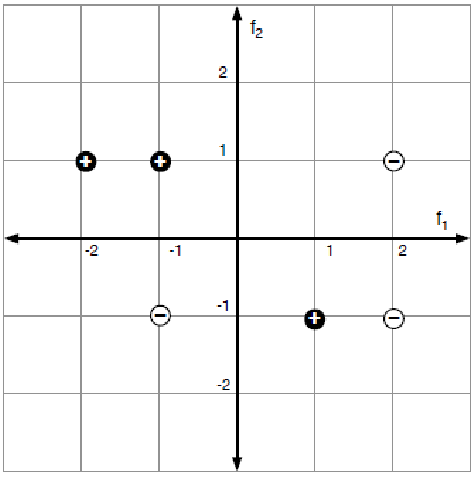
Université Cadi Ayyad
Faculté des sciences Semlalia
Département Informatique

05/05/20

Devoir à rendre

Exercise 1:

Considérons le graphique suivant (ensemble de points selon f1 et f2)



On donne 6 points avec leurs classes correspondantes (+ et -) :
(-1, -1), (2, 1) et (2, -1) de classe +
(-2, 1), (-1, 1) et (1, -1) de classe -
Soit **E**, l'ensemble de ces 6 points

1) Tracer la droite (D1) : **f1 = 1.5**. Est-ce que cette droite sépare nettement les deux classes ?
   Pour quel intervalle (I_1) de f1, le sous-ensemble ($E_1 \subset E$) d'exemples est homogène ? Quelle est sa classe ?
   Pour quel intervalle (I'_1) de f1 le sous-ensemble d'exemples ($E_2 \subset E$) est non-homogène ?
2) Tracer ensuite la droite(D2) : **f2 = 0.** Est-ce que cette droite sépare nettement les deux classes de E_2?
   Pour quel intervalle (I_2) de f2, le sous-ensemble ($F_1 \subset E_2$) d'exemples est homogène ? Quelle est sa classe ?
   Pour quel intervalle (I'_2) de f2, le sous-ensemble ($F_2 \subset E_2$) d'exemples est non-homogène ?

3) Tracer enfin la droite(D3) : **f1 = 0.**
   Montrer que cette droite sépare nettement les deux classes de F_2 .
   Noter les intervalles de f1 : (I_3) pour la classe + et (I_4) pour la classe –

4) Ecrire alors les règles de décision à l'aide des droites D1, D2 et D3 (selon les intervalles de I_1, I'_1, I_2, I'_2 ,I_3 et I_4) pour la classification.
5) Tracer l'arbre de décision correspondante.
6) Construire la table (attributs/classe) des données utilisées selon 3 attributs A, B et C (de valeurs possibles vrai ou faux) tels que :
   A = vrai si f1 $\in I_1$, faux sinon
   B= vrai si f2 $\in I_2$, faux sinon
   C= vrai si f1 $\in I_3$, faux sinon

7) Calculer L'entropie de la table selon ID3. (Le tableau précédent peut être utile).
8) Expliquez comment vous choisissez la racine dans l'arborescence selon ID3
9) Quelle classe l'arbre de décision prévoit-il pour le nouveau point (1,1) ?

Exercice 2: L'algorithme d'apprentissage Perceptron

Considérons un perceptron avec **X** le vecteur d'entrée (avec x_0=1 pour seuil), **W** est le vecteur poids (w_0, w_1, w_2,…) et **y** est la sortie désirée (+1 ou – 1).

La fonction d'activation est : f(**X.W**) =+1 si **X.W** $>$ 0, -1 sinon

X.W est le produit scalaire entre **X** et **W**. (peut s'écrire aussi : XW^T)

1) Montrer que quand on classifie correctement on a toujours : **y*(X.W)** $>$ **0** et quand on ne classifie pas correctement on a toujours : **y*(X.W) < 0**.
2) Posons **E(W)** comme coût (erreur) qui doit être minimisé. Pour cela on utilise la descente du gradient pour la mise à jour des poids (η taux d'apprentissage non nul) :

   $$\mathbf{W} = \mathbf{W} - \eta * \frac{\partial E}{\partial W} \qquad (1)$$

   a) Dans le cas où on classifie correctement, les valeurs des poids ne se modifient pas. Quelle condition doit remplir **E(W)** pour que l'équation (1) reste valable dans cas ?
   b) Dans le cas où on ne classifie pas correctement, on pose **E(W)= -y*(X.W)**, Montrer alors que :

      $$\mathbf{W} = \mathbf{W} + \eta * \mathbf{y} * \mathbf{X} \qquad (2)$$

   c) Dans la procédure d'entrainement, il est évident de ne tenir compte que des exemples mal classifiés (pas correctement classifiés). Supposons qu'on effectue plusieurs itérations où un exemple peut être réutilisé plusieurs fois.
      Soit α_i le nombre de fois où un exemple **i** est mal classifié.
      Justifier l'équation (3) de mise à jour des poids en fin d'apprentissage (pour **n exemples**) :

      $$W = W^{(0)} + \eta \sum_{i=1}^{n} \alpha_i \, y_i * X_i \qquad (3)$$

   W est le vecteur poids actuel, $\mathbf{W^{(0)}}$ est le vecteur poids initial.
3) Exemple d'application :

