Introduction à l'apprentissage machine (GIF-4101 / GIF-7005)
Département de génie électrique et de génie informatique
Automne 2018

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 30% de la note finale.

Question 1 (14 points sur 100)

Une matrice de décision $\mathbf{W}$, de taille $K \times L$, permet de combiner les décisions d'un ensemble de L classifieurs à deux classes, pour faire du classement de données à K classes. L'équation de décision basée sur cette matrice est la suivante :

$$\bar{\mathrm{h}}_i(\mathbf{x}) = \sum_{j=1}^{L} w_{i,j} \mathrm{h}_{j,i}(\mathbf{x}),$$

où :
— $\mathrm{h}_{j,i}(\mathbf{x})$ est le j-ème classifieur de base de l'ensemble ;
— $w_{i,j}$ est l'élément à la position (i,j) dans la matrice de décision $\mathbf{W}$;
— $\bar{\mathrm{h}}_i(\mathbf{x})$ est la décision combinée de l'ensemble pour la classe C_i.

(4) (a) Supposons que l'on veut résoudre un problème à $K = 5$ classes à l'aide d'un ensemble de classifieurs à deux classes combinés selon la méthode *un contre tous* (en anglais, *one against all*). Donnez le nombre de classifieurs à deux classes à utiliser ainsi que la matrice de décision $\mathbf{W}$ correspondant à cette configuration.

Solution: Avec un ensemble de classifieurs de type *un contre tous*, le nombre de classifieurs à utiliser correspond au nombre de classes traitées, soit $L = K = 5$ classifieurs. La matrice de décision à utiliser pour cette configuration est la suivante :

$$\mathbf{W} = \begin{bmatrix} +1 & -1 & -1 & -1 & -1 \\ -1 & +1 & -1 & -1 & -1 \\ -1 & -1 & +1 & -1 & -1 \\ -1 & -1 & -1 & +1 & -1 \\ -1 & -1 & -1 & -1 & +1 \end{bmatrix}.$$

(4) (b) Supposons maintenant que l'on veut résoudre ce problème à $K = 5$ classes toujours à l'aide d'un ensemble de classifieurs à deux classes, mais cette fois en combinant les classifieurs selon la méthode de *séparation par paires* (en anglais, *pairwise separation*). Donnez le nombre de classifieurs à deux classes à utiliser ainsi que la matrice de décision $\mathbf{W}$ correspondant à cette configuration.

Solution: Le nombre de classifieurs à utiliser correspond au nombre de combinaisons de classes possibles, en tenant compte de la symétrie entre les classifieurs (discriminer C_i de C_j correspondant à discriminer C_j de C_i à un signe près), soit $L = K(K-1)/2 = 10$ classifieurs. La matrice de décision correspondante est la suivante :

$$\mathbf{W} = \begin{bmatrix} +1 & +1 & +1 & +1 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 0 & +1 & +1 & +1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & -1 & 0 & 0 & +1 & +1 & 0 \\ 0 & 0 & -1 & 0 & 0 & -1 & 0 & -1 & 0 & +1 \\ 0 & 0 & 0 & -1 & 0 & 0 & -1 & 0 & -1 & -1 \end{bmatrix}.$$

(6) (c) Finalement, supposons que l'on veut résoudre ce problème à $K = 5$ classes d'un ensemble redondant de $L = 9$ classifieurs, avec une matrice de décision basée sur un code à correction d'erreur (en anglais, *error code output correction*). Donnez la matrice de décision $\mathbf{W}$ correspondant à cette configuration qui maximise la robustesse des décisions d'ensemble. Déterminez également le nombre d'erreurs de classement des classifieurs de base que cette configuration de système peut tolérer sans se tromper.

Solution: Une configuration possible de matrice de décision à code de correction d'erreur est la suivante :

$$\mathbf{W} = \begin{bmatrix} +1 & -1 & -1 & -1 & -1 & -1 & -1 & +1 & -1 \\ -1 & +1 & -1 & +1 & -1 & -1 & +1 & +1 & +1 \\ -1 & -1 & -1 & +1 & +1 & +1 & -1 & -1 & +1 \\ -1 & +1 & +1 & -1 & -1 & +1 & +1 & -1 & -1 \\ +1 & -1 & +1 & -1 & +1 & -1 & +1 & -1 & +1 \end{bmatrix}.$$

La distance de Hamming minimale entre chaque combinaison de lignes est de 5. Ceci indique donc que cette configuration de système peut tolérer deux erreurs de classement ($\lfloor (5-1)/2 \rfloor = 2$) par les classifieurs de base sans prendre une mauvaise décision de classement.

