Chapitre 4

Perceptron avec apprentissage supervisé en tant que classificateur

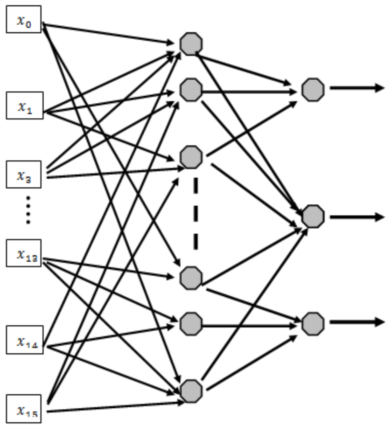


Figure 4.1 : Réseau à propagation avant ayant deux couches et plusieurs sorties

Nous savons qu'un réseau de neurones de McCulloch-Pitts correctement construit a des possibilités de calcul universel.

Mais nous savons aussi qu'un tel réseau n'a pas la faculté d'apprentissage.

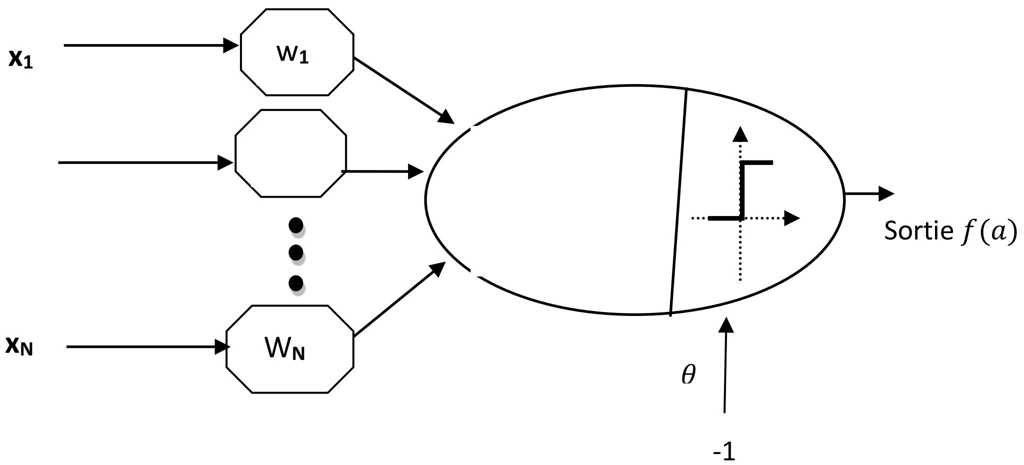
Le perceptron est un réseau de McCulloch-Pitts mais qui est doté de la possibilité d'apprentissage.

Cet apprentissage se fait avec **l'algorithme d'apprentissage du perceptron**.

Donc, les neurones dans le perceptron sont des neurones à seuil qui fonctionnent ensemble.

Le réseau du perceptron apprend à partir des exemples et les poids changent durant l'apprentissage.

Dans cette section nous allons examiner un perceptron à un seul neurone, tel que montré dans la figure suivante :



Ce neurone reçoit de multiples entrées (inputs) et les traite pour produire une réponse (output).

L'apprentissage requiert un ensemble de couples (entrées, cibles), qui est appelé **ensemble d'entraînement**.

$$E = \{(\boldsymbol{X}_1, t_1); (\boldsymbol{X}_2, t_2); \dots ; (\boldsymbol{X}_p, t_p)\}$$

A chaque entrée $\boldsymbol{X_i}$ correspond sa cible $\mathbf{t_i}$.

$$\mathbf{X_i} = \begin{pmatrix} \mathbf{x_{i1}} \\ \mathbf{x_{i2}} \\ \cdot \\ \vdots \\ \mathbf{x_{in}} \end{pmatrix}$$

Le traitement de ce perceptron se fait suivant le processus ci après.

1- Processus d'apprentissage pour un perceptron pour effectuer une tâche de classification.

Nous avons ici **11** étapes :

Etape 1 :

On affecte des valeurs aléatoires aux poids.

$$w_1 = une\ valeur\ aléatoire\ (petite)$$

$$w_2 = une\ autre\ valeur\ aléatoire$$

$$\dots$$

$$w_n = une\ autre\ valeur\ aléatoire$$

Etape 2 :

Le premier vecteur $(x_1, x_2, \dots, x_n)$ est fourni,

et chacune de ses composantes est multipliée par le poids correspondant

$$w_1.x_1$$

$$w_2.x_2$$

$$\dots$$

$$w_n.x_n$$

Etape 3 :

La somme des résultats est effectuée

$$u = w_1.x_1 + w_2.x_2 + \cdots + w_n.x_n$$

Etape 4 :

Cette somme est présentée à une fonction seuil

$$u - \theta = ?$$

Etape 5 :

Si cette différence est supérieure à zéro, alors la fonction seuil donne la valeur 1

$$y = 1$$

Etape 6 :

Sinon, la fonction seuil donne la valeur 0

$$y = 0$$

(ainsi, un perceptron à un neurone ne peut indiquer que deux classes (1 ou 0).

Etape 7 :

La réponse du perceptron ***y***, est comparée à la cible ***t***.

$$t = y \ ?$$

Etape 8 :

Si le résultat de la classification est faux $t \neq y$

l'algorithme d'apprentissage ajuste les poids pour que, par la suite, l'observation donnée soit affectée à la classe correcte.

Une même observation peut être présentée plusieurs fois au réseau pour que la réponse correcte soit obtenue.

Lorsque l'affectation de l'observation est correcte, les poids ne changent pas.

Etape 9 :

Le vecteur suivant est présenté à l'entrée.

Etape 10 :

Le processus se répète pour ce vecteur

Etape 11 :

Et ainsi de suite, pour toutes les données, jusqu'à ce que l'apprentissage soit achevé

L'algorithme d'apprentissage qui intervient dans l'étape 8 est illustré dans ce qui suit dans le cas où le seuil θ est fixé à 0

2- Algorithme d'apprentissage du perceptron

L'entrée nette **u** pour un vecteur d'entrée **X** est :

$$u = w_1.x_1 + w_2.x_2 + \cdots + w_n.x_n$$

La fonction seuil, positionnée à l'origine produit un output y, tel que

$$y = \begin{cases} 0 \quad si \quad u < 0 \\ 1 \quad si \quad u > 0 \end{cases}$$

Si la classification est correcte

C'est-à-dire si

$$y = t$$

cela veut dire que le perceptron a correctement classé et les poids ne seront pas ajustés.

