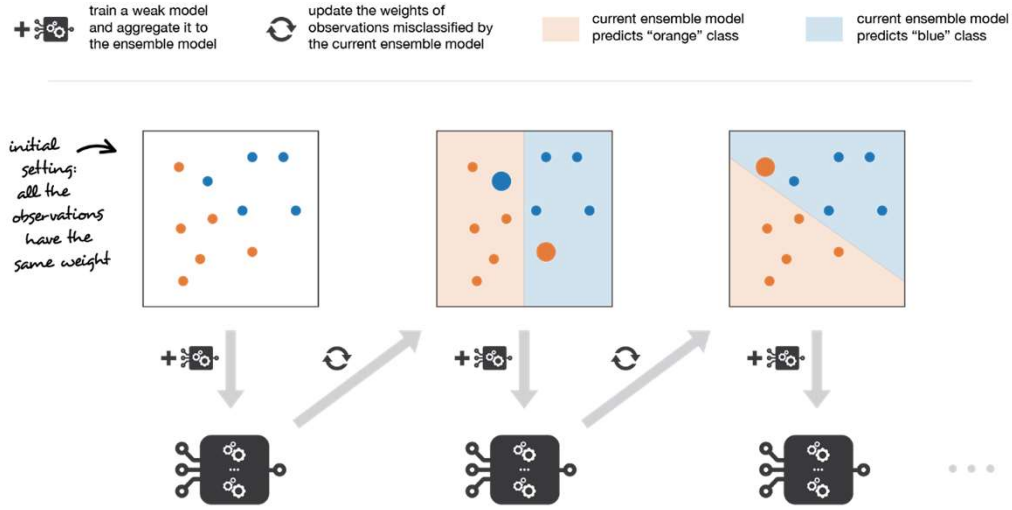
Optimisation globale difficile ➔ optimisation itérative locale en ajoutant les apprenants faibles un par un, en recherchant à chaque itération la meilleure paire possible (coefficient, apprenant faible) à ajouter au modèle d'ensemble actuel

$$(c_l, w_l(.)) = \arg\min_{c,w(.)} E(s_{l-1}(.) + c \times w(.)) = \arg\min_{c,w(.)} \sum_{n=1}^{N} e(y_n, s_{l-1}(x_n) + c \times w(x_n))$$

où E(.) est l'erreur d'ajustement du modèle donné et e(.,.) est la fonction de perte/erreur.

L itération :

- ajuster le meilleur modèle faible possible avec les poids des observations courantes
- calculer la valeur du coefficient de mise à jour qui est une sorte de métrique d'évaluation scalaire de l'apprenant faible qui indique dans quelle mesure cet apprenant faible doit être pris en compte dans le modèle d'ensemble
- mettre à jour l'apprenant fort en ajoutant le nouvel apprenant faible multiplié par son coefficient de mise à jour
- calculer les nouveaux poids des observations qui expriment les observations sur lesquelles nous souhaitons nous concentrer à la prochaine itération (les poids des observations mal prédites par le modèle agrégé augmentent et les poids des observations correctement prédites diminuent).



5. Modèles – Boosting - AdaBoost

Algorithme :

- Au départ, toutes les observations ont le même poids.
- Un modèle est construit sur un sous-ensemble de données.
- En utilisant ce modèle, des prédictions sont faites sur l'ensemble des données.
- Les erreurs sont calculées en comparant les prédictions et les valeurs réelles.
- Lors de la création du modèle suivant, des poids plus élevés sont attribués aux points de données qui ont été prédits de manière incorrecte.
- Les pondérations peuvent être déterminées en utilisant la valeur de l'erreur. Par exemple, plus l'erreur est élevée, plus le poids attribué à l'observation est important.

Ce processus est répété jusqu'à ce que la fonction d'erreur ne change pas, ou que la limite maximale du nombre d'estimateurs soit atteinte.

Hyperparamètres :

base_estimators : spécifie le type d'estimateur de base, c'est-à-dire l'algorithme à utiliser comme apprenant de base.

n_estimators : Il définit le nombre d'estimateurs de base, la valeur par défaut est 10 mais vous pouvez l'augmenter afin d'obtenir une meilleure performance.

learning_rate : même impact que dans l'algorithme de descente de gradient. Ce paramètre contrôle la contribution des estimateurs dans la combinaison finale. Il existe un compromis entre le taux d'apprentissage et les n_estimateurs.

max_depth : profondeur maximale de l'estimateur individuel. Réglez ce paramètre pour obtenir les meilleures performances.

n_jobs : indique au système le nombre de processeurs qu'il est autorisé à utiliser. La valeur '-1' signifie qu'il n'y a pas de limite

random_state : rend la sortie du modèle reproductible. Il produira toujours les mêmes résultats si vous lui donnez une valeur fixe ainsi que les mêmes paramètres et données d'entraînement.