Question 2 (34 points sur 100)

Supposons les données suivantes en deux dimensions :

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

$$\mathbf{x}^4 = \begin{bmatrix} 1{,}5 \\ -1{,}2 \end{bmatrix}, \quad \mathbf{x}^5 = \begin{bmatrix} 0{,}5 \\ -0{,}9 \end{bmatrix}, \quad \mathbf{x}^6 = \begin{bmatrix} 2{,}0 \\ -0{,}4 \end{bmatrix}.$$

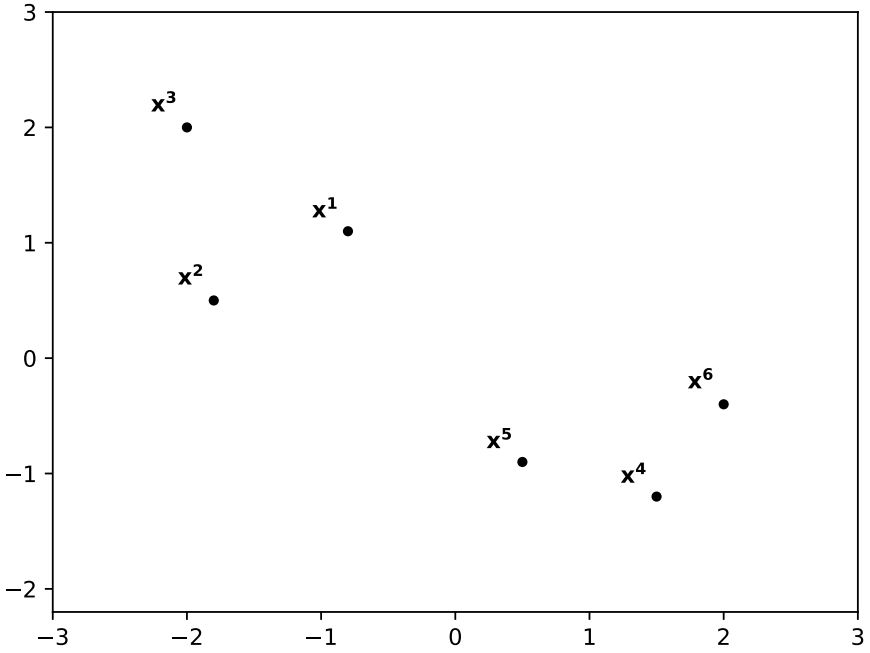
Le vecteur moyen $\mathbf{m}$ et la matrice de covariance $\mathbf{S}$ estimés de ces données sont les suivants :

$$\mathbf{m} = \begin{bmatrix} -0{,}1000 \\ 0{,}1833 \end{bmatrix}, \quad \mathbf{S} = \begin{bmatrix} 2{,}864 & -1{,}744 \\ -1{,}744 & 1{,}534 \end{bmatrix}.$$

Les vecteurs propres et valeurs propres associés de cette matrice de covariance sont :

$$\lambda_1 = 4{,}0654, \quad \mathbf{c}_1 = \begin{bmatrix} 0{,}8235 \\ -0{,}5673 \end{bmatrix}, \quad \lambda_2 = 0{,}3323, \quad \mathbf{c}_2 = \begin{bmatrix} 0{,}5673 \\ 0{,}8235 \end{bmatrix}.$$

Finalement, les données sont tracées dans la figure suivante.



(8) (a) Donnez l'équation permettant d'effectuer une transformation blanchissante de ces données, sous la forme d'une équation linéaire ayant la formulation suivante :

$$\mathbf{z} = \mathbf{A}\,\mathbf{x} + \mathbf{b},$$

en précisant les valeurs numériques de $\mathbf{A}$ et $\mathbf{b}$.

