28. Pour un réseau de neurones à au moins deux couches cachées, ajouter des neurones dans une couche ne permet pas d'augmenter les performances.
29. Un coût $max(0, 1 - f(x)y)$ est généralement meilleur que $max(0, 0.1 - f(x)y)$ car il permet d'augmenter la marge des données avec les frontière de décision.
30. Pour apprendre un réseau de neurones, il suffit de connaître le gradient de la fonction de coût par rapport aux entrées de chaque couche et par rapport aux paramètres de chaque couche.

1-F, 2-V, 3-V, 4-F,5-F,6-F,7-V,8-F,9-F,10-V,11-V,12-V,13-F,14-F,15-F,16-ambigu,17-V,18-F,19-F,20-V,21-F,22-V,23-F,24-V,25-F,26-F,27-V,28-F,29-F, 30-V

Exercice 2 (4 points) — **Régression logisique économe**

On considère le problème de régression logistique linéaire (sans biais pour simplifier) pour la classification binaire entre deux classes $\{-1, +1\}$. On rappelle l'hypothèse dans ce contexte : $p(y = +1|\mathbf{x}) = \frac{1}{1+e^{-f_{\mathbf{w}}(\mathbf{x})}}$. On considère un ensemble de données $D = \{(\mathbf{x}^i, y^i)\}_{i=1}^N$ avec $\mathbf{x}^i \in \mathbb{R}^d$ et $y^i \in \{-1, +1\}$, avec $\mathbf{x}^i = (x_1^i, \ldots, x_d^i)$.

Q 2.1 (0.5 points) Donner l'expression de la vraisemblance $L(D; \mathbf{w})$ sur le jeu de données D, puis celle du logarithme de la vraisemblance $LL(D; \mathbf{w})$ en fonction de $f_{\mathbf{w}}(\mathbf{x}^i)$ et de y^i. Cherche-t-on à la maximiser ou à la minimiser ?

$L(\mathbf{w}, D) = \prod_{i|y_i=+} \frac{1}{1+e^{-f_{\mathbf{w}}(\mathbf{x}^i)}} \prod_{i|y_i=-} \frac{1}{1+e^{f_{\mathbf{w}}(\mathbf{x}^i)}} = \prod_i \frac{1}{1+e^{-y^i f_{\mathbf{w}}(\mathbf{x}^i)}}$ à maximiser (on rappelle que $1 - \sigma(\mathbf{x}) = \sigma(-\mathbf{x})$).

$NL(\mathbf{w}, D) = -\sum_i log(1 + e^{-y^i f_{\mathbf{w}}(\mathbf{x}^i)})$

Q 2.2 (0.5 points) On veut pénaliser la log-vraisemblance $LL(D; \mathbf{w})$ du modèle par une pénalité $L2$ sur le vecteur de poids $\mathbf{w}$, soit par $\frac{\lambda}{2}\|\mathbf{w}\|^2$ avec λ la constante de pénalisation. Donner le coût à minimiser en fonction de y^j, x_i^j et des w_i les coordonnées de $\mathbf{w}$ (on rappelle que $w_0 = 0$ pour simplifier).

$L(\mathbf{w}, D) = -\sum_i log(1 + e^{-y^i f_{\mathbf{w}}(\mathbf{x}^i)}) + \lambda\|\mathbf{w}\|^2 = -\sum_i log(1 + e^{-y^i(\sum_j w_j x_j^i)}) + \lambda(\sum_j w_j^2)$

On considère dans la suite que les exemples $\mathbf{x}^i$ sont très parcimonieux : pour chaque exemple, il n'y a que s dimensions en moyenne non nulles, avec s très inférieure à d.

Q 2.3 (0.5 points) Donner dans le cas $\lambda = 0$ la règle de mise-à-jour pour l'algorithme de descente de gradient pour un pas de gradient ϵ sur l'exemple $(\mathbf{x}^i, y^i)$ (descente stochastique). Quelle est la complexité computationnelle en fonction de s ?

$w_j = w_j + \epsilon \frac{y^i x_j^i}{1+e^{y^i(\sum_j w_j x_j)}}$

Complexité de s en moyenne, la maj est à 0 pour $d - s$ et le calcul de $f_{\mathbf{w}}(\mathbf{x})$ est également en s.

Q 2.4 (0.5 points) On considère $\lambda > 0$, donner dans ce cas la règle de mise-à-jour.

$w_j = w_j + \epsilon \frac{y^i x_j^i}{1+e^{y^i(\sum_j w_j x_j)}} - \epsilon\lambda w_j$

Q 2.5 (0.5 points) Soit $\mathbf{w}^t$ le paramètre à l'itération t, on considère k itérations supplémentaire de descente de gradient stochastique jusqu'à $t + k$. Exprimer $\mathbf{w}_i^{t+k}$ en fonction de $\mathbf{w}_i^t$, k, ϵ et λ si on ne considère que des exemples tels que $\mathbf{x}_i^j = 0$ durant ces k itérations.

