Corrige IFT6266 - TP 1

20 septembre 2006

1. *Expliquez pourquoi selon vous* **généraliser** *correctement uniquement à partir d'un ensemble d'exemples est fondamentalement un problème qui est mathématiquement mal défini.*

   Le problème de généraliser consiste à trouver une fonction f qui minimise

   $$C(f) = E_X[\mathcal{L}(f, X)] = \int_x \mathcal{L}(f, x)p(x)dx \tag{1}$$

   où $\mathcal{L}(f, x)$ est le coût d'appliquer la fonction f à l'exemple x, et où X est la variable aléatoire dont sont tirés les exemples, selon une distribution p.

   Malheureusement, cette formulation mathématique n'est pas utilisable en pratique puisque la distribution $p(x)$ est inconnue, et ne peut qu'être **approximée** à partir des exemples disponibles pour l'apprentissage. C'est le fait que cette approximation (par exemple la moyenne empirique du coût, sur les exemples d'apprentissage) diffère de la véritable intégrale qui peut donner lieu au phénomène de l'overfitting: notre approximation pourra être minimisée sans que la véritable intégrale la soit aussi.

   La formulation mathématique (1) du problème de généralisation ne peut en tout cas pas être appliquée directement pour optimiser $C(f)$, ce qui permet de dire que c'est un problème mathématiquement mal défini.

2. *Supposons que le seul algorithme d'apprentissage que vous connaissiez soit la régression linéaire, qui estime $E[Y|X]$ à partir de paires (x_i, y_i). Si on vous demande de vous attaquer à un problème de classification probabiliste, i.e. l'estimation de $P(Y|X)$, avec les $y_i \in \{1, \ldots, N\}$, comment est-ce que vous pourriez vous y prendre en utilisant seulement votre algorithme de régression linéaire? Commencez par considérer le cas plus simple de $N = 2$ classes. Selon vous, quelles sont les défauts de cette approche?*

   Commençons par le cas binaire $N = 2$. Dans ce cas, l'espérance peut s'écrire

   $$E[Y|X = x] = P(Y = 1|X = x) + 2P(Y = 2|X = x) = p_1(x) + 2p_2(x).$$

   D'autre part on a aussi que $p_1(x) + p_2(x) = 1$, donc :

   $$E[Y|X = x] = p_1(x) + 2(1 - p_1(x)) = 2 - p_1(x)$$

   ce qui donne les probabilités de chaque classe en fonction de l'espérance conditionnelle de Y étant donné $X = x$:

   $$p_1(x) = 2 - E[Y|X = x] \tag{2}$$
   $$p_2(x) = E[Y|X = x] - 1 \tag{3}$$

   On voit donc que la connaissance de $E[Y|X = x]$ est suffisante pour calculer les probabilités de chaque classe.

   On aurait pu aussi utiliser notre algorithme pour estimer simplement $E[1_{Y=2}|X = x]$, donc en utilisant la fonction indicatrice (0 ou 1) $1_{Y=2}$ comme "cible" pour notre régresseur linéaire. Soit $f(x)$ le résultat de cette régression au point x, alors on pourrait estimer nos probabilités par $p_2(x) = f(x)$ et $p_1(x) = 1 - f(x)$.

   Malgré tout, cette méthode présente plusieurs inconvénients :

- Le plus important est que rien ne garantit que l'on obtiendra pas des probabilités négatives ou plus grandes que 1. Une manière basique de régler ce problème consisterait à tronquer $p_1(x)$ et $p_2(x)$ pour les forcer à rester dans $[0, 1]$.
- Même si la surface de décision pour la classification est linéaire (définie par $\hat{E}[Y|X = x] = 1.5$) , on voit que les exemples éloignés de la surface de décision on une grande influence sur la fonction de coût, même s'ils sont bien classifiés. Cela risque de nuire aux performances de classification des exemples plus proches de la surface de décision.
- Dans le cas de classes débalancées, cette surface de décision risque même d'être très mal placée (et mener à prédire la même classe partout, par exemple).

Au final on ferait mieux de faire de la classification logistique...

