Contrôle écrit INF442
— Algorithmes pour l'analyse de données en C++ —
École Polytechnique
Département d'informatique

Durée : 3 heures

5 Juin 2019

Imformations générales :

- Les problèmes sont tous **indépendants** les uns des autres et peuvent donc être traités dans n'importe quel ordre.
- On appréciera en premier lieu la **clarté** et l'**exhaustivité** des réponses aux questions.
- Les énoncés sont écrits en français, vos réponses peuvent être rédigées en français ou en anglais.
- Tous les **documents imprimés** du cours (transparents d'amphi, polycopié, notes personnelles) sont autorisés, en revanche les appareils électroniques (téléphone, ordinateur, calculatrice, etc.) sont interdits.
- Prenez le temps de bien lire les énoncés avant de commencer. Pensez également à garder 5 à 10 minutes en fin d'examen pour relire votre copie.

Barème : Les problèmes ci-dessous comptent pour la note finale de la façon suivante :

Problème 1 : Regroupement des k-médianes sur la droite réelle	30%
Problème 2 : Optimalité du classifieur 1-NN	20%
Problème 3 : Convergence de l'algorithme du perceptron	20%
Problème 4 : Réseaux de perceptrons et formules booléennes	20%
Problème 5 : Questions de cours et de C++	10%

Pour chaque problème, le pourcentage du barème de chaque question est donné **relativement au problème**. Ainsi, les pourcentages des questions d'un même problème se somment à 100%.

Problème 1 : Regroupement des k-médianes sur la droite réelle

Étant donné un nuage de points $P = \{p_1, \cdots, p_n\}$ sur la droite réelle $\mathbb{R}$ et un entier $k > 0$, le regroupement des k-médianes (*k-medians clustering* en anglais) consiste à choisir k centres $c_1, \cdots, c_k \in \mathbb{R}$ et une partition $\sigma : P \to \{1, \cdots, k\}$ tels que l'énergie

$$E(c_1, \cdots, c_k, \sigma) = \sum_{i=1}^{n} |p_i - c_{\sigma(p_i)}|$$

soit minimale, où $|\cdot|$ désigne la valeur absolue (notez l'absence du carré dans les termes de la somme).

Question 1.1 (10%)**.** Considérons le cas où les centres $c_1, \cdots, c_k$ sont fixes. Montrez qu'alors le minimum est atteint par la partition de Voronoï σ_{NN} vue en cours.

Question 1.2 (10%)**.** Considérons à présent le cas particulier où $k = 1$ et $n = 2$, et trions les points de P de telle façon que $p_1 \leq p_2$. Montrez qu'alors le minimum d'énergie est atteint pour tout $c_1 \in [p_1, p_2]$.

Question 1.3 (10%)**.** Considérons maintenant le cas où $k = 1$ et $n = 3$, et trions encore une fois les points de P de telle façon que $p_1 \leq p_2 \leq p_3$. Montrez qu'alors le minimum d'énergie est atteint en posant $c_1 = p_2$.

Question 1.4 (25%)**.** Regardons maintenant le cas plus général où $k = 1$ et $n \geq 2$ est quelconque, et trions une fois de plus les points de P de manière à ce que $p_1 \leq p_2 \leq \cdots \leq p_n$. Montrez qu'alors le minimum d'énergie est atteint en plaçant c_1 à la médiane de l'ensemble P, plus précisément en posant

$$c_1 = \begin{cases} p_{\frac{n}{2}+1} & \text{si } n \text{ est pair} \\ p_{\frac{n+1}{2}} & \text{si } n \text{ est impair} \end{cases}$$

Vous pourrez par exemple procéder par récurrence sur $n \geq 2$.

Les éléments ci-dessus nous permettent d'adapter l'algorithme de Lloyd vu en cours, simplement en remplaçant, à chaque itération de l'algorithme, le calcul de la moyenne de chaque cluster par le calcul de la médiane telle que définie à la question 1.4.

