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

X2017

Correction

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

Question 1.1. Le raisonnement est le même que pour les k-moyennes. Pour un p_i donné, le choix de $\sigma(p_i)$ n'intervient que dans un seul terme de la somme dans l'expression de $E(c_1, \cdots, c_k, \sigma)$, terme qui en retour ne dépend que de $\sigma(p_i)$ et peut donc être optimisé indépendamment des autres en choisissant pour $\sigma(p_i)$ l'indice du centre le plus proche de p_i.

Question 1.2. Pour tout $c_1 \in [p_1, p_2]$ on a $|p_1 - c_1| + |p_2 - c_1| = c_1 - p_1 + p_2 - c_1 = p_2 - p_1$, qui est indépendant de c_1. Par ailleurs, pour $c_1 \notin [p_1, p_2]$ on a $\max\{|p_1 - c_1|, |p_2 - c_1|\} > p_2 - p_1$. Ainsi, le minimum d'énergie est atteint par chaque point $c_1 \in [p_1, p_2]$.

Question 1.3. Pour tout $c_1 \in [p_1, p_3]$, le même calcul que précédemment donne $\sum_{i=1}^{3} |p_i - c_1| = p_3 - p_1 + |p_2 - c_1|$, qui est minimal quand $c_1 = p_2$ et alors égal à $p_3 - p_1$. Par ailleurs, pour tout $c_1 \notin [p_1, p_3]$ on a $\sum_{i=1}^{3} |p_i - c_1| > p_3 - p_1$, ce qui donne le résultat.

Question 1.4. Les cas de base $n = 2$ et $n = 3$ sont traités dans les questions précédentes. Considérons maintenant le cas $n \geq 4$, sous l'hypothèse de récurrence que le résultat est vrai pour toute valeur de n strictement plus petite. Pour tout $c_1 \in [p_1, p_n]$ on a $\sum_{i=1}^{n} |p_i - c_1| = p_n - p_1 + \sum_{i=2}^{n-1} |p_i - c_1|$. Le premier terme est indépendant de c_1, tandis que par l'hypothèse de récurrence (appliquée à l'ensemble $\{2, \cdots, n-1\}$) la somme est minimale lorsque $c_1 = p_{1+\frac{n-2}{2}+1} = p_{\frac{n}{2}+1}$ si n est pair et lorsque $c_1 = p_{1+\frac{n-2+1}{2}} = p_{\frac{n+1}{2}}$ si n est impair. Par ailleurs, lorsque $c_1 \notin [p_1, p_n]$ on a $\sum_{i=1}^{n} |p_i - c_1| > p_n - p_1 + \sum_{i=2}^{n-1} |p_i - c_1|$, donc (par l'hypothèse de récurrence encore une fois) le minimum dans ce cas-là ne peut être que plus grand que dans le cas précédent. D'où le résultat.

Question 1.5. L'argument est similaire à celui des k-moyennes, mais pas tout à fait identique :

- chaque étape de type E non-triviale change l'affectation d'au moins un point de donnée et donc décroît strictement l'énergie ;
- chaque étape de type M non-triviale décroît l'énergie au sens large en replaçant au moins un centre de cluster à la médiane de son cluster (non-unique minimiseur d'après les questions précédentes).

Donc **l'énergie décroît seulement au sens large**. Toutefois, à la fin de chaque itération k de l'algorithme on se retrouve avec une partition σ_k telle que chaque centre est situé à la médiane de son cluster comme défini dans la question précédente. Par conséquent, en cas de mise à jour

lors de l'itération $k+1$, l'étape E doit être non-triviale et la partition doit changer ($\sigma_{k+1} \neq \sigma_k$). Par ailleurs, l'énergie atteinte à la fin de l'itération k est minimale pour la partition σ_k, et donc l'énergie atteinte à la fin de l'itération $k+1$ doit être strictement plus petite (du fait que l'étape E non-triviale induit une diminution stricte de l'énergie, comme on l'a vu). Il découle que $\sigma_k \neq \sigma_l$ pour tout $l > k$, et donc l'algorithme ne peut effectuer plus d'itérations qu'il y a de partitions distinctes de l'ensemble de points, soit un nombre fini.

