MASTER : Traitement de l'Information et Exploitation des Données
Aymeric Chazottes, Fouad Badran, Cécile Mallet, Awa Niang, Charles Sorror, Sylvie Thiria

TPB01 : Régression par Perceptron Multicouches

I - Les Objectifs

Le but de ce TP est de mettre en œuvre une méthodologie constructive pour déterminer une fonction de régression à l'aide de perceptrons multicouches. Ce TP comporte 2 Parties :

- Un PMC réalisant une régression à partir de données brutes simulées pour approximer l'espérance.
- Une approximation de la variance de données brutes également effectuée par une régression à l'aide d'un PMC.

==========

Le rapport de TP devra être synthétique. Il doit montrer la démarche suivie, et ne faire apparaître que les résultats nécessaires. Il s'agit de quantifier les résultats tout en rédigeant un rapport qui les analyse et les commente. Les paramètres utilisés devront être indiqués, Les graphiques des expériences doivent être insérés dans le rapport. Les résultats présentés devront être analysés et commentés.

II - Les Données

Le T.P. se fera sur données simulées. La manière dont on génèrera les données doit permettre d'illustrer la méthode et les résultats auxquels on peut parvenir. **Dans la suite, on verra, entre autres, l'impact du nombre de données sur l'apprentissage et la généralisation.** Cela sera fait en jouant sur le nombre total de données mais aussi en jouant sur la répartition entres les ensembles d'apprentissage et de validation.
Il ne faut pas perdre de vu que l'**on travaillera avec un jeu de données jouet** dont la fonction sous-jacente est très régulière dans un espace de faible dimension (une dimension en entrée et une dimension en sortie).
L'objectif est de comprendre les mécanismes d'apprentissage. Il faudra obtenir des bonnes performances mais surtout comprendre comment on les a obtenues. (On connait déjà la fonction à apprendre, elle a servi à simuler les données. Si on voulait uniquement de bonnes performances on l'utiliserait directement.)
On pourra afficher la fonction à apprendre au milieu du nuage des données apprises. Cela n'est pas possible en général. Ce choix vous permet de mieux comprendre les performances du réseau vous pourrez ainsi vérifier la qualité des indicateurs de performance. Pour un jeu de données réaliste cela ne sera pas possible. **Il faudra donc principalement se baser sur les performances (et tout autre indicateur) pour justifier vos propos.**
Dans ce TP, on verra comment mener un apprentissage et on illustrera un certain nombre de cas de figure. (Il faudra garder à l'esprit que le problème posé est très simple.)

III - Eléments pour la réalisation du TP

En plus de l'énoncé et des données mises à disposition un certain nombre de fonctions sont fournies. Il faudra les comprendre. Dans la seconde partie du TP et à tout moment si nécessaire, il faudra réaliser quelques programmes/scripts nécessaires à la réalisation et à l'analyse. **On fera attention à présenter des figures lisibles et munies de légendes claires.**

IV - 1ère Partie du TP : Régression : Approximation de l'espérance

(Ci-dessous on indique ce qui doit être traité lors de la séance. Une partie des questions/indications correspond à ce qu'il faut réaliser l'autre partie à ce qui devra être traité spécifiquement dans le compte-rendu.)

On effectuera cette 1ère partie de TP uniquement en utilisant le jeu de données « **schioler** » définie de la façon suivante :

- Sur les abscisses les points suivent une distribution uniforme sur $]-2,\ 2[$.
- $y = f(x) + delta$ où :

$$f(x)\begin{cases} = sin(\pi x) & sur\ ]-1, 1[\\ = 0 & sur\ ]-2,-1] \ \cup\ [1, 2[\end{cases}$$

et $delta$ est un bruit qui suit une distribution normale $\mathcal{N}(0, \sigma^2)$ avec l'écart type $\sigma = 0{,}2$.

Dans la suite, on considérer un jeu de 1000 points qui servira pour apprendre les poids du PMC. Il sert à constituer deux ensembles : le premier, **ensemble d'Apprentissage**, obtenu en choisissant un point sur deux, et le second, **ensemble de Validation**, obtenu en prenant les 500 points restant. **On prévoit par ailleurs un autre ensemble de 500 points pour le Test.**

1°) Sur deux figures présenter d'une part les données d'apprentissage et de validation (avec des couleurs différentes) et d'autre part les données de Test.

Rappeler ce qu'est le sur-apprentissage. Indiquer pourquoi on a généré trois ensembles. (Il faudra indiquer l'usage de chacun de ces ensembles.)
Décrire la répartition des données pour ces trois ensembles. L'information de ces différents ensembles diffère-elle ? Que dire de l'échantillonnage des espaces d'entrées et de sorties apprises par le PMC.