Question 1.5 (25%)**.** Montrez que l'énergie décroît lors de l'exécution de cette variante de l'algorithme, et que celle-ci termine après un nombre fini d'étapes.

Question 1.6 (20%)**.** Reprenons maintenant l'approche par programmation dynamique vue en cours. Étant donnés deux entiers $m \leq n$ et $l \leq k$, on dénote par $\textsc{opt}(m, l)$ l'énergie minimale obtenue avec l centres sur le sous-ensemble $\{p_1, \cdots, p_m\}$ des points de P. Montrez la récurrence suivante :

$$\textsc{opt}(m, l) = \min_{1 \leq j \leq m} \left\{ \textsc{opt}(j-1, l-1) + \sum_{i=j}^{m} |p_i - q| \right\},$$

où q désigne la médiane de l'ensemble $\{p_j, \cdots, p_m\}$ telle que définie à la question 1.4, et où on pose par convention $\textsc{opt}(m, l) = 0$ lorsque $m = 0$ ou $l = 0$.

Ainsi, on peut adapter l'algorithme par programmation dynamique pour résoudre exactement le regroupement des k-médianes sur la droite réelle, avec une complexité inchangée en $O(n^3 k)$.

Problème 2 : Optimalité du classifieur 1-NN

Soit (X, Y) un couple de variables aléatoires prenant leurs valeurs respectivement dans $\mathbb{R}^d$ et dans $\{1, \cdots, \kappa\}$. On s'intéresse au comportement du classifieur k-NN avec $k = 1$ sur les réalisations de (X, Y). Pour cela on regarde le risque avec la fonction de perte 0-1.

Soit $x \in \mathbb{R}^d$ un point fixé une fois pour toutes. Pour toute classe $y \in \{1, \cdots, \kappa\}$, on note $p_y(x)$ la probabilité postérieure $\mathbb{P}(Y = y \mid X = x)$. Soit y^* la classe qui maximise la probabilité postérieure, c'est-à-dire

$$y^* = \underset{y \in \{1, \cdots, \kappa\}}{\operatorname{argmax}} \; p_y(x)$$

Question 2.1 (15%)**.** Montrez que l'erreur de Bayes en x est $1 - p_{y^*}(x)$. On rappelle que l'erreur de Bayes en x est l'espérance, sur l'ensemble des tirages possibles de $(Y \mid X = x)$, de la perte 0-1 du classifieur de Bayes.

Question 2.2 (25%)**.** On tire indépendamment selon $(Y \mid X = x)$ une réponse y_1 pour l'entraînement et une réponse y_2 pour le test. Le classifieur 1-NN prédit alors la réponse y_1 pour x. Montrez que l'erreur associée, c'est-à-dire l'espérance (sur les tirages possibles de y_1 et y_2) de la perte 0-1, est

$$\sum_{y=1}^{\kappa} p_y(x)\,(1 - p_y(x))$$

Question 2.3 (15%)**.** En déduire que l'erreur du classifieur 1-NN est supérieure ou égale à l'erreur de Bayes en x.

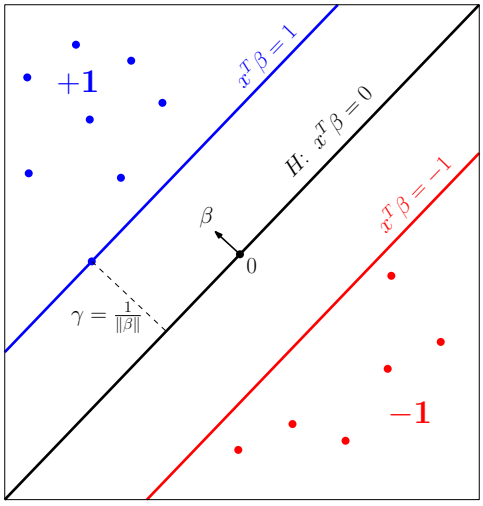
Question 2.4 (20%)**.** Dans le cas binaire ($\kappa = 2$), montrez que l'erreur du classifieur 1-NN est égale à $2\,p_{y^*}(x)\,(1 - p_{y^*}(x))$. En déduire que cette erreur n'est pas plus de 2 fois l'erreur de Bayes.

