Apprentissage et reconnaissance (GIF-4101 / GIF-7005)
Département de génie électrique et de génie informatique
Automne 2016

UNIVERSITÉ LAVAL

EXAMEN FINAL

Instructions : – Une feuille aide-mémoire recto verso manuscrite est permise ;
– Durée de l'examen : 2 h 50.

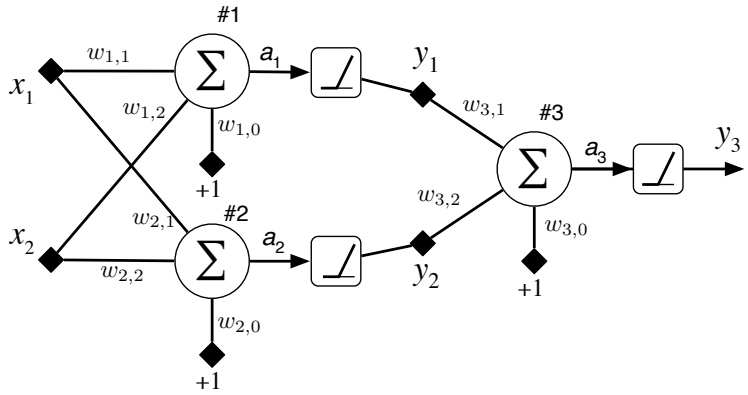
Pondération : – Cet examen compte pour 35 % de la note finale ;
– La note est saturée à 100 % si le total des points avec bonus excède cette valeur.

Question 1 (20 points sur 100)

Soit un réseau de neurones avec comme fonction de transfert la fonction ReLU (*rectified linear unit*) :

$$\mathrm{f}_{\mathrm{ReLU}}(a) = \max(0,a) = \begin{cases} 0 & \text{si } a < 0 \\ a & \text{autrement} \end{cases} .$$

(8) (a) Supposons que l'on veut entraîner le réseau suivant à trois neurones avec activation ReLU, pour résoudre le problème du OU exclusif (XOR).



Données du XOR :

$$\mathbf{x}^1 = [0\ 0]^T \quad r^1 = 0$$
$$\mathbf{x}^2 = [0\ 1]^T \quad r^2 = 1$$
$$\mathbf{x}^3 = [1\ 0]^T \quad r^3 = 1$$
$$\mathbf{x}^4 = [1\ 1]^T \quad r^4 = 0$$

Déterminez les poids et biais des trois neurones du réseau permettant d'obtenir une erreur de classement nulle sur ce problème.

Solution: Une configuration de poids possible est :

$$w_{1,1} = 1, \quad w_{1,2} = 1, \quad w_{1,0} = 0,$$
$$w_{2,1} = 2, \quad w_{2,2} = 2, \quad w_{2,0} = -2,$$
$$w_{3,1} = 1, \quad w_{3,2} = -1, \quad w_{3,0} = 0.$$

(12) (b) Supposons maintenant que l'on fait un apprentissage en ligne (mise à jour pour une instance à la fois) selon l'erreur quadratique moyenne, par une rétropropagation des erreurs. Développez les équations pour mettre à jour les poids sur la couche de sortie de neurones utilisant la fonction d'activation ReLU.

Conseil : Ne vous formalisez pas de la discontinuité de la fonction ReLU pour $a = 0$, vous pouvez formuler la fonction de la dérivée en deux morceaux, comme fait plus haut dans l'énoncé.

Solution: Lors d'un apprentissage en ligne, les poids de la couche de sortie sont ajustés par l'équation suivante :

$$\Delta w_{j,i} = -\eta \frac{\partial E^t}{\partial w_{j,i}}.$$

Selon le chaînage des dérivées, on développe les équations suivantes :

$$\begin{aligned}
\frac{\partial E^t}{\partial w_{j,i}} &= \frac{\partial E^t}{\partial e_j^t}\frac{\partial e_j^t}{\partial y_j^t}\frac{\partial y_j^t}{\partial a_j^t}\frac{\partial a_j^t}{\partial w_{j,i}} \\
\frac{\partial E^t}{\partial w_{j,0}} &= \frac{\partial E^t}{\partial e_j^t}\frac{\partial e_j^t}{\partial y_j^t}\frac{\partial y_j^t}{\partial a_j^t}\frac{\partial a_j^t}{\partial w_{j,0}}
\end{aligned}$$

