Ensemble

On se rend compte qu'en présence de grands jeux de données, il est plus facile d'avoir une classification performante en combinant les prévisions de plusieurs classificateurs faibles qu'en construisant un seul classificateur très complexe. On appelle méthode d'ensemble une méthode qui consiste à déduire une prévision unique à partir des prévisions de plusieurs modèles.

Il existe plusieurs stratégies pour combiner les prévisions de modèles multiples : * Moyenne des probabilités prescrites par chaque modèle. * Vote de majorité * Approche de type "sélection de variables" utilisée en régression, mais on prend un modèles au lieu d'une variable. * Bagging, boosting, forêts aléatoires

Bagging

La méthode consiste à

1. Sélectionner B échantillons "bootstrap" (tirage avec remise).
2. Construire un arbre de classification avec chacun des échantillons "bootstrap"
3. Prédire la classe d'une nouvelle observation avec chacun des B arbres
4. Attribuer à la nouvelle observation la classe dans laquelle elle est prédite le plus souvent. On peut calculer un score qui est la moyenne des probabilités prédites par chaque arbre.

La méthode du bagging augmente la stabilité des prédictions. Cependant, il est plus difficile d'interpréter l'importance de chaque variable dans le processus de classification. On peut procéder ainsi : * additionner la réduction de l'indice de Gini pour toutes les divisions basées sur une certaine variable ; * calculer la moyenne de ces réductions sur tous les arbres * exprimer la moyenne de réduction de chaque variable en pourcentage de la moyenne de réduction maximale observée sur l'ensemble des variables.

Forêts aléatoires

Très similaire à la méthode du bagging ; la méthode des forêts aléatoires considère elle aussi B arbres obtenus à partir de B échantillons bootstrap.

L'idée est que lors de la construction des B arbres, dans la méthode des forêts aléatoires avant chaque division on choisit aléatoirement $m < p$ des variables $X_1, \dots, X_p$ et on ne considère que ces m variables pour la division optimale. Un choix commun est $m \approx \sqrt{p}$.

L'avantage de procéder ainsi est de décorréler les arbres obtenus, puisque le gros point faible de la méthode du bagging est que les B arbres obtenus se ressemblent souvent beaucoup.

Boosting

L'idée derrière cette approche est de créer des arbres de façon successive en donnant un poids plus élevé aux observations mal classées par les arbres précédents. Il est important d'utiliser des arbres peu performants (très, très simples)} et de laisser le boosting déterminer les bons poids à fournir à ces arbres.

Il existe plusieurs variantes à ce type d'approche * boosting adaptatif (AdaBoost) ; * boosting par descente du gradient (gradient boosting).

Descente du gradient stochastique

On doit choisir un nombre M d'itérations, la taille J de l'arbre de classification à construire à chaque itération (on suggère quelque chose de petit, comme $4 \leq J \leq 8$), la fraction f des données d'entrainement à échantillonner "au hasard" à chaque itération (on suggère $0.5 \leq f \leq 0.8$).

On construit un premier arbre de taille J au jeu de données, disons T_1. Pour $m = 2, \dots, M$: * On calcule les erreurs de prévisions (pseudo-résidus) obtenus avec l'arbre T_{m-1} pour chaque observation. * On échantillonne une fraction f du jeu de données d'entrainement et on construit un nouvel arbre de taille J, disons t_m, pour ces données, mais ce coup-ci en utilisant les pseudo-résidus comme variable à prédire. * On calcule un multiplicateur γ_m qui minimise une certaine fonction de perte. * On pose $T_m = T_{m-1} + \gamma_m t_m$.

- Le mode d'échantillonnage aléatoire à l'étape 3 : donner une probabilité d'être sélectionnée plus élevée aux observations pour lesquelles le pseudo-résidu est plus loin de 0
- La fonction de perte calculée à l'étape 4, la façon dont elle est optimisée, le poids qu'elle donne à chaque observation
- Inclusion de pénalités de régularisation pour prévenir le sur-ajustement
- Implémentation informatique (XGBoost vs LightGBM, utilisation de GPU, etc.)

Tuning

Toute m'ethode de classification requiert qu'on règle (tune) la valeur des hyper-paramètres.

Même si ce n'est pas un théorème, on semble très souvent observer le phénomène que plus la méthode est puissante, plus son réglage est difficile (p.ex. plusieurs hyper-paramètres, sensibilité à la valeur des hyper-paramètres, etc.)

- Analyse discriminante : linéaire ou quadratique ? ; probabilités a priori
- Arbres de classification : indice d'impureté ; nombre de feuilles maximal ; profondeur maximale ; nombre minimal d'observations dans un noeud avant de tenter un scission ; nombre minimal d'observations dans une feuille
- Bagging : Arbres de classification + nombre d'échantillons bootstrap (B)
- Forêts aléatoires : Bagging + nombre de variables à considérer pour chaque division (m)
- Boosting : Arbres de classification + fraction du jeu de données à échantillonner (f) + quelques autres hyper-paramètres selon la variante …