Question 2.5 (25%)**.** Dans le cas général ($\kappa \geq 2$), montrez que l'erreur du classifier 1-NN reste bornée supérieurement par 2 fois l'erreur de Bayes.

Problème 3 : Convergence de l'algorithme du perceptron

Considérons un ensemble d'observations et de réponses $(x_1, y_1), \cdots, (x_n, y_n) \in \mathbb{R}^d \times \{-1, 1\}$ tel qu'il existe un hyperplan H séparant strictement les deux classes $+1$ et -1. Sans perte de généralité (quitte à faire une translation et une homothétie), on suppose que H passe par l'origine et que les observations $x_1, \cdots, x_n$ sont toutes situées dans la boule unité ($\|x_i\| \leq 1$ pour tout i, où $\|\cdot\|$ désigne la norme euclidienne).

On paramétrise H par l'équation $x^T\beta = 0$, où β est un vecteur de $\mathbb{R}^d$. Sans perte de généralité encore une fois (l'équation étant homogène), on fixe $\|\beta\| = \frac{1}{\gamma}$, où $\gamma > 0$ désigne la marge associée à l'hyperplan H. Ainsi on a $y_i\, x_i^T \beta \geq 1$ pour toute observation x_i. Voir la figure ci-dessous pour une illustration.



Rappelons que l'algorithme du perceptron vu en cours pose $\hat{\beta} = 0 \in \mathbb{R}^d$ puis itère sur les observations x_i, et pour chacune d'elles, teste si elle est mal classifiée ($y_i\, x_i^T \hat{\beta} \leq 0$). En cas de mauvaise classification de x_i, l'algorithme met à jour $\hat{\beta}$ immédiatement par une étape de descente de gradient stochastique :

$$\hat{\beta} \longleftarrow \hat{\beta} + y_i\, x_i \tag{1}$$

L'algorithme répète ce processus d'itération sur les observations et de mise à jour de $\hat{\beta}$ à la volée jusqu'à convergence (c'est-à-dire jusqu'à ce qu'il n'y ait plus d'observations mal classifiées). Chaque itération sur l'ensemble des observations est appelée une *époque*.

Question 3.1 (35%)**.** Lorsqu'une observation x_i considérée est mal classifiée, montrez que

$$\left\|\hat{\beta}^{\text{new}} - \beta\right\|^2 \leq \left\|\hat{\beta}^{\text{old}} - \beta\right\|^2 - 1,$$

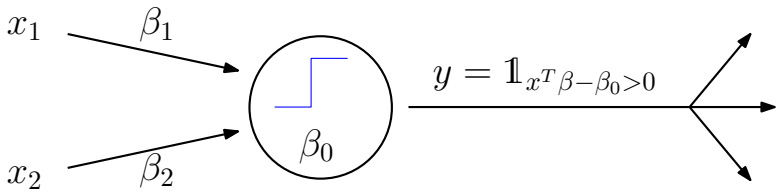
où $\hat{\beta}^{\text{old}}$ et $\hat{\beta}^{\text{new}}$ désignent le vecteur $\hat{\beta} \in \mathbb{R}^d$ respectivement avant et après sa mise à jour décrite dans l'équation (1).

Question 3.2 (25%)**.** En déduire que l'algorithme converge après un nombre fini d'époques, et donner une borne supérieure sur ce nombre.

Question 3.3 (40%)**.** Donnez une configuration de points et de labels dans le plan ($d = 2$) telle que l'algorithme décrit ci-dessus (qui recherche uniquement des hyperplans séparateurs passant par l'origine en optimisant le vecteur $\hat{\beta}$ comme décrit dans l'équation (1)) boucle indéfiniment. Vous devez justifier votre réponse en détaillant le déroulement de l'algorithme sur votre input.

