CHAPITRE 4 KOHONEN

Nous sommes des nains assis sur des épaules de géants. Si nous voyons plus de choses et plus lointaines qu'eux, ce n'est pas à cause de la perspicacité de notre vue, ni de notre grandeur, c'est parce que nous sommes élevés par eux.

BERNARD DE CHARTRES , XII^{e}siècle*

EN reconnaissance des formes ou en intelligence artificielle, le problème le plus difficile actuellement est la visualisation de données complexes. En effet, on en cherche généralement une représentation simple, basée sur un petit nombre de traits caractéristiques mais qui permet néanmoins de préserver les relations importantes qui les relient.

En d'autres termes, on cherche à créer une représentation simplifiée des observations, ayant un certain niveau d'abstraction et permettant de prendre certaines décisions.

Ce problème est d'autant plus difficile que la base de données est de grande taille.

Il s'agit en fait d'un problème géométrique qui traite de la proximité de points dans un certain espace métrique.

Une première solution à ce problème est de construire un modèle qui sera utilisé par la suite. C'est l'approche paramétrique.

Une autre solution, dite non paramétrique, est basée sur les données elles-mêmes pour effectuer la classification désirée. Une nouvelle donnée est comparée aux anciennes, et c'est cette comparaison qui permet de donner la classe d'appartenance. Il n'est pas nécessaire de passer par l'étape intermédiaire qui consiste à identifier un modèle statistique.

L'algorithme de Kohonen fait partie de ces méthodes non paramétriques. Il est basé sur l'apprentissage compétitif. Un de ses intérêts est qu'il est non supervisé (du moins dans sa version clusterisation de données).

4.1 Préliminaires

Le but est d'associer des vecteurs donnés, appartenant éventuellement à un espace de très grande dimension, avec un certain nombre de vecteurs de référence, encore appelés *vecteurs prototypes*

** Rapportée par Jean de Salisbury ((vers 1124-1130), évêque de Chartres d'origine anglaise), cette image suggère comment la scolastique du moyen-âge se situe elle-même par rapport à la patristique.*

ou encore *codebooks*. La distribution de probabilité de ces vecteurs prototypes doit refléter (selon une certaine mesure à préciser) la distribution de probabilité des signaux d'entrée mais, en général, cette dernière n'est pas connue car on ne dispose le plus souvent que d'un nombre fini (généralement petit) de vecteurs d'exemples.

Soit N le nombre (donné a priori) de vecteurs prototypes. Chacun de ces vecteurs sera affecté à une unité (qu'on peut, suivant le contexte, appeler neurone ou encore centre par exemple). Les unités appartiennent donc à $\mathcal{U} = \{u_1, \ldots, u_N\}$ et à chaque unité u_i est associé le vecteur de référence $\mathbf{w}_i$.

Les unités sont reliées entre elles par des connexions et il faut souligner que ces connexions n'ont rien à voir avec celles que l'on trouve dans les MLP. Elles décrivent quelles unités sont voisines d'une unité donnée ; en d'autres termes elles fournissent les voisinages topologiques des unités. L'ensemble des connexions est $\mathcal{C}$ tel que :

$$\mathcal{C} \subset \mathcal{U} \times \mathcal{U}\,, \qquad \text{une connexion sera notée} \quad (u_i, u_j) \in \mathcal{C} \tag{4.1}$$

L'ensemble des voisins d'une unité i sera donc :

$$N_i = \{u_j \in \mathcal{U} \,|\, (u_i, u_j) \in \mathcal{C}\} \tag{4.2}$$

Soient alors $\mathbf{x}$ les signaux d'entrée (vecteurs de dimension n). Deux cas peuvent alors se présenter :

- Ces entrées sont générées à partir d'une loi de distribution continue :

$$p(\mathbf{x}),\ \mathbf{x} \in \boldsymbol{R}^n \tag{4.3}$$

- Ces entrées sont tirées d'un ensemble fini $\mathcal{D}$ de M données :

$$\mathcal{D} = \{\mathbf{x}_1, \ldots, \mathbf{x}_M\} \quad \mathbf{x}_i \in \boldsymbol{R}^n \tag{4.4}$$

