# Statistique en Grande Dimension et Apprentissage
# TP "Arbres/Random Forests/Boosting"

Ce TP fera usage des modules `pandas`, du module `tree` de `sklearn` (en particulier de `DecisionTreeClassifier`), des modules `BaggingClassifier`, `RandomForestClassifier` et `GradientBoostingClassifier` de `sklearn.ensemble`.

**Exercice 1** (Illustration sur un problème simple de classification binaire)**.**

1. **Arbres de décision "simples"**
   (a) Chargez les données `spam7.csv`.
   (b) La réponse est donnée par la colonne `yesno`. Fabriquez $X$ et $Y$ puis faites le split habituel en données train et test.
   (c) Utilisez le module `DecisionTreeClassifier` avec les paramètres par défaut pour prédire votre échantillon test. Consulter la documentation associée et l'ensemble des arguments de ce classifieur.
   (d) Prédire les classes de l'échantillon test et fournir les scores associés à chaque individu (à l'aide de `.predict_proba`). Affichez les résultats des premiers individus de l'échantillon test.
   (e) Affichez les scores d'entrainement et de test.
   (f) Affichez l'arbre à l'aide de la fonction `tree.plot_tree`.
   (g) Faites une sélection du meilleur estimateur (par `GridSearchCV`) en faisant varier la profondeur de l'arbre, le nombre minimal d'individus par noeud, la "métrique" d'hétérogénéité (ou `impurity`) et le coefficient d'élagage. Affichez votre meilleur résultat : classifieur optimal, erreur d'entraînement et test associées.
   (h) Affichez l'arbre de décision optimal.
   (i) Tracez un graphe d'importance des variables à l'aide de `.feature_importances_`.
   (j) Faites un scatterplot des données dans le plan des deux variables les plus influentes (avec des couleurs relatives à la prédiction).
2. **Bagging**
   (a) Appliquez l'algorithme de bagging sur les données `spam7.csv` avec les paramètres par défaut.
   (b) Calculez l'erreur *out-of-bag* et comparez-la à l'erreur test.
   (c) Sélectionnez le meilleur modèle dans une gamme de modèles de votre choix (attention, ça met plus de temps).
   (d) Quand on fixe `max_features=0.8`, cela ressemble à la Random Forest mais ça n'en est pas tout à fait une. Expliquez.
3. **Random Forest**
   (a) Appliquez l'algorithme de Random Forest sur les données `spam7.csv` avec les paramètres par défaut.

(b) Analysez les paramètres de l'algorithme.

(c) Sélectionnez le meilleur modèle dans une gamme de modèles de votre choix.

(d) Affichez le graphe d'importance des variables associé au meilleur modèle.

4. **Gradient Boosting**

(a) Appliquez l'algorithme de Gradient Boosting sur les données `spam7.csv` avec les paramètres par défaut.

(b) Analysez les paramètres de l'algorithme et faites un test de performances.

*N.B. Dans les méthodes d'agrégation, il existe bien sûr beaucoup d'autres méthodes. La méthode dite de Stacking par exemple est basée sur l'idée d'empiler/mélanger différents types d'algorithmes pour optimiser les performances. La fonction* `sklearn.ensemble.StackingClassifier` *est prévue à cet effet.*

**Exercice 2** (DecisionTreeRegressor)**.** Les arbres de décision et leurs extensions agrégées sont bien sûr utilisables en régression. On propose dans cet exercice d'utiliser sur la base classique `BostonHousing` (considérée. Testez rapidement le `DecisionTreeRegressor` et une alternative agrégée (random forest par exemple).

**Exercice 3** (XGboost)**.**

1. Consultez la documentation de XGboost à ce lien.
2. Testez rapidement sur `spam7.csv` (cadre bien sûr un peu simple pour utiliser les performances de cet algorithme).

**Exercice 4.** Ecrire "à la main" un algorithme de gradient boosting dans le cadre des arbres de décision, pour la classification multi-classes avec fonction de perte ?déviance ?. Fabriquer également une fonction qui génère un graphe de l'évolution de l ?erreur d ?entraînement et de l ?erreur test en fonction des itérations du gradient boosting.

**Exercice 5** (Classification de cancers)**.** Dans cet exercice, il s'agit de mettre en oeuvre les algorithmes vus dans ce chapitre pour la classification de cancers en fonction de données d'expression de gènes relatives. Il y a 16063 genes, 144 observations d'entraînement, 54 échantillons test. Les classes de cancer sont numérotées de 1 à 14 (sein/prostate/langue/collorectal/lymphome /vessie /mélanôme /utérus /leucémie /rein /pancréas /ovaires/mésothéliome/cerveau).

Vous rendrez un fichier notebook relatif à votre étude incluant arbres de décision standard, random forests, gradient boosting et éventuellement `Xgboost`.