Problème 4 : Réseaux de perceptrons et formules booléennes

Considérons la variante suivante du neurone perceptron, qui renvoie la valeur $+1$ si $x^T\beta - \beta_0 > 0$ et 0 si $x^T\beta - \beta_0 \leq 0$ (avec $x = \left[\begin{smallmatrix} x_1 \\ x_2 \end{smallmatrix}\right]$ et $\beta = \left[\begin{smallmatrix} \beta_1 \\ \beta_2 \end{smallmatrix}\right]$) :



Supposons que les entrées x_1, x_2 appartiennent elles-mêmes à l'ensemble $\{0, 1\}$.

Question 4.1 (20%)**.** Donnez des poids $\beta_1, \beta_2 \in \mathbb{R}$ et un biais $\beta_0 \in \mathbb{R}$ qui permettent à ce perceptron de simuler la porte logique `NAND` ("non-et"), c'est-à-dire que la valeur de sortie y vérifie la table de vérité ci-dessous :

x_1	x_2	x_1 `NAND` x_2
1	1	0
1	0	1
0	1	1
0	0	1

Montrez que le neurone ainsi obtenu simule bien la porte `NAND`.

Question 4.2 (15%)**.** Donnez une valeur constante $x_2 \in \{0, 1\}$ qui fasse que le perceptron de la question 4.1 simule la porte logique `NOT` ("non"), dont la table de vérité est donnée ci-dessous :

x_1	`NOT` x_1
1	0
0	1

Montrez que le neurone ainsi obtenu simule bien la porte `NOT`.

Grâce à ces résultats, on peut simuler toute formule booléenne sans quantificateurs par un réseau de perceptrons.

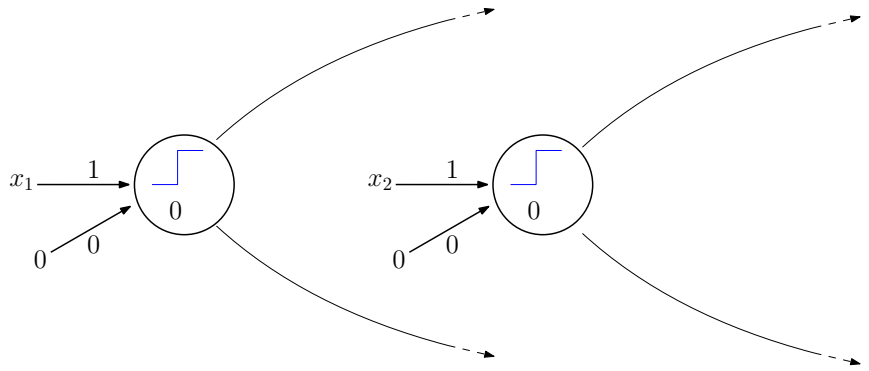
Question 4.3 (30%)**.** Proposez un réseau de perceptrons qui simule la porte logique `XOR` ("ou-exclusif"), dont la table de vérité est donnée ci-dessous :

x_1	x_2	x_1 `XOR` x_2
1	1	0
1	0	1
0	1	1
0	0	0

Pour vous aider, on rappelle l'identité logique suivante :

$$x_1 \texttt{ XOR } x_2 = (x_1 \texttt{ NAND } (\texttt{NOT } x_2)) \texttt{ NAND } ((\texttt{NOT } x_1) \texttt{ NAND } x_2)$$

Vous présenterez votre réseau de perceptrons sous forme graphique, en indiquant les poids sur les arêtes du réseau et les biais à l'intérieur des nœuds. Pour éviter les croisements d'arêtes dans le dessin, vous pourrez présenter le début du réseau comme suit :



Vous n'avez pas besoin de montrer que votre réseau simule bien la porte `XOR`.

Question 4.4 (35%)**.** Montrez qu'on ne peut pas simuler la porte logique `XOR` avec un seul perceptron, quel que soit le choix de poids β_1, β_2 et de biais β_0. Pour cela vous pourrez par exemple vous placer dans l'espace $\{0,1\}^2$ des valuations des variables (x_1, x_2) et donner un argument géométrique.