On calcule les dérivées partielles suivantes :

$$\begin{aligned}
\frac{\partial E^t}{\partial e_j^t} &= \frac{\partial}{\partial e_j^t}\frac{1}{2}\sum_{j=1}^{K}(e_j^t)^2 = e_j^t, \\
\frac{\partial e_j^t}{\partial y_j^t} &= \frac{\partial}{\partial y_j^t} r_j^t - y_j^t = -1, \\
\frac{\partial a_j^t}{\partial w_{j,i}} &= \frac{\partial}{\partial w_{j,i}}\sum_{i=1}^{R} w_{j,i}y_i^t + w_{j,0} = y_i^t, \\
\frac{\partial a_j^t}{\partial w_{j,0}} &= \frac{\partial}{\partial w_{j,0}}\sum_{i=1}^{R} w_{j,i}y_i^t + w_{j,0} = 1.
\end{aligned}$$

Il ne reste qu'à développer la dérivée de la fonction d'activation ReLU :

$$\frac{\partial y_j^t}{\partial a_j^t} = H(a) = \begin{cases} 0 & \text{si } a < 0 \\ 1 & \text{autrement} \end{cases},$$

où la fonction $H(a)$ est nommée fonction *Heaviside*.

Ce qui nous donne pour la couche de sortie :

$$\begin{aligned}
\Delta w_{j,i} &= -\eta \frac{\partial E^t}{\partial w_{j,i}} = -\eta \frac{\partial E^t}{\partial e_j^t} \frac{\partial e_j^t}{\partial y_j^t} \frac{\partial y_j^t}{\partial a_j^t} \frac{\partial a_j^t}{\partial w_{j,i}} \\
&= \eta e_j^t H(a_j^t) y_i^t = \begin{cases} 0 & \text{si } a < 0 \\ \eta e_j^t y_i^t & \text{autrement} \end{cases} \\
\Delta w_{j,0} &= -\eta \frac{\partial E^t}{\partial w_{j,0}} = -\eta \frac{\partial E^t}{\partial e_j^t} \frac{\partial e_j^t}{\partial y_j^t} \frac{\partial y_j^t}{\partial a_j^t} \frac{\partial a_j^t}{\partial w_{j,0}} \\
&= \eta e_j^t H(a_j^t) = \begin{cases} 0 & \text{si } a < 0 \\ \eta e_j^t & \text{autrement} \end{cases} .
\end{aligned}$$

Question 2 (17 points sur 100)

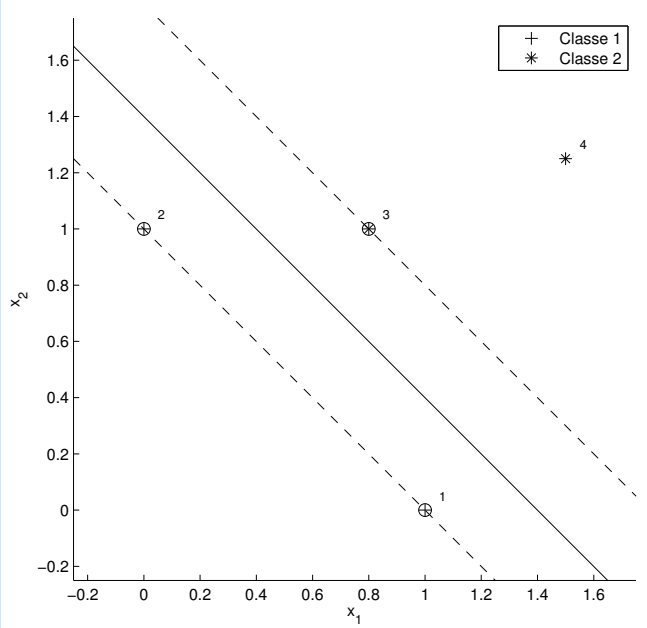
Soit le jeu de données $\mathcal{X} = \{\mathbf{x}^t, r^t\}_{t=1}^4$ présenté ici-bas.

$$\mathbf{x^1} = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \quad \mathbf{x^2} = \begin{bmatrix} 0 \\ 1 \end{bmatrix}, \quad \mathbf{x^3} = \begin{bmatrix} 0{,}8 \\ 1 \end{bmatrix}, \quad \mathbf{x^4} = \begin{bmatrix} 1{,}5 \\ 1{,}25 \end{bmatrix},$$
$$r^1 = -1, \qquad r^2 = -1, \qquad r^3 = 1, \qquad r^4 = 1.$$

Supposons que l'on veut classer ces données avec un classifieur de type Séparateur à vastes marges (SVM) utilisant un noyau linéaire ($K(\mathbf{x},\mathbf{x}') = \langle \mathbf{x},\mathbf{x}' \rangle = (\mathbf{x}')^T \mathbf{x}$), sans marge floue.

