Invățare nesupervizată

- Specificul învățării nesupervizate
- Problematica grupării datelor
- Gruparea datelor cu rețele neuronale
- Cuantizare (discretizare) vectorială
- Mapare topologică și rețele de tip Kohonen

Specificul învățării nesupervizate

Invățare supervizată:

- Setul de antrenare conține atât date de intrare cât și răspunsurile corecte aferente acestora
- Exemplu: clasificarea în clase predefinite pentru care se cunosc exemple de elemente ce aparțin claselor
- Invățarea supervizată este echivalentă cu optimizarea unei funcții de eroare care măsoară diferența dintre răspunsurile pe care ar trebui să le producă rețeaua și cele pe care le produce efectiv

Invățare nesupervizată:

- Setul de antrenare conține doar date de intrare
- Exemplu: gruparea datelor în categorii în funcție de similaritatea dintre ele
- Se bazează pe proprietățile statistice ale datelor
- Nu se bazează pe conceptul de funcție de eroare ci pe cel de calitate a modelului extras din date, care trebuie maximizată prin procesul de antrenare

Problematica grupării datelor

Gruparea datelor (clustering) = identificarea de grupări naturale în cadrul datelor caracterizate prin faptul că

- Datele din aceeași clasă (cluster) sunt suficient de similare
- Datele din clase diferite sunt suficient de disimilare

Obs: nu există o etichetare apriori a datelor (cum există în cazul clasificării supervizate) iar uneori nu este cunoscut apriori nici măcar numărul de clase

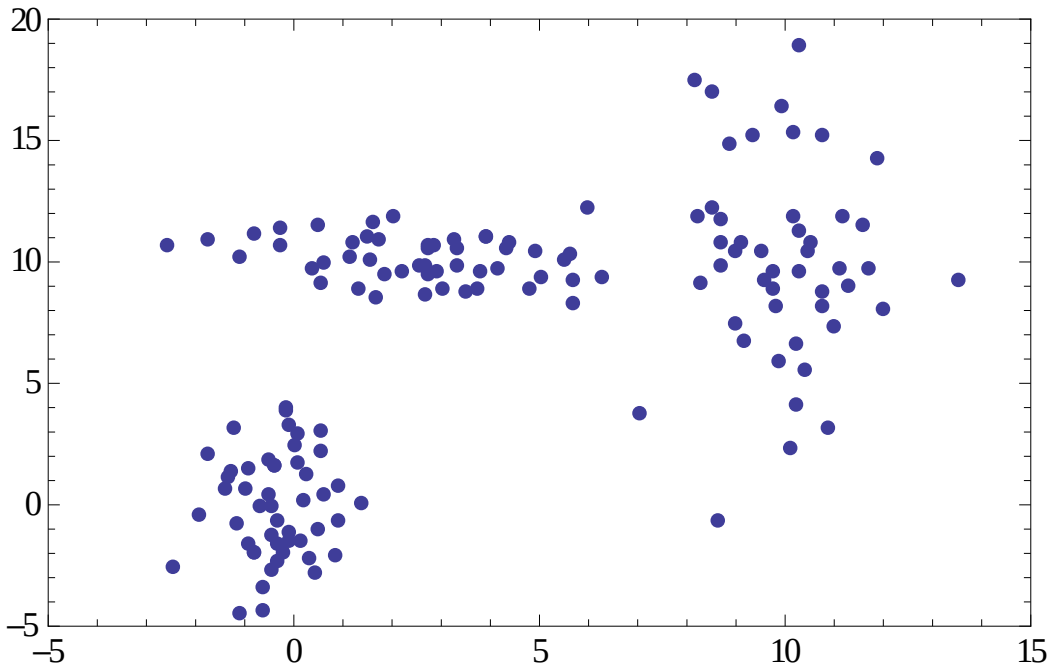
Aplicații:

- Identificarea unor profile de utilizatori ai unor resurse Web (e-commerce, e-learning etc.)
- Identificarea unor regiuni omogene în imagini digitale (segmentarea imaginilor)
- Categorizarea documentelor electronice
- Analiza datelor biologice

Problematica grupării datelor

Exemplu: date bidimensionale generate artificial (folosind trei repartitii normale cu parametri diferiți)

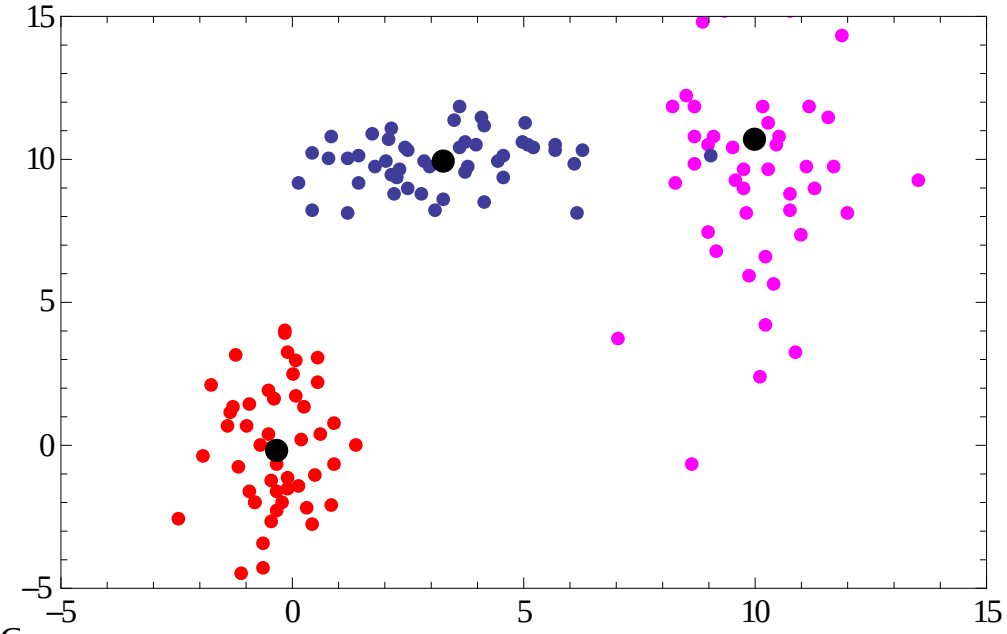
Problemele reale se caracterizează prin faptul ca datele sunt descrise prin multe atribute



Problematica grupării datelor

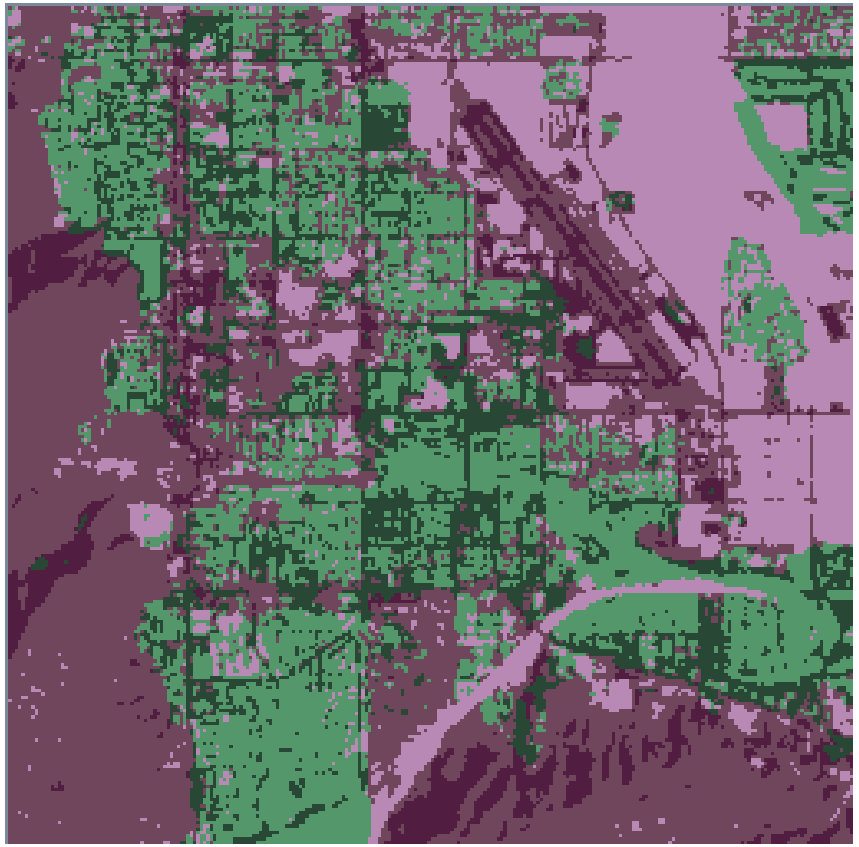
Exemplu: date bidimensionale generate artificial (distribuirea în categorii)

Obs: datele generate conform cu modelul corespunzator unei categorii pot fi în regiunea corespunzatoare celeilalte categorii (ex: punctul albastru din zona datelor mov) => identificarea grupurilor este o problema dificila.