Sinon (c'est à dire $y \neq t$)

les poids w_i sont ajustés de la manière suivante :

$$w_{i,new} = w_{i,old} + \beta . x_i . E$$

$w_{i,new}$ est la nouvelle valeur du poids w_i

$w_{i,old}$ est l'ancienne valeur du poids w_i

x_i est la composante numéro i du vecteur input

β est le taux d'apprentissage

E est l'erreur :

$$Erreur = E = t - y$$

Ce qui donne pour les trois conditions possibles de l'erreur, E :

$$w_{i,new} = \begin{cases} w_{i,old} & si \quad E = 0\ (i.e., t = y) \\ w_{i,old} + \beta x_i & si \quad E = 1\ (i.e., t = 1, y = 0) \quad Règle\ 1 \\ w_{i,old} - \beta x_i & si \quad E = -1\ (i.e., t = 0, y = 1) \quad Règle\ 2 \end{cases}$$

le taux d'apprentissage β est une constante comprise entre 0 et 1.

Son rôle est d'ajuster la vitesse d'apprentissage.

Les petites valeurs conduisent à un ajustement lent des poids, ce qui exige de plus longues périodes pour achever l'entraînement.

Les grandes valeurs accélèrent le taux d'incrémentation des poids. Des ajustements accélérés des poids ne sont pas nécessairement meilleurs car ils peuvent faire que la solution (i.e. les poids) oscille autour de l'optimum, cela conduit à l'instabilité.

EXEMPLE :

Nous allons, dans ce qui suit, utiliser cet algorithme d'apprentissage du perceptron dans une forme modifiée pour entraîner un perceptron classificateur simple ayant deux entrées et une sortie, tel que montré dans la figure 2.11b, dans laquelle x_1 et x_2 sont les variables d'entrée et y est la sortie du perceptron. La tâche est de classer les objets bidimensionnels représentés dans la figure 2.11a, qui appartiennent à deux classes de sortie (△ indique la classe A et ■ indique la classe B).

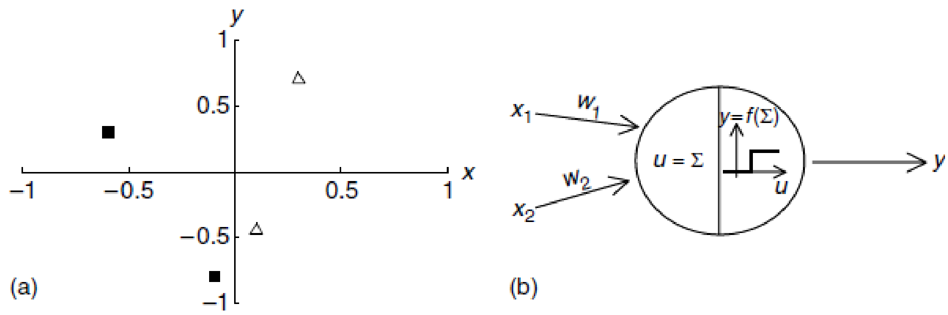


Figure 4.2: Apprentissage du perceptron

(a) représentation des données d'entrée appartenant aux deux classes soumise à apprentissage au perceptron.

(b) Configuration du perceptron pour cette tâche.

Pour cet exemple posons $\beta = 0.5$

Des valeurs initiales aléatoires seront données aux poids ; $w_1^0 = 0.8$ et $w_2^0 = -0.5$

(l'exposant 0 est pour désigner les valeurs initiales).

Donc le vecteur poids initial est $W^0 = \{0.8, -0.5\}$ et il est superposé aux données dans la figure 2.12. La longueur du vecteur initial L^0, qui est notée $||W^0||$ est

$$\mathbf{L}^0 = \sqrt{(w_1^0)^2 + (w_2^0)^2} = \sqrt{(0.8)^2 + (-0.5)^2} = 0.94.$$

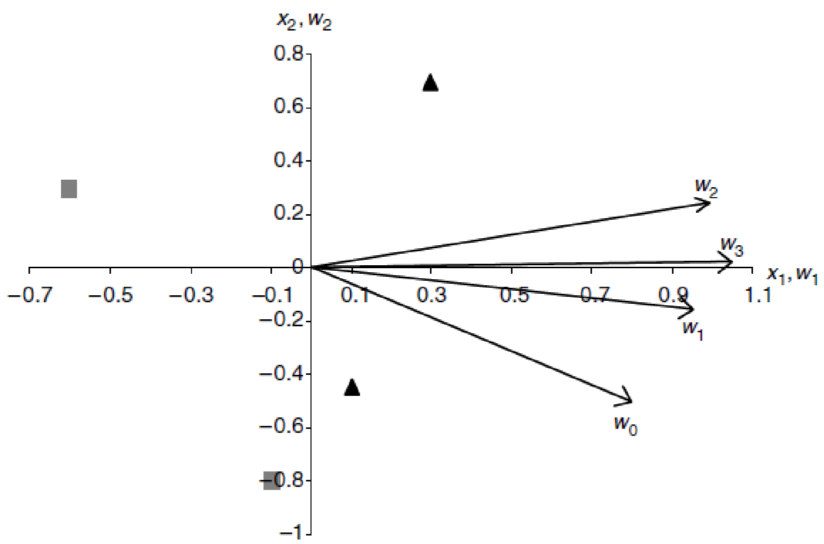


Figure 4.3: Vecteurs des poids

Présentons le vecteur d'entrée 1-A1 :

$$x_1 = 0.3, \quad x_2 = 0.7, \qquad t = 1; \quad w_1^0 = 0.8\,, \quad w_2^0 = -0.5$$

L'input pondéré est

$$u = (0.8)(0.3) + (-0.5)(0.7) = -0.11$$

$$u < 0 \Rightarrow y = 0$$

⇒ classification incorrecte

⇒ les poids doivent être ajustés en utilisant la règle 1 de l'équation 2.12.