Dans la suite on va chercher à apprendre la fonction sous-jacente aux données. On pourra déterminer la qualité de la régression via différents indicateurs. On discuter ces indicateurs. On pourra, dans un second temps, poursuivre cette discussion en se basant sur le fait que l'on connaît la fonction à estimer. (Cette seconde partie ne pourrait pas avoir lieu pour un jeu de donné réel.) Cette seconde partie permettra de mieux comprendre les performances de l'apprentissage.

Pour évaluer la qualité de l'apprentissage, on va devoir déterminer un indicateur permettant de l'évaluer. Cet indicateur devra être déterminé sur chacun des trois ensembles (apprentissage, validation et test).

Ici, on considérera la RMS ($\sqrt{\frac{1}{N}\sum_{i=1:N}\left(Y_{i\,calculé} - Y_{i\,désiré}\right)^2}$).

2°) **Pour le modèle appris à quoi correspondrait la valeur de la RMS obtenue pour le jeu de données simulées ?** (Faire le lien avec les paramètres du bruit simulé.)
De façon plus général, indiquer les propriétés que doivent respecter les erreurs d'apprentissage et de validation pour une architecture « optimale » apprise sur un jeu de donné quelconque. Il faudra aussi préciser ce qui est attendu de l'erreur de généralisation (erreur calculée sur l'ensemble de test).

On va apprendre plusieurs réseaux de neurones en faisant croître le nombre **m** de neurones sur la couche cachée. On pourra partir de **m=1** et se limiter par exemple à un maximum de **9** neurones en progressant par pas de **2**.

3°) Pour les différentes valeurs de **m** (nombre de neurones dans la couche cachée), **on présentera les valeurs des erreurs d'apprentissage et de validation** dans un tableau. On les présentera aussi sur une même figure (dans un seul repère) en fonction du nombre **m** de cellules cachées.
(On gardera à l'esprit l'ordre de grandeur du bruit des données simulées.)

Discuter ces valeurs et indiquer quelle(s) architecture(s) correspond(ent) à un estimateur « optimal » ? Expliquer.
Au-delà de l'architecture optimale obtenue, on devra détailler le cheminement permettant de la déterminer. On devra indiquer la qualité de l'estimateur relativement aux ensembles d'apprentissage et de validation mais aussi en quoi cette architecture est optimale relativement aux autres.
(Dans un cas plus réaliste, pour choisir une architecture « optimale », on apprendrait chaque architecture pour plusieurs initialisations. Cela n'est pas fait ici mais il faudra le garder à l'esprit.)

4°) **Une fois l'architecture optimale retenue,** rappeler l'intérêt de l'erreur de généralisation. Qu'attend-on, en termes d'erreur de généralisation, pour cette architecture optimale ?
Pour les différentes valeurs de m, **présenter les valeurs des erreurs sur l'ensemble de test** dans un tableau.
Vérifier si l'erreur en généralisation est minimale pour la ou les architectures optimales. Discuter cette valeur.

Avant de passer à la suite reprendre la définition du sur-apprentissage déjà demandée et identifié les architectures de la partie précédente qui y seraient sensibles.
(Discuter les erreurs des trois ensembles.)

Que dire de la qualité d'un estimateur qui aurait été appris uniquement sur l'ensemble d'apprentissage ? (On considérera l'architecture avec les meilleures performances pour cet ensemble.)

5°) On considère maintenant la qualité de l'apprentissage visuellement. Pour cela, on va s'intéresser aux différences entre les fonctions apprises et la fonction vraie. Présenter et discuter les différents cas de figure (en indiquant les architectures correspondantes).

Dans la suite, on s'intéresse aux effets du nombre de données et de leur répartition sur la qualité de l'apprentissage et donc de la généralisation.

6°) Générer de nouveaux ensembles d'apprentissage et validation en changeant la proportion des données de ces deux ensembles. On prendra, par exemple, un point sur n avec n = 3, 4, 5, 10, 30, 100. (En gardant un ensemble de Test similaire au précédent.)
Déterminer à chaque fois l'architecture optimale. On travaillera avec les mêmes éléments (figures et tableaux) que ceux des questions 3° et 5°. On indiquera à chaque fois les éléments de décision.

Considérer d'éventuels liens entre l'architecture optimale retenue et le nombre de points de l'apprentissage. Que remarquez-vous ? Expliquer.

Dans la suite, on va s'intéresser aux notions d'interpolation et d'extrapolation.

7°) On va utiliser des ensembles d'apprentissage et de validation dont l'échantillonnage est irrégulier c'est-à-dire dont certains intervalles de la fonction ne sont pas représentés :

- 1er cas : [-0.75 -0.25] U [1 1.25]
- 2ème cas : [-0.75 -0.25] U [1 2.00]

On continuera d'utiliser un ensemble Test similaire au précédent.
Ici, on procédera comme précédemment pour les différents affichages et indicateurs.

Indiquer les effets de l'échantillonnage sur l'apprentissage.
On discutera ce qui est obtenu relativement à ce qui avait été fait précédemment.
On présentera les éléments (figures et tableaux) permettant d'illustrer le propos.
Pour l'ensemble de Test calculer l'erreur intervalle par intervalle. Que remarque-t-on ?