Problematica grupării datelor

Exemplu: segmentarea imaginilor = identificarea regiunilor omogene din imagine



Problematica grupării datelor

Un element esențial în rezolvarea problemelor de grupare îl reprezintă măsura de similaritate/disimilaritate utilizată

In alegerea unei măsuri adecvate trebuie să se țină cont de natura atributelor:

- atribute cantitative -> vectori cu componente numerice
- atribute calitative -> vectori cu componente simbolice
- mixte -> vectori cu componente codificate după mai multe reguli

Problematica gruparii datelor

- Masuri specifice datelor numerice

Masura de similaritate (cos) :

$$S(X,Y) = \frac{X^T Y}{\|X\|\|Y\|}$$

Masura de disimilaritate (distanta Euclidiana) :

$$d(X,Y) = \|X\text{-}Y\|$$

Pentru vectori normalizati :

$$d^2(X,Y) = 2(1 - S(X,Y))$$

Problematica gruparii datelor

Metode de grupare a datelor:

- Partiționale:
  - Conduc la o partiționare a datelor în clase disjuncte sau parțial suprapuse
  - Fiecare clasă este caracterizată prin unul sau mai multe prototipuri (centrii sau centre)

- Ierarhice:
  - Conduc la o ierarhie de partiționări ce poate fi vizualizată sub forma unei structuri arborescente numită dendrogramă

Problematica grupării datelor

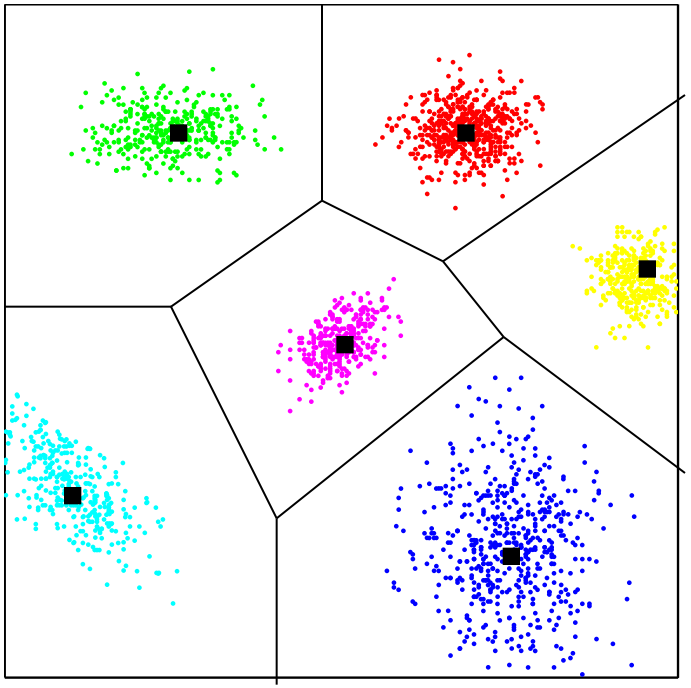
Prototipurile claselor pot fi:

- Media datelor din clasă

(algoritmul K-means)

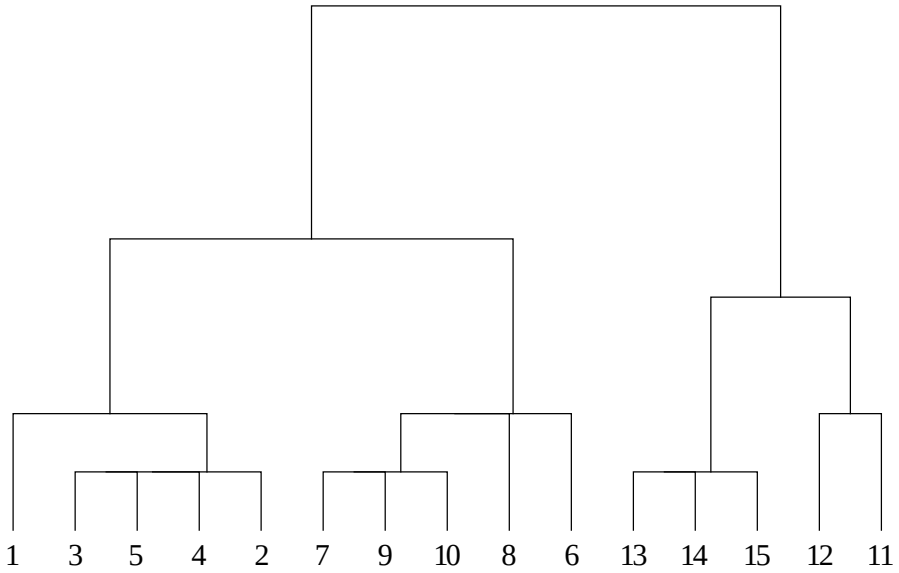
- Mediana datelor din clasă

Asignarea datelor la clase se face pe baza criteriului celui mai apropiat (similar) prototip

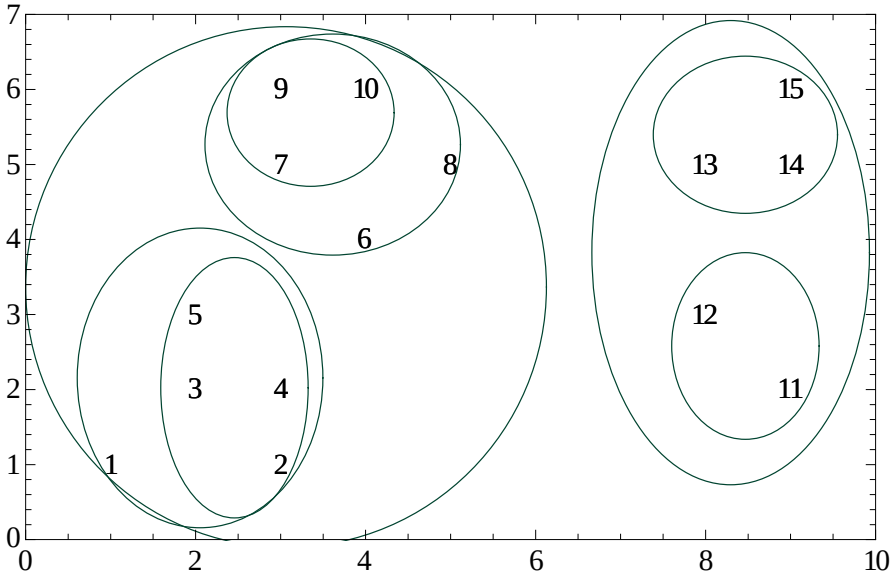


Problematica grupării datelor

Algoritmi ierarhici



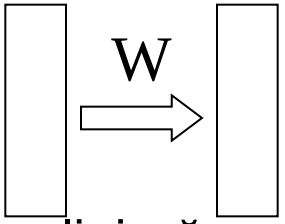
Dendrograma



Partitionarile corespunzătoare dendrogramei

Gruparea datelor cu rețele neuronale

Problema: clasificarea nesupervizata a unor date N-dimensionale în K clase



Arhitectura:

- Un nivel cu N unități de intrare
- Un nivel cu K unități de iesire cu funcție de activare liniară
- Conectivitate totală între cele două nivele (matricea ponderilor, W, conține pe linia k prototipul corespunzător clasei k)

Funcționare:

Pt o dată de intrare X se determină distanțele față de toate prototipurile (liniile matricii W) și se identifică prototipul cel mai apropiat; acesta va indica clasa căreia îi aparține X

Unitatea având vectorul de ponderi cel mai apropiat de X este denumită unitate învingătoare

Gruparea datelor cu rețele neuronale

Antrenare:

Set de antrenare: $\{X^1, X^2, \ldots, X^L\}$

Clase de algoritmi:

- numărul claselor (unităților de ieșire) este cunoscut
  - Algoritm de tipul WTA – " the Winner Takes All"
- numărul claselor (unităților de ieșire) este necunoscut
  - Algoritm de tipul ART – "Adaptive Resonance Theory"

Specificul algoritmilor:

- se ajustează doar ponderile unităților învingătoare
- rata de învățare este descrescătoare (pentru a induce stabilizarea valorilor ponderilor)

Gruparea datelor cu rețele neuronale

Exemplu: algoritm de tip WTA

Etapa 1. Inițializarea prototipurilor.