Pour un signal d'entrée donné $\mathbf{x}$, on défini l'unité gagnante $g(\mathbf{x})$ comme l'unité qui possède le vecteur de référence le plus proche de $\mathbf{x}$:

$$g(\mathbf{x}) = \text{argmin}_{u_i \in \mathcal{U}} \parallel \mathbf{x} - \mathbf{w}_i \parallel \tag{4.5}$$

où $\parallel \cdot \parallel$ est la norme euclidienne.

Nous allons maintenant introduire deux concepts fondamentaux qui sont la *tessellation de Voronoï* et son dual, la *triangulation de Delaunay*.

4.1.1 Tessellation de Voronoï

Considérons un ensemble de points $\{\mathbf{w}_1, \ldots, \mathbf{w}_N\}$ de $\boldsymbol{R}^n$ (voir figure 4.1-a). La cellule de Voronoï relative au vecteur $\mathbf{w}_i$ est définie comme l'ensemble des points de $\boldsymbol{R}^n$ pour lesquels $\mathbf{w}_i$ est le vecteur le plus proche :

$$V_i = \{\mathbf{x} \in \boldsymbol{R}^n \,|\, i = \text{argmin}_{j \in \{1,\ldots,N\}} \parallel \mathbf{x} - \mathbf{w}_j \parallel\} \tag{4.6}$$

Chaque cellule définit donc une classe d'équivalence.

On sait que les cellules de Voronoï sont des domaines convexes, qui respectent donc la relation suivante :

$$(\mathbf{x}_1 \in V_i \wedge \mathbf{x}_2 \in V_i) \quad \Rightarrow \quad (\mathbf{x}_1 + \alpha(\mathbf{x}_2 - \mathbf{x}_1) \in V_i) \qquad \forall \alpha, \quad 0 \leq \alpha \leq 1 \tag{4.7}$$

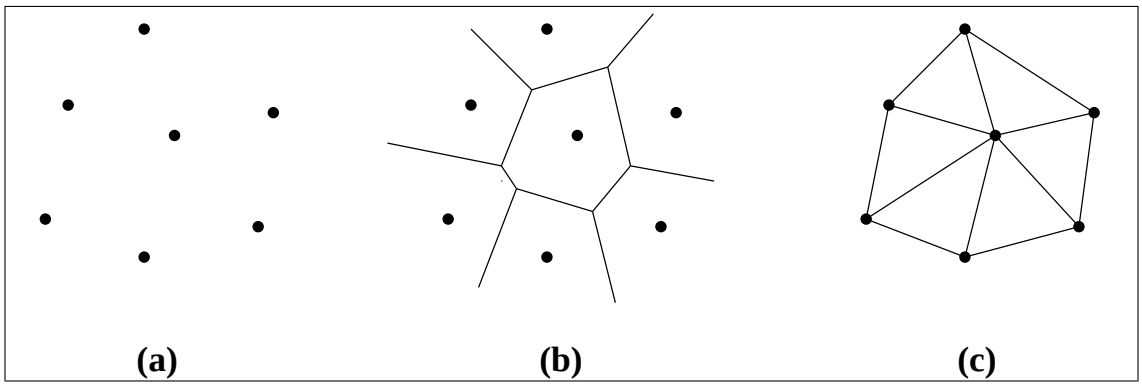

FIG. 4.1 – Tessellation de Voronoï et triangulation de Delaunay.

La partition de $\boldsymbol{R}^n$ formée de tous les polygones de Voronoï est appelée *Tessellation de Voronoï* ou encore *Tessellation de Dirichlet*. Elle est dessinée sur la figure 4.1-b. Pour la construire, on trace les médiatrices des segments joignant deux points voisins. C'est facile lorsque la figure est simple, mais cela devient très complexe dès qu'il y a beaucoup de points. Il existe des programmes qui font cela pour des données en deux dimensions seulement, mais le concept est valide quelle que soit cette dimension.