Dans le cas plus général où $N > 2$, on n'a plus la bijection entre $E[Y|X = x]$ et les $P(Y = i|X = x)$, ce qui rend les choses plus compliquées, mais on peut généraliser l'idée d'utiliser des indicateurs comme cibles.

Une méthode consisterait à associer à chaque y_i une cible vectorielle pour notre régression linéaire (multiple, maintenant), un vecteur "one-hot", i.e. un vecteur de dimension N contenant des zéros partout, sauf un 1 en position y_i. Notre algorithme de régression linéaire pourrait alors être appliqué sur ces nouvelles valeurs cibles, ce qui correspond à utiliser l'algorithme binaire vu ci-dessus en mode "one vs. rest". La probabilité de chaque classe (vs. les autres classes) estimée à partir du vecteur de sortie de l'algorithme souffre des mêmes défauts que ceux vus dans le cas binaire, et il faut de plus trouver une manière de les combiner afin d'avoir pour chaque classe une probabilité positive, dont la somme sur les classes est égale à 1. On peut encore tronquer et normaliser, mais ce n'est pas très rassurant et enlève les garanties que l'estimateur est correct en espérance (i.e. non-biaisé).

3. *Considérons un classifieur affine, i.e.* $f(x) = \text{sign}(b + w'x)$*, avec paramètres* b *(un scalaire) et* w *(un vecteur), et entrée* x*. Montrez que* $\frac{|b+w'x|}{||w||}$ *est la distance entre* x *et la surface de décision* $b + w'x = 0$*, et que le signe (*$f(x)$*) indique de quel côté de la surface* x *se trouve.*

   Soit S un plan: $b + w'x = 0$, alors le vecteur normal au plan est $w = (w_1, \ldots, w_n)'$
   Soit P un point dans le plan S, alors $b + w'P = 0 \rightarrow b + \sum_i w_i P_i = 0 \rightarrow \sum_i w_i P_i = -b$
   Le vecteur $\mathbf{PX}$ entre P et X est $\mathbf{PX} = (x_1 - P_1, \ldots, x_n - P_n)'$
   Notons $Proj_w\mathbf{PX}$ la projection du vecteur $\mathbf{PX}$ sur le vecteur w. La longueur de cette projection est $D = |Proj_w\mathbf{PX}| = \frac{|w \cdot \mathbf{PX}|}{||w||}$

$$\rightarrow D = \frac{|\sum_i w_i(x_i - P_i)|}{||w||} = \frac{|\sum_i w_i x_i - \sum_i w_i P_i|}{||w||} = \frac{|\sum_i w_i x_i + b|}{||w||} = \frac{|b + w'x|}{||w||}$$

   Pour se convaincre de la formule de la longueur de la projection d'un vecteur sur un autre, on peut considérer Z le point de S le plus proche de X, et l'angle θ formé par les vecteurs PX et ZX, dont le cosinus est $\frac{PX.w}{||PX||||w||}$, qui égale aussi par définition le ratio des longueurs du côté adjacent, $||XZ||$, à l'hypothénuse $||PX||$, dans le triangle rectangle PXZ. On a donc bien $||XZ|| = \frac{PX.w}{||w||}$.

   Considérons la question du signe. Une solution moins complète aurait obtenu tous les points.

   La fonction $b + w'x$ étant continue par rapport à x, son signe ne peut qu'être constant dans un demi-espace délimité par S (un côté de S). En effet, si deux points x_1 et x_2 situés du même côté de S avaient des signes différents, par le théorème des valeurs intermédiaires, quel que soit le chemin continu reliant x_1 à x_2, il existerait un point de ce chemin appartenant à S. Or, c'est évidemment faux par exemple si l'on considère la droite reliant x_1 à x_2, qui est entièrement contenue dans le même demi-espace (ce dernier étant convexe).

   D'autre part, il existe des valeurs de $\alpha \in \mathbb{R}$ telles que $f(\alpha w) = \text{sign}(b + \alpha\|w\|^2)$ atteigne 1 ou -1, donc les deux côtés de S ont nécessairement des signes différents, et $f(x)$ indique bien de quel côté de S le point x se trouve.

