HPC-Big Data 2021 : Apprentissage Statistique TP 4

Agrégation de modèles

Description des données :

Les outils testés dans ce TP seront les méthodes d'agrégation d'arbres binaires (Bagging, Random Forest et Boosting) en régression puis discrimination.

Un producteur éolien utilise des prévisions de force de vent pour en déduire, par exploitation de la courbe de réponse des éoliennes, sa production électrique du lendemain. Afin d'assurer l'équilibre du réseau national de transport d'électricité, cette prévision de production est en effet exigée par le gestionnaire RTE du réseau français.
Non satisfait des prévisions de vent calculées par un modèle météorologique, ce producteur vous demande de lui fournir un modèle statistique capable d'améliorer les scores de prévisions de force de vent sur son parc éolien. Il aurait également besoin, pour son activité, de prévisions d'occurrence de dépassement du seuil de **13 m/s**.

Vous disposez d'une archive de prévisions de différentes variables, issues du modèle météorologique exploité jusqu'alors par votre client, ces prévisions constituant de potentiels prédicteurs pour vos modèles statistiques, ainsi que de l'archive correspondante des mesures de force de vent effectuées sur le parc éolien du producteur.

Le fichier ***DataTP.txt*** contient les 11 variables suivantes :

HU : humidité relative prévue en %

N : nébulosité (= couverture nuageuse) prévue en octas (entiers de 0 à 8)

P : pression prévue en hPa

u : composante zonale du vent prévue en m/s

v : composante méridienne du vent prévue en m/s

hel : hélicité prévue en m^2/s^2 (indice de vorticité)

DD : direction du vent prévue en rad

mois : mois de validité de la prévision

heure : heure de validité de la prévision

FFp : force du vent prévue en m/s

FFo : force du vent **observée** en m/s.

1. **Chargement des librairies et des données :**

Après installation, charger les packages : ***rpart, ipred, randomForest, gbm***

Charger les données dans une data.frame :
data=read.table("DataTP.txt",header=TRUE)

2. **Bagging :**

- La fonction ***bagging*** de la librairie ***ipred*** exploite une famille d'arbres élagués (par défaut ***cp***=0.01) : ***bagreg.out = bagging(FFo ~ . , data, coob=T)***
  L'analyse de l'erreur **OOB** (out-of-bag) permet de choisir le nombre d'arbres.
  Analyser l'influence du nombre d'arbres (***nbagg*** par défaut égal à 25) sur l'erreur **OOB** (out-of-bag).
  Analyser l'influence de la complexité des arbres sur le score **RMSE** en faisant varier le paramètre cp : ***control=rpart.control(cp=)***. L'élagage a-t-il un intérêt en bagging ?

  Confronter la technique de bagging à un arbre binaire individuel élagué en régression puis en discrimination.

3. **Random Forest:**

- La fonction ***randomForest*** de la librairie ***randomForest*** permet d'estimer des modèles de forêts aléatoires : ***?randomForest***.
  rfreg.out = randomForest(FFo ~ . , data, importance=T)
  Le modèle obtenu est ininterprétable mais des coefficients estiment les contributions des différents prédicteurs : ***sort(importance(rf.out)[,1],dec=T)***

  Etudier l'influence du nombre d'arbres agrégés (***ntree***, par défaut égal à 500).
  Etudier l'influence du paramètre ***mtry***, nombre de variables tirées au hasard en chaque nœud parmi les p prédicteurs potentiels (par défaut ***mtry***=E(p/3) en régression et $E(p^{1/2})$ en discrimination, avec E=partie entière).
  Piloter également la complexité des arbres via le paramètre ***maxnodes***, nombre maximal de feuilles.

  Confronter forêts aléatoires, arbre individuel et bagging en régression comme en discrimination. Conclure sur ces techniques d'agrégation.

4. **Boosting :**

- La fonction ***gbm*** de la librairie ***gbm*** permet d'estimer des modèles par gradient boosting. Le nombre d'itérations, ou nombre d'arbres (***n.trees***, par défaut 100), est paramétré ainsi que leur complexité (***interaction.depth***, par défaut 1) et qu'un coefficient de rétrécissement (***shrinkage***, par défaut 0.1) contrôlant le taux d'apprentissage. La qualité est visualisée par un graphe représentant l'évolution de l'erreur d'apprentissage. D'autre part, une procédure de validation croisée est incorporée, elle fournit un nombre optimal d'itérations à considérer.
  boostreg.out = gbm(FFo ~ . , data, distribution="gaussian", n.trees=5000, cv.folds=10)
  plot(boostreg.out$cv.error)
  gbm.perf(boostreg.out)

  Etudier les influences des paramètres ***n.trees***, ***interaction.depth*** et ***shrinkage*** sur les performances de la méthode.
  Confronter boosting, forêts aléatoires, arbre individuel et bagging en régression comme en discrimination (en discrimination le prédictand doit être de type numérique codé 0 et 1 et non de type factor). Conclure.

5. **Conclusion :**

- Confronter finalement les différentes méthodes d'agrégation aux méthodes des TP précédents en régression puis discrimination. Conclure.