Supposons que les variations des deux poids soient notées par Δw_1^1 et Δw_2^1 , l'exposant 1 dénote la première variation du poids. Alors

$$\Delta w_1^1 = \beta x_1 = (0.5)(0.3) = 0.15$$
$$\Delta w_2^1 = \beta x_2 = (0.5)(0.7) = 0.35.$$

Les nouveaux poids après la première incrémentation seront notés par w_1^1 et w_1^2. Donc

$$w_1^1 = w_1^0 + \Delta w_1^1 = 0.8 + 0.15 = 0.95$$
$$w_2^1 = \Delta w_2^0 + w_2^1 = -0.5 + 0.35 = -0.15.$$

Sous la forme vectorielle :

$$\mathbf{w}^1 = [w_1^1, w_2^1] = [0.8, -0.5] + [0.15, 0.35] = [0.95, -0.15].$$

Ainsi le nouveau vecteur des poids est $\{0.95, -0{,}15\}$ et sa longueur est

$$\mathbf{L}^1 = \sqrt{(w_1^1)^2 + (w_2^1)^2} = \sqrt{0.95^2 + (-0.15)^2} = 0.96.$$

Dans la figure 2.12, les poids initiaux et les poids modifiés sont représentés avec les données d'entrée. La mauvaise classification provoque le changement des poids.

Nous pouvons voir que si le perceptron sous-estime (i.e., t = 1, y = 0), les poids s'approchent du vecteur d'entrée ; dans notre cas, ils bougent dans la direction du point qui se trouve dans le coin haut droit.

Présentons maintenant le vecteur d'entrée suivant au perceptron modifié : Vecteur d'entrée 2-B2 :

$$x_1 = -0.6, \;\; x_2 = 0.3, \;\; t = 0; \quad w_1^1 = 0.95\,, \;\; w_2^1 = -0.15$$

$$u = (-0.6)(0.95) + (0.3)(-0.15) = -0.1305 < 0$$

$$u < 0 \Rightarrow y = 0$$

⇒ classification correcte

⇒ les poids ne seront pas modifiés

Ainsi le neurone a correctement classé les deux vecteurs d'entrée.

Présentons-lui le troisième vecteur 3-B1 :

$$x_1 = -0.1, \qquad x_2 = -0.8, t = 0; \quad w_1^1 = 0.95\,, w_2^1 = -0.15$$

$$u = (-0.1)(0.95) + (-0.8)(-0.15) = 0.025$$

$$u > 0 \Rightarrow y = 1$$

⇒ classification incorrecte

⇒ le perceptron surestime (i.e., t = 0, y = 1) ; les poids doivent être ajustés en utilisant la règle 2 de l'équation 2.12.

Supposons que les variations des deux poids soient notées par Δw_1^2 et Δw_2^2, l'exposant 2 dénote la deuxième variation du poids. Alors

$$\Delta w_1^2 = -\beta x_1 = -(0.5)(-0.1) = 0.05$$

$$\Delta w_2^2 = -\beta x_2 = -(0.5)(-0.8) = 0.4$$

Les nouveaux poids après la première incrémentation seront notés par w_1^2 et w_2^2 ,

$$w_1^2 = w_1^1 + \Delta w_1^2 = 0.95 + 0.05 = 1.0$$
$$w_2^2 = w_2^1 + \Delta w_2^2 = -0.15 + 0.4 = 0.25.$$

Sous forme vectorielle

$$\mathbf{w}^2 = \left[w_1^2, w_2^2\right] = [0.95, -0.15] + [0.05, 0.4] = [1.0, 0.25].$$

Le nouveau vecteur poids est $W^2 = (w_1^2, w_2^2) = \{1.0, 0.25\}$; sa norme notée par $||w^2||$ est

$$\mathbf{L}^2 = \|\mathbf{w}^2\| = \sqrt{1.0^2 + 0.25^2} = 1.03.$$

Le nouveau vecteur des poids est superposé aux données dans la figure 2.12 qui montre que la surestimation (i.e., t= 0, y = 1) pousse ce vecteur poids loin du vecteur d'entrée en question.

Présentons le quatrième vecteur d'entrée 4-A2 :

$$x_1 = 0.1, \;\; x_2 = -0.45, \;\; t = 1; \quad w_1^2 = 1.0\,, \;\; w_2^2 = 0.25$$
$$u = (0.1)(1.0) + (-0.45)(0.25) = -0.0125$$
$$u < 0 \Rightarrow y = 0$$

$\Rightarrow$ classification incorrecte

il y a sous estimation, la classification est incorrecte. Donc les poids doivent être ajustés en utilisant la règle 1 de l'équation 2.12

Supposons que les variations des deux poids soient notées par Δw_1^3 et Δw_2^3, l'exposant 3 dénote la troisième variation du poids. Alors

$$\Delta w_1^3 = \beta x_1 = (0.5)(0.1) = 0.05$$
$$\Delta w_2^3 = \beta x_2 = (0.5)(-0.45) = -0.225.$$

Notons les nouveaux poids après la troisième variation par w_1^3 et w_2^3

$$w_1^3 = w_1^2 + \Delta w_1^3 = 1.0 + 0.05 = 1.05$$
$$w_2^3 = w_2^2 + \Delta w_2^3 = 0.25 - 0.225 = 0.025$$

Sous forme vectorielle

$$\mathbf{W}^3 = [w_1^3, w_2^3] = [1.0, 0.25] + [0.05, -0.225] = [1.05, 0.025].$$

Le nouveau vecteur des poids est

$$\mathbf{W}^3 = (w_1^3, w_2^3) = (1.05, 0.025).$$

La longueur $\boldsymbol{L}^3$de ce vecteur poids est

$$\mathbf{L}^3 = \sqrt{1.05^2 + 0.025^2} = 1.05.$$

Le nouveau vecteur des poids est superposé aux données dans la figure 2.12, qui montre encore une fois que la sous estimation pousse ce vecteur vers le vecteur d'entrée.