4. *Vous allez ici généraliser la régresssion linéaire ordinaire dans deux directions. Premièrement vous allez considérer la possibilité que l'erreur sur certains exemple soit plus importante que l'erreur sur d'autre. On va donc supposer qu'on nous donne un poids v_t sur l'erreur de l'exemple t. Par ailleurs on va supposer que certaines solutions (w, b) sont préférables à d'autres (technique de la régularisation) et on va pénaliser les moins préférables. Plus précisement vous allez considérer que l'on veut minimiser λ fois la norme au carré de w ($\lambda \sum_i w_i^2$) en plus des erreurs de prédiction. Le coût d'apprentissage est donc $C = \lambda \sum_i w_i^2 + \sum_t v_t(b + w'x_t - y_t)^2$. Dérivez la solution analytique à ce problème de minimisation (et donc donnez une formule pour w et b qui minimise C).*

Commençons par considérer le coût à w fixe, et tâchons de minimiser par rapport à b

$$C_w(b) = \frac{1}{2}\lambda\|w\|^2 + \frac{1}{2}\sum_t v_t(b + w'x_t - y_t)^2$$

où on a rajouté des facteurs $\frac{1}{2}$ pour simplifier les notations dans les dérivées futures. $C_w(b)$ est une fonction quadratique en b et le coefficient de b^2 est $\frac{1}{2}\sum_t v_t > 0$, donc il existe un unique minimum à cette fonction qui annule la dérivée $\frac{\partial C_w}{\partial b}$. Annuler cette dérivée s'écrit

$$\begin{aligned}\frac{\partial C_w}{\partial b} = 0 &= \sum_t v_t(b + w'x_t - y_t) \\ &= \left(\sum_t v_t\right) b + w'\left(\sum_t v_t x_t\right) - \sum_t v_t y_t\end{aligned}$$

ce qui donne pour b :

$$\begin{aligned}b &= \frac{\sum_t v_t y_t}{\sum_t v_t} - w'\frac{(\sum_t v_t x_t)}{\sum_t v_t} \\ &= \bar{y} - w'\bar{x} \\ &= b^*(w)\end{aligned} \tag{4}$$

en notant $\bar{x}$ et $\bar{y}$ les moyennes pondérées des x et des y sur l'ensemble d'entraînement.

Comme le raisonnement ci-dessus est valide quel que soit le vecteur w, le minimum de $C(w, b)$ est aussi le minimum de $C(w, b^*(w))$, qui s'écrit

$$\begin{aligned}C(w, b^*(w)) &= \frac{1}{2}\lambda\|w\|^2 + \frac{1}{2}\sum_t v_t(w'(x_t - \bar{x}) - (y_t - \bar{y}))^2 \\ &= \frac{1}{2}\lambda\|w\|^2 + \frac{1}{2}\sum_t v_t(w'\bar{x}_t - \bar{y}_t)^2\end{aligned}$$

en notant $\bar{x}_t = x_t - \bar{x}$ et $\bar{y}_t = y_t - \bar{y}$.

Calculons la dérivée seconde de $C(w, b^*(w))$ par rapport à w afin de vérifier que cette fonction est strictement convexe en w et donc qu'annuler sa dérivée première nous donnera son unique minimum. Pour rester cohérent avec le reste du devoir, je note la dérivée première sous la forme d'un vecteur rangée, comme dans l'exercice 5.

$$\begin{aligned}\frac{\partial C(w, b^*(w))}{\partial w} &= \lambda w' + \sum_t v_t(w'\bar{x}_t - \bar{y}_t)\bar{x}_t' \\ &= \lambda w' + w'\left(\sum_t v_t \bar{x}_t \bar{x}_t'\right) - \sum_t v_t \bar{y}_t \bar{x}_t' \\ &= w'\left(\lambda I + \sum_t v_t \bar{x}_t \bar{x}_t'\right) - \sum_t v_t \bar{y}_t \bar{x}_t'.\end{aligned} \tag{5}$$