$w_j^{t+1} = w_j^t - \epsilon\lambda w_j^t$, donc $w_j^{t+k} = w_j^t(1 - \epsilon\lambda)^k$.

Q 2.6 (1.5 points) Proposer un algorithme d'apprentissage efficace dans le cas d'exemples parcimonieux. Quel est le temps moyen par exemple ?

Mémoriser la derniere fois qu'on a mis à jour l'indice j.

- initialisation de la mémoire $c_j = 0$ pour toutes les dimensions
- Pour choix stochastique d'exemple x^t
  - Calculer $g = 1/(1 + f_w(\mathbf{x}^t))$ (en O(s))
  - Pour tout j tq $x_j^t! = 0$: (en O(s))
    - $k = t - c_j$
    - $w_j = w_j(1 - \epsilon\lambda)^k$
    - $w_j = w_j + \epsilon x_j^t g$
    - $c_j = t$

Temps moyen en O(s)

Exercice 3 (4 points) – **Risque balancé**

On considère un problème de classification à deux classes C_1 et C_2, on suppose $p(x|C_k) = \frac{2x}{a_k^2} e^{-\frac{x^2}{a_k^2}}$ avec $a_k \in \mathbb{R}$ les deux paramètres inconnus et $x \in \mathbb{R}^+ - \{0\}$. Soit $\mathcal{D} = \{(x^i, y^i)\}_{i=1}^N$ un ensemble d'apprentissage. On notera N_k le nombre d'exemples de la classe C_k.

Q 3.1 Pour une classe C_k donnée, on veut déterminer à partir des données le paramètre a_k de sa loi par maximum de vraisemblance.

Q 3.1.1 (0.5 point) Donner l'expression de la log-vraisemblance pour le paramètre a_k de la classe C_k.

$L(D|C_k) = \frac{2^{N_k}}{a_k^{2N_k}} \prod_{i|y^i=C_k} x^i \exp(\frac{-x^{i^2}}{a_k^2})$ donc $LL(D|C_k) = N_k \log(\frac{2}{a_k^2}) + \sum_{i|y^i=C_k} log(x^i) - \frac{1}{a_k^2}\left(\sum_{i|y^i=C_k} x^{i^2}\right)$

Q 3.1.2 (1 point) En déduire l'estimation de $\hat{a}_k$.

On dérive $dL/da_k = -2N_k/a_k + 2\frac{\sum_{i|y^i=C_k} x^{i^2}}{a_k^3}$

On cherche la racine : $2N_k a_k{}^2 = 2(\sum_{i|y^i=C_k} x^{i^2})$, soit $\hat{a}_k = \sqrt{\frac{\sum_i x^{i^2}}{N_k}}$

Q 3.2 On utilise un classifieur bayésien pour classer les données, mais avec des coûts asymétriques : $l_{12} = 2\beta$ si l'exemple était de classe C_2 mais qu'on a prédit C_1 et $l_{21} = \beta$ dans le cas contraire. On considère par ailleurs les classes équiprobables.

Q 3.2.1 Donner l'expression du risque $R(C_k|x)$ pour les deux classes.

$R(C_1|x) = 2\beta p(C_2|x)$, $R(C_2|x) = \beta p(C_1|x)$.

Q 3.2.2 Quelle est la condition sur $P(C_1|x)$ et $P(C_2|x)$ pour classer x comme C_1 ?

$R(C_2|x)/R(C_1|x) = \frac{P(C_1|x)}{2P(C_2|x)}$ donc on classe en C_2 si le rapport est supérieur à 2.

Q 3.2.3 Donner l'expression de la frontière de décision en fonction de x et $\hat{a}_k$.

La frontière : $P(C_2|x) = 2P(C_1|x)$, $P(x|C_2)P(C_2) = 2P(x|C_1)P(C_1)$, $\frac{x}{a_2^2}e^{-x^2/a_2^2} = 2\frac{x}{a_1^2}e^{-x^2/a_1^2}$ Soit $e^{-x^2(1/a_2^2 - 1/a_1^2)} = 2a_2^2/a_1^2$, soit $x^2 = -a_2^2 a_1^2 log(2a_2^2/a_1^2)/(a_1^2 - a_2^2)$

Exercice 4 (6 points) – SVM ordonné

On considère un problème d'ordonnancement : on considère une question et un ensemble de documents $\{\mathbf{x}^i\}_{i=1}^N$; certains documents sont plus pertinents que d'autres pour cette question. On note $\mathbf{x}^i \succ \mathbf{x}^j$ lorsque le document i est plus pertinent que le document j. On souhaite trouver une fonction $f : \mathbb{R}^d \to \mathbb{R}$ qui permette d'inférer l'ordonnancement à partir d'un document $\mathbf{x}$. En particulier, on souhaite que sur le jeu de données d'entraînement $f(\mathbf{x}^i) > f(\mathbf{x}^j)$ si $\mathbf{x}^i \succ \mathbf{x}^j$.