Ainsi le vecteur des poids final est

$$\mathbf{W}^3 = (w_1^3, w_2^3) = (1.05, 0.025).$$

Le vecteur final ainsi que tous les précédents vecteurs sont superposés aux données dans la figure 2.12.

Il est maintenant possible de voir si tous les vecteurs d'entrée dont correctement classés en utilisant les poids finaux. Les résultats sont présentés dans la table 2.3, qui montre que le perceptron a correctement classé toutes les données.

Table 4.1 : Exactitude de classification du perceptron entraîné

Point D'entrée	x_1	x_2	t	U	y	Exactitude de la classification
A1	0.3	0.7	1	0.3x1.05+0.7x0.025 =0.3325>0	1	Correct
B1	-0.6	0.3	0	-0.6x1.05+0.3x0.025 =-0.6225<0	0	Correct
B2	-0.1	-0.8	0	-0.1x1.05+(-0.8)x0.025 =-0.125<0	0	Correct
A2	0.1	-0.45	1	0.1x1.05+(-0.45)x0.025 =0.09375>0	1	Correct

Cet exemple montre comment le perceptron apprend en ajustant les poids. Dans ce cas-ci, l'apprentissage est complet en une seule itération de l'ensemble des données, cela est dû à la nature linéaire du problème.

Une présentation ou itération de toutes les données est appelée une époque. Les poids bougent jusqu'à ce que le perceptron classe correctement toutes les données.

Pour tout vecteur d'entrée que le neurone sous estime, le nouveau vecteur poids est poussé vers ce vecteur, et pour tout vecteur d'entrée que le neurone surestime, le vecteur poids est poussé loin de ce vecteur.

D'autre part, l'apprentissage pas à pas au moyen de l'algorithme de l'apprentissage du perceptron résulte en un accroissement de la longueur du vecteur des poids après chaque ajustement. Ceci ne sera pas une qualité désirable lorsque nous avons un grand nombre des vecteurs d'entrée que nous devons utiliser pour l'apprentissage. Ceci risque de conduire à des problèmes calculatoires. La solution à cela existe dans des réseaux plus sophistiqués que nous verrons plus tard.

Quelles sont les caractéristiques de ce perceptron entraîné ?

3- Frontière de classification

Examinons la frontière de classification qui sépare les deux groupes.

Rappelons que la frontière est définie par la position de la fonction seuil qui, dans ce cas-ci est égal à u = 0.

$$u = w_1 x_1 + w_2 x_2 = 0$$

$$x_2 = -\left(\frac{w_1}{w_2}\right) x_1.$$

Ainsi, la ligne de frontière est donnée par les équations précédentes définissent une droite. Chaque ajustement des poids donne une frontière de classification différente ; ainsi, l'apprentissage est un processus qui cherche un ensemble de poids (w_1 et w_2) qui donnent la frontière de classification correcte.

la frontière finale de classification pour ce problème est obtenue en insérant les valeurs des poids finaux dans l'équation 2.13 :

$$x_2 = -(1.05/0.025)x_1 = -42x_1,$$
$$x_2 = -42x_1.$$

La pente de cette droite est (- 42).

La ligne de frontière est superposée aux données dans la figure 2.13.(suivante).

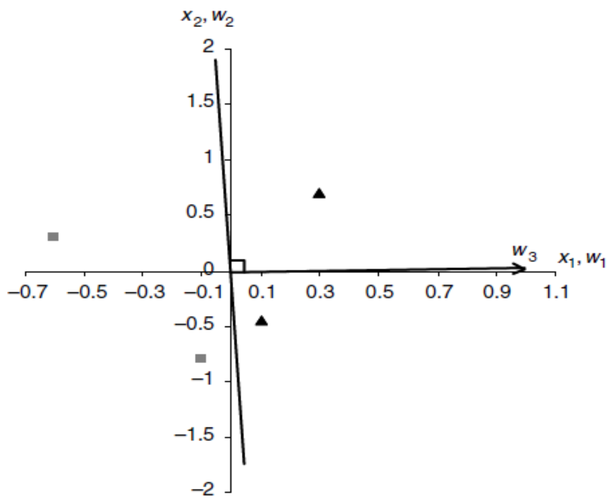


Figure 4.4: la frontière de classification du perceptron et le vecteur des poids finaux

Nous pouvons voir que la ligne de frontière sépare les données en deux catégories par une ligne droite, et le fait que la frontière de classification soit linéaire signifie que le perceptron peut être entraîné à résoudre tout problème de classification en deux classes dans lequel les classes peuvent être séparées par une ligne droite (i.e. linéairement séparable). Dans les problèmes de plus grandes dimensions, dans lesquels il y a plus de deux variables d'entrée, les classes sont séparables par un hyper plan ; l'algorithme et les concepts étudiés ici continuent de s'appliquer. Cependant, au delà de trois dimensions, nous ne pouvons pas les représenter visuellement.

Remarquons sur la figure 4.4 que le vecteur final des poids est perpendiculaire à la frontière de classification.

Ainsi, l'apprentissage trouve le vecteur des poids optimal afin de fixer une ligne de frontière qui lui est perpendiculaire et qui divise correctement le domaine des entrées en deux régions pour aboutir à une classification exacte.

Nous pouvons voir comment les poids définissent toujours une droite qui est perpendiculaire à leur vecteur.

Par exemple, pour un vecteur d'entrée $\boldsymbol{x} = \{x_1, x_2\}$ et un vecteur des poids $W = \{w_1, w_2\}$

$$u = x_1 w_1 + x_2 w_2,$$

Qui peut être exprimé comme suit, en utilisant les vecteurs

$$u = \|x\|.\|w\|.\cos(\theta)$$

Où $\|x\|$ et $\|w\|$ sont les modules du vecteur d'entrée et du vecteur poids, et θ est l'angle entre les deux.

Sur la frontière $u = 0$.