La dérivée seconde est donc :

$$\frac{\partial^2 C(w, b^*(w))}{\partial w^2} = \lambda I + \sum_t v_t \bar{x}_t \bar{x}_t'$$

qui est une matrice définie positive pour $\lambda > 0$, puisque pour tout vecteur $z \neq 0$, on a

$$z' \left(\lambda I + \sum_t v_t \bar{x}_t \bar{x}_t' \right) z = \lambda \|z\|^2 + \sum_t v_t \left(z' \bar{x}_t \right)^2 > 0.$$

Cela démontre la convexité stricte de $C(w, b^*(w))$ par rapport à w. Annulons donc (5) pour trouver le minimum :

$$\begin{aligned} \frac{\partial C(w, b^*(w))}{\partial w} = 0 \quad &\Leftrightarrow \quad \left(\lambda I + \sum_t v_t \bar{x}_t \bar{x}_t' \right) w = \sum_t v_t \bar{y}_t \bar{x}_t \\ &\Leftrightarrow \quad w = \left(\lambda I + \sum_t v_t \bar{x}_t \bar{x}_t' \right)^{-1} \sum_t v_t \bar{y}_t \bar{x}_t. \end{aligned} \tag{6}$$

Les équations (6) et (4) donnent l'unique solution à notre problème de régression linéaire.

5. *Un algorithme d'optimisation est dit invariant linéaire si une transformation linéaire de la paramétrisation ne change pas la solution. Plus précisément, soit $C(\theta)$ le coût à minimiser en θ, et soit $\tilde{\theta} = A\theta$ avec une matrice A inversible. Soit θ^* la solution obtenue par un algorithme d'optimisation qui cherche à minimiser $C(\theta)$ en θ. Si l'algorithme est appliqué à minimiser $C(A^{-1}\tilde{\theta})$ en $\tilde{\theta}$ et qu'il obtient $\tilde{\theta}^* = A\theta^*$, on dit que l'algorithme est invariant linéaire. Dans le cas d'un algorithme itératif, cela doit être vrai après chaque itération de l'algorithme. Chaque itération de la descente de gradient de $C(\theta)$ sur le vecteur $\theta \in \mathbb{R}^n$ modifie θ ainsi:*

$$\theta \leftarrow \theta - \epsilon \frac{\partial C}{\partial \theta}$$

avec ϵ une petite constante positive. Chaque itération de la méthode de Newton modifie θ ainsi:

$$\theta \leftarrow \theta - (\frac{\partial^2 C}{\partial \theta^2})^{-1} \frac{\partial C}{\partial \theta}$$

Est-ce que la descente de gradient est invariante linéaire? Est-ce que la méthode de Newton est invariante linéaire? Donnez le développement. Si vous n'y arrivez pas avec la méthode de Newton, considérez au moins le cas où θ est un scalaire.

Afin de pouvoir appliquer la formule des dérivées en chaîne sans risque de nous tromper, nous allons adopter la convention que la dérivée d'une fonction $f : \mathbb{R}^{m\times 1} \mapsto \mathbb{R}^{n\times 1}$ par rapport à son argument $x \in \mathbb{R}^{m\times 1}$ (un vecteur colonne m-dimensionnel) est, par définition, une matrice $(n \times m)$ dont l'élément (i, j) est $\frac{\partial f_i}{\partial x_j}$. De plus, par définition, si $f : \mathbb{R}^{m\times 1} \mapsto \mathbb{R}$, la dérivée seconde de f par rapport à son argument $x \in \mathbb{R}^{m\times 1}$ sera représentée par une matrice $(m \times m)$ dont l'élément (i, j) est $\frac{\partial^2 f}{\partial x_i \partial x_j}$, ce qui, d'après la définition de la dérivée première ci-dessus, peut aussi s'écrire

$$\frac{\partial^2 f}{\partial x^2} = \frac{\partial}{\partial x} \left(\left(\frac{\partial f}{\partial x} \right)' \right).$$