4.1.2 Triangulation de Delaunay

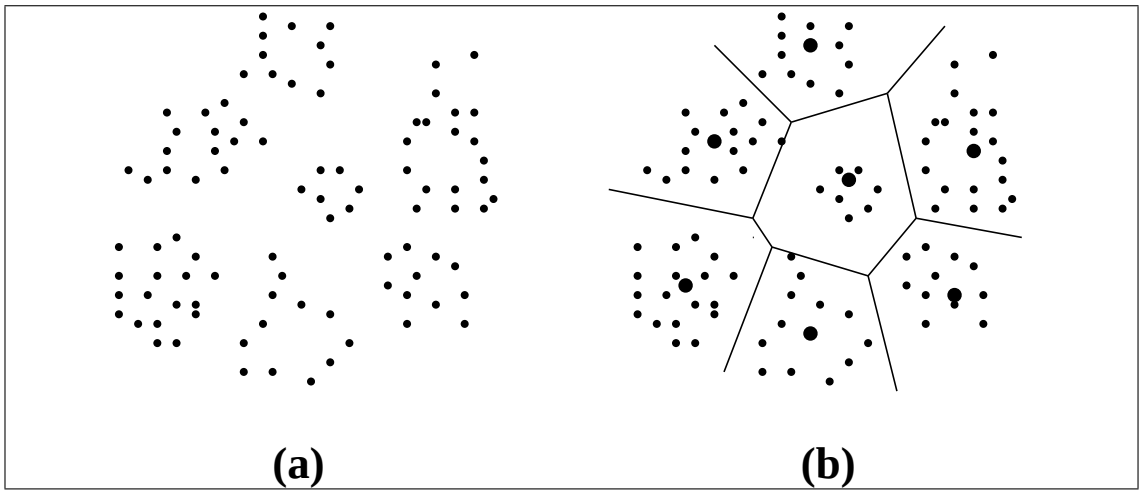

FIG. 4.2 – Tessellation de Voronoï sur des nuages de points formant des amas

Si nous relions toutes les paires de points qui partagent une frontière de Voronoï commune, nous obtenons la *triangulation de Delaunay* (voir figure 4.1-c). Parmi toutes les triangulations possibles celle-ci possède des propriétés particulières de rapidité (à calculer) et d'optimalité. On l'utilise par exemple en robotique et en traitement d'images.

4.1.3 Utilisation de la tessellation de Voronoï

Supposons que les données soient à deux dimensions et qu'elles soient représentées par la figure 4.2-a. On *voit* qu'il y a 7 classes dont les centres de gravité sont représentés par les gros points.

La tessellation de Voronoï est la partition de l'espace qui sépare *au mieux* (au sens de la métrique euclidienne) ces données en 7 classes.

On appelle souvent chacun des polyèdres de Voronoï V_i, le champ récepteur de l'unité i (dans une métaphore réseau de neurones bien sûr).

Cette tessellation peut être obtenue à l'aide de l'algorithme des *K-moyennes* par exemple, un algorithme simple et général, mais qui souffre de plusieurs graves défauts comme une sensibilité très forte aux outliers et des résultas très mauvais dès que les nuages de points se recouvrent.

On connaît beaucoup de théorèmes concernant les propriétés des diagrammes de Voronoï et des triangulation de Delaunay, mais ils ne sont valides que pour le plan (la raison profonde est fondée sur le fait que les formules d'Euler ne sont valables que dans ce cas [MS94]).

L'algorithme de Kohonen effectue lui aussi une tessellation de Voronoï. Nous allons voir dans les sections suivantes comment il fonctionne.

4.2 Algorithme de Kohonen

4.2.1 Minimiser une erreur au sens de Bayes

Supposons pour commencer que l'on ait deux classes C_1 et C_2 et que les motifs de C_1 et de C_2 aient les densité de probabilité $p_1(x)$ et $p_2(x)$. Supposons que l'on connaisse aussi les probabilité a priori π_1 et π_2 des deux classes. Un motif appartenant à C_1 sera noté m_1 (m_2 pour C_2). Alors on peut construire un classifieur bayésien.