(7) (a) Tracez dans votre cahier de réponse les données du jeu $\mathcal{X}$, l'hyperplan séparateur que l'on obtient avec un SVM sur ces données, les marges géométriques maximales correspondantes et encerclez les données agissant comme vecteurs de support.

Solution:



(10) (b) Donnez les valeurs correspondant aux poids $\mathbf{w}$ et biais w_0 du discriminant linéaire maximisant les marges géométriques tracées en (a).

Solution: Trois données sont identifiées comme vecteurs de support : $\mathbf{x}^1$, $\mathbf{x}^2$ et $\mathbf{x}^3$. En travaillant directement dans l'espace d'entrée, ceci nous donne trois équations et trois inconnus.

$$\begin{aligned} \mathrm{h}(\mathbf{x}^1) &= w_2 x_2^1 + w_1 x_1^1 + w_0 = 1w_2 + 0w_1 + w_0 = -1 \\ \mathrm{h}(\mathbf{x}^2) &= w_2 x_2^2 + w_1 x_1^2 + w_0 = 0w_2 + 1w_1 + w_0 = -1 \\ \mathrm{h}(\mathbf{x}^3) &= w_2 x_2^3 + w_1 x_1^3 + w_0 = 1w_2 + 0{,}8w_1 + w_0 = 1 \end{aligned}$$

La résolution de ce système d'équations peut se faire par la suivante.

$$\begin{aligned} \text{EQ1} &: w_2 + w_0 = -1 \\ \text{EQ2} &: w_1 + w_0 = -1 \\ \text{EQ3} &: w_2 + 0{,}8w_1 + w_0 = 1 \\ \text{EQ3} - \text{EQ1} - 0.8 \times \text{EQ2} &: (1-1)w_2 + (0{,}8 - 0{,}8)w_1 + (1 - 1 - 0{,}8)w_0 = 1 + 1 + 0{,}8 \\ &\rightarrow w_0 = -3{,}5 \\ \text{EQ1} - \text{L4} &: (1-0)w_2 + (1-1)w_0 = -1 + 3{,}5 \\ &\rightarrow w_2 = 2{,}5 \\ \text{EQ2} - \text{L4} &: (1-0)w_1 + (1-1)w_0 = -1 + 3{,}5 \\ &\rightarrow w_1 = 2{,}5 \end{aligned}$$

La résolution de ce système d'équations nous donne les valeurs suivantes :

$$\mathbf{w} = [2{,}5 \;\; 2{,}5]^T,\ w_0 = -3{,}5.$$

Question 3 (15 points sur 100)

Soit les réseaux de neurones profonds de type autoencodeur, tel que présenté en classe.

(5) (a) La première phase de l'entraînement d'un autoencodeur est dite non-supervisée. Expliquez en quoi consiste cette phase d'entraînement, en précisant les éléments suivants :
- — Topologie du réseau, en précisant ce à quoi correspond la partie encodeur et la partie décodeur ;
- — Données utilisées pour l'entraînement, incluant valeurs cibles en sortie ;
- — Critère d'erreur utilisé pour l'entraînement ;
- — Fonctionnement de l'algorithme d'entraînement (procédure d'apprentissage et poids modifiés à chaque étape).

Solution:
- — **Topologie** : Un autoencodeur est organisé en deux parties symétriques, soit l'encodeur et le décodeur. L'encodeur est composé de plusieurs couches de neurones

faisant une projection d'un espace d'entrée à D dimensions vers un espace intermédiaire. Le décodeur effectue la transformation inverse de l'espace intermédiaire vers un espace de sortie de la même taille que l'espace d'entrée (D dimensions). Les poids du décodeur sont généralement liés à ceux de l'encodeur, étant en fait la transposée, la correspondance se faisant selon le niveau d'éloignement des entrées sorties, les poids de l'encodeur sur la première couche connectée à l'entrée sont liés aux poids du décodeur sur la dernière couche correspondant à la sortie, et ainsi de suite.

