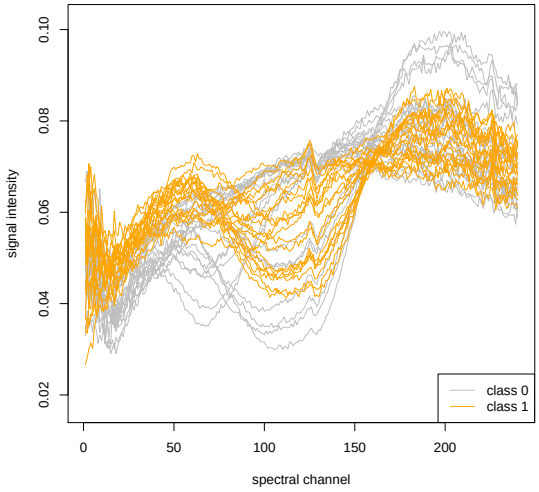
Projet n°4 : validation croisée et Random Forests

M1 parcours SSD - UE Apprentissage Statistique I

L'algorithme des Random Forests (forêts aléatoires) est un algorithme d'apprentissage supervisé qui fournit en général de bonnes performances de prédiction. C'est une méthode dite ensembliste, car elle procède par aggrégation de classifieurs, en l'occurence des arbres de décision, qui sont construits de manière stochastique. Dans sa forme la plus standard, il met en jeu deux paramètres : le nombre d'arbres à considérer (`ntree`) et un paramètre (`mtry`) contrôlant le caractère aléatoire introduit lors de leur construction [1]. Ces paramètres sont liés à la complexité du modèle sous-jacent. Il est donc crucial de les optimiser avec soin.

L'objectif de cet exercice est d'implémenter une procédure de validation croisée visant à optimiser le paramètre `ntree` sur un jeu de données réel. Le paramètre `mtry` sera, dans un premier temps en tout cas, laissé à sa valeur par défaut. Ce jeu de données vise à reconnaitre deux catégories de bactéries (notées ici "0" et "1"), en fonction d'un signal spectral illustré dans la figure ci-dessous.



Ce jeu de données est enregistré dans le fichier `bacteria-dataset.Rdata`, contenant les objets suivants : une matrice `X` de taille 2403×240 contenant les spectres, et un vecteur `Y` de longueur 2403 contenant leurs catégories (1839 de la classe "0" et 564 de la classe "1").

1. `ntree` et `mtry` sont les noms de ces paramètres dans le package `randomForest` mentionné à la fin du document.

Objectifs

L'objectif est en premier lieu d'implémenter **par vos propres moyens** (i.e., sans utiliser d'éventuels packages dédiés) une procédure de validation croisée permettant d'estimer les performances de classification de l'algorithme des RandomForest, pour des valeurs du paramètre `ntree` prises dans $\{25, 50, 100, 250, 500, 1000\}$. On considèrera 10 folds pour la validation croisée.

Pour mesurer les performances de classification on s'intéressera aux indicateurs suivants : le taux de bonne classification global ("accuracy"), la sensibilité et la spécificité. Pour définir ces deux derniers critères, on considèrera que la classe "1" est la classe positive (et la classe "0" la classe négative). Là aussi, vous calculerez ces indicateurs **par vos propres moyens** (i.e., sans utiliser d'éventuels packages dédiés).

Pour juger de la performance des Random Forests, on appliquera une procédure similaire pour estimer les performances de l'algorithme des K plus proches voisins, pour des valeurs de K prises entre 1 et 9.

Enfin, pour aller plus loin, on pourra consolider cette analyse en faisant également varier le paramètre `mtry`.

Le minimum

Voici les consignes qui empêcheraient d'avoir la moyenne si elles ne sont pas respectées :

- Le compte-rendu ne doit pas faire plus de 6 pages (sans compter les éventuelles annexes).
- Il doit contenir le nom des auteurs, un titre explicite, une introduction et une conclusion.
- Il doit également contenir une rapide description de l'algorithme des RandomForest, et du rôle joué par les paramètres `ntree` et `mtry`. On cherchera dans cette description à faire le lien avec les notions vues en cours (e.g., risque empirique, complexité, ...), et on précisera la valeur prise par défaut par le paramètre `mtry`.
- Une figure présentant l'évolution des trois critères de performances considérés (taux de bonne classification, sensibilité, spécificité) en fonction de `ntree` doit être incluse et commentée : quelle valeur de `ntree` retenez-vous ?
- La même figure présentant les résultats obtenus par K plus proches voisins doit également y figurer, et les performances des deux méthodes doivent être comparées.
- Enfin, on cherchera éventuellement à évaluer l'impact du paramètre `mtry`.
- Le code mis en oeuvre doit apparaître, commenté un minimum, en annexe (et uniquement en annexe).

Bibliographie

- Pour une introduction aux Random Forests on pourra par exemple se référer à Wikipedia : `https://fr.wikipedia.org/wiki/For%C3%AAt_d%27arbres_d%C3%A9cisionnels` et (plus complet) `https://en.wikipedia.org/wiki/Random_forest`
- Pour réaliser l'apprentissage d'un modèle Random Forest et s'en servir pour faire des prédiction, on pourra utiliser le package `randomForest` via les commande suivantes :
  - apprendre un modèle : `model = randomForest(X, Y, ntree = n)`
  - l'utiliser pour faire des prédictions : `preds = predict(model, X)`