Solution: L'équation d'une transformation blanchissante est la suivante :

$$\mathbf{z} = \mathbf{S}^{-0,5}(\mathbf{x} - \mathbf{m}),$$

ce qui revient à dire que $\mathbf{A} = \mathbf{S}^{-0,5}$ et $\mathbf{b} = -\mathbf{S}^{-0,5}\mathbf{m}$ selon la forme de l'équation linéaire donnée dans l'énoncé. On a donc :

$$\mathbf{C} = [\mathbf{c}_1\ \mathbf{c}_2] = \begin{bmatrix} 0{,}8235 & 0{,}5673 \\ -0{,}5673 & 0{,}8235 \end{bmatrix},$$
$$\mathbf{D}^{-0,5} = \begin{bmatrix} \lambda_1^{-0,5} & 0 \\ 0 & \lambda_2^{-0,5} \end{bmatrix} = \begin{bmatrix} 0{,}4950 & 0 \\ 0 & 1{,}7347 \end{bmatrix},$$
$$\mathbf{A} = \mathbf{S}^{-0,5} = \mathbf{C}\mathbf{D}^{-0,5}\mathbf{C}^\top = \begin{bmatrix} 0{,}8946 & 0{,}5787 \\ 0{,}5787 & 1.3361 \end{bmatrix},$$

et :

$$\mathbf{b} = -\mathbf{S}^{-0,5}\mathbf{m} = \begin{bmatrix} -0{,}0166 \\ -0{,}1871 \end{bmatrix}.$$

(16) (b) Supposons un réseau de neurones de type autoencodeur avec une couche d'encodage à K neurones, une couche de décodage et une fonction de transfert linéaire ($\mathrm{f}_{\text{lin}}(x) = x$). La sortie d'un neurone i de la couche d'encodage se modélise comme suit :

$$z_i^t = (\mathbf{w}_i^{\text{enc}})^\top \mathbf{x}^t + w_{i,0}^{\text{enc}}, \quad i = 1, \ldots, K.$$

La sortie de la couche de décodage se modélise comme suit, ce qui correspond à la donnée d'entrée reconstruite selon les D dimensions de l'espace d'origine :

$$\hat{x}_j^t = (\mathbf{w}_j^{\text{dec}})^\top \mathbf{z}^t + w_{j,0}^{\text{dec}}, \quad j = 1, \ldots, D.$$

Une analyse en composantes principales (ACP) peut être utilisée comme modèle de base pour estimer les paramètres de ce modèle. Expliquez en mots comment peut-on utiliser l'ACP pour déterminer les valeurs numériques du réseau autoencodeur.

Déterminez également les valeurs précises de l'autoencodeur ($\mathbf{w}_i^{\text{enc}}$, $w_{i,0}^{\text{enc}}$, $\mathbf{w}_j^{\text{dec}}$ et $w_{j,0}^{\text{dec}}$) devant être utilisées selon les informations données en préambule de la question, si l'encodage se fait pour une valeur de $K = 1$.

Solution: Avec l'ACP on apprend une transformation linéaire qui fait la compression d'un espace d'entrée à D dimensions vers un espace compressé à $K < D$ dimensions. Cette compression se fait en utilisant les vecteurs propres correspondants aux K valeurs propres les plus élevées, comme l'utilisation de celles-ci dans la transformation linéaire maximise la capture d'information. Comme une couche de neurones avec fonction de transfert linéaire correspond à une transformation linéaire, l'application de l'ACP de cette façon permet d'obtenir la partie encodeur du réseau.

La partie décodeur se fait en appliquant la transformation inverse de l'ACP. Comme les vecteurs propres représentés dans $\mathbf{C}$ sont orthonormaux, on sait que la transformation

inverse consiste à utiliser la transposée $\mathbf{C}^\top$. Dans le cas où une réduction de la dimensionnalité est effectuée, la transformation inverse sera d'utiliser $\mathbf{W}^\top$, où $\mathbf{W}$ correspond aux K premières colonnes de $\mathbf{C}$ (vecteurs propres associés aux valeurs propres les plus élevées).