- **Données** : Les données d'entraînement correspondent à des données du problème que l'on veut résoudre. Toutes les données n'ont pas à avoir d'étiquettes. Les valeurs cibles sont en fait les données d'entrée que l'on veut reconstruire dans le processus d'encodage-décodage de l'autoencodeur.
- **Critère d'erreur** : L'erreur de reconstruction, $\sum_t \|\mathbf{x}^t - \hat{\mathbf{x}}^t\|^2$, est généralement utilisée pour guider l'apprentissage de l'autoencodeur.
- **Algorithme d'entraînement** : L'entraînement est effectué de façon itérative, en débutant par les deux couches les plus externes de l'encodeur/décodeur, pour ensuite y aller vers les deux couches qui suivent (2e couche après l'entrée avec 2e couche qui précède la sortie) et ainsi de suite. Les poids de chaque couche de l'encodeur et du décodeur sont liés, généralement en utilisant la transposée des poids de l'encodeur pour le décodeur.

(5) (b) La deuxième phase de l'entraînement d'un autoencodeur est dite de raffinement (*fine-tuning)*. Expliquez en quoi consiste cette deuxième phase, en précisant les éléments suivants :

- Topologie du réseau, en référant aux parties présentées à la question précédente ;
- Données utilisées pour l'entraînement, incluant valeurs cibles en sortie ;
- Critère d'erreur utilisé pour l'entraînement ;
- Fonctionnement de l'algorithme d'entraînement (procédure d'application et poids modifiés à chaque étape).

Solution:

- **Topologie** : Dans la deuxième phase, seule la partie encodeur est conservée. Une couche de sortie supplémentaire est ajoutée pour transformer les données dans la représentation intermédiaire de l'encodeur dans la forme voulue pour la sortie. Cette couche de sortie est initialisée aléatoirement et peut être interprétée comme un classifieur, alors que l'encodeur comporte la représentation apprise des données.
- **Données** : Les données utilisées dans la 2e phase doivent comporter des étiquettes de classes, qui sont utilisées pour comme valeurs cibles en sorties.
- **Critère d'erreur** : Un critère d'erreur standard en classement (erreur quadratique, entropie croisée) est utilisé comme critère d'erreur pour l'apprentissage.
- **Algorithme d'entraînement** : L'apprentissage s'effectue par une rétropropagation standard. Les poids de la nouvelle couche de sortie sont initialisés aléatoirement, alors que les poids de l'encodeur appris à l'étape précédente sont utilisés

pour les autres couches. L'apprentissage se fait sur la couche de sortie et les autres couches, qui sont en fait raffinées par la rétropropagation.

(5) (c) Une force des réseaux profonds est la capacité d'apprendre une représentation à partir des données. Expliquez à quoi correspond cette représentation dans un autoencodeur.

Également, il est courant d'effectuer des transferts de représentation en apprentissage profond. De quelle façon peut-on faire cela à l'aide d'un autoencodeur ? Expliquez cela en termes concrets, en disant comment on peut transférer la représentation d'un autoencodeur apprise sur un certain problème vers un autre problème distinct, mais relaté, en présentant les différentes étapes nécessaires pour y arriver.

Solution: La représentation des données avec des réseaux profonds (incluant autoencodeurs) correspond à l'extraction d'information pertinente à partir des données brutes, généralement pas l'extraction de concepts de bas niveau (sur les couches d'entrées) vers des concepts plus complexes (dans des couches plus profondes). Les valeurs de la représentation intermédiaire devraient représenter des concepts d'assez haut niveau relativement à la tâche à résoudre.

Le transfert de représentation se fait par l'apprentissage d'un autoencodeur sur les données du problème initial, possiblement suivi d'un raffinement sur les données du même problème. Ensuite, la partie encodeur peut être transférée vers le nouveau problème. On peut alors utiliser une nouvelle couche cachée et refaire un nouvel apprentissage (phase de raffinement) avec les données du nouveau problème. Les éléments de représentation appris avec le problème initial devraient se retrouver toujours dans la partie encodeur, étant ajustés à la nouvelle tâche durant le raffinement du réseau.

Question 4 (48 points sur 100)

Répondez aussi brièvement et clairement que possible aux questions suivantes.

(3) (a) Expliquez en quoi diffère l'objectif principal d'un apprentissage de modèles discriminatifs de classement relativement à l'apprentissage de modèles génératifs.

Solution: Dans l'apprentissage de modèles génératifs, l'objectif principal est d'apprendre la densité des données selon les classes, c'est-à-dire le $P(\mathbf{x}|C_i,\theta)$, qui pourra ensuite être utilisé dans les règles de décision. Avec l'apprentissage de modèles discriminatifs, on vise à apprendre directement la règle de décision, modélisé par la fonction $\mathrm{h}(\mathbf{x}|\theta)$ dans le cas à deux classes, sans passer par une étape intermédiaire d'estimation des densités de probabilité par classe.