Question 1.6. Le raisonnement est le même que dans le cas des k-moyennes :

- D'abord, la convexité des régions de Voronoï implique que chaque cluster optimal est contigu. On peut donc ne considerer que des clusters $C_1, C_2, \cdots, C_k$ contigus et les trier en fonction de l'ordre total sur leurs points.

- Ensuite, en notant j l'indice du plus petit point dans le cluster C_k, on remarque que les clusters $C_1, \cdots, C_{k-1}$ minimisent forcément l'énergie pour $k-1$ clusters sur l'ensemble $\{1, \cdots, j-1\}$, car sinon une autre partition de moindre énergie sur $\{1, \cdots, j-1\}$ pourrait être complétée en une partition de moindre énergie que $C_1, \cdots, C_k$ sur $\{1, \cdots, n\}$ en lui adjoignant simplement C_k. Dès lors, trouver la partition optimale pour (n, k) revient à choisir l'indice j tel que l'énergie atteinte par la partition optimale pour $(j-1, k-1)$, additionnée à l'énergie du k-ème cluster (exprimée comme la somme des distances à sa médiane, comme vu précemment), donne le montant minimal. D'où la récurrence.

Problème 2 : Optimalité du classifieur 1-NN

Question 2.1. Ici la variable Y prend ses valeurs dans l'ensemble $\{1, \cdots, \kappa\}$, et la perte est nulle si $Y = y^*$ et égale à 1 si $Y \neq y^*$. D'où l'espérance :

$$\sum_{y \neq y^*} p_y(x) = 1 - p_{y^*}(x).$$

Question 2.2. La perte est nulle si $y_1 = y_2$ et égale à 1 si $y_1 \neq y_2$. Comme les deux tirages y_1, y_2 sont indépendants, la probabilité que $y_1 \neq y_2$ est

$$\mathbb{P}(y_1 \neq y_2) = \sum_{y=1}^{\kappa} \mathbb{P}(y_1 = y \text{ and } y_2 \neq y) = \sum_{y=1}^{\kappa} p_y(x)(1 - p_y(x)).$$

Question 2.3. Pour tout y on a $1 - p_y(x) \geq 1 - p_{y^*}(x)$ donc

$$\sum_{y=1}^{\kappa} p_y(x)(1 - p_y(x)) \geq (1 - p_{y^*}(x)) \sum_{y=1}^{\kappa} p_y(x) = 1 - p_{y^*}(x).$$

Question 2.4. On a $p_1(x) + p_2(x) = 1$ donc, en supposant sans perte de géneralité que $y^* = 1$, on a

$$\begin{aligned} p_1(x)(1 - p_1(x)) + p_2(x)(1 - p_2(x)) &= p_1(x)(1 - p_1(x)) + (1 - p_1(x))(1 - (1 - p_1(x))) \\ &= p_1(x)(1 - p_1(x)) + (1 - p_1(x))p_1(x) \\ &= 2\, p_1(x)(1 - p_1(x)) = 2\, p_{y^*}(x)(1 - p_{y^*}(x)). \end{aligned}$$

Ainsi, comme $p_{y^*}(x) \leq 1$, l'erreur obtenue est majorée par $2(1 - p_{y^*}(x))$, qui correspond à 2 fois l'erreur de Bayes.

Question 2.5. On sépare le terme en y^* des autres :

$$\sum_{y=1}^{\kappa} p_y(x)(1 - p_y(x)) = p_{y^*}(x)(1 - p_{y^*}(x)) + \sum_{y \neq y^*} p_y(x)(1 - p_y(x)).$$

On remarque que $p_{y^*}(x) \leq 1$ dans le premier terme et $1 - p_y(x) \leq 1$ dans le deuxième, donc :

$$\sum_{y=1}^{\kappa} p_y(x)(1 - p_y(x)) \leq 1 - p_{y^*}(x) + \sum_{y \neq y^*} p_y(x) = 2\,(1 - p_{y^*}(x))\,.$$

