# Partiel

Notations : on considérera généralement un espace de description des exemples $X \in \mathbb{R}^d$, un ensemble de labels binaire $Y = \{-1, +1\}$, un ensemble d'apprentissage de $n$ exemples $E = \{(x^i, y^i) \in (X, Y)\}, i \in \{1, \ldots, n\}$. Pour une formule booléenne *expr*, Nous noterons $\mathbf{1}_{expr}$ la fonction caractéristique, qui renvoie 1 si l'expression est vraie, 0 sinon.

## Exercice 1 (6 points) – Questions indépendantes

Soit $\mathcal{F} = \{f_{\mathbf{w}} : \mathbb{R}^d \to \mathbb{R}\}$ une famille de fonctions paramétrées par un paramètre $\mathbf{w}$, une fonction de coût $L(y, \hat{y}) : \mathbb{R} \times \mathbb{R} \to \mathbb{R}^+$ et des données issues d'une distribution jointe $p(\mathbf{x}, y)$ avec $\mathbf{x} \in X$ et $y \in Y$.

**Q 1.1** Famille de fonctions

**Q 1.1.1** Rappelez la formalisation du problème de classification.

**Q 1.1.2** Donnez une définition brève des concepts de sur-apprentissage et sous-apprentissage.

**Q 1.1.3** Donnez une représentation graphique de la frontière de décision pour chaque famille de fonctions ci-dessous (avec *sign* la fonction de décision appliquée à la sortie de chacune d'entre elles). Classez-les selon leur risque de sur-apprentissage (de la moins expressive à la plus expressive) :

- $\mathcal{F}_a = \{f_{\mathbf{w}}(\mathbf{x}) =< \mathbf{w}, \mathbf{x} > +w_0,\ \mathbf{w} \in \mathbb{R}^d\}$
- $\mathcal{F}_b = \{f_{i,\theta}(\mathbf{x}) = 2 \times \mathbf{1}_{x_i > \theta} - 1,\ i \in [1, d],\ \theta \in \mathbb{R}\}$
- $\mathcal{F}_c = \{f_{\mu,r}(\mathbf{x}) = 2 \times \mathbf{1}_{||\mathbf{x}-\mu||^2 < r} - 1,\ \mu \in \mathbb{R}^d,\ r \in \mathbb{R}\}$
- $\mathcal{F}_d = \{f_{\mu,\mathbf{r},\mathbf{w}}(\mathbf{x}) = \sum_{k=1}^{p} w_k f_{\mu_\mathbf{k}, r_k}(\mathbf{x}),\ f_{\mu_\mathbf{k}, r_k} \in \mathcal{F}_c, \mathbf{w} \in \mathbb{R}^p,\ p \in \mathbb{N}\}$ combinaison linéaire de fonctions de $\mathcal{F}_c$
- $\mathcal{F}_e = \{f_{\mathbf{w},\mathbf{w}'}(\mathbf{x}) =< \mathbf{x}, \mathbf{w} > + < \mathbf{w}', \mathbf{x} >,\ \mathbf{w}, \mathbf{w}' \in \mathbb{R}^d\}$

**Q 1.1.4** Parmi les familles précédentes, lesquelles sont capables de sur-apprendre n'importe quel jeu de données ?

**Q 1.2** Classification bayésienne

**Q 1.2.1** Rappelez la définition du classifieur bayésien et le principe du classifieur naïf bayésien.

**Q 1.2.2** Est-il possible de construire un classifieur bayésien équivalent à un classifieur de $\mathcal{F}_b$ ? $\mathcal{F}_c$ ?

**Q 1.2.3** Même question pour un classifieur naïf bayésien.

**Q 1.3** Répondre par oui ou non en justifiant brièvement.

**Q 1.3.1** La solution optimale de la régression linéaire avec un critère moindres carrés est unique.

**Q 1.3.2** La solution optimale d'un perceptron est en général unique.

**Q 1.3.3** La régression logistique est une régression non linéaire.

**Q 1.3.4** Augmenter le nombre de couches d'un réseau de neurones décroît toujours l'erreur en test.

**Q 1.3.5** Il est toujours possible de construire un réseau de neurones qui permet de reconnaître une partition quelconque d'exemples en deux classes dans $X$.