(3) (b) Expliquez comment on peut interpréter la fonction sigmoïde dans un contexte de classement binaire avec densités normales.

Solution: La fonction sigmoïde peut s'interpréter comme une probabilité *a posteriori* de classe $P(C_i|\mathbf{x})$, dans un contexte où les densités normales par classe partagent la même matrice de covariance.

(3) (c) Expliquez l'intérêt d'utiliser une régularisation l_1, tel qu'utilisé avec LASSO, comparativement à une régularisation l_2, utilisée dans une régression d'arête (*ridge regression*).

Solution: Une régularisation l_1 a l'avantage de favoriser des poids w_i nuls, désactivant ainsi l'effet les variables x_i, permettant ainsi une sélection de variables simultanément à l'apprentissage de la fonction de régression.

(3) (d) Dans le développement des équations pour l'apprentissage des paramètres du SVM, expliquez pourquoi on passe de la forme primale à la forme duale relativement à l'optimisation du modèle.

Solution: Le passage à la forme duale permet d'éliminer le paramètre vectoriel $\mathbf{w}$ de l'optimisation, optimisant uniquement les valeurs α_i et le biais w_0 selon la fonction duale, éliminant par le fait le point de selle dans la fonction objective.

(3) (e) Dans les méthodes à noyau, expliquez le sens de la mesure effectuée avec la fonction noyau entre deux instances.

Solution: La fonction noyau $K(\mathbf{x}_i,\mathbf{x}_j)$ mesure la similarité entre deux instances, c'est-à-dire que plus la valeur est élevée plus les données sont proche ou similaires.

(3) (f) Présentez les principales similarités et différences entre les SVM avec noyau gaussien et les réseaux de neurones RBF.

Solution: Similarité : le modèle de traitement résultant est similaire, sous la forme $\sum_i w_i \exp(\|\mathbf{x} - \mathbf{m}^i\|^2/\sigma^2)$.
Différence : les centres $\mathbf{m}$ des réseaux RBF sont synthétiques et appris à partir des données, alors qu'ils correspondent à des données d'entraînement pour le SVM. L'apprentissage avec les SVM se fait par une méthode déterministe (programmation quadratique), alors que l'apprentissage avec les réseaux RBF est stochastique, sans garantie de convergence à l'optimum global.

(3) (g) Indiquer la taille des vecteurs correspondant aux composantes principales obtenues lors d'une analyse en composantes principales (ACP) **à noyau**.

Solution: Chaque vecteur correspondant à une composante principale obtenue par une ACP à noyau est de taille N, soit le nombre de données utilisées pour le calcul de la matrice de Gram sur laquelle les composantes ont été extraites.

(3) (h) Expliquez pourquoi le critère d'arrêt hâtif (*early stopping*) est particulièrement intéressant et important pour favoriser la généralisation dans l'entraînement d'un perceptron multicouche.

Solution: Le critère d'arrêt hâtif permet d'interrompre l'apprentissage lorsque l'on observe une dégradation significative des performances sur un ensemble de validation (estimation de la généralisation), ce qui est particulièrement utile avec un perceptron

multicouche, qui a tendance à surapprendre significativement s'il est entraîné trop longuement.

(3) (i) Indiquez les trois conditions principales favorisant l'émergence des réseaux profonds dans les dix dernières années.

Solution:

1. Disponibilité de très grands jeux de données (*big data*)
2. Disponibilité d'une capacité de calcul faramineuse (GPU)
3. Nouveaux modèles d'apprentissage très flexibles, avec des *a priori* permettant de bien gérer la malédiction de la dimensionnalité

(3) (j) Expliquez en quoi consiste la méthode d'entraînement dropout avec les réseaux profonds.

Solution: Le dropout consiste à désactiver aléatoirement des neurones du réseau (typiquement, 50 % des neurones) lors de la présentation de chaque donnée pour l'entraînement (choix aléatoire différent de neurones actifs pour chaque donnée).

(3) (k) Expliquez ce que représente le taux d'erreur bayésien optimal en classement.

Solution: La taux d'erreur bayésien optimal représente le meilleur taux d'erreur qu'il est possible d'obtenir si l'on connaît parfaitement la véritable distribution des données selon les classes.

(3) (l) Expliquez le principal avantage associé à l'utilisation de codes à correction d'erreurs pour le traitement des données à plusieurs classes avec un ensemble de classifieurs binaires (à deux classes), comparativement à des approches de type *un contre tous* ou de *décisions par paires*.