Problème 3 : Convergence de l'algorithme du perceptron

Question 3.1. Comme x_i est mal classifié par $\hat{\beta}^{\text{old}}$, on remplace ce dernier par $\hat{\beta}^{\text{new}} = \hat{\beta}^{\text{old}} + y_i\, x_i$, ce qui donne :

$$\left\|\hat{\beta}^{\text{new}} - \beta\right\|^2 = \left\|\hat{\beta}^{\text{old}} + y_i\, x_i - \beta\right\|^2.$$

On développe ensuite le terme de droite :

$$\begin{aligned}\left\|\hat{\beta}^{\text{old}} + y_i\, x_i - \beta\right\|^2 &= \left\|\hat{\beta}^{\text{old}} - \beta\right\|^2 + 2\, y_i\, x_i^T\, (\hat{\beta}^{\text{old}} - \beta) + y_i^2\, \|x_i\|^2 \\ &= \left\|\hat{\beta}^{\text{old}} - \beta\right\|^2 + 2\, y_i\, x_i^T\, \hat{\beta}^{\text{old}} - 2\, y_i\, x_i^T\, \beta + y_i^2\, \|x_i\|^2.\end{aligned}$$

Le deuxième terme de cette somme est négatif ou nul car x_i est mal classifié par $\hat{\beta}^{\text{old}}$. Le troisième terme est majoré par -2 car d'après la définition de β on a $y_i\, x_i^T\, \beta \geq 1$. Enfin le dernier terme est majoré par 1 car $y_i \in \{-1, 1\}$ et par hypothèse $\|x_i\| \leq 1$. Au total, on obtient :

$$\left\|\hat{\beta}^{\text{new}} - \beta\right\|^2 \leq \left\|\hat{\beta}^{\text{old}} - \beta\right\|^2 + 0 - 2 + 1 = \left\|\hat{\beta}^{\text{old}} - \beta\right\|^2 - 1.$$

Question 3.2. Ainsi, chaque fois que le vecteur $\hat{\beta}$ est mis à jour par l'algorithme, il se rapproche du vecteur β d'au moins 1 unité en norme 2 au carré. Comme $\|\hat{\beta} - \beta\|^2 = \|\beta\|^2 < +\infty$ initialement, l'algorithme effectue au plus $\lceil \|\beta\|^2 \rceil$ mises à jour de $\hat{\beta}$. Et comme au moins une mise à jour intervient par époque avant convergence, le nombre total d'époques traversées est au plus $\lceil \|\beta\|^2 \rceil + 1$, le $+1$ provenant de la toute dernière époque — celle où la convergence est constatée par l'algorithme.

Question 3.3. On pose $n = 4$ et on place les observations de la manière suivante dans le plan :

- $x_1 = (1, 0)$ avec label $y_1 = +1$
- $x_2 = (0, 1)$ avec label $y_2 = -1$
- $x_3 = (-1, 0)$ avec label $y_3 = +1$
- $x_4 = (0, -1)$ avec label $y_4 = -1$

On a alors la séquence suivante de vecteurs $\hat{\beta}$:

$$0 \longrightarrow \left[\begin{smallmatrix}1\\0\end{smallmatrix}\right] \longrightarrow \left[\begin{smallmatrix}1\\-1\end{smallmatrix}\right] \longrightarrow \left[\begin{smallmatrix}0\\-1\end{smallmatrix}\right] \longrightarrow 0,$$

séquence qui se répète indéfiniment.