Dans le cas précis actuel, ceci veut dire que l'ACP avec $K = 1$ se fait selon la transformation linéaire suivante, qui vise à centrer les données sur la moyenne et les projeter la composante principale.

$$z = \mathbf{c}_1^\top \mathbf{x} - \mathbf{c}_1^\top \mathbf{m}$$
$$\hat{\mathbf{x}} = \mathbf{c}_1 \cdot z + \mathbf{m}$$

Donc, selon la modélisation faite de l'autoencodeur, ceci veut dire que l'on utilise comme valeurs :

$$\begin{array}{ll} \mathbf{w}_1^{\text{enc}} = \mathbf{c}_1, & w_{1,0}^{\text{enc}} = -\mathbf{c}_1^\top \mathbf{m}, \\ w_{1,1}^{\text{dec}} = c_{1,1}, & w_{1,0}^{\text{dec}} = m_1, \\ w_{2,1}^{\text{dec}} = c_{1,2}, & w_{2,0}^{\text{dec}} = m_2. \end{array}$$

(10) (c) Exécutez l'algorithme K-means (avec $K = 2$ groupes) à partir d'une initialisation utilisant les données $\mathbf{x}^2$ et $\mathbf{x}^6$ comme centres initiaux, jusqu'à convergence de l'algorithme.

Solution: En partant des centres $\mathbf{m}_1(0) = \mathbf{x}^1$ et $\mathbf{m}_2(0) = \mathbf{x}^5$, nous recalculons les distances des données aux centres.

	$\|\mathbf{x}^t - \mathbf{m}_1(0)\|$	$\|\mathbf{x}^t - \mathbf{m}_2(0)\|$	$b^t(1)$
$\mathbf{x}^1$	1,6619	3,1765	1
$\mathbf{x}^2$	0,0000	3,9051	1
$\mathbf{x}^3$	1,5133	4,6648	1
$\mathbf{x}^4$	3,7121	0,9434	2
$\mathbf{x}^5$	2,6926	1,5811	2
$\mathbf{x}^6$	3,9051	0,0000	2

Les nouveaux centres correspondants sont :

$$\mathbf{m}_1(1) = \begin{bmatrix} -1{,}533 \\ 1{,}200 \end{bmatrix}, \quad \mathbf{m}_2(1) = \begin{bmatrix} 1{,}333 \\ -0{,}833 \end{bmatrix}.$$

Comme les centres ont changé, on recalcule les distances et l'assignation aux centres.

	$\|\mathbf{x}^t - \mathbf{m}_1(1)\|$	$\|\mathbf{x}^t - \mathbf{m}_2(1)\|$	$b^t(2)$
$\mathbf{x}^1$	0.7401	2.8790	1
$\mathbf{x}^2$	0.7491	3.4052	1
$\mathbf{x}^3$	0.9262	4.3748	1
$\mathbf{x}^4$	3.8680	0.4028	2
$\mathbf{x}^5$	2.9230	0.8360	2
$\mathbf{x}^6$	3.8787	0.7951	2

L'assignation aux centres $b^t(2)$ n'a pas changé donc l'algorithme a convergé. Les centres $\mathbf{m}_1(1)$ et $\mathbf{m}_2(1)$ ainsi que l'assignation aux centres $b^t(1)$ correspond donc au résultat final.

Question 3 (52 points sur 100)

Répondez aussi brièvement et clairement que possible aux questions suivantes.

(4) (a) Dans le cours, le fonctionnement remarquable de l'apprentissage profond est expliqué par sa capacité à faire de la composition de modèles. Expliquez en quoi consiste la « compositionnalité » dans ce contexte. Expliquez également pourquoi une méthode basée sur les distances telles que le classement par le plus proche voisin ne permet pas d'exploiter la compositionnalité.

Solution: La compositionnalité vise à définir des éléments de base de modélisation, qui sont composés de façon hiérarchique pour faire des descriptions d'entités complexes.

Une méthode telle que le classement par le plus proche voisin utilise une description complète des données sur laquelle une mesure de distance est appliquée. Ce genre d'approche ne permet pas de décrire des concepts nouveaux, non vus de façon explicite dans des instances du jeu de données, car les concepts de base ne sont pas décomposés dans des représentations hiérarchiques.

(4) (b) 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).

(4) (c) Un avantage certain des outils logiciels modernes pour faire des réseaux de neurones profonds est l'utilisation de gradient automatique. Expliquez en quoi ceci consiste précisément, en mentionnant l'impact pour la programmation de réseaux profonds et l'efficacité des traitements informatiques.