C'est-à-dire :

$$u = \|x\|.\|w\|.\cos(\theta)$$

Puisque $\|x\|$ et $\|w\|$ sont différents de zéro, alors $\cos(\theta)$ doit être égal à zéro.

Ce qui signifie que $\theta = \pm 90°$.

- Nous savons que $\cos(\theta)$ est positif entre $-90°$ et $+90°$. Donc toute entrée qui fait un angle compris entre $-90°$ et $+90°$ donne $u \geq 0$ et donc $y = 1$.

- Et toute entrée qui fait un angle supérieur à $+90°$ ou inférieur à $-90°$ donne $u < 0$ et donc $y = 0$.
- Le vecteur poids et la droite de discrimination sont perpendiculaires.

Un exemple pratique de perceptron sur des données réelles :

Identification de l'origine des poissons à partir du diamètre de l'anneau de croissance des écailles.

Un problème simple mais réaliste sera résolu à l'aide du perceptron. Les autorités gouvernementales concernées par l'épuisement des stocks de saumon décident de réguler la pêche.

Pour cette fin, il est nécessaire d'identifier quand est-ce que l'origine d'un poisson est l'Alaska ou bien le Canada.

Cinquante poissons de chaque lieu d'origine ont été capturés, et le *diamètre de l'anneau de croissance des écailles* a été mesuré pour le temps où les poissons ont vécu en eau *douce* puis pour le temps suivant où ils ont vécu en eau *salée*.

Le but est d'identifier l'origine d'un poisson à partir du diamètre de l'anneau de croissance dans l'eau douce et dans l'eau salée.

Cette section va étudier comment le perceptron va classer le poisson comme originaire de l'Alaska ou du Canada.

La figure 4.5 montre la relation entre les deux variables, qui sont séparées en deux groupes distincts.

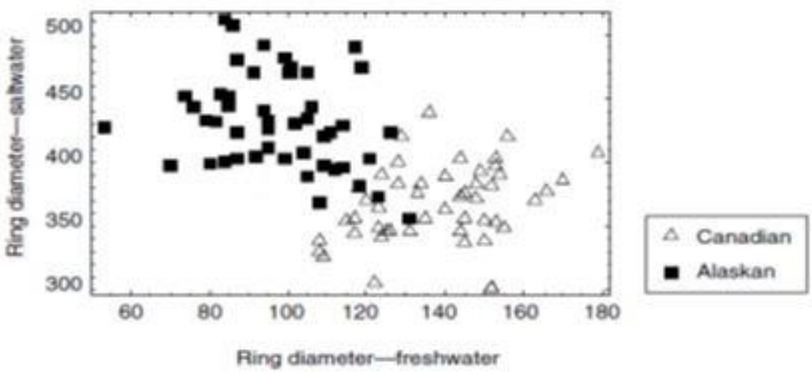


Figure 4.5

Parce qu'il y a deux entrées, un classificateur à deux entrés doit être entraîné. Il y a 50 vecteurs inputs dans chacune des deux classes, classe 1 (Canadien) et classe 2 (Alaskien).

Un seul nœud d'output représente les deux classes avec un output 0 pour la première classe et un output 1 pour représenter l'autre classe.

A cause du grand nombre des vecteurs seuls les résultats seront montrés, mais le processus est exactement le même que celui que nous avons déjà étudié dans la section précédente.

L'évolution des frontières de classification pour ces données en quelques points du processus d'entraînement est montrée dans la figure 4.6.

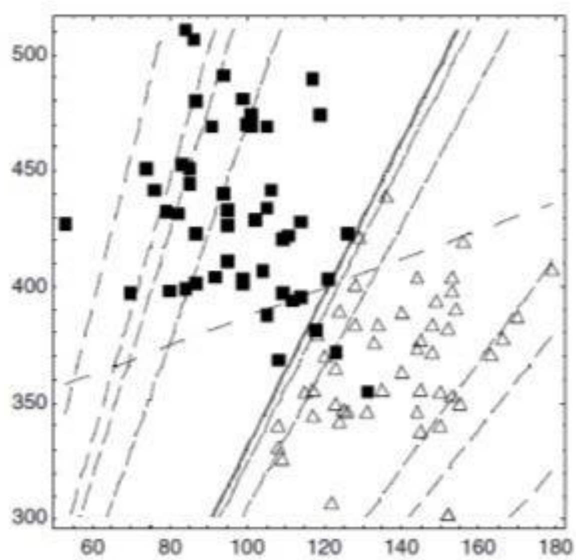


Figure 4.6 Progrès de la classification

La figure 4.6 décrit les résultats de l'entraînement.

On peut voir que la frontière initiale de classification est loin de ce qu'elle devrait être (au coin bas droit de la figure 4.6) et l'apprentissage repositionne incrémentalement la frontière jusqu'à ce que la classification correcte soit obtenue.

Montrant maintenant comment la classification s'améliore avec chaque époque.

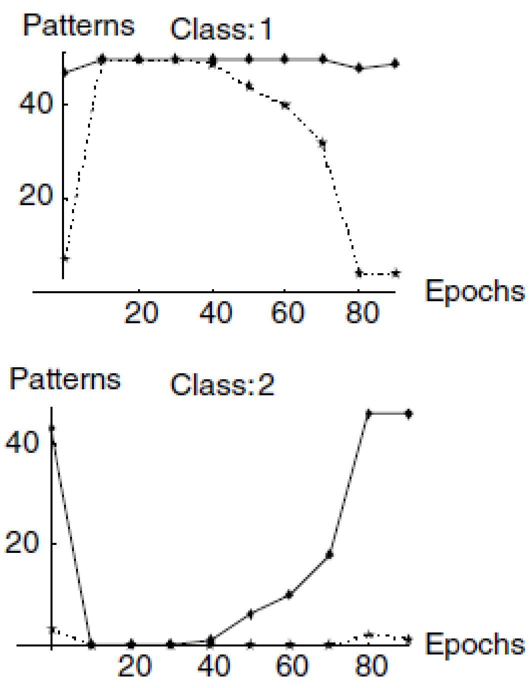


Figure 4.7 : Données classées correctement/incorrectement

Dans beaucoup de problèmes réels, l'ensemble des données d'entrée doit être présenté d'une manière répétée sur plusieurs époques, et les poids sont ajustés jusqu'à ce que la classification correcte soit atteinte. Cela est dû au fait que chaque vecteur mal classé repositionne la frontière et peut annuler les changements causés par d'autres vecteurs qui l'ont précédé.