Le classifieur peut faire deux types d'erreur : il peut déclarer C_1 alors que c'est le motif m_2 qui est observé ou C_2 alors que c'est le motif m_1 qui est observé. Il fait obligatoirement des erreurs quand les probabilités des motifs des deux classes se recouvrent.

Le but d'une classification optimale est de minimiser l'erreur due aux mauvaises classifications. La règle de décision bayésienne est [LaV89] :

$$\begin{aligned} \text{Coût}(C_1, C_2) &= \int_{C_1} p_2(x)\pi_2 \, dx + \int_{C_2} p_1(x)\pi_1 \, dx && (4.8) \\ &= \pi_1 + \int_{C_1} (p_2(x)\pi_2 - p_1(x)\pi_1) \, dx && (4.9) \end{aligned}$$

Ce coût est minimum quand tous les points pour lesquels l'expression sous l'intégrale est positive sont déclarés appartenant à la région C_2. La surface de décision optimale est ainsi définie par :

$$\begin{aligned} C_2 &= \{x | p_2(x)\pi_2 - p_1(x)\pi_1 > 0\} && (4.10) \\ C_1 &= \text{le complémentaire} && (4.11) \end{aligned}$$

4.2.2 Minimiser une erreur sur un critère de distance

Afin de classer des données, on cherche à minimiser une erreur qui, si les données $\mathbf{x}$ sont issues d'une loi de probabilité continue de densité $p(\mathbf{x})$, s'écrit (voir [Fri97] par exemple) :

$$E(p(\mathbf{x}), \mathcal{U}) = \sum_{u \in \mathcal{U}} \int_{V_u} \| \mathbf{x} - \mathbf{w}_u \|^2 \; p(\mathbf{x}) \, d\mathbf{x} \qquad (4.12)$$

où V_u est la région de Voronoï de l'unité u.

Si on ne dispose que d'un ensemble fini de données $\mathcal{D}$, l'erreur à minimiser est :

$$E(\mathcal{D},\mathcal{U}) = \frac{1}{|\mathcal{D}|} \sum_{u \in \mathcal{U}} \sum_{\mathbf{x} \in R_u} \| \mathbf{x} - \mathbf{w}_u \|^2 \tag{4.13}$$

où R_u est l'ensemble des points de la région de Voronoï associée à l'unité u. C'est généralement dans ce cas que nous nous plaçons.

Les équations (4.12) et (4.13) montrent toute la difficulté qu'il y a à trouver à la fois :
- les bons vecteurs de référence,
- les domaines de Voronoï,
- tout en tenant compte si possible de la distribution de probabilité des données,
- et en minimisant l'erreur.

4.2.3 Utiliser un espace de représentation de petite dimension

Il est possible de projeter l'espace des données sur un espace de petite dimension avec la contrainte que les relations existant entre les données (on pense à des structures d'agrégats par exemple) se retrouvent dans le plan de projection. L'interprétation visuelle des données devient alors plus aisée.

Si, de plus, le plan de projection est défini comme une grille de dimension finie, où chacun des nœuds de la grille est en relation de voisinage avec un certain nombre de voisins, on construit alors une projection qui conserve (au moins en partie) les voisinages topologiques des données.

L'algorithme de Kohonen [Koh89] possède ces propriétés. Néanmoins, quelquefois, si la structure topologique des données n'est pas connue a priori, ou si elle est trop complexe, la conservation parfaite de la topologie n'est pas obtenue [MS94].

4.2.4 Cartes auto-organisatrices de Kohonen

Généralement la grille est à deux dimensions et à chaque nœud de cette grille se trouve une unité u_{ij} mais on peut utiliser aussi une ficelle où chaque unité u_i possède donc deux voisins, un à gauche et un à droite.

