# Oraux blancs MPI
# type CCINP

Florian Bourse

## Exercice A.

Voici un ensemble de données $E$ :

| V | W | X | Y |
|---|---|---|---|
| F | F | F | F |
| F | V | F | V |
| V | F | F | V |
| V | V | F | F |
| V | V | V | F |

On souhaite construire un arbre de décision pour classifier la variable $Y$, contenant le moins de nœuds possible. *On rappelle qu'un arbre de décision est un arbre binaire dont les nœuds internes sont étiquetés par les attributs et les feuilles par $\{V, F\}$. Les fils gauches correspondent à une réponse $F$ et les fils droits à une réponse $V$.*

1. Rappeler le principe de l'apprentissage supervisé.
2. L'entropie d'un ensemble $S$ d'exemples esst définie par :

$$H(S) = -\frac{n_+}{n}\log_2(\frac{n_+}{n}) - \frac{n_-}{n}\log_2(\frac{n_-}{n})$$

   où $n_+$, $n_-$ et $n$ désignent respectivement le nombre d'éléments de $S$ dont l'étiquette est $+$, le nombre d'éléments de $S$ dont l'étiquette est $-$ et enfin le nombre total d'éléments de $S$. Dans le cas où $k = 0$, on prend la convention $k\log_2(k) = 0$. Par exemple, l'entropie de l'ensemble de toutes les données $E$ ci-dessus est $H(E) = 0,918\,296$ (valeur approchée).
   Étant donné un attribut $A$, on définit le gain de $A$ par rapport à $S$ par

$$IG(S, A) = H(S) - \frac{n_{A=V}}{n}H(S_{A=V}) - \frac{n_{A=F}}{n}H(S_{A=F})$$

   où $S_{A=V}$ désigne le sous ensemble des éléments de $S$ dont l'attribut $A$ est $V$ et $n_{A=V}$ désigne son cardinal, de même pour $F$ et $n$ désigne toujours le cardinal de $S$. Par exemple, le gain d'information de l'attribut $X$ par rapport à $E$ est $IG(E, X) = 0,118\,296$ (valeur approchée).
   Calculer les gains d'information $IG(E, V)$ et $IG(E, W)$.
   Quel attribut serait sélectionné en premier par l'algorithme ID3 ?
3. Donner l'arbre de décision construit par ID3 en entier, sans élagage.
4. Une idée pour élaguer l'arbre est de commencer à la racine, et d'enlever les tests pour lesquels le gain d'information est plus petit qu'un certain seuil $\varepsilon$. On parle d'élagage de haut en bas. Quel est l'arbre de décision renvoyé pour $\varepsilon = 0,000\,1$ ?
   Tracer la matrice de confusion de cet arbre sur les données d'entrainement.

5. Une autre possibilité est de commencer aux feuilles, et d'élaguer les sous-arbres dont le gain d'information d'un test est plus petit qu'un certain seuil $\varepsilon$. Avec cette méthode, aucun ancêtre d'enfants avec un haut gain d'information ne sera élagué. On parle d'élagage de bas en haut. Quel est l'arbre de décision renvoyé pour $\varepsilon = 0,000\,1$ ? Tracer la matrice de confusion de cet arbre sur les données d'entrainement.
6. Comparer les taux d'erreurs et la complexité des deux approches et discuter dans quels cas l'élagage de bas en haut serait préférable à l'élagage de haut en bas et vice versa.
7. Quelle est la profondeur de l'arbre renvoyé par ID3 avec élagage de bas en haut ? Peut-on trouver un arbre de profondeur plus petite qui classifie aussi parfaitement $Y$ sur le jeu de données d'entrainement ? Quelles conclusions peut-on en tirer sur les performances de l'algorithme ID3 ?

# Exercice B. L'exercice suivant est à traiter dans le langage C

On s'intéresse dans cet exercice à l'implémentation en C de files de priorité, et à l'insertion de tous les éléments d'un tableau dans une file de priorité. Pour simplifier l'analyse, on considèrera des files de priorités dont la priorité d'un élément est égale à sa valeur, l'élément plus petit étant prioritaire.
Dans un premier temps, on implémente ces files de priorités à l'aide de listes chaînées triées dans l'ordre croissant. Puis, nous analyserons la tassification d'un tableau par le bas.

1. Le code qui vous est fourni essaye d'implémenter des listes chaînées dont l'insertion et la suppression se font nécessairement en tête de liste. Pourquoi cette implémentation ne fonctionne-t-elle pas ? Proposez une solution et l'implémenter.
2. Modifier la fonction d'insertion pour insérer le nouvel élément à la position attendue dans la liste, afin qu'elle reste triée dans l'ordre croissant. La solution proposée doit avoir une complexité en temps dans le pire des cas linéaire en la taille de la liste.
3. Quelle est la complexité pour insérer successivement tous les éléments d'un tableau de taille $n$ dans une file vide ?
4. Rappeler la définition de la propriété de tas.
5. On utilise un tableau de taille $2^h - 1$ pour représenter les étiquettes d'un arbre binaire complet de profondeur $h$, les nœuds étant numérotés dans l'ordre d'un parcours en largeur de gauche à droite. Pour $0 < i < 2^h$, quelle case correspond au parent du nœud en case $i$ ? Démontrer-le.
6. Pour tassifier notre tableau, nous allons pour chacun des nœuds en partant du dernier échanger sa valeur avec celle de son père si celle-ci est plus grande. Implémenter cette technique.
7. Montrer que cette technique est correcte, et en analyser sa complexité.