La figure 4.7a montre le nombre de vecteurs d'entrée classés comme classe 1, et la figure 4.7b montre le nombre de vecteurs classés comme classe 2 après chaque époque.

La ligne continue représente les vecteurs correctement classés dans la classe en question et la ligne en pointillés montre les vecteurs classés à tord comme appartenant à cette même classe.

Par exemple, l’image du haut dans la figure 4.7 montre que le perceptron apprend immédiatement à classer les données de la classe 1, mais il classe à tord quelques données classe 2 comme appartenant à la classe 1.

Cependant, incrémentalement le perceptron apprend à classer les données classe 2 correctement, et après 80 époques, il classe la majorité (presque toutes) les données correctement.

L’image du bas de la figure 4.7 montre la qualité de la classification pour la classe 2. Dans les itérations du début, le perceptron a des difficultés pour différentier et classer les vecteurs, mais plus tard il apprend à classer correctement les données classe 2 et élimine les données classe 1 qui ont été classé à tord comme appartenant à la classe 2.

La figure 4.8 montre la frontière finale de classification superposée sur les données, elle montre que le perceptron a trouvé la meilleure frontière de classification linéaire pour ce problème. Après que l’entraînement soit fini, l’exactitude de la classification peut être évaluée.

Dans cet exemple, il y avait 50 vecteurs dans chacune des deux classes, et le perceptron a correctement classé 97 d’entre eux, avec seulement trois mauvaises classifications. Ainsi, l’exactitude de la classification pour les saumons canadiens est 100 pour cent, et pour les saumons alaskiens est de 94 pour cent ; l’exactitude de la classification globale est 97 pour cent.

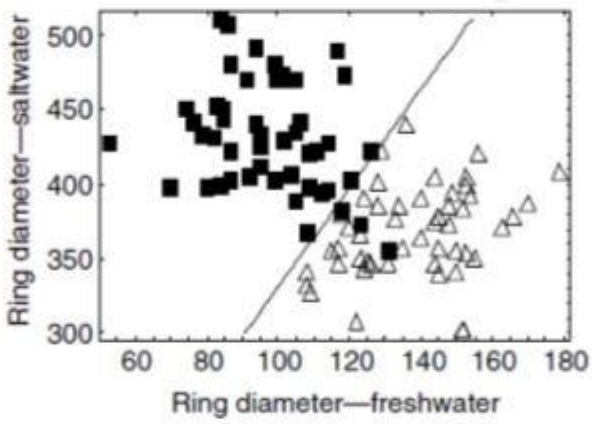


Figure 4.8 : la frontière finale de classification produite par le perceptron.

4- Comparaison entre le perceptron et l'analyse discriminante linéaire de la statistique.

L'analyse discriminante est une méthode statistique multivariée utilisée pour analyser simultanément la différence entre des catégories en termes de plusieurs variables numériques indépendantes.

Comme classificateurs, le perceptron et l'analyse discriminante linéaire sont **équivalents.** Pour montrer cela, l'analyse discriminante a été faite sur l'ensemble des données du saumon utilisé pour la classification par le perceptron qu'on a vu précédemment.

La frontière de classification résultante est montrée en même temps avec la frontière de classification donnée par le perceptron qui ont été montré dans la figure 4.9.

La ligne en pointillés représente la frontière de classification par l'analyse discriminante et la ligne en continu représente la ligne du perceptron.

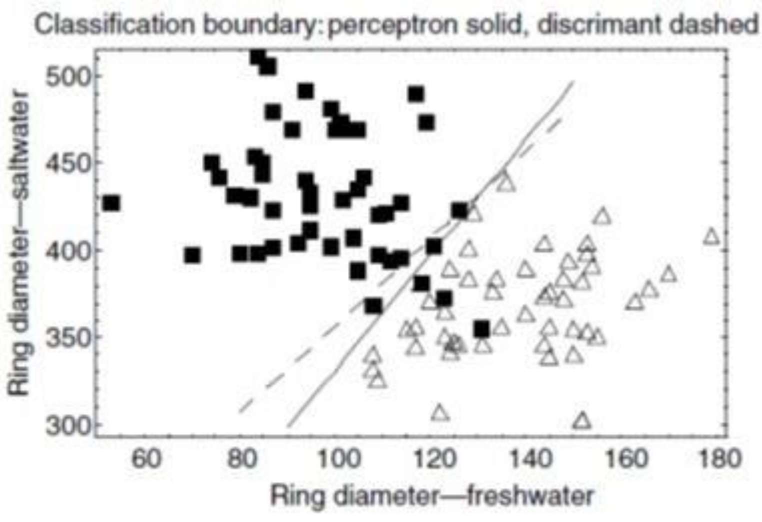


Figure 4.9 : Frontière de classification : perceptron (trait plein), droite de discrimination (tirets)

La frontière de classification par le perceptron est

$$x_2 = 0.026 + 3.31x_1.$$

La frontière de classification par la fonction discriminante est

$$x_2 = 106.9 + 2.5x_1$$

La figure 4.9 montre la similarité entre les deux classificateurs. Cependant, la qualité de la classification par le perceptron est légèrement meilleure que celle par la fonction discriminante.

1. Pour la discrimination linéaire (en pointillés dans la figure 4.9) l’exactitude de la classification pour le saumon d’Alaska est 90%, avec cinq mauvaises classifications, l’exactitude pour le saumon du Canada est 100% et l’exactitude de la classification globale pour l’ensemble des données est 95%.
2. Ces valeurs pour le perceptron (ligne en continue dans la figure 4.9) sont 94%, 100% et 97% respectivement.

5- Perceptron à sortie multiples pour la classification à plusieurs catégories

Les problèmes de classification impliquant plus de deux classes de sortie peuvent être traités avec un perceptron à sorties multiples qui, pour chaque classe, possède un neurone de sortie dans la couche de sortie.