## Exercice 2 (4 points) – Réseau de neurones

**Q 2.1** On suppose un réseau de neurones à deux entrées, une couche cachée de deux neurones et un neurone de sortie. Pour cette question on suppose une fonction d'activation linéaire.

**Q 2.1.1** Dessinez ce réseau de neurone en précisant les poids.

**Q 2.1.2** Est-il possible de représenter ce réseau par un perceptron ? Si oui donner son équivalent (et les valeurs des poids associés), sinon expliquez pourquoi.

**Q 2.2** On considère pour cette question que les exemples sont décris sur $d$ variables binaires $x_i \in \{0, 1\}$. Peut-on pour les quatre familles de classifieurs ci-dessous construire un réseau de neurones équivalent en ne considérant que des fonctions d'activations linéaires ou à seuil ($f_\theta(x) = \mathbf{1}_{x>\theta}$) ? (vous avez le droit de transformer les entrées en considérant par exemple le logarithme des entrées). Explicitez dans le cas affirmatif sur quelques lignes le principe de construction du réseau de neurone.

**Q 2.2.1** Formule logique quelconque qui utilise comme opérateur des conjonctions, disjonctions et négations entre les entrées.

**Q 2.2.2** Arbres de décision binaires.

**Q 2.2.3** 1-plus proche voisin.

## Exercice 3 (6 points) – Matching pursuit

Un algorithme classique et très simple pour trouver une solution linéaire $f_{\mathbf{w}}(\mathbf{x}) = \sum_{j=1}^{d} x_j w_j$ $\mathbf{w} \in \mathbb{R}^d$ pour la classification d'un ensemble de données $E$ est l'algorithme itératif du *Matching Pursuit*. Il suppose que l'on dispose d'une fonction $\Omega$ capable de mesurer *l'intérêt* d'une dimension $j$ de l'espace de description de $X$.

1. Au départ : $w_j = 0, \forall j$
2. A chaque itération, la fonction $\Omega$ permet de sélectionner la dimension $j$ la plus *intéressante*.
3. Le paramètre $w_j$ correspondant est mis à jour selon une méthode définie plus tard.

Dans ce type de stratégie, $d$ est souvent très grand et on souhaite conserver beaucoup de $w_j$ nuls. On limitera donc le nombre d'itération de l'algorithme.

**Q 3.1** Nous allons bâtir notre raisonnement sur la fonction coût des moindres carrés $L(y, f_{\mathbf{w}}(\mathbf{x}))$. Rappelez l'expression de cette fonction sous forme analytique et matricielle.

**Q 3.2** A chaque itération $t$ nous noterons $\mathbf{w}^t$ le vecteur des paramètres et nous sélectionnerons un poids $w^t_{j^t}$ grâce à la fonction $\Omega(\mathbf{w}^t) = X^T(X\mathbf{w}^t - Y) : j^t = argmax_i |\Omega(\mathbf{w}^t)_i|$, sélection de la dimension $j$ de $\Omega(\mathbf{w}^t)$ qui a la plus grande valeur, quelque soit son signe.

**Q 3.2.1** Montrer que $\Omega(\mathbf{w})$ appartient à $\mathbb{R}^d$ à l'aide d'un schéma représentant les différentes matrices et leurs dimensions dans l'équation ci-dessus.

**Q 3.2.2** A quoi correspond la fonction $\Omega$ présentée ci-dessus ? Comment interpréter la sélection d'un $w_j$ ?

**Q 3.2.3** Dans ce type d'algorithme, il est très important de normaliser les données, c'est à dire de pondérer les dimensions du problèmes de sorte que les variables aient une moyenne de 0 et un écart-type de 1. Les données sont donc pre-traitées et votre matrice $X$ satisfait les conditions suivantes :

$$\forall j, \quad \bar{x}_j = \frac{1}{n}\sum_{i=1}^{n} x_{ij} = 0, \quad \frac{1}{n}\sum_{i=1}^{n} (\bar{x}_j - x_{ij})^2 = 1$$

Quel problème se pose-t-il si une des variables a une moyenne nettement supérieure aux autres ?