Solution: Le gradient automatique permet de calculer automatiquement les gradients analytiques de réseaux de neurones profonds et autres structures en graphe similaires. Ceci est donc particulièrement utile pour les développeurs, car des modifications à des structures existantes ou l'utilisation de fonctions de pertes personnalisées, par exemple, ne nécessitent pas de reprogrammer l'algorithme d'apprentissage par descente du gradient, permettant ainsi des développements rapides et la personnalisation des réseaux pour des tâches précises. De plus, le calcul de gradients automatique permet une optimisation des traitements bien adaptée au matériel, par exemple pour exploiter efficacement la capacité de calcul offerte par les cartes graphiques.

(4) (d) Classez en ordre chronologique d'apparition les architectures suivantes de réseaux de neurones : AlexNet, ResNet, perceptron multicouche, LeNet-5.

Solution: Perceptron multicouche (∼1985), LeNet-5 (∼1998), AlexNet (2012) et ResNet (2015).

(4) (e) Expliquez une façon simple d'obtenir l'incertitude sur une décision faite avec une forêt aléatoire.

Solution: La forêt aléatoire résulte d'un ensemble varié d'arbres de décision. La décision est prise selon la valeur moyenne obtenue des arbres formant la forêt. On peut évaluer l'incertitude sur cette décision en calculant la variance sur ces décisions, une grande variance indiquant une incertitude élevée sur la décision.

(4) (f) Expliquez pourquoi dit-on que le *bagging* induit de façon passive de la diversité entre les membres formant les ensembles, alors que AdaBoost le fait de façon active.

Solution: Le bagging génère une diversité dans les ensembles en utilisant des jeux d'entraînement différents pour chaque modèle entraîné, générés par un tirage aléatoire avec remise du jeu d'origine. Cette méthode est dite passive, comme on se base sur le fait que les modèles de base sont instables, produisant des modèles significativement différents seulement en variant la composition des jeux d'entraînement.

AdaBoost génère de la diversité en modulant les probabilités d'échantillonnage de chaque donnée du jeu d'entraînement, selon les performances observées jusqu'à présent. En ce sens, la procédure est active, comme la modulation tient compte des performances observées jusqu'à présent, avec une rétroaction directe sur la procédure de génération des jeux de données utilisés pour entraîner chaque membre de l'ensemble.

(4) (g) Expliquez pourquoi la standardisation des données, qui ramène la distribution des données pour chaque variable à une moyenne nulle et écart-type unitaire, n'est pas équivalente à une transformation blanchissante.

Solution: Avec une standardisation des données, on ne retire pas la covariance entre les variables, la transformation étant effectuée de façon indépendante pour chaque variable pour avoir une moyenne nulle et écart-type unitaire. Avec une transformation blanchissante, les données sont tournées pour retirer toute covariance, en plus d'être transformées pour obtenir une moyenne nulle et un écart-type unitaire.

(4) (h) Dans le cours, les méthodes de sélection séquentielle avant et arrière pour la sélection de caractéristiques sont présentées comme étant des heuristiques. Expliquez ce que cela signifie en définissant le terme *heuristique* et indiquez les implications pour la sélection de caractéristiques en général.

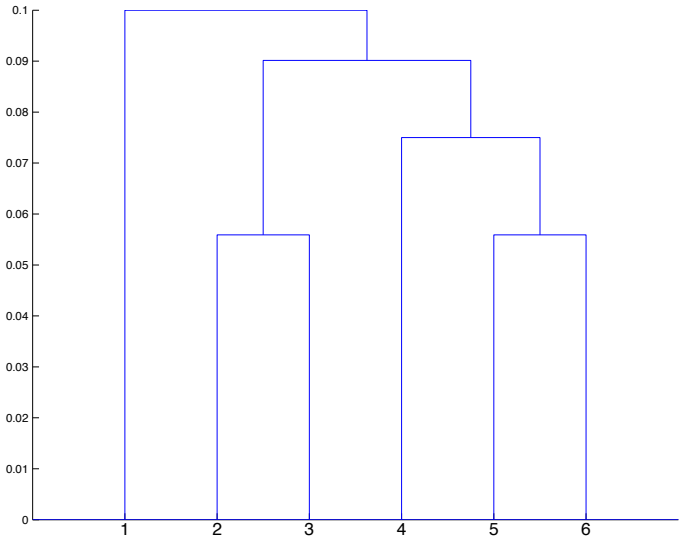
Solution: Une heuristique est une méthode relativement simple permettant d'obtenir une solution satisfaisante, mais pas nécessairement optimale à un problème dans des temps de traitement raisonnables.