$w_{ij} = \text{rand}(-1, 1), \quad i = \overline{1, K}, \quad j = \overline{1, N}$

$t = 1$

Etapa 2. Proces iterativ de ajustare a prototipurilor.

REPEAT

FOR $l = \overline{1, L}$ DO

Determină k^* cu proprietatea $d(X^l, W^{k^*}) \leq d(X^l, W^i)$, $i = \overline{1, K}$

$W^{k^*} = W^{k^*} + \eta(t)(X^l - W^{k^*})$

$t = t + 1$

UNTIL $(\eta(t) < \epsilon)$ OR $(t > t_{max})$.

Gruparea datelor cu rețele neuronale

Observații:

- Rata de invățare descrescătoare satisfacând următoarele proprietăți

$$\lim_{t\to\infty}\eta(t)=0,\quad \sum_{t=1}^{\infty}\eta(t)=\infty,\quad \sum_{t=1}^{\infty}\eta^2(t)<\infty$$

… dar nici foarte lent

Exemple:

$\eta(t)=\eta(0)/t^{\alpha},\ \alpha\in[0.5,1]$

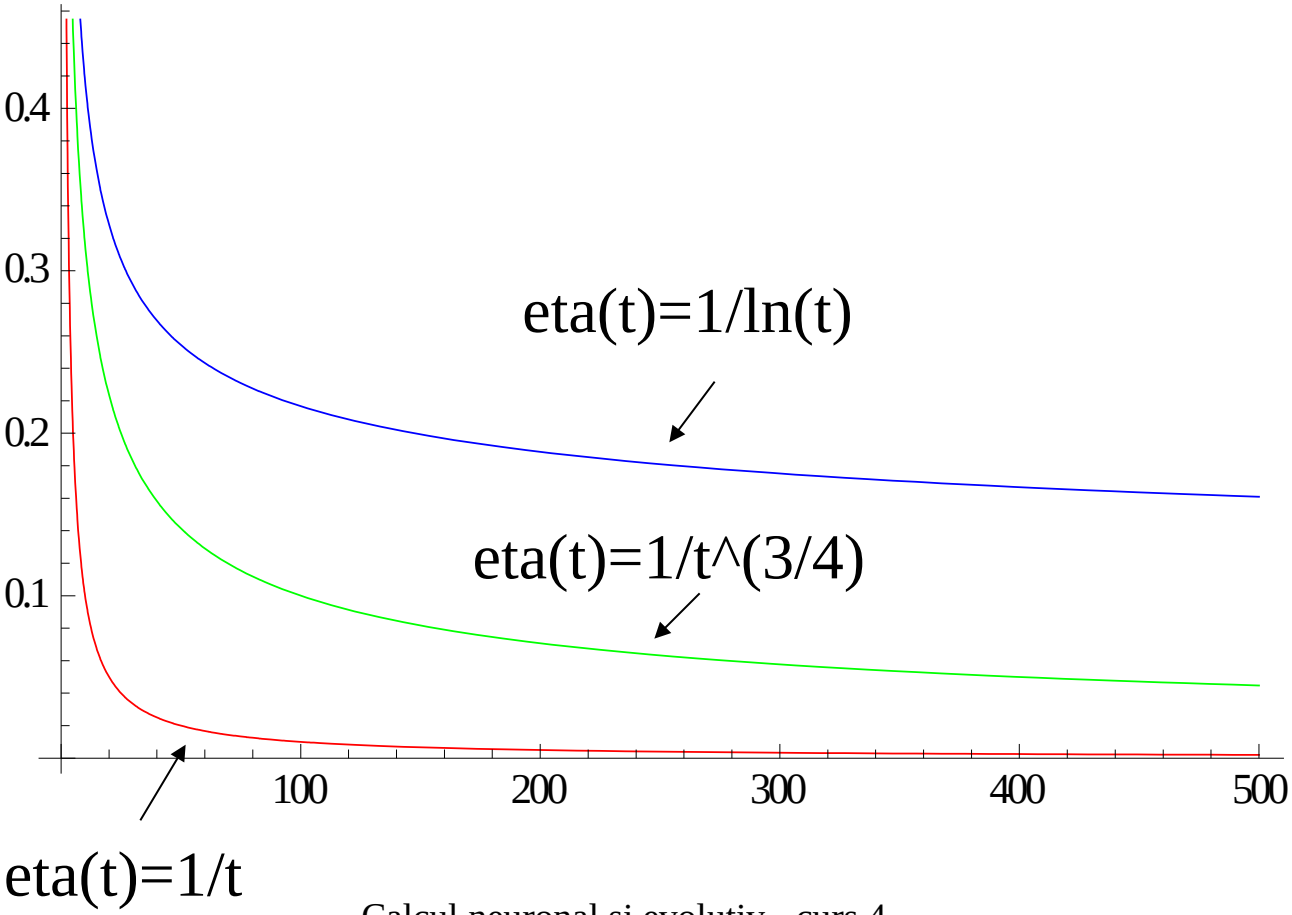
… nu foarte rapid

$\eta(t)=\eta(0)/\ln(t+1)$

tinde catre zero

Gruparea datelor cu rețele neuronale

Exemple de rata de invățare descrescătoare



Gruparea datelor cu rețele neuronale

Observații:

- Problema unităților inactive (moarte): unități care nu devin niciodată învingătoare

Cauze: inițializarea inadecvată a ponderilor

Solutii:

- Inițializarea prototipurilor cu vectori din setul de antrenare
- Ajustarea nu doar a ponderilor corespunzatoare unității învingătoare ci și a celorlalte (însă cu o rată de învățare mai mică)
- Penalizarea unităților care ies prea frecvent învingătoare

Gruparea datelor cu rețele neuronale

- Penalizarea unităților care ies prea frecvent învingătoare: se modifică criteriul de stabilire a unității învingătoare

$$d(X, W^{k*}) - \theta_{k*} \leq d(X, W^{k}) - \theta_{k}$$

θ_k – prag care este invers proportional cu frecventa situatiilor in care unitatea este invingatoare

Gruparea datelor cu rețele neuronale

Este util ca atât datele de intrare cât și ponderile să fie normalizate:

Variante:

- Normalizarea datelor din setul de antrenare se face înainte de inițierea antrenării
- Normalizarea ponderilor se face pe parcursul antrenării:

$$W^{k^*} = \frac{W^{k^*} + \eta(t)(X^l - W^{k^*})}{\|W^{k^*} + \eta(t)(X^l - W^{k^*})\|}.$$

Gruparea datelor cu rețele neuronale

Adaptive Resonance Theory: oferă soluții pentru următoarele două probleme ce apar la proiectarea sistemelor de clasificare nesupervizată:

- Adaptabilitate
  - Capacitatea sistemului de a asimila noi date și de a identifica noi clustere (impune număr variabil de clustere)

- Stabilitate
  - Capacitatea sistemului de a conserva structura de clustere astfel încât pe parcursul procesului de adaptare sistemul să nu își schimbe radical răspunsul pentru o anumită dată de intrare

Gruparea datelor cu rețele neuronale

Exemplu: algoritmul ART2 (pt date din R^n)

Inițializări:

Se stabileşte numărul iniţial de unităţi funcţionale, $K < L$.

Se iniţializează vectorii ponderilor cu elemente distincte selectate aleator din setul de antrenare.