Soit alors un vecteur de donnée $\mathbf{x}$, l'unité gagnante de la grille $u_{ij} = g(\mathbf{x})$ et une unité quelconque u_{kl}.

La distance entre ces deux unités sera mesurée par la norme L_1 encore appelée distance de Manhattan :

$$d_1(u_{ij}, u_{kl}) = |i - k| + |j - l| \tag{4.14}$$

On procédera de manière itérative, avec un nombre d'itérations t_{max} fixé à l'avance.

L'algorithme de Kohonen se déroule de la façon suivante :

1. Initialiser les $N = N_1 \times N_2$ unités de la grille avec des vecteurs de référence $u_{ij} \in \boldsymbol{R}^n$ choisis aléatoirement.
2. Fixer les connexions (par exemple les voisins d'une unité sont les unités nord, sud, est et ouest).
3. Initialiser le temps : $t = 0$.
4. Tirer au hasard un vecteur de donnée $\mathbf{x}$.
5. Déterminer le gagnant $g = g(\mathbf{x})$

$$g(\mathbf{x}) = \text{argmin}_{u \in \mathcal{U}} \| \mathbf{x} - \mathbf{w}_u \| \tag{4.15}$$

6. Modifier chaque unité v de la quantité :

$$\Delta \mathbf{w}_v = \alpha(t)\, h_{gv}(\mathbf{x} - \mathbf{w}_v) \tag{4.16}$$

où :

$$\alpha(t) = \alpha_{init}(\frac{\alpha_{final}}{\alpha_{init}})^{\frac{t}{t_{max}}} \tag{4.17}$$

et :

$$h_{gv} = exp(-\frac{d_1(gv)}{2\sigma(t)^2}) \tag{4.18}$$

$$\text{avec} \quad \sigma(t) = \sigma_{init}(\frac{\sigma_{final}}{\sigma_{init}})^{\frac{t}{t_{max}}} \tag{4.19}$$

7. Incrémenter le temps d'une unité.
8. Si $t < t_{max}$ retourner à l'étape 4

4.2.5 Exemple

Soit un espace d'entrée à deux dimensions. Tirons aléatoirement des vecteurs d'entrée à l'intérieur d'un triangle de cet espace. Nous allons placer une ficelle (une dimension) dans cet espace de façon à conserver *au mieux* les relations de voisinage.

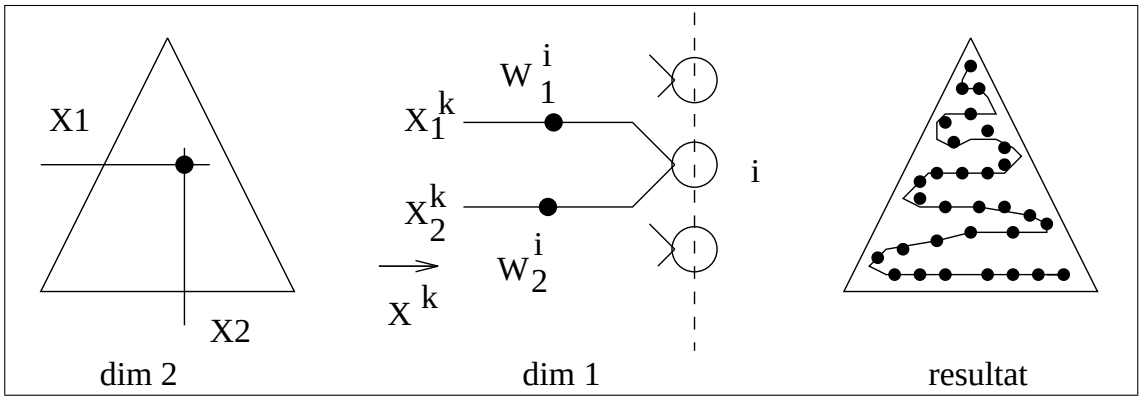

FIG. 4.3 – Exemple.

Un vecteur k de l'espace d'entrée est donc $\mathbf{x^k} = {}^{\mathrm{T}}(x_1^k, x_2^k)$ (voir FIG. 4.3).