Problème 4 : Réseaux de perceptrons et formules booléennes

Question 4.1. Posons $\beta_1 = \beta_2 = -1$ et $\beta_0 = -1.5$. On a alors

$$1.5 - (x_1 + x_2) \leq 0 \iff x_1 + x_2 \geq 1.5 \iff x_1 = x_2 = 1,$$

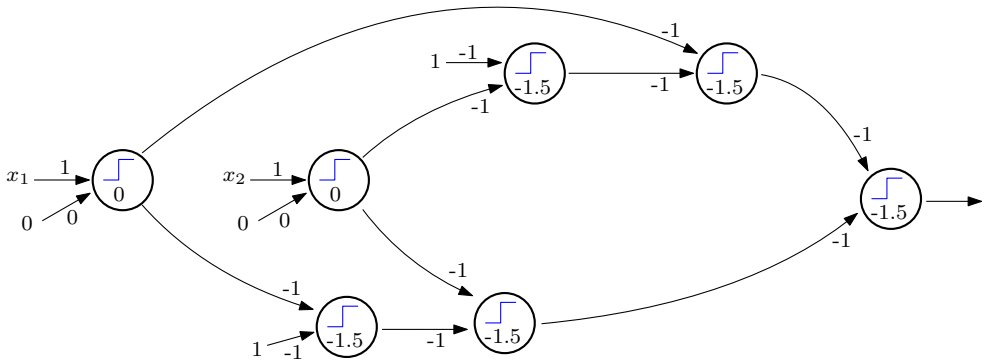
ce qui fait que $y = \mathbb{1}_{1.5-(x_1+x_2)>0}$ simule bien la porte logique `NAND`.

Question 4.2. Posons $x_2 = 1$, ce qui donne

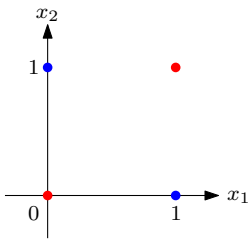
$$1.5 - (x_1 + 1) \leq 0 \iff x_1 \geq 0.5 \iff x_1 = 1.$$

Ainsi le neurone simule bien la porte `NOT`.

Question 4.3.



Question 4.4. On se place dans l'espace $\{0,1\}^2$, que l'on plonge dans le plan $\mathbb{R}^2$:



En bleu on marque les valuations de (x_1, x_2) telles que x_1 `XOR` $x_2 = 1$, en rouge les valuations telles que x_1 `XOR` $x_2 = 0$. On remarque immédiatement qu'il n'existe pas de droite séparant les deux classes (bleue et rouge). Dès lors, un unique perceptron (qui produit forcément un séparateur linéaire) ne peut discriminer les valuations de `XOR` positif des valuations de `XOR` nul.

Problème 5 : Questions de cours et de C++

1. **A.** Effectivement, les prédicteurs k-NN ne requièrent aucun entraînement.

2. **D.** Effectivement, comme la variable i n'est pas initialisée elle contient une valeur arbitraire.

3. **C**. Effectivement, par élimination : le perceptron requiert des labels, qu'on n'a pas ici ; k-means produit forcément des clusters convexes et donc ne peut discriminer les spirales complètement l'une de l'autre ; single-linkage souffre du chaining effect du fait de l'échantillonnage. Reste DBSCAN, qui n'est pas troublé par le chaining effect puisque ce dernier génère une relativement faible densité locale, et permet donc de bien séparer les spirales. C'est la méthode de choix ici.

4. **C**. Effectivement, l'application directe de la formule du cours pour le F-score donne 16/22.

5. **D**. Effectivement, c'est le fameux problème du diamant en cas d'héritage non-virtuel. Attention à l'ordre des appels des constructeurs des classes-mères.

6. **B**. Effectivement, par élimination : k-means est non-supervisé donc peu adapté au problème considéré ; le classifieur k-NN en grande dimension (200 ici) requiert un temps quasi-linéaire en la taille du training set (ici 10 000) et en la dimension pour une seule prédiction, soit un nombre total d'opérations de l'ordre de $10^4 \times 2.10^2 \times (10^6 - 10^4) \approx 10^{12}$ pour effectuer l'ensemble des prédictions, ce qui n'est pas envisageable en première instance ; le perceptron semble adapté au problème mais il donnera n'importe quoi si les deux classes ne sont pas linéairement séparables. Reste la machine à vecteurs de supports, qui est bien adaptée au problème et dont le comportement sera plus stable en cas de non-séparabilité linéaire des classes. C'est la méthode de choix ici.

7. **E**. Sans commentaire.