Le processus d'entraînement pour un perceptron à plusieurs sorties crée une application qui affecte chaque vecteur input à la classe correcte en ajustant itérativement les poids pour produire une sortie égale à 1 pour le neurone de sortie correspondant et la valeur 0 pour tous les autres neurones.

Cette section va examiner la performance du réseau du perceptron dans la projection de deux dimensions vers des classes multiples.

La figure 4.10 montre des données appartenant à trois classes ; ce qui demande trois neurones perceptron de sortie, chacun pour une classe, comme montré dans la figure 4.11. Il y a 20 points de données dans chaque classe.

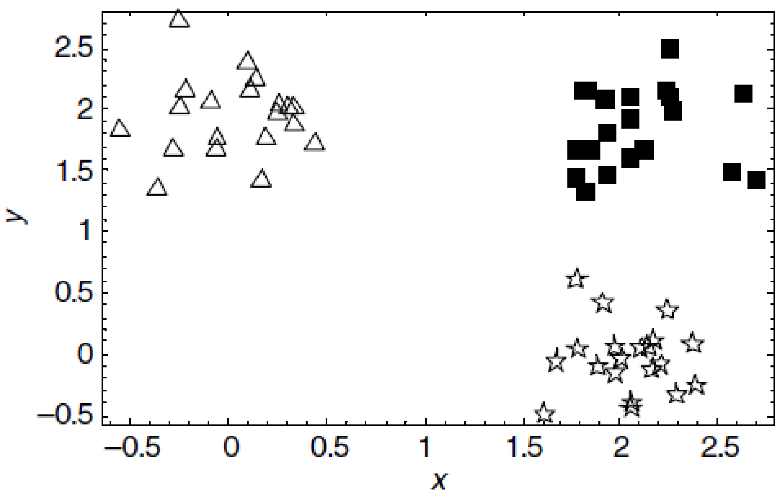


Figure 4.10

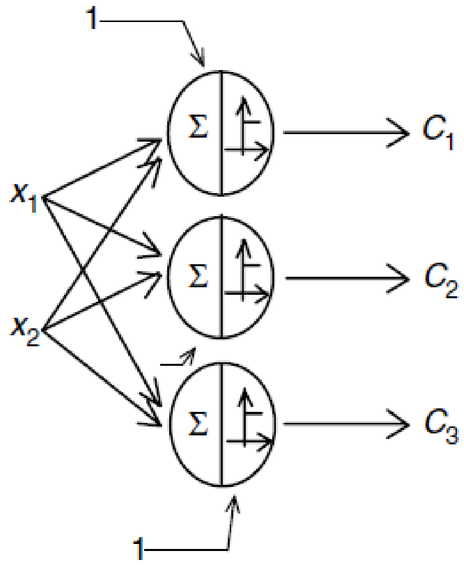


Figure 4.11

Le processus d’entraînement est similaire à celui utilisé pour un seul neurone, sauf que pour la sortie requise des trois perceptrons pour la classe 1 doit être 1, 0, 0 et pour la classe 2 elles seront 0, 1, 0 et pour la classe 3 elles seront 0, 0, 1 respectivement.

Durant l’entraînement, la sortie des trois perceptrons est comparée avec les sorties cibles, et tout perceptron qui produit des classifications incorrectes trouve ses poids ajustés au moyen de la règle d’apprentissage sur plusieurs époques jusqu’à ce que tous les perceptrons classent correctement les données qu’ils représentent et ne classent pas à tord les données qui ne leurs appartiennent pas.

Les frontières finales de classification obtenues de l’entraînement sont superposées aux données dans la figure 4.12.

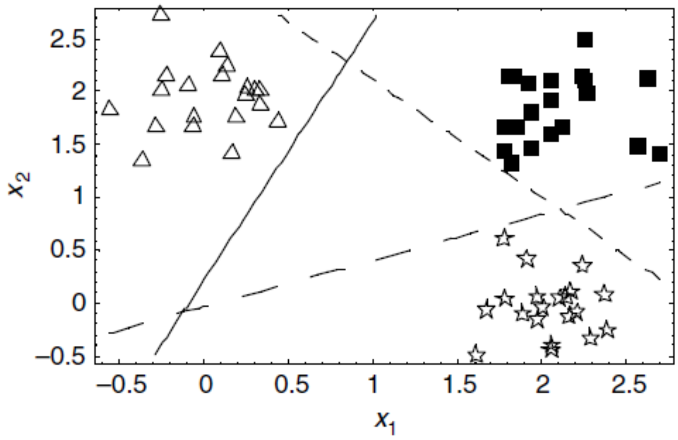


Figure 4.12.

Du fait que chaque perceptron défini une frontière de classification linéaire, il y a trois frontières linéaires, comme montré dans la figure. Chaque frontière de classification sépare une classe des deux autres. A cause du fait que ce problème est adapté à la classification linéaire, le réseau du perceptron classe correctement toutes les données.

Pour montrer que le réseau a appris à classer correctement, un vecteur input a été choisi de chaque classe aléatoirement pour être présenté au réseau. Les inputs et les réponses correspondantes sont montrés dans la table 4.2.

Entrées	Sorties	Cibles
(-0.56, 1.84)	(1, 0, 0)	(1, 0, 0)
(2.23, 2 .15)	(0, 1, 0)	(0, 1, 0)
(2.00, -0.04)	(0, 0, 1)	(0, 0, 1)

Table 4.2 : Les entrées et les sorties correspondantes fournies par le perceptron à multiples sorties

Ainsi, le réseau entraîné classe les trois vecteurs input. La frontière de classification pour chaque neurone montré dans la figure 4.12 peut être facilement construite à partir des poids correspondants du réseau entraîné.

Rappelons que chaque neurone est un neurone à seuil et que le processus qui a été utilisé pour extraire la ligne de frontière pour un perceptron s'applique ici pour chacun des trois perceptrons. L'équation extraite pour chaque ligne de classification est donnée par

$$-3.8 - 42.8x_1 + 17.5x_2 = 0$$
$$-78.2 + 26.9x_1 + 24.3x_2 = 0$$
$$-1.35 + 16.6x_1 - 38.0x_2 = 0.$$

Les coefficients dans les équations représentent les poids w_1 et w_2 pour chaque neurone, et l'ordonnée à l'origine représente le poids associé au biais dont l'input est +1.