Chaque unité i possède un vecteur de référence $\mathbf{w} = {}^{\mathrm{T}}(w1, w_2)$.

On cherche l'unité gagnante, c'est-à-dire celle dont le vecteur de référence $\mathbf{w}$ est le plus proche de $\mathbf{x}^k$.

On modifie un peu le vecteur de référence selon les formules ci-dessus, pour l'unité gagnante et pour le voisinage (mais un peu moins au fur et à mesure qu'on s'éloigne de l'unité gagnante). Cette modification est de plus en plus faible, au fur et à mesure que le nombre d'itérations augmente.

Si on trace les vecteurs de référence dans l'espace d'entrée, on observe que la ficelle s'est déployée de façon à couvrir cet espace en conservant une partie des relations de voisinage.

4.2.6 Remarques

Ce n'est pas seulement le vecteur de référence de l'unité gagnante qui est significatif, mais aussi la localisation de celle-ci. En effet, ses voisines correspondent aussi à des entrées voisines. C'est pour cela que la grille de Kohonen s'appelle souvent *carte topologique* de Kohonen car elle préserve les relations de voisinage.

Ces cartes topologiques, comme toute méthode de quantification vectorielle sont destinées à approximer des signaux d'entrée, ou leurs densité de probabilité, par des vecteurs de référence, un codebook, placé dans l'espace d'entrée de manière à minimiser une fonction d'erreur.

Mais si les signaux doivent être classés de manière supervisée en un certain nombre de catégories, il est nécessaire d'adapter l'algorithme de Kohonen pour faire de la *quantification vectorielle* (LVQ).

4.3 Learning Vector Quantization (LVQ)

Supposons que chaque classe soit caractérisée par un ensemble fixe d'unités calculant des distances. Pour classer un vecteur $\mathbf{x}$, on sélectionne le vecteur de référence le plus proche $\mathbf{w}_i$ et on regarde la classe $\Phi(\mathbf{w}_i)$ qui lui est associée.

Dans le diagramme de Voronoï associé aux divers vecteurs de référence, les frontières de séparation entre classes sont constituées des portions de frontières du diagramme de Voronoï séparant deux vecteurs de référence associés à deux classes différentes.

On réalise ainsi des frontières non linéaires, éventuellement non convexes s'il y a plusieurs vecteurs de référence par classe (voir FIG. 4.4).

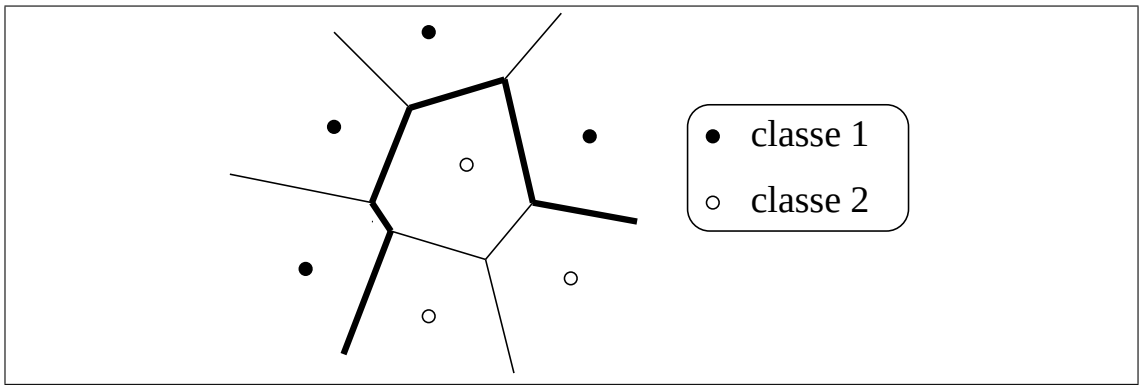

FIG. 4.4 – Frontières de séparation entre classes.