Solution: L'utilisation de codes à correction d'erreurs permet l'utilisation de grands ensembles de classifieurs binaires comportant assez de redondance pour tolérer quelques erreurs de classement de membres de l'ensemble.

(3) (m) Expliquez la principale différence entre le *Bagging* et le *Random subspace*.

Solution: Avec le *bagging* on effectue un tirage aléatoire avec remise des données d'entraînement différent pour chaque membre de l'ensemble, alors qu'avec le *random subspace*, c'est le choix des variables d'entrées du classifieur qui est effectué aléatoirement pour chaque membre de l'ensemble.

(3) (n) Quelle est la condition minimale exigée d'un classifieur de type *weak learner* pour l'utiliser avec l'algorithme AdaBoost ?

Solution: Un classifieur de type *weak learner* doit faire mieux que le hasard pour le classement de jeux de données, soit un taux d'erreur inférieur à 50 %.

(3) (o) Lors de planification d'expérimentations, dans quel contexte la stratégie *un facteur à la fois* peut-elle être utilisée ?

Solution: La stratégie *un facteur à la fois* peut être utilisée lorsque les facteurs sont indépendants les uns des autres, c'est-à-dire lorsque les valeurs des autres facteurs n'ont pas d'effets relatifs sur la variation en performance associée à la variation de la valeur d'un facteur particulier.

(3) (p) Dans un graphique de courbe ROC, quel doit être la tracé de la courbe pour s'assurer que la méthode évaluée fait mieux que le hasard en tous points d'opération ?

Solution: La courbe doit être dans le triangle supérieur gauche du tracé, avec des valeurs toujours au-dessus ou égales à la droite allant du point $(0,0)$ au point $(1,1)$.

Question 5 (10 points bonus)

Supposons que la fonction A nous permet d'effectuer l'apprentissage d'une paramétrisation θ d'un classifieur sur le jeu de données $\mathcal{X}$, en utilisant γ comme hyperparamètre de la méthode d'apprentissage :

$$\theta = A(\mathcal{X},\gamma).$$

On dispose également d'une fonction d'erreur $E(\mathcal{X}|\theta)$ pour évaluer la performance du classifieur entraîné $\mathrm{h}(\cdot|\theta)$ sur le jeu de données $\mathcal{X}$.

Supposons maintenant que pour un problème donné nous avons sous la main deux jeux de données, soit un ensemble d'entraînement $\mathcal{S}$ et un ensemble de test $\mathcal{T}$. On veut tester L valeurs différentes d'hyperparamètres données dans l'ensemble $\Gamma = \{\gamma_1,\gamma_2,\ldots,\gamma_L\}$. Pour évaluer la performance de chaque paramétrisation, on veut effectuer une validation croisée à K plis. Donnez le pseudo-code correspondant une méthodologie valide, permettant d'évaluer chacune de ces paramétrisations γ_i et de sélectionner la paramétrisation γ^* qui apparaît optimale en termes d'erreur en généralisation. Tentez d'être aussi formel que possible dans l'énonciation mathématique de votre pseudo-code.

Vous pouvez utiliser la fonction $\mathrm{Part}(\mathcal{X},K)$, qui fait une partition aléatoire stratifiée de l'ensemble de données $\mathcal{X}$ en K sous-ensembles distincts :

$$\{\mathcal{X}_1,\mathcal{X}_2,\ldots,\mathcal{X}_K\} = \mathrm{Part}(\mathcal{X},K).$$

Solution:

$\{\mathcal{S}_1,\mathcal{S}_2,\ldots,\mathcal{S}_K\} = \mathrm{Part}(\mathcal{S},K)$
for all $k \in \{1,\ldots,K\}$ **do**
    $\mathcal{X}_k \leftarrow \mathcal{S}_1 \cup \cdots \cup \mathcal{S}_{k-1} \cup \mathcal{S}_{k+1} \cup \cdots \cup \mathcal{S}_k$
    for all $i \in \{1,\ldots,L\}$ **do**
        $\theta_{i,k} \leftarrow A(\mathcal{X}_k,\gamma_i)$
        $e_{i,k} \leftarrow E(\mathcal{S}_k|\theta_{i,k})$
    end for

8: **end for**
9: $e_i \leftarrow \frac{1}{K} \sum_{k=1,\ldots,K} e_{i,k}, \quad i = 1,\ldots,L$
10: $j \leftarrow \arg\min_{i=1,\ldots,L} e_i$
11: $\gamma^* \leftarrow \gamma_j$