Dans le contexte de sélection de caractéristiques, ceci veut dire que les méthodes de sélection séquentielle peuvent fournir un sous-ensemble de caractéristiques offrant de bonnes performances avec ce nombre de variables, sans nécessairement être le meilleur sous-ensemble possible pour ce sous-problème. De plus, le temps de traitement est relativement raisonnable, de l'ordre de $O(KD)$, si on suppose le temps d'évaluation de la performance d'un sous-ensemble comme étant constant.

(4) (i) Selon la présentation faite en classe de l'algorithme EM, expliquez avec précision ce à quoi consistent les appartenances h_i^t des données. Expliquez la distinction entre ces variables h_i^t et les variables cachées z_i^t de l'algorithme.

Solution: L'appartenance h_i^t est définie comme étant la probabilité que la donnée $\mathbf{x}^t$ provienne du groupe $\mathcal{G}_i$ obtenu selon la configuration actuelle Φ, $h_i^t \equiv P(\mathcal{G}_i|\mathbf{x}^t,\Phi)$. Elles sont en fait les observations probabilistes des valeurs z_i^t, qui représentent la véritable association entre les données et les groupes, et qui ne sont pas observables.

(4) (j) Soit le dendrogramme donné ici-bas.



Supposons que l'on veut former quatre groupes, donnez les indices des instances formant chacun de ces groupes.

Solution: $\{1\}$, $\{2,3\}$, $\{4\}$, $\{5,6\}$

(4) (k) Expliquez la différence entre le problème de surapprentissage et le problème de sur-recherche en apprentissage machine.

Solution: Le surapprentissage provient du fait que le modèle entraîné est probablement trop complexe pour le problème que l'on veut modéliser, de sorte qu'avec un entraînement poussé, par exemple sur plus d'itérations que nécessaire, le modèle peut très bien performer sur les données d'entraînement, sans nécessairement obtenir de bonnes performances sur de nouvelles données non vues en entraînement.

Le problème de sur-recherche porte plutôt sur le fait qu'un très grand nombre de configurations de modèles a été testé, et que le modèle recherché, qui semble bien fonctionner sur des données distinctes utilisées pour mesurer empiriquement ses performances en généralisation, ne généralise pas bien en pratique. Ce problème survient souvent lorsqu'un trop grand nombre de configurations est testé.

(4) (l) Expliquez dans quelles circonstances une recherche aléatoire d'hyperparamètres semble mieux fonctionner qu'une recherche en grille, avec un nombre égal de configurations testées.

Solution: Lorsque des hyperparamètres pour lesquels l'optimisation est effectuée ont peu ou pas d'influence sur la performance du système, une recherche aléatoire pourra bien exploiter le nombre de configurations testées selon les hyperparamètres ayant une influence, en faisant ainsi une meilleure recherche selon ces facteurs. À l'opposé, une recherche en grille va « gaspiller » des configurations pour évaluer différentes valeurs des hyperparamètres à négliger, en conservant les mêmes valeurs pour les hyperparamètres significatifs.

(4) (m) Expliquez pourquoi dit-on qu'avec un nombre suffisamment grand de données et un modèle d'une complexité suffisante, on peut obtenir des performances en généralisation se rapprochant du taux d'erreur bayésien optimal. Expliquez également pourquoi il n'est pas possible d'obtenir un meilleur taux de classement que le taux d'erreur bayésien optimal.

Solution: Le taux d'erreur bayésien optimal est le taux que l'on peut obtenir si l'on connaît la véritable distribution des données de notre problème. Avec cette connaissance et selon les variables d'entrées utilisées, il n'est pas possible de faire mieux, même si ce taux d'erreur n'est pas nul, comme la modélisation faite du problème est réputée être fidèle aux véritables phénomènes sous-jacents.

Avec un très grand nombre de données, on peut en arriver à avoir une bonne estimation de la distribution des données, si le modèle d'apprentissage permet de faire cette modélisation, comme l'échantillonnage devrait être alors assez dense. Cependant, étant donné la taille des espaces mathématiques en jeu pour des dimensionnalités moindrement élevées, le nombre d'échantillons nécessaires à un échantillonnage assez dense risque d'être extrêmement élevé.