Avec ces notations, la descente de gradient s'écrit, avec $\theta \in \mathbb{R}^{m\times 1}$:

$$\theta \leftarrow \theta - \epsilon \left(\frac{\partial C}{\partial \theta} \right)'.$$

La question est de savoir si, avec une initialisation telle que $\tilde{\theta} = A\theta$, cette égalité est encore préservée après une mise à jour des paramètres. En notant $\tilde{C}(\tilde{\theta}) = C(A^{-1}\tilde{\theta}) = C(\theta)$, c'est le cas si et seulement si

$$\tilde{\theta} - \epsilon \left(\frac{\partial \tilde{C}}{\partial \tilde{\theta}} \right)' \quad = \quad A \left(\theta - \epsilon \left(\frac{\partial C}{\partial \theta} \right)' \right)$$

$$\begin{aligned}
\Leftrightarrow A\theta - \epsilon\left(\frac{\partial C(A^{-1}\tilde{\theta})}{\partial(A^{-1}\tilde{\theta})}\frac{\partial(A^{-1}\tilde{\theta})}{\partial\tilde{\theta}}\right)' &= A\theta - \epsilon A\left(\frac{\partial C}{\partial\theta}\right)' \\
\Leftrightarrow \left(\frac{\partial C}{\partial\theta}A^{-1}\right)' &= A\left(\frac{\partial C}{\partial\theta}\right)' \\
\Leftrightarrow \left(A^{-1}\right)'\left(\frac{\partial C}{\partial\theta}\right)' &= A\left(\frac{\partial C}{\partial\theta}\right)'.
\end{aligned}$$

Cette dernière égalité est vraie dans le cas général (i.e. pour toute valeur de C et θ) si et seulement si $\left(A^{-1}\right)' = A$, i.e. A est une matrice orthonormale. C'est donc la condition nécessaire et suffisante pour que la descente de gradient soit invariante linéaire. C'est donc dire que la **descente de gradient n'est pas invariante linéaire**, sauf si la transformation linéaire est une matrice orthonormale.

La méthode de Newton, elle, s'écrit (toujours en utilisant les notations définies en début d'exercice) :

$$\theta \leftarrow \theta - \left(\frac{\partial^2 C}{\partial\theta^2}\right)^{-1}\left(\frac{\partial C}{\partial\theta}\right)'$$

et est invariante linéaire si et seulement si pour tout C et θ, est vérifiée l'égalité

$$\begin{aligned}
\tilde{\theta} - \left(\frac{\partial^2\tilde{C}}{\partial\tilde{\theta}^2}\right)^{-1}\left(\frac{\partial\tilde{C}}{\partial\tilde{\theta}}\right)' &= A\left(\theta - \left(\frac{\partial^2 C}{\partial\theta^2}\right)^{-1}\left(\frac{\partial C}{\partial\theta}\right)'\right) \\
\Leftrightarrow A\theta - \left(\frac{\partial}{\partial\tilde{\theta}}\left(\left(\frac{\partial\tilde{C}}{\partial\tilde{\theta}}\right)'\right)\right)^{-1}\left(\frac{\partial C}{\partial\theta}A^{-1}\right)' &= A\theta - A\left(\frac{\partial^2 C}{\partial\theta^2}\right)^{-1}\left(\frac{\partial C}{\partial\theta}\right)' \\
\Leftrightarrow \left(\frac{\partial}{\partial\tilde{\theta}}\left(\left(A^{-1}\right)'\left(\frac{\partial C}{\partial\theta}\right)'\right)\right)^{-1}\left(A^{-1}\right)'\left(\frac{\partial C}{\partial\theta}\right)' &= A\left(\frac{\partial^2 C}{\partial\theta^2}\right)^{-1}\left(\frac{\partial C}{\partial\theta}\right)' \\
\Leftrightarrow \left(\left(A^{-1}\right)'\frac{\partial}{\partial\theta}\left(\left(\frac{\partial C}{\partial\theta}\right)'\right)\frac{\partial\theta}{\partial\tilde{\theta}}\right)^{-1}\left(A^{-1}\right)'\left(\frac{\partial C}{\partial\theta}\right)' &= A\left(\frac{\partial^2 C}{\partial\theta^2}\right)^{-1}\left(\frac{\partial C}{\partial\theta}\right)' \\
\Leftrightarrow \left(\left(A^{-1}\right)'\frac{\partial^2 C}{\partial\theta^2}A^{-1}\right)^{-1}\left(A^{-1}\right)'\left(\frac{\partial C}{\partial\theta}\right)' &= A\left(\frac{\partial^2 C}{\partial\theta^2}\right)^{-1}\left(\frac{\partial C}{\partial\theta}\right)' \\
\Leftrightarrow A\left(\frac{\partial^2 C}{\partial\theta^2}\right)^{-1}A'\left(A^{-1}\right)'\left(\frac{\partial C}{\partial\theta}\right)' &= A\left(\frac{\partial^2 C}{\partial\theta^2}\right)^{-1}\left(\frac{\partial C}{\partial\theta}\right)'
\end{aligned}$$