La figure 4.13 montre comment la classification s'améliore après chaque époque pour chacun des neurones représentant les classes 1, 2 et 3 respectivement.

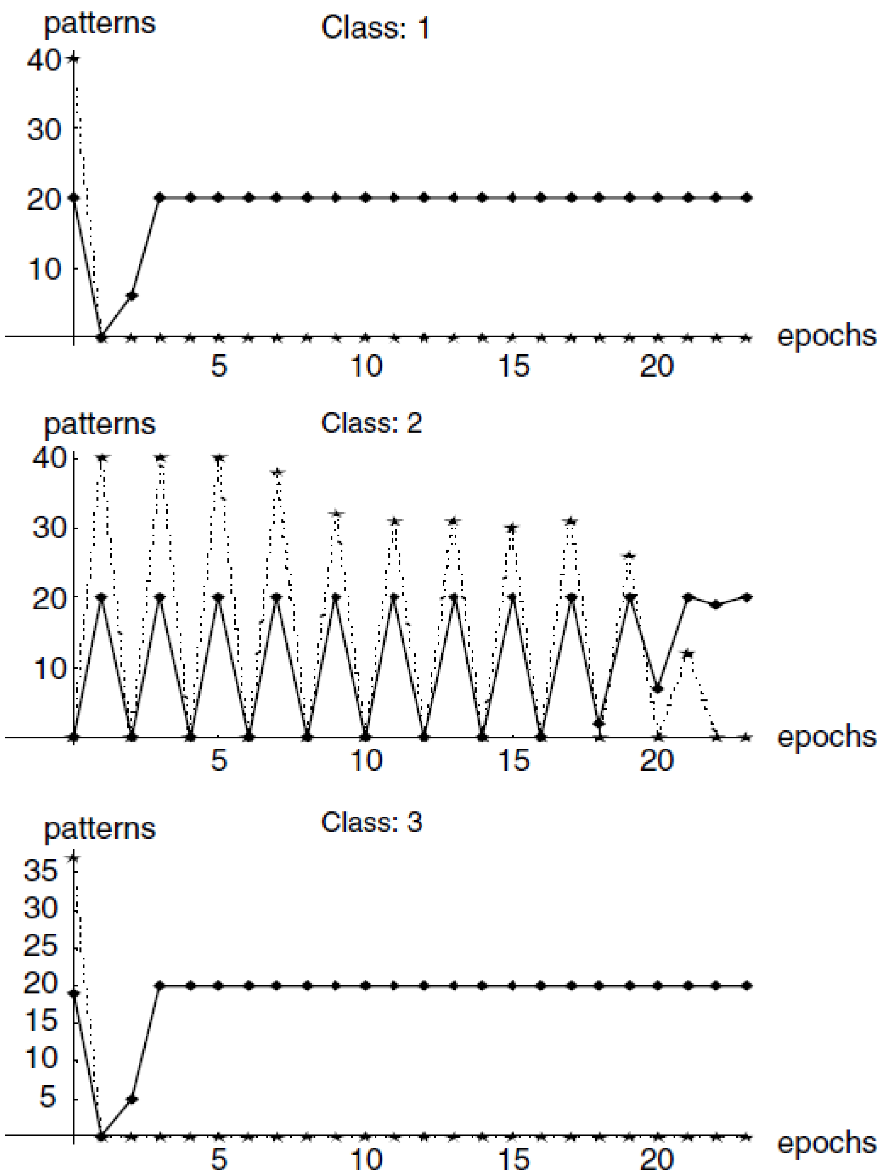


Figure 4.13

Les neurones 1 et 3 commencent par mal classer puis apprennent à classer correctement assez rapidement, tandis que le neurone 2 apprend les données de la classe 2 avec un peu de difficulté. Le neurone 2 classe à tord un grand nombre de vecteurs appartenant aux autres classes puis les élimine lentement durant les époques.

Après 25 époques, les trois neurones produisent une classification correcte sur les 20 points de données représentées par chaque neurone, et produisent les lignes de frontières individuelles.

Comparaison avec l'analyse discriminante multiple

Les fonctions de discriminations multiples sont fonctionnellement similaires au classificateur du perceptron à sorties multiples dans le fait que les fonctions tentent de séparer une classe des deux autres.

Globalement, la performance du perceptron est parfaite dans cet exemple et la performance de l'analyse discriminante est légèrement inférieure.

Les équations pour les fonctions discriminantes de Fisher sont :

$$-31.4 + 29.7x_1 - 0.953x_2 = 0$$
$$-49.5 + 30.0x_1 + 18.5x_2 = 0$$
$$-20.6 - 0.231x_1 + 20.1x_2 = 0$$

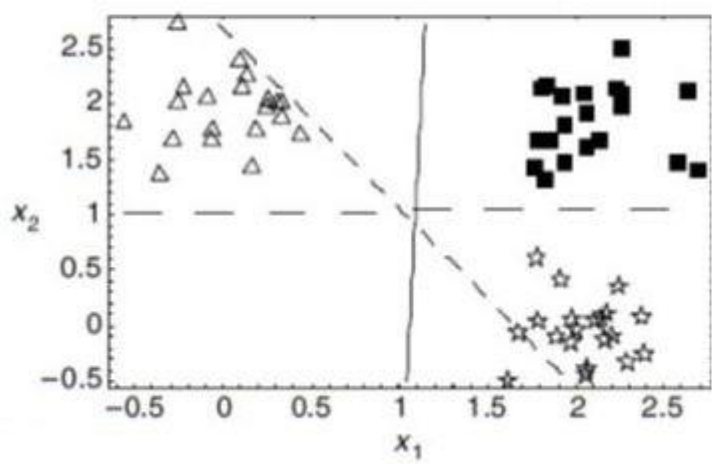


Figure 4.14