   Le tableau ci-dessous est une liste d'exemples de points dans R^2. Supposons que nous exécutons l'algorithme perceptron, avec une dimension fictive (plusieurs fois), sur ces points d'échantillonnage. Nous enregistrons le nombre total de fois que chaque point participe à une étape de descente de gradient stochastique car il est mal classé, tout au long de l'exécution de l'algorithme.

x_1	x_2	y	Nbre de fois mal-classifié
-3	2	+1	0
-1	1	+1	0
-1	-1	-1	2
2	2	-1	1
1	-1	-1	0

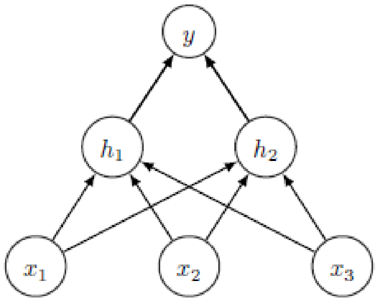
(a) Supposons que le taux d'apprentissage est **η = 1** et que le vecteur de poids initial est $\mathbf{W^{(0)} =)=(w^{(0)}_0, w^{(0)}_1, w^{(0)}_2) = (1, -3, 2)}$ où la première composante est le biais (poids seuil) .
Donner le vecteur poids résultant **W** selon l'équation (3) puis donner **l'équation de la droite séparatrice des 2 classes** (en fonction de x_1 et x_2).

(b) Dans certains cas, la suppression d'un seul point peut modifier la frontière de décision apprise par l'algorithme perceptron.

Pour quel (s) point (s), le cas échéant, dans notre jeu de données la frontière de décision apprise changerait-elle si nous le(s) supprimons ? Expliquez votre réponse.

Exercice 3: Propagation avant et arrière

Le graphique suivant montre la structure d'un réseau neuronal simple avec une seule couche cachée. La couche d'entrée se compose de trois dimensions x = [x_1, x_2, x_3]. La couche cachée comprend deux unités h = [h_1, h_2]. La couche de sortie comprend une unité y. **Nous ignorons les termes biaisés (seuils) pour plus de simplicité.**



Nous utilisons **g(z) = max (0,z)** (pratique en deepLearning, appelée **ReLu** : **Re**ctified **L**inear **U**nit) comme fonction d'activation pour la couche cachée et la couche de sortie.

De plus, notons

$l(y,t) = \frac{1}{2}(y-t)^2$, l'erreur à la sortie (appelée la **fonction de perte** ou « loss function »).

Ici, t est la valeur cible (désirée) pour l'unité de sortie **y**. l dépends des poids de connexion.

Notons les matrices de poids W et V reliant respectivement l'entrée à la couche cachée, et la couche cachée à la sortie. Ils sont initialisés comme suit :

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

$$V = \begin{bmatrix} 0 & 1 \end{bmatrix}$$

et **x = [1, 2, 1]** avec **t = 1**:

Supposons également que nous avons au moins un échantillon **(x, t)** donné par les valeurs ci-dessus.

1. Tracez le graphique de g(z) ainsi que celui de sa dérivée.
2. Donnez l'expression de **y** en fonction de **g, V, W** et de **x**.
3. Supposons que l'entrée actuelle est **x = [1, 2, 1]**. La valeur cible est **t = 1**. Calculez la sortie **y** *(donnez clairement toutes les étapes intermédiaires).*
   Vous pouvez réutiliser les résultats de la question précédente. L'utilisation de la notation matricielle est recommandée mais pas obligatoire.
4. La mise à jour des poids se fait à l'aide des gradients **(**$\frac{\partial l}{\partial V}$ et $\frac{\partial l}{\partial W}$**)** de la **fonction de perte** par rapport aux poids.
   a) Donner les expressions développées des termes :
      - Le gradient par rapport à V, soit $\frac{\partial l}{\partial V}$
      - Le gradient par rapport à W, soit $\frac{\partial l}{\partial W}$

   b) Donner la version numérique de $\frac{\partial l}{\partial V}$