et cette dernière égalité étant évidemment vérifiée, on peut en conclure que la méthode de Newton est invariante linéaire pour toute matrice A inversible.

6. *Considérons le problème de prédiction non-paramétrique dans des espaces discrets. Soit la variable d'entrée $X = (X_1, \ldots, X_d)$ avec $X_i \in \{1, \ldots, V\}$ et la variable de sortie $Y \in \{0, 1\}$. On estime $E[Y|X = x_{n+1}]$ par la moyenne des y_t observés dans le sous-ensemble $\mathcal{V}(x_{n+1})$ des n exemples d'apprentissage (x_t, y_t) pour lesquels $\sum_i 1_{x_{t,i} \neq x_{n+1,i}} < \epsilon$. Si on suppose les X uniformément distribués, on voudrait pouvoir dire quelque chose d'intéressant sur la probabilité que $\mathcal{V}(x_{n+1})$ soit vide. On s'intéresse à ce qui se passe quand d devient grand.*

   (a) *Considérez d'abord le cas plus simple $\epsilon = 1$ (égalité) et $V = 2$ (symboles binaires). (Si vous avez de la difficulté avec les maths, essayer d'y répondre de manière expérimentale comme en* **(c)***).*
   Avec $\epsilon = 1$ et $V = 2$, la probabilité qu'un exemple x_i $(1 \leq i \leq n)$ soit dans le voisinage de x_{n+1} est

$$P(x_i \in \mathcal{V}(x_{n+1})) = P(x_i = x_{n+1}) = \Pi_{j=1}^{d} P(x_{i,j} = x_{n+1,j}) \tag{7}$$

   puisque les variables sont tirées indépendamment les unes des autres. Or pour toute variable j, on a

$$P(x_{i,j} = x_{n+1,j}) = \frac{1}{2}$$

puisque les exemples x_i et x_{n+1} sont indépendants et chaque variable est distribuée uniformément dans $\{1, 2\}$. Donc (7) devient:

$$P(x_i \in \mathcal{V}(x_{n+1})) = \frac{1}{2^d}.$$

Finalement, tous les exemples étant tirés indépendamment, on obtient que la probabilité que $\mathcal{V}(x_{n+1})$ soit vide est donnée par

$$\begin{aligned} P(\mathcal{V}(x_{n+1}) \text{ vide}) &= P(\forall i \in \{1, \ldots, n\}\ x_i \notin \mathcal{V}(x_{n+1})) \\ &= \Pi_{i=1}^{n} P(x_i \notin \mathcal{V}(x_{n+1})) \\ &= \Pi_{i=1}^{n} \left(1 - \frac{1}{2^d}\right) \\ &= \left(1 - \frac{1}{2^d}\right)^n. \end{aligned}$$

Lorsque le nombre n d'exemples d'entraînement est fixe, cette probabilité converge vers 1 lorsque $d \to +\infty$ à une vitesse exponentielle. Lorsque d est grand, on peut aussi approximer cette relation par

$$P(\mathcal{V}(x_{n+1}) \text{ vide}) \simeq 1 - \frac{n}{2^d}.$$

En haute dimension, on a donc très peu de chance d'avoir dans notre ensemble d'entraînement des points similaires aux exemples de test (où la notion de similarité est définie par $\mathcal{V}(x_{n+1})$).