Il existe plusieurs algorithmes, basés sur les cartes topologiques de Kohonen, ajustant de façon itérative au mieux ces vecteurs de référence : LVQ1, LVQ2 et LVQ3 [Koh90].

Une procédure classique pour l'initialisation est d'utiliser un algorithme des K-moyennes sur tous les exemples pour placer les références, puis de leur associer la classe dominante par un vote majoritaire. Une autre méthode consiste à utiliser la même méthode sur tous les exemples de chaque classe pour placer le vecteur de référence associé. On peut bien entendu utiliser les cartes topologiques de Kohonen à la place des K-moyennes.

Remarque : Le choix du nombre de vecteurs de référence est plutôt arbitraire. Une règle empirique, que l'on utilise aussi dans d'autres méthodes non paramétriques, est d'en prendre $\sqrt{N}$ où N est le nombre de vecteurs d'apprentissage.

4.3.1 LVQ1

On tire au hasard un exemple $\mathbf{x}$ et on détermine le vecteur de référence le plus proche. Si ce vecteur de référence est associé à la bonne classe, il est rapproché de $\mathbf{x}$, sinon il est éloigné :

$$\text{si} \quad \phi(\mathbf{x}) = \phi(\mathbf{w}_i) \qquad \text{alors} \qquad \Delta\mathbf{w}_i^{t+1} = \alpha(t)(\mathbf{x} - \mathbf{w}_i^t) \tag{4.20}$$

$$\text{si} \quad \phi(\mathbf{x}) \neq \phi(\mathbf{w}_i) \qquad \text{alors} \qquad \Delta\mathbf{w}_i^{t+1} = -\beta(t)(\mathbf{x} - \mathbf{w}_i^t) \tag{4.21}$$

On ne touche pas aux autres vecteurs de référence.

Comme ordres de grandeur, on prend $\alpha(0)$ et $\beta(0)$ de l'ordre de 0,01 que l'on fait décroître vers 0 en 100 000 itérations.

Après convergence, on observe que les frontières de décision sont placées aux mêmes endroits que les placerait un classifieur bayésien.

4.3.2 LVQ2

Dans LVQ2, si la classe associée au vecteur de référence est différente de la classe de $\mathbf{x}$, on ne repousse $\mathbf{w}_i$ que si un autre vecteur de référence associé à la bonne classe se trouve dans le voisinage.

Soit donc $\mathbf{w}_j$ le second vecteur de référence le plus proche. Si celui-ci est associé à la bonne classe et si $\mathbf{x}$ est assez près de la médiatrice de $\mathbf{w}_i$ et $\mathbf{w}_j$ (donc dans une certaine fenêtre), on rapproche $\mathbf{w}_j$ et on éloigne $\mathbf{w}_i$. Dans tous les autres cas, on ne fait rien.

$$\text{si} \quad \phi(\mathbf{x}) \neq \phi(\mathbf{w}_i) \quad \text{et} \quad \phi(\mathbf{x}) = \phi(\mathbf{w}_j) \quad \text{et} \quad (\mathbf{x} - \mathbf{w}_j)^2 < (1+\delta)(\mathbf{x} - \mathbf{w}_i)^2 \tag{4.22}$$

$$\text{alors} \qquad \Delta\mathbf{w}_i = -\beta(\mathbf{x} - \mathbf{w}_i) \tag{4.23}$$

$$\text{et} \qquad \Delta\mathbf{w}_j = \alpha(\mathbf{x} - \mathbf{w}_j) \tag{4.24}$$

δ correspond à 10 % ou 20 % de la distance séparant $\mathbf{w}_i$ de $\mathbf{w}_j$. C'est une grandeur à déterminer expérimentalement et qui dépend du nombre d'exemples car il faut bien qu'il y ait un nombre suffisant d'exemples à tomber dans la fenêtre pour faire de bonnes statistiques.