Cette dernière limitation fut mise en avant par Minsky et Papert dans leur ouvrage de 1969 sur les perceptrons, qui mena au premier "hiver de l'IA".

Problème 5 : Questions de cours et de C++

Pour chacune des questions suivantes, indiquez la réponse (**unique**) qui convient parmi celles proposées. Note : toutes les questions sauf la dernière ont le même poids dans la notation.

1. Étant données n observations dans $\mathbb{R}^d$, fournies sous la forme d'une matrice de coordonnées $n \times d$ et d'un vecteur de réponses $n \times 1$, quelle est la complexité en temps de l'entraînement d'un prédicteur k-NN sur ces données ?

   A. 0 **B.** $O(nd)$ **C.** $O(n^d)$

2. Qu'affiche le programme suivant ?

```
#include <iostream>

void incr (int* p) { p++; }

int main() {
  int i; incr(&i);
  std::cout << i << std::endl;
  return 0;
}
```

   A. rien, il plante à l'exécution **B.** 0 **C.** 1 **D.** n'importe quoi

3. Vous récupérez des données sans label, sur lesquelles vous souhaitez faire de la classification binaire. Les données sont 2-dimensionnelles, et leur affichage donne le résultat suivant :

   Quelle approche parmi les suivantes vous paraît-elle la plus adaptée ?

   A. k-means avec $k = 2$ **B.** single-linkage **C.** DBSCAN **D.** un perceptron

4. On se donne la matrice de confusion suivante entre deux classes labellisées 1 et 2 (dans cet ordre) :

$$\begin{bmatrix} 0.4 & 0.1 \\ 0.2 & 0.3 \end{bmatrix}$$

   Quel est le F-score associé à la classe 1 ?

   A. 2/9 **B.** 4/11 **C.** 16/22 **D.** 18/27

5. Qu'affiche le programme suivant ?

```
#include <iostream>

class Cours { protected: Cours() { std::cout << "Cours "; } };

class Data: protected Cours { public: Data() { std::cout << "Data "; } };

class Cpp: public Cours { protected: Cpp() { std::cout << "Cpp "; } };

class INF442: private Cpp, protected Data { public: INF442() { std::cout << "INF442 "; } };

int main() {
  INF442 c;
  return 0;
}
```

   A. "Cours Cpp Data INF442"
   B. "Cours Data Cpp INF442"
   C. "Cours Data Cours Cpp INF442"
   D. "Cours Cpp Cours Data INF442"
   E. "INF442 Data Cours Cpp Cours"

6. Un ami vous apporte ses données, composées d'un million d'observations en dimension 200, réparties en 2 classes, dont seulement 10 000 observations ont des labels. Il souhaiterait tester certaines des méthodes que vous avez vues en cours pour faire de la classification binaire sur ses données. Pour cela il envisage d'effectuer l'entraînement sur le sous-échantillon de 10 000 observations avec labels, puis la prédiction sur le reste des données. Il soupçonne que ses deux classes sont échantillonnées selon deux distributions gaussiennes distinctes, toutefois il n'est pas certain que les classes soient linéairement séparables. Quelle méthode allez-vous tester en priorité parmi celles citées ci-dessous ?

   A. k-means avec $k = 2$
   B. une machine à vecteurs de support
   C. un perceptron
   D. un classifieur k-NN

7. Question subsidiaire (sur 0 point) : quel est l'intérêt de l'apprentissage du langage C++ dans le cadre d'un cours d'introduction à la science des données ?

   A. aucun
   B. augmenter artificiellement la difficulté du cours
   C. pouvoir dire qu'on fait ça à l'X et nulle part ailleurs
   D. faire de l'histoire des sciences et découvrir comment nos parents faisaient de l'analyse de données dans les années 1970
   E. ne pas être simple utilisateur de bibliothèques et pouvoir, le cas échéant, entrer dans le code et l'adapter ou l'optimiser selon ses besoins

*

* *