(b) **BONUS**. *Trouvez une formule pour le cas plus général $\epsilon > 1$ (match inexact).*
Dans le cas plus général, commençons par reformuler (7):

$$\begin{aligned} P(x_i \in \mathcal{V}(x_{n+1})) &= P\left(\sum_{j=1}^{d} 1_{x_{i,j} \neq x_{n+1,j}} < \epsilon\right) \\ &= \sum_{k=0}^{\epsilon-1} P\left(\sum_{j=1}^{d} 1_{x_{i,j} \neq x_{n+1,j}} = k\right) \\ &= \sum_{k=0}^{\epsilon-1} \binom{d}{k} \left(1 - \frac{1}{V}\right)^k \left(\frac{1}{V}\right)^{d-k} \end{aligned}$$

où la dernière ligne est obtenue en remarquant que le nombre de variables qui diffèrent entre x_i et x_{n+1} suit une loi binômiale de paramètre $1 - \frac{1}{V}$ (la probabilité que deux éléments tirés uniformément parmi V valeurs possibles soient différents).
On peut maintenant en déduire la probabilité que $\mathcal{V}(x_{n+1})$ soit vide comme au (a) :

$$\begin{aligned} P(\mathcal{V}(x_{n+1}) \text{ vide}) &= \Pi_{i=1}^{n} P(x_i \notin \mathcal{V}(x_{n+1})) \\ &= \left(1 - \sum_{k=0}^{\epsilon-1} \binom{d}{k} \left(1 - \frac{1}{V}\right)^k \left(\frac{1}{V}\right)^{d-k}\right)^n \end{aligned}$$

(c) **BONUS**. *Étudiez empiriquement le problème en faisant un histogramme de cette probabilité en fonction de ϵ, pour plusieurs valeurs de d de plus en plus grandes. Vous pouvez soit le faire avec la formule trouvée en* **(b)** *ou bien en faisant des tirages aléatoires des X_i et du point test X.*

La figure 1 montre comment évolue $P(\mathcal{V}(x_{n+1})$ vide) lorsque ϵ augmente, pour différentes valeurs de d (et $V = 5$ et $n = 10000$ fixés). On observe que lorsque d augmente, il faut un ϵ de plus en plus grand (proportionnellement à d) pour avoir de bonnes chances d'avoir des exemples d'entraînement dans le voisinage d'un point de test.

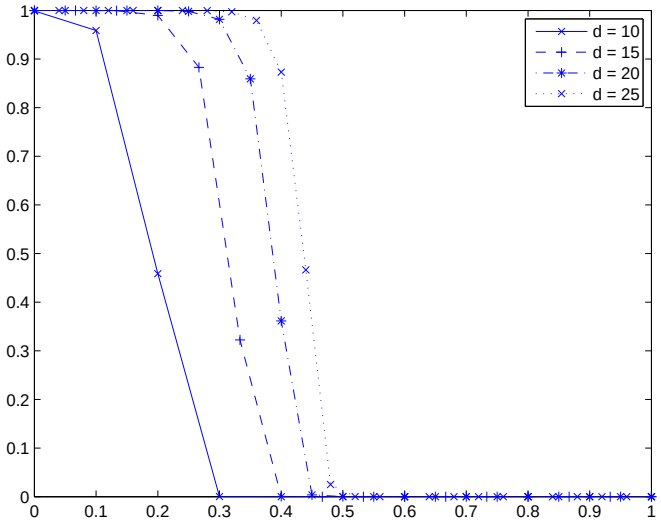

Figure 1: Probabilité que $\mathcal{V}(x_{n+1})$ soit vide en fonction de $\frac{\epsilon-1}{d}$ (la fraction des variables autorisées à être différentes pour être considéré dans le voisinage d'un point), pour différentes valeurs de d. Ici, sont fixés $V = 5$ et $n = 10000$.