Un premier problème de LVQ2 est que les corrections effectuées sont proportionnelles à la différence entre $\mathbf{x}$ et $\mathbf{w}_i$ et entre $\mathbf{x}$ et $\mathbf{w}_j$, et que la correction sur $\mathbf{w}_j$ (la bonne classe) est plus grande que la correction sur $\mathbf{w}_i$ (la mauvaise classe), diminuant ainsi la distance $\| \mathbf{w}_i - \mathbf{w}_j \|$. On peut compenser cet effet en acceptant tous les vecteurs de la fenêtre pour lesquels un des vecteurs de référence $\mathbf{w}_i$ ou $\mathbf{w}_j$ appartient à la bonne classe et l'autre non.

Un deuxième problème de LVQ2 est que si les itérations se poursuivent trop longtemps, la frontière de décision dépasse la limite de Bayes ; il est nécessaire de n'applique l'algorithme que pendant un temps court (10 000 itérations par exemple).

Ces considérations nous amènent à LVQ3.

4.3.3 LVQ3

Si $\mathbf{w}_i$ et $\mathbf{w}_i$ sont les deux plus proches vecteurs de référence de $\mathbf{x}$ et que $\mathbf{x}$ et $\mathbf{w}_j$ appartiennent à la même classe et que $\mathbf{x}$ et $\mathbf{w}_i$ appartiennent à deux classes différentes et que $\mathbf{x}$ tombe dans la

fenêtre, alors :

$$\mathbf{w}_j^{t+1} = \mathbf{w}_j^t + \alpha(t)(\mathbf{x} - \mathbf{w}_j^t) \tag{4.25}$$

$$\mathbf{w}_i^{t+1} = \mathbf{w}_i^t - \alpha(t)(\mathbf{x} - \mathbf{w}_i^t) \tag{4.26}$$

$$\tag{4.27}$$

Si $\mathbf{x}$, $\mathbf{w}_i$ et $\mathbf{w}_j$ appartiennent à la même classe :

$$\mathbf{w}_k^{t+1} = \mathbf{w}_k^t + \epsilon\alpha(t)(\mathbf{x} - \mathbf{w}_k^t) \quad k \in \{i, j\} \tag{4.28}$$

La bonne valeur de ϵ semble dépendre de la taille de la fenêtre et être d'autant plus petite que la fenêtre est petite. On prend ϵ entre 0,1 et 0,5.

4.4 Conclusions

Ces méthodes, cartes topologiques, LVQ1, LVQ2 et LVQ3 sont simples et permettent d'obtenir rapidement de bons résultats.

4.5 Notes

Les méthodes de classification ou de typologies ont pour but de regrouper des individus en un certain nombre de classes homogènes. Leur science s'appelle la taxinomie.

-processus de recherche de classes = classification, clustering, partitionnement, catégorisation en IA la traduction correcte du terme anglais classification devrait être discrimination

non supervisées : pas de label a priori

non hiérarchiques : centres mobiles, nuées dynamiques (k-means aux USA) hiérarchiques : CHA (agregation Ward -> indicée)

-processus de classement = analyse discriminante, recherche de séparation entre classes existantes.

Bibliographie

[Fri97] Bernd Fritzke. Some competitive learning methods. Technical report, Systems Biophysics, Institute for Neural Computation, Ruhr-Universität Bochum, 1997. available at `http ://www.neuroinformatik.ruhr-uni-bochum.de/ini/VDM/research/gsn/JavaPaper/`.

[Koh89] T. Kohonen. *Self-Organization and Associative Memory*. Springer-Verlag, Berlin, 3 edition, 1989.

[Koh90] Teuvo Kohonen. The self-organizing map. *Proceedings of the IEEE*, 78(9) :78–87, September 1990.

[LaV89] Anthony LaVigna. *Nonparametric Classification Using Learning Vector Quantization*. PhD thesis, Univ. Maryland, 1989. available at `http ://www.isr.umd.edu/TechReports/ISR/1990/TR_90-8/TR_90-8.phtml`.

[MS94] Thomas Martinez and Klaus Schulten. Topology representing networks. *Neural Networks*, 7(3) :507–522, 1994.