$t = 1$

REPEAT

FOR $l = \overline{1, L}$ DO

se determină $k^* \in \{1, \ldots, K\}$ cu $d(X^l, W^{k^*}) = \min_k d(X^l, W^k)$;

IF $d(X^l, W^{k^*}) < \rho$ OR $K = K_{max}$ THEN $W^{k^*} = (X^l + W^{k^*} \text{card}(\omega_{k^*}))/(1 + \text{card}(\omega_{k^*}))$

($\text{card}(\omega_{k^*})$ reprezintă numărul de elemente din clasa k^*)

ELSE se adaugă o nouă unitate ascunsă: $K = K + 1$ şi $W^{K+1} = X^l$

Incrementează indicatorul de iteraţie, $t = t + 1$

UNTIL $t > t_{max}$

Gruparea datelor cu rețele neuronale

Observatii:

- Parametrul rho este denumit prag de vigilenta; valoarea lui influențează numărul de clase identificate:
  - Cu cât valoarea este mai mică cu atât crește numărul de clase
  - Cu cât valoarea este mai mare cu atât scade numărul de clase
- Prezintă dezavantajul că prototipurile identificate depind de ordinea în care sunt prezentate datele
- Diferă de algoritmul incremental de determinare a centrelor retelelor RBF doar prin forma relației de ajustare (nu utilizează rata de învățare ci calculează prototipul ca fiind media datelor asignate lui)
- Există și variante specifice datelor binare (alg ART1)

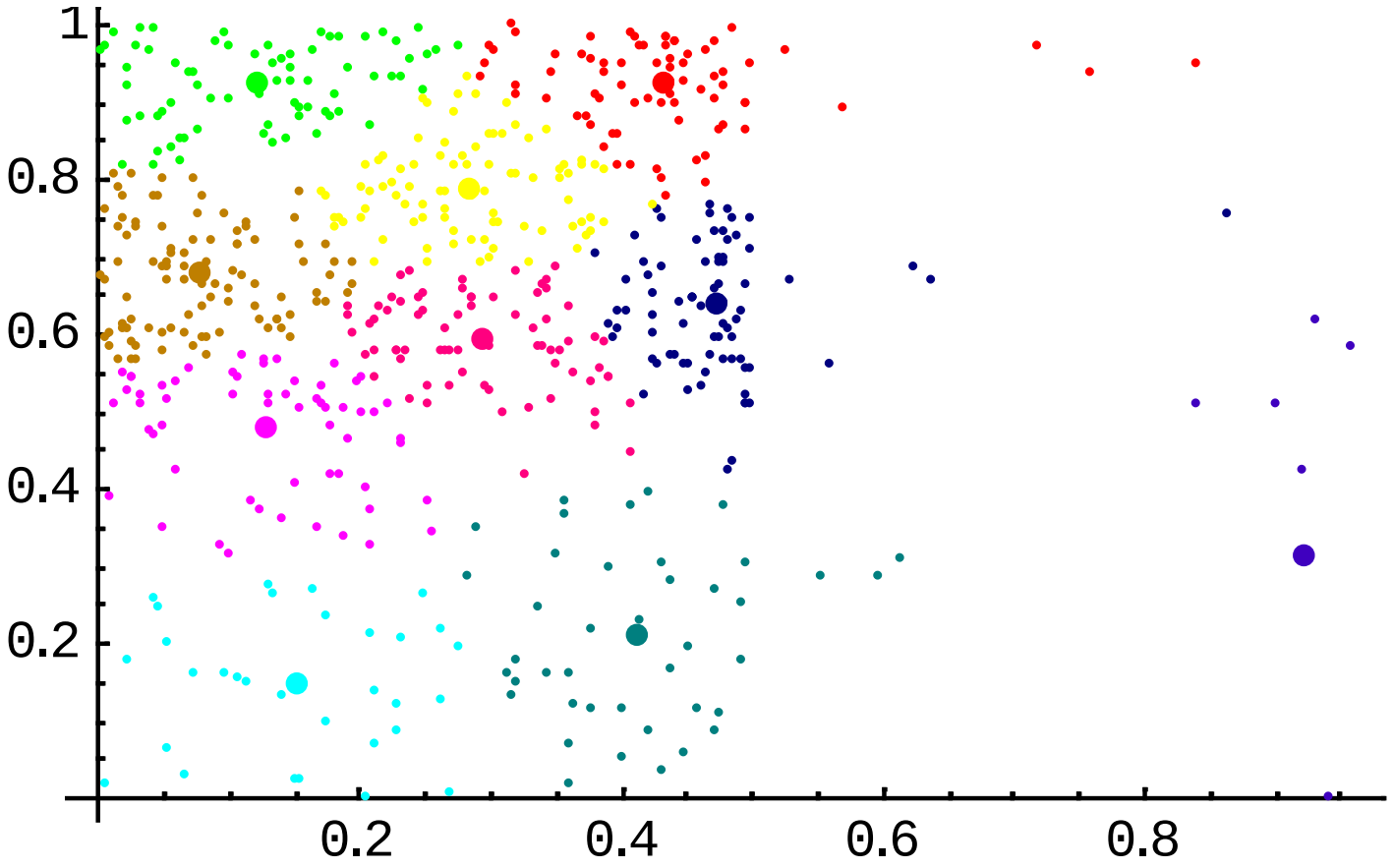
Cuantizare vectorială

Scop cuantizare vectoriala (Vectorial Quantization)

- Maparea unui domeniu din R^N intr-o multime finită de prototipuri
- Permite partiționarea unui domeniu vectorial într-un număr finit de regiuni astfel incât fiecare regiune să fie identificată printr-un prototip
- Cuantizarea permite înlocuirea unui vector cu indicele prototipului regiunii din care face parte, conducând la o metodă simplă de compresie cu pierdere de informație. Scopul urmărit este minimizarea informației pierdute.
- Numărul prototipurilor este mare în zonele in care datele sunt dense și mic in zonele cu densitate redusa

Cuantizare vectorială

Exemplu



Cuantizare vectorială

- Daca numărul de regiuni este cunoscut atunci se poate folosi un algoritm simplu de tip WTA

- Există și varianta supervizată a cuantizării vectoriale (LVQ - Learning Vector Quantization)

  Set de antrenare: $\{(X_1,d_1),\ldots,(X_L,d_L)\}$

Algoritmul LVQ:

1. Inițializare prototipuri aplicând un algoritm de tip WTA asupra setului $\{X_1,\ldots,X_L\}$
2. Identificarea claselor pe criteriul distantei minime si stabilirea indicatorului de clasa folosind etichetele $d_1,\ldots d_L$: pentru fiecare clasa se alege eticheta cea mai frecventă

Cuantizare vectorială

- Se ajustează iterativ prototipurile într-o manieră similară celei de la algoritmul de antrenare al perceptronului. La fiecare epocă de ajustare se efectuează:

FOR $l := 1, L$ DO

determina k* cu : $d(X^l, W^{k*}) \leq d(X^l, W^k)$

IF $c(k*) = d_l$ THEN $W^{k*} := W^{k*} + \eta(t)(X^l - W^{k*})$

ELSE $W^{k*} := W^{k*} - \eta(t)(X^l - W^{k*})$

ENDIF

ENDFOR

Invățare nesupervizată

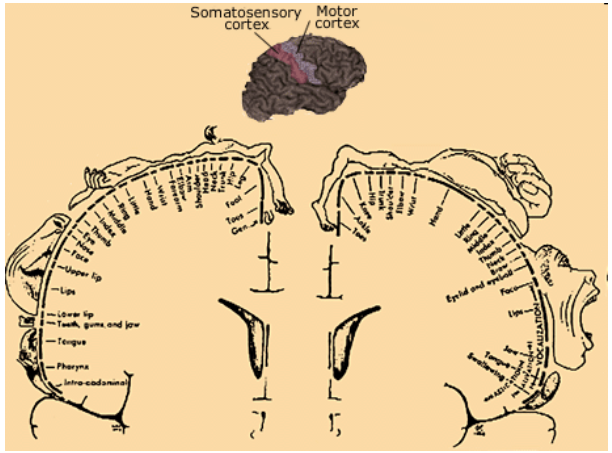
- Mapare topologică
- Rețele Kohonen (hărți cu auto-organizare)
- Analiza componentelor principale
- Algoritmi de învățare de tip Oja, Sanger

Mapare topologica

- Este o variantă de cuantizare vectoriala care asigură conservarea relațiilor de vecinătate din domeniul datelor de intrare:
  - Date de intrare similare vor face parte fie din aceeași clasă fie din clase vecine
  - Apare necesitatea definirii unei relații de ordine între prototipuri prin urmare și între unitățile de ieșire ale rețelei
  - Arhitectura rețelelor care realizează mapare topologică se caracterizează prin existența unei structuri geometrice a nivelului de ieșire corespunzătoare unei grile uni, bi sau tridimensionale
  - Rețelele cu astfel de arhitectură sunt cunoscute sub numele de rețele Kohonen sau hărți cu auto-organizare (self-organizing maps)