**Q 3.3** A l'itération $t$, une fois que le paramètre $w_{j^t}^t$ est sélectionné, il est mis à jour selon la formule :

$$w_{j^t}^{t+1} \leftarrow w_{j^t}^t - \sum_{i=1}^{n} x_{ij^t}(f_{\mathbf{w}^t}(x_i) - y_i)$$

Donner une interprétation de cette formule en quelques lignes. A quoi correspond la valeur calculée ?

**Q 3.4** Proposer au moins 2 critères d'arrêt basiques pour cet algorithme itératif (dont un par rapport à $\Omega$)

**Q 3.5** Proposer une implémentation python pour l'algorithme du Matching Pursuit.

## Exercice 4 (6 points) – Perceptron avec rejet

Dans certains domaines d'application, il est préférable pour un classifieur de décider qu'il n'est pas capable de classer un exemple plutôt que de prendre le risque de prédir un mauvais label : un tel exemple est considéré comme *rejeté* par le classifieur. Dans cet exercice, on propose d'étudier une adaptation du perceptron à ce contexte [1].
Pour cela, on considère la famille de classifieurs de la forme :

$$h_{\mathbf{w},\lambda}(\mathbf{x}) = \begin{cases} +1 & \text{if} \quad < \mathbf{x}, \mathbf{w} > \geq \lambda_{+1} \\ -1 & \text{if} \quad < \mathbf{x}, \mathbf{w} > \leq -\lambda_{-1} \\ 0 & \text{sinon Rejet} \end{cases}$$

avec $\mathbf{w} \in \mathbb{R}^d$ et $\lambda = (\lambda_{+1}, \lambda_{-1}) \in \mathbb{R}^+ \times \mathbb{R}^+$.
La zone de rejet est délimitée dans ce cas par deux séparatrice appelées marges. Dans la suite de l'exercice, on considérera un problème de classification binaire, à deux classes, $\{-1, +1\}$.

**Q 4.1** Représentez sur un exemple en 2d la séparatrice et les marges. Que représentent $\mathbf{w}$, $\lambda_{-1}$, $\lambda_{+1}$ ?

Nous allons considérer par la suite le coût suivant $L(h(\mathbf{x}), y) = \begin{cases} 0 & \text{if} \quad h(\mathbf{x}) = y \\ \gamma & \text{if} \quad h(\mathbf{x}) = 0, \quad 0 < \gamma < 1 \\ 1 & \text{if} \quad h(\mathbf{x}) \neq y, h(\mathbf{x}) \neq 0 \end{cases}$

**Q 4.2** Donnez un exemple concret de problème de classification où un tel contexte est justifié.

**Q 4.3** Donnez la formulation du problème d'apprentissage sur une base d'exemples $\{(\mathbf{x}^i, y^i)\}$, $i \in \{1, ..., n\}$, en particulier les paramètres à optimiser. Est-il possible d'optimiser cette fonction ? Justifiez.

**Q 4.4** Une manière de relaxer le problème d'optimisation est de considérer un coût *surrogate*, c'est-à-dire un coût approché non optimal mais qui peut être optimisé (c'est par exemple le cas du coût *hinge loss* du perceptron par rapport au coût 0-1). On propose de considérer la fonction de coût suivante :

$$L(h(\mathbf{x}), y) = \begin{cases} -\gamma\tau_c + (1-\gamma)\tau_r & \text{si } h(\mathbf{x}) = 0 \\ -\tau_c + (1-\gamma)\tau_r & \text{si } h(\mathbf{x}) \neq 0 \\ 0 & \text{sinon} \end{cases} \quad \text{avec} \quad \begin{array}{l} \tau_c = \lambda_y - y < \mathbf{w}, \mathbf{x} > \\ \tau_r = \lambda_{-y} + y < \mathbf{w}, \mathbf{x} > \end{array}$$

1. Ramasubramanian et al. : 2004, *WSEAS TS*

pour un exemple $\mathbf{x}$ de la classe $y$

**Q 4.4.1** Que représentent $\tau_c$ et $\tau_r$ ? Quelles est leur signe en fonction de la bonne ou mauvaise classification d'un exemple, ou d'un rejet de l'exemple ?

**Q 4.4.2** Que représentent les différents termes du coût selon la valeur de $h(\mathbf{x})$ ?

**Q 4.5** Proposez un algorithme pour optimiser le coût précédent.

**Q 4.6** Donnez l'implémentation python de votre algorithme.