Pour simplifier le problème, on considère qu'en fait les documents sont partitionnés en deux ensembles : les documents pertinents $S_+ = \{\mathbf{u}^i\}_{i=1}^{n_+}$ et les documents non pertinents $S_- = \{\mathbf{v}^j\}_{j=1}^{n_-}$ avec $n_+ + n_- = n$. La comparaison de deux documents de S_+ ou de deux documents de S_- ne nous importent pas, par contre on souhaite que pour $\mathbf{u}^i \in S_+$ et $\mathbf{v}^j \in S_-$, $f(\mathbf{u}^i) - f(\mathbf{v}^j) > 0$. On considère dans la suite que f est linéaire et paramétrée par $\mathbf{w}$: $f_\mathbf{w}(\mathbf{x}) = \langle \mathbf{w}, \mathbf{x} \rangle$.

On définit le problème d'optimisation suivant :

$$\min_{\mathbf{w}, \xi_{ij}} \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{n_+}\sum_{j=1}^{n_-} \xi_{ij}$$

avec les contraintes $\langle \mathbf{w}, \mathbf{u}^i \rangle - \langle \mathbf{w}, \mathbf{v}^j \rangle \geq 1 - \xi_{ij}$ et $\xi_{ij} \geq 0 \; \forall i = 1, \ldots, n_+, \forall j = 1, \ldots, n_-$

Rappel : Pour un problème d'optimisation $min_{\theta \in \mathbb{R}^d} J(\theta)$ sous les m contraintes $g_j(\theta) \leq 0$ pour $j = 1 \ldots m$, le Lagrangien associé est $\mathcal{L}(\theta, \mu) = J(\theta) + \sum_{j=1}^{m} \mu_j g_j(\theta)$ avec $\mu_j \geq 0$. Les conditions KKT spécifient qu'à l'optimum, nécessairement $\nabla \mathcal{L}(\theta, \mu) = 0$, $g_j(\theta) \leq 0$, $\mu_j \geq 0$, et $\mu_j g_j(\theta) = 0$ pour $j = 1 \ldots m$.

Q 4.1 (1 point) En considérant chaque terme du problème d'optimisation, expliquez en quoi cette formulation répond bien au problème. À quoi sert C ?

- la première contrainte permet de traduire la contrainte de bon ordonnancement avec une marge et une tolérance ξ_{ij}.
- Minimiser les ξ permet controler l'erreur.
- La constance C est une constante de pénalisation pour régler le sur-apprentissage.

Q 4.2 (1 point) Donner le Lagrangien correspondant à ce problème.

$$L(\mathbf{w}, C, \xi, \alpha, \beta) = \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{n_+}\sum_{j=1}^{n_-} \xi_{ij} - \sum_{i=1}^{n_+}\sum_{j=1}^{n_-} \beta_{ij}\xi_{ij} + \sum_{i=1}^{n_+}\sum_{j=1}^{n_-} \alpha_{ij}(1 - \xi_{ij} - \langle \mathbf{w}, \mathbf{u}^i \rangle + \langle \mathbf{w}, \mathbf{v}^j \rangle)$$

Q 4.3 (1.5 point) Écrire les conditions d'optimalité par rapport aux variables $\mathbf{w}$, $\mathbf{u}^i$ et $\mathbf{v}^j$ et en déduire l'expression de $\mathbf{w}$.

- $\nabla_\mathbf{w} L = \mathbf{w} - \sum_{i=1}^{n_+}\sum_{j=1}^{n_-} \alpha_{ij}(-\mathbf{u}^i + \mathbf{v}^j) = 0$
- $\nabla_{\xi_{ij}} = C - \beta_{ij} - \alpha_{ij} = 0$
- $\xi_{ij}\beta_{ij} = 0$
- $\alpha_{ij}(1 - \xi_{ij} - \langle \mathbf{w}, \mathbf{u}^i \rangle + \langle \mathbf{w}, \mathbf{v}^j \rangle) = 0$

Donc $\mathbf{w} = \sum_{i=1}^{n_+}\sum_{j=1}^{n-1} \alpha_{ij}(-\mathbf{u}^i + \mathbf{v}^j)$

Q 4.4 (1.5 point) Quel est le problème dual correspondant ?

On remplace $\mathbf{w}$:

$$L = \frac{1}{2}\| \sum_{ij}^{n_+ n_-} \alpha_{ij}(\mathbf{v}^j - \mathbf{u}^i)\|^2 + C\sum_{ij} \xi_{ij} - \sum_{ij} \beta_{ij}\xi_{ij} - \sum_{ij} \alpha_{ij}(1 - \xi_{ij} + \langle \sum_{ij} \alpha_{i'j'}(\mathbf{u}^{i'} - \mathbf{v}^{j'}), \mathbf{u}^i - \mathbf{v}^j \rangle)$$