Harti cu auto-organizare

- SOM – Self Organizing Maps
- Au fost dezvoltate inițial pentru a modela așa-numitele hărți corticale (porțiuni din scoarța cerebrală care reacționează la anumiți stimuli):
  - Hărti topografice corespunzătoare simțului vizual
  - Hărți tonotopice corespunzătoare simțului auditiv
  - Hărți senzoriale asociate cu simțul tactil (întreagă suprafață a pielii este mapată într-o regiune astfel încât receptori tactili apropiați activează neuroni vecini ai hărții)



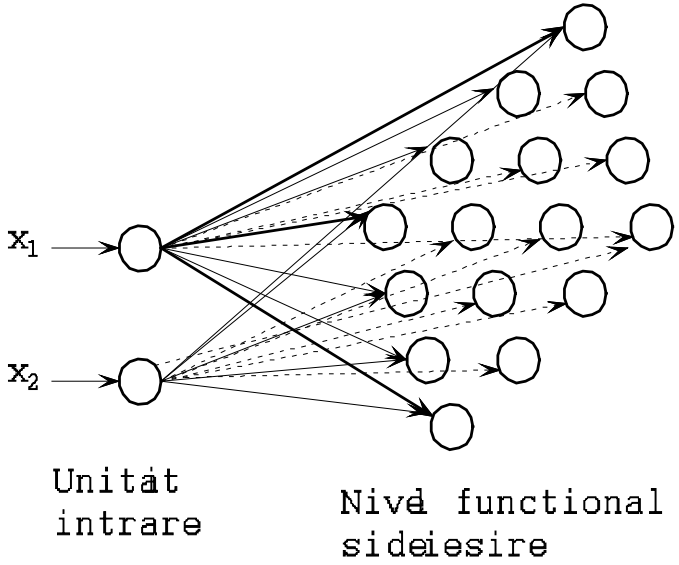
Hartă senzorială

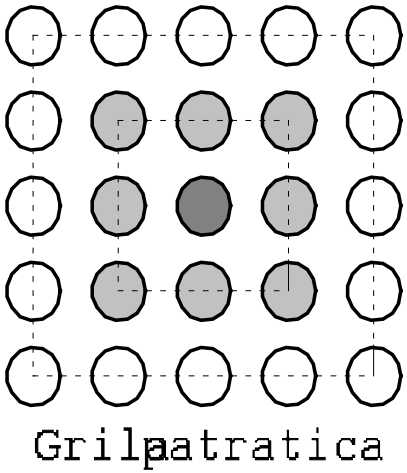
Hărți cu auto-organizare

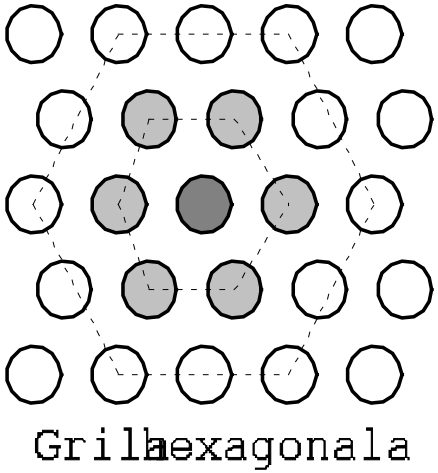
- Aplicațiile curente ale hărților cu auto-organizare cuprind:
  - Gruparea datelor
  - Compresia datelor (maparea datelor dintr-un spațiu de dimensiune mai mare într-unul de dimensiune mai mică astfel incât să fie conservate relațiile de vecinătate)
- Aplicații specifice: http://www.cis.hut.fi/research/som-research/
  - Explorarea colecțiilor de documente electronice
  - Căutare după conținut în baze de date cu imagini

Rețele Kohonen

- Arhitectura:
  - Un nivel de unități de intrare
  - Un nivel de unități funcționale plasate în nodurile unei grile (permite definirea unor distanțe între unități și a unui sistem de vecinătăți)







- Tipuri de grile:

Dpdv al dimensiunii:
- Unidimensionale
- Bidimensionale
- Tridimensionale

Dpdv al structurii:
- Pătratice
- Hexagonale
- Arbitrare (graf planar)

Rețele Kohonen

- Structura de vecinătăți asociată nivelului de unități de ieșire:
  - Fiecărei unități funcționale (p) i se asociază un vector de poziție (r_p)
  - In cazul grilelor cu n dimensiuni acest vector conține n componente
  - Se alege o distanță pe domeniul vectorilor de poziție

$$d_1(r_p, r_q) = \sqrt{\sum_{i=1}^{n} (r_p^i - r_q^i)^2} \quad \text{(distanta euclidiana)}$$

$$d_2(r_p, r_q) = \sum_{i=1}^{n} | r_p^i - r_q^i | \quad \text{(distanta Manhattan)}$$

$$d_3(r_p, r_q) = \max_{i=\overline{1,n}} | r_p^i - r_q^i |$$

Rețele Kohonen

- Vecinătatea de ordin (raza) s a unității p este:

$$V_s(p) = \{q \mid d(r_p, r_q) \leq s\}$$

- Exemplu: In cazul unei grile bidimensionale vecinătățile de ordin 1 ale unității având poziția (i,j) corespunzătoare celor trei distanțe sunt

$$V_1^{(1)}(i, j) = V_1^{(2)}(i, j) = \{(i-1, j), (i+1, j), (i, j-1), (i, j+1)\}$$

$$V_1^{(3)}(i, j) = \{(i-1, j), (i+1, j), (i, j-1), (i, j+1), (i-1, j-1),$$
$$(i-1, j+1), (i+1, j-1), (i+1, j+1)\}$$

Rețele Kohonen

- Funcționare:
  - Pentru un vector de intrare, X, se determină unitatea funcțională învingătoare folosind criteriul distanței minime față de vectorii cu ponderi
  - Rezultatul poate fi vectorul de poziție al unității învingătoare sau vectorul cu ponderi asociat acesteia
- Antrenare:
  - De tip nesupervizat
  - Set de antrenare: $\{X^1,\ldots,X^L\}$
  - Specific: similară cu antrenarea de tip WTA însă o dată cu ajustarea ponderilor unității învingătoare se ajustează și ponderile unităților vecine

Rețele Kohonen

- Algoritm de antrenare

Initializare $W, t, \eta(t), s(t)$

Repeat

For $l := 1, L$ do

determina p^* a.i. $d(X^l, W^{p^*}) \leq d(X^l, W^p)$, pt orice p

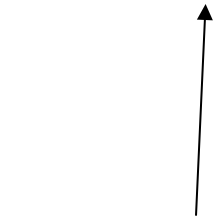
$W^p := W^p + \eta(t)(X^l - W^p)$, pt. $p \in N_{s(t)}(p^*)$

Endfor

t := t +1

calculeaza $s(t), \eta(t)$

Until $t > t_{max}$



Vecinatate de
Raza s(t) a unitatii p*

Rețele Kohonen

- Algoritm de antrenare

  – Ajustarea unităților vecine celei învingătoare asigură conservarea relațiilor de vecinătate astfel că date de intrare similare vor fi asociate cu unități învecinate

  – Atât rata de învățare cât și dimensiunea vecinătății descresc în timp

  – Maniera de descreștere a ratei de învățare este similară celei de la algoritmii de tip WTA

  – Dimensiunea vecinătății se alege inițial suficient de mare pentru a “acoperi” întregul set de unități funcționale

Rețele Kohonen

- Algoritmul de antrenare se caracterizează prin două faze

  – Faza de ordonare: corespunde iterațiilor în care dimensiunea vecinătății este semnificativă și are ca efect stabilirea ponderilor unităților astfel încât unor date de intrare similare să le fie asociate unități vecine

  – Faza de ajustare fină: corespunde iterațiilor în care dimensiunea vecinătății este mică (chiar redusa la un singur element) și are ca rol ajustarea fină a ponderilor pentru ca vectorii de ponderi să devină prototipuri cât mai reprezentative pentru datele de intrare

Obs: pentru a diferenția modul de ajustare a ponderilor unității învingătoare față de cel al ponderilor celorlalte unități se folosește conceptul de funcție de vecinătate