V - 2ème Partie du TP : Régression : Approximation de la Variance

Dans cette partie, on va déterminer des intervalles de confiance en estimant la variance associée à chaque observation. On utilise ici le jeu de données « **silverman** ».

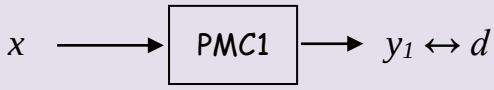
- Sur les abscisses les points suivent une distribution uniforme sur $]-0.25,\ 1[$.
- $y = f(x) + delta(x)$ où :

$$f(x) \begin{cases} = sin\,(2\pi\,(1-x)^2) & \text{sur }]\,0,\ 1[\\ = 0 & \text{sur }]-0.25,\ 0] \end{cases}$$

$delta(x)$ est un bruit qui suit un distribution normale $\mathcal{N}(0, sigma(x)^2)$ avec la variance $sigma(x)^2$

$$\begin{cases} = 0.0025 & si\ x <= 0.05 \\ = x^2 & si\ x > 0.05 \end{cases}$$

Principe : La méthode proposée ici pour l'estimation de la variance est très simple.
♣) Un premier PMC (PMC1) est optimisé pour modéliser la fonction $y=f(x)$ à partir d'une base d'apprentissage $\{(x_i,d_i)\ i=1,N\}$. Il s'agit d'une simple régression.



Après apprentissage, la sortie y de ce premier réseau donne une estimation de l'espérance de d/x : $y \approx E[d/x]$. L'erreur (noté $Eapp$) commise par ce 1er réseau est :

$$Eapp = (d/x - y) \approx (d/x - E[d/x])$$

♣) Pour chaque valeur x, un second réseau (PMC2) va permettre d'approximer la variance des données autour de la moyenne. Cette variance s'écrit :

$$Var(d/x) = E[(d/x - E[d/x])^2] \approx E[Eapp^2/x].$$

On voit donc que les sorties de ce second réseau devront approximer l'erreur quadratique du 1er réseau (qui représente la variance empirique sous l'hypothèse que le PMC1 est un bon estimateur de la moyenne conditionnelle) :

$$x \longrightarrow \boxed{\text{PMC2}} \longrightarrow y_2 \leftrightarrow Eapp^2$$

Remarque : Cette méthode d'estimation de la variance est sous-optimale car la moyenne estimée par PMC1 ne tiens pas compte de la variance. Il existe des architectures spécifiques et des méthodes d'apprentissage complète (comme celle vue en cours) qui permettent un apprentissage simultané de $E(d/x)$et $Var(d/x)$ qui sont alors deux sorties d'un même réseau pour lequel la fonction de coût est définie de manière ad hoc (maximum de vraisemblance).

Travail à faire :

Dans un 1er temps, on utilisera un ensemble d'Apprentissage de taille Napp=1000 données « **silverman** » simulées. Il conviendra de paramétrer un nombre d'itérations d'apprentissages suffisant.

1. - Déterminer PMC1 (5 neurones cachés devraient suffire). Visualiser ensuite dans le même repère :
   - l'ensemble des données
   - la fonction de régression y_1 estimé par PMC1 que l'on pourra comparer à
   - la courbe théorique (c'est-à-dire la courbe de la vraie fonction sans ajout de bruit).

2. - Identifier la ligne de code qui simule la fonction de la variance du bruit.

3. - Déterminer PMC2 en choisissant un nombre de neurones cachés approprié. **Les variances étant positives, c'est une contrainte qui devra être respectée par le PMC2**. Selon ce qu'il est possible de réaliser, utiliser une fonction de sortie exponentielle ou **faire apprendre le logarithme des sorties par le PMC.** Dans les deux cas, cela permettra d'assurer des valeurs de sortie positives.
Comme pour la question 1, on vous demande de présenter dans le même repère :
   - les données d'apprentissage de la variance empirique que vous avez utilisées,
   - la fonction de régression y_2 estimé par PMC2 (en sortie) que l'on pourra comparer à la courbe théorique correspondant à la variance théorique du bruit identifiée à la question 2.

4. - On représentera sur un même graphique les mêmes éléments qu'à la question 1.
On y ajoutera des courbes d'encadrement à plus ou moins deux écarts types :
   - avec la variance estimée par PMC2 (correspondant à $y_1 - 2\sqrt{y_2}$ et $y_1 + 2\sqrt{y_2}$)
   - avec la variance théorique pour permettre la comparaison.

   Vérifier si plus de 96,4% des données au moins se situent dans cet intervalle.

5. – Dans un second temps, on s'interrogera sur la validité de la procédure dans le cas où l'on disposerait de moins de données. On recommence donc ces expériences avec Napp=50 puis Napp=200. Que peut-on en dire ?