Rețele Kohonen

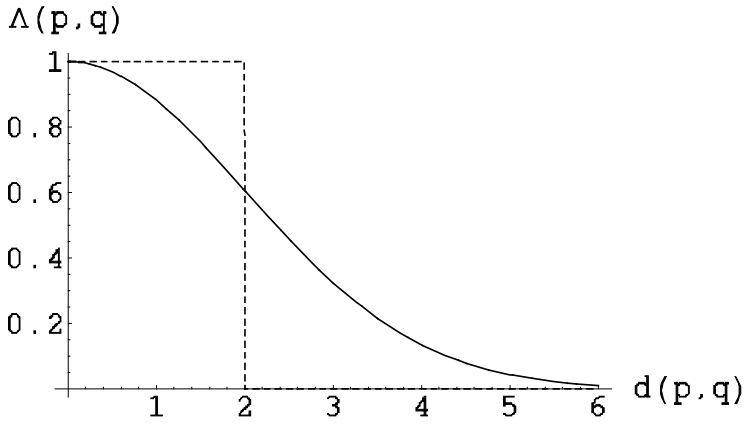
- Utilizarea unei funcții de vecinătate:

$$W^p = W^p + \eta(t)\Lambda(p^*, p)(X - W^p).$$

- Exemple de funcții de vecinătate:

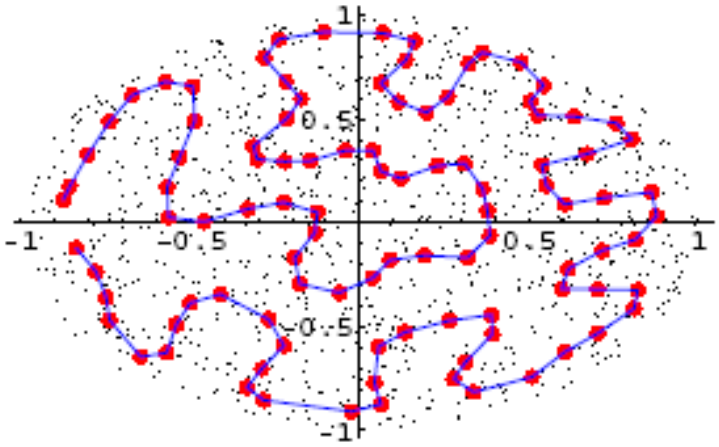
$$\Lambda(p,q) = \begin{cases} 1 & \text{dacă } q \in V_s(p) \\ 0 & \text{altfel} \end{cases}$$

$$\Lambda(p,q) = \exp\left(-\frac{d^2(r_p, r_q)}{2s^2}\right)$$

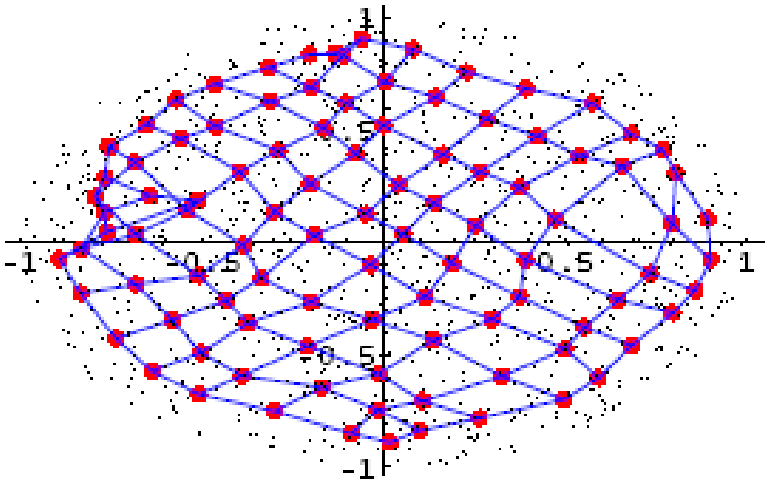


Rețele Kohonen

- Ilustrarea mapării topologice (date de intrare bidimensionale generate uniform aleator in interiorul unui cerc)
  - Se vizualizează punctele corespunzatoare vectorilor de ponderi ale tuturor unităților
  - Se unesc între ele punctele corespunzătoare unor unități vecine (in funcție de structura nivelului de unități funcționale fiecare punct este unit cu două sau mai multe alte puncte



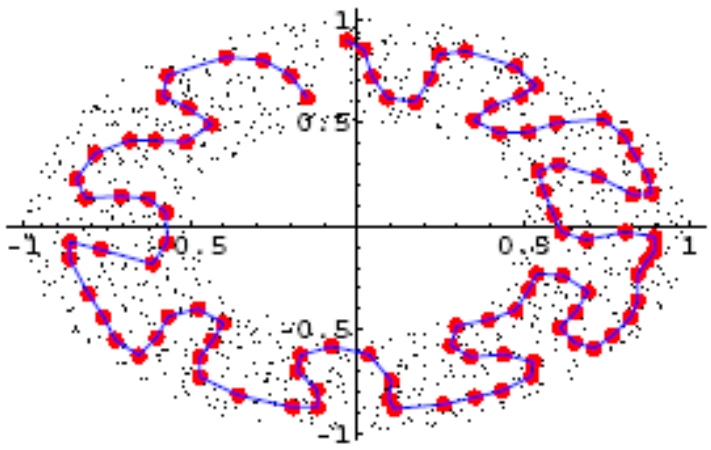
Grila 1D

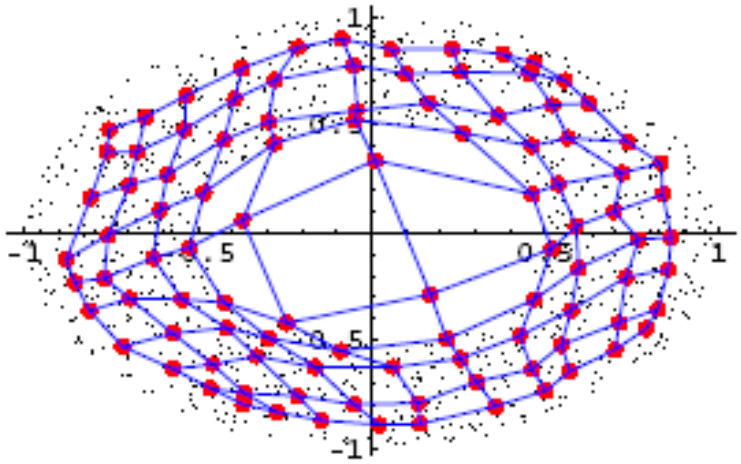


Grila 2D

Rețele Kohonen

- Ilustrarea mapării topologice
  - date de intrare bidimensionale generate uniform aleator în interiorul unei coroane circulare



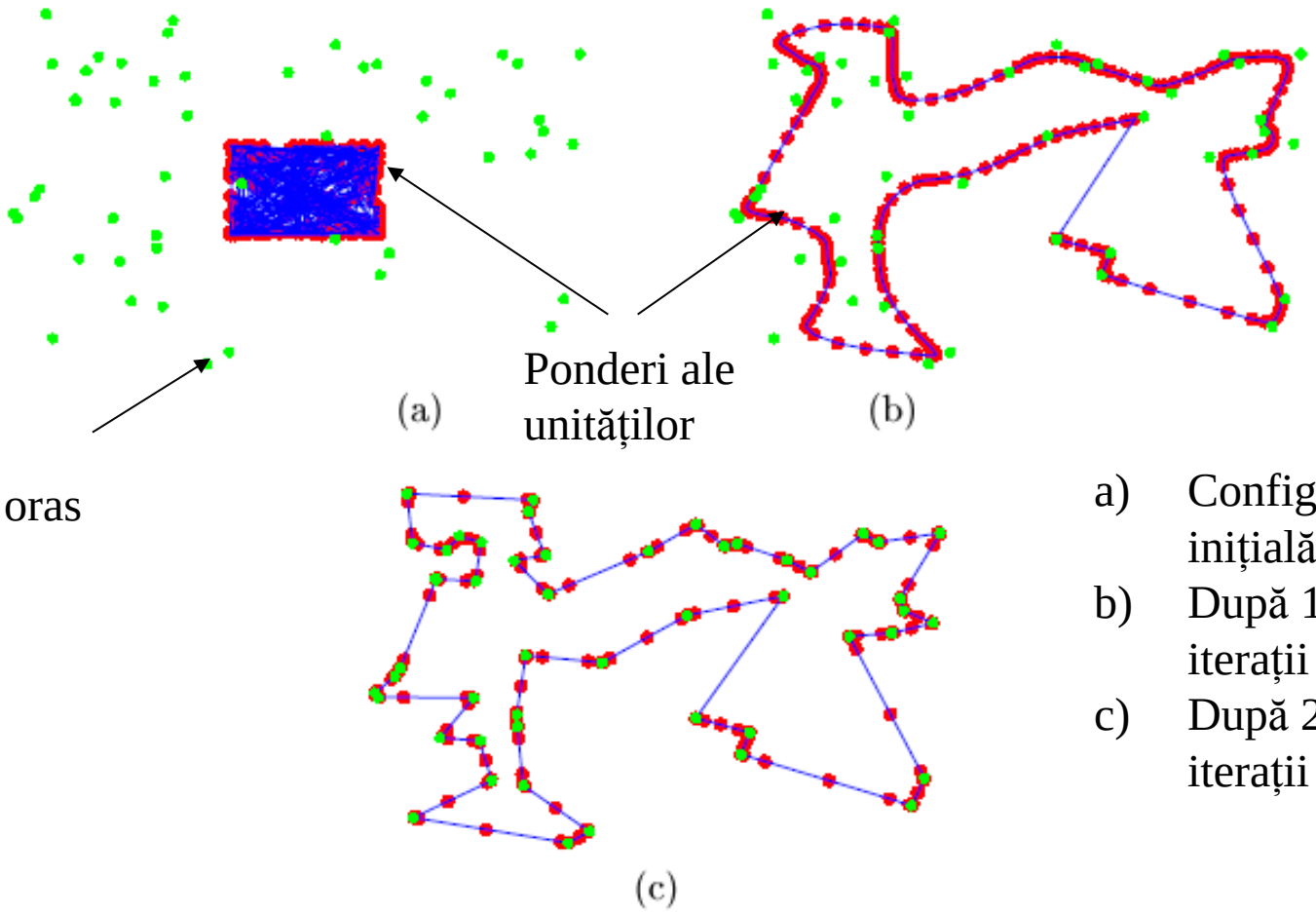


Rețele Kohonen

- Problema comis voiajorului:
  - determinarea unui traseu de cost minim care asigură vizitarea o singură a dată a fiecărui oraș (circuit hamiltonian de cost minim într-un graf complet etichetat cu n noduri)
  - se folosește o rețea cu două unități de intrare și n unități funcționale dispuse într-o grila unidimensională circulară (unitatea n este considerată vecină cu unitatea 1) – rețea elastică
  - datele de intrare vor fi coordonatele orașelor
  - Pe parcursul antrenării, vectorii cu ponderi ale unităților tind să se apropie de coordonatele orașelor iar relația de vecinătate dintre unități va indica ordinea de parcurgere a orașelor
  - Intrucât mai multe unități pot să tindă către același oraș, numărul unităților trebuie să fie mai mare (de două, trei ori) decât numărul orașelor

Rețele Kohonen

- Problema comis voiajorului:



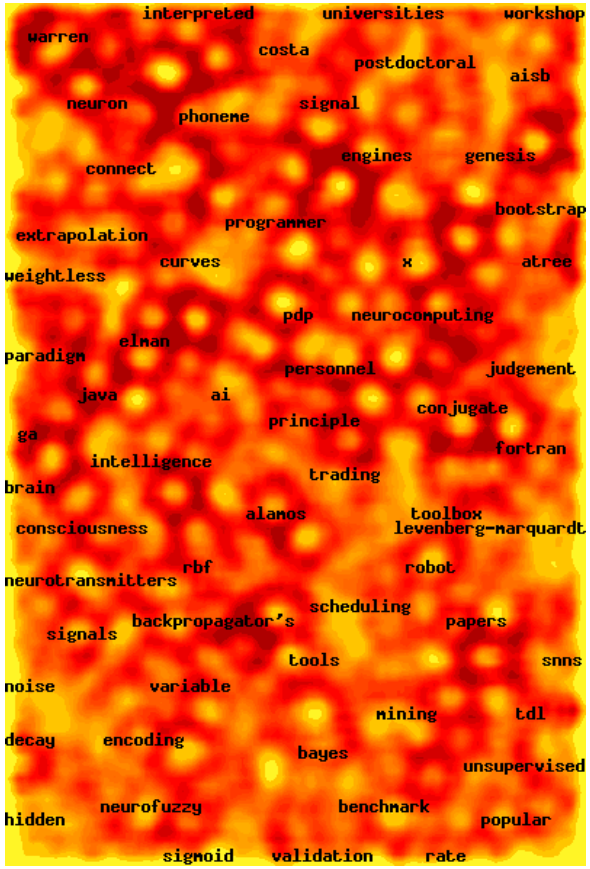
a) Configurația inițială
b) După 1000 iterații
c) După 2000 iterații

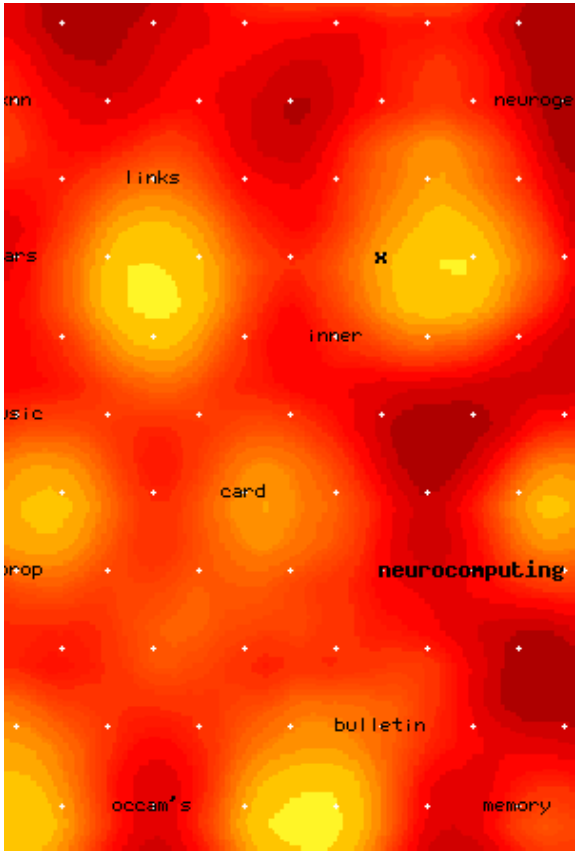
Rețele Kohonen

- Alte aplicatii:
  - Controlul robotilor autonomi: robotul este antrenat cu date de intrare corespunzatoare regiunilor din zona de deplasare in care nu sunt obstacole (robotul va invata “harta” zonei)
  - Categorizarea documentelor electronice: WebSOM

Rețele Kohonen

- WebSOM (http://websom.hut.fi/websom/)





Analiza componentelor principale

- Scop: reducerea dimensiunii datelor vectoriale cu păstrarea a cât mai mult din informația pe care o poartă

- Utilitate: reducerea dimensiunii datelor în vederea efectuării unor prelucrări (clasificare, grupare); în datele reale pot exista atribute cu varianță mică (fără suficientă putere de discriminare deci nerelevante din punctul de vedere al unei clasificări)

- Principiu: se realizează o transformare (liniară) a datelor care asigură reducerea dimensiunii de la N la M componente (M<N)

$$Y=WX$$

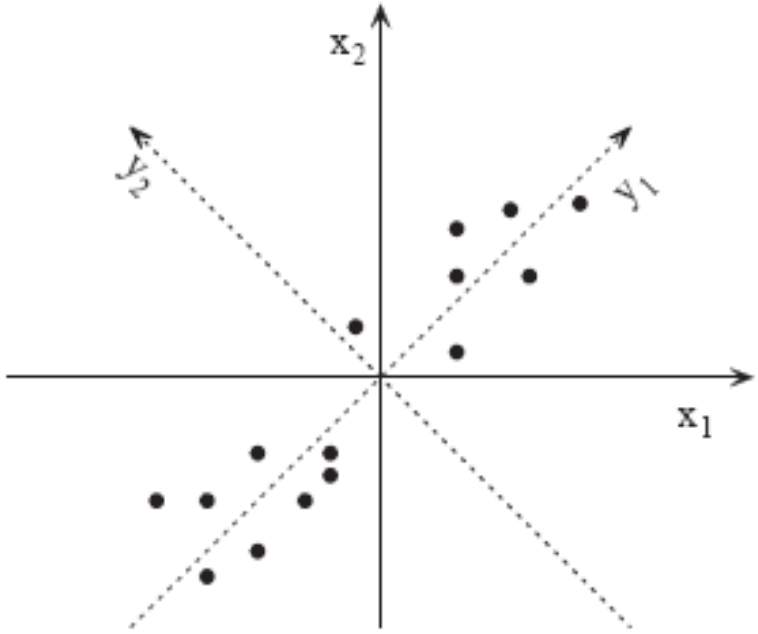
X = vector cu N componente
Y = vector cu M componente
W = matrice cu M linii și N coloane

Analiza componentelor principale

- Ilustrare: N=2, M=1

Sistemul de axe de coordonate x_1Ox_2 este transformat in sistemul y_1Oy

Oy_1 - este direcția de-a lungul căreia se înregistrează cea mai mare variabilitate a datelor: se poate reține doar componenta y_1



Analiza componentelor principale

Formalizare:

- Datele de intrare pot fi interpretate ca realizări ale unui vector aleator N-dimensional cu o anumită distribuție (se poate considera că media vectorului aleator este 0)

- Se caută o pereche de transformări

$T:R^N \to R^M$ si $S:R^M \to R^N$

$X \xrightarrow{T} Y \xrightarrow{S} X'$

care au proprietatea ca vectorul reconstruit $X'=S(T(X))$ este cât mai apropiat de X (eroarea la reconstruire este cât mai mica)

Analiza componentelor principale

Formalizare: se poate demonstra că matricea W (M linii și N coloane) care asigură eroarea minimă la reconstruire conține pe linii vectorii proprii ai matricii de covarianță asociată distribuției datelor de intrare (ce corespund celor mai mari M valori proprii)

Vector aleator de medie nula :

$$X = (X_1,...,X_N),\ E(X_i) = 0$$

Matricea de covarianta : C(X)

$$c_{ij} = E((X_i - E(X_i))(X_j - E(X_j)) = E(X_i X_j)$$

$C(X)$ este matrice simetrica si (semi)pozitiv definita $\Rightarrow$

toate valorile sale proprii sunt reale si pozitive

$$\lambda_1 \geq \lambda_2 \geq ... \geq \lambda_M ... \geq \lambda_N \geq 0$$

Analiza componentelor principale

Construirea matricii de transformare T (metoda statistică):

- Se determină matricea de covarianță asociată distribuției datelor de intrare
  - Analitic (în cazul în care distribuția este cunoscută)
  - Prin estimare pe baza datelor de intrare (se folosește covarianța de selecție)
- Se determină valorile și vectorii proprii ai matricii C
  - Dacă nu se pot determina exact se pot folosi metode numerice de aproximare
- Se ordoneaza descrescător valorile proprii ale matricii C și se rețin vectorii proprii corespunzători celor mai mari M valori proprii

Analiza componentelor principale

Dezavantaje ale metodei statistice:

- Are un cost ridicat (determinat de necesitatea construirii matricii de covarianță și de calculul valorilor și vectorilor proprii) în special în cazul în care N este mare

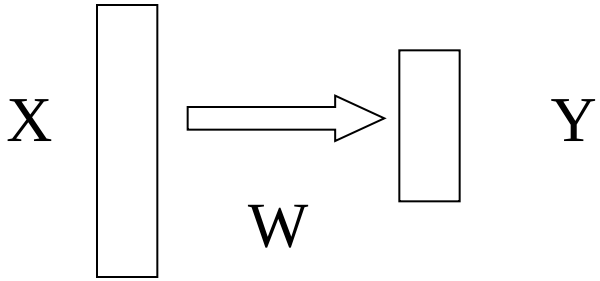
- Nu are caracter incremental
  - Daca matricea de covarianță este estimată pe baza datelor de intrare, apariția unor noi date poate să o modifice, ceea ce conduce la necesitatea recalculării valorilor și vectorilor proprii

- Alternativa: utilizarea unei rețele neuronale cu arhitectură simplă și algoritm incremental de învățare

Rețele neuronale pt. ACP

Arhitectura:

- N unități de intrare
- M unități de ieșire (liniare)
- Conectivitate totala între nivele



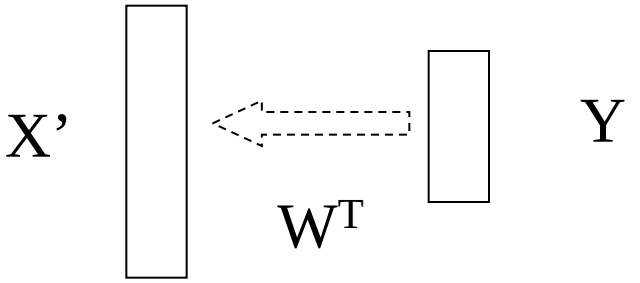
Funcționare:

- Extragerea componentelor principale

$$Y=WX$$

- Reconstruirea datelor inițiale

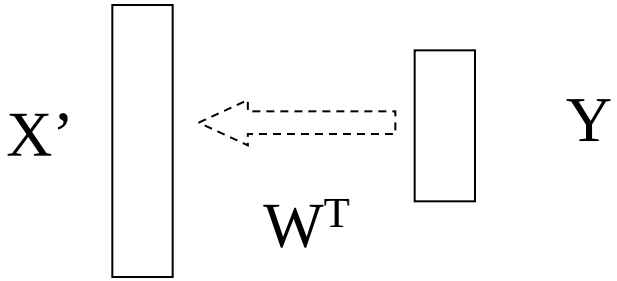
$$X'=W^TY$$



Retele neuronale pt. ACP

Antrenare:

- Nesupervizată
- Set: {X1,X2,…} (antrenarea este incrementala, rețeaua se ajustează pe măsură ce primește noi date de intrare)



Scopul antrenării: reducerea erorii la reconstruire (diferența dintre X și X’ trebuie minimizată)

- Poate fi interpretată ca problema antrenării autosupervizate a rețelei utilizate pentru reconstruirea semnalului de intrare

Rețele neuronale pt. ACP

Antrenare autosupervizata:

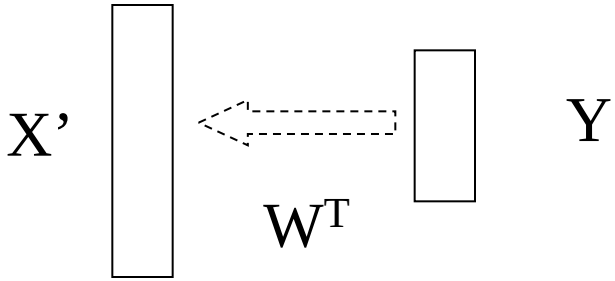
Set de antrenare:

$\{(X^1,X^1), (X^2,X^2),....\}$

Eroare pătratică la reconstruire (pentru un exemplu):

$$E(W) = \frac{1}{2}\sum_{j=1}^{N}(x_j - x'_j)^2 =$$

$$= \frac{1}{2}\sum_{j=1}^{N}(x_j - \sum_{i=1}^{M} w_{ij} y_i)^2$$



Aplicand ideea de la algoritmul Widrow-Hoff se ajunge la relatia de ajustare:

$$w_{ij} := w_{ij} + \eta \cdot (x_j - x')y_i$$

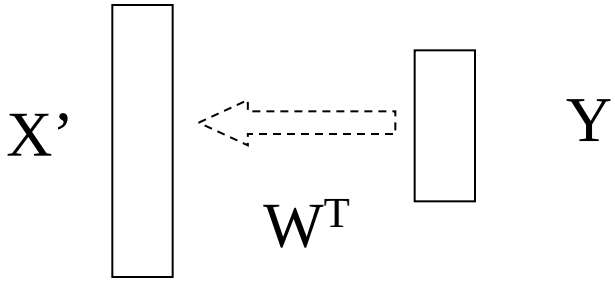
$$w_{ij} := w_{ij} + \eta \cdot (x_j - \sum_{k=1}^{M} w_{kj} y_k)y_i$$

Rețele neuronale pt. ACP

Algoritmul Oja:

Set de antrenare:

$\{(X^1,X^1), (X^2,X^2),....\}$



Initializari : $w_{ij} := rand(-1,1);\ t := 1$

REPEAT // pt. fiecare exemplu X

$$y_i = \sum_{j=1}^{N} w_{ij} x_j,\ \ i = 1..M$$

$$w_{ij} := w_{ij} + \eta \cdot (x_j - \sum_{k=1}^{M} w_{kj} y_k) y_i,\ i = 1..M,\ j = 1..N$$

$$t := t + 1$$

UNTIL $t > t_{max}$

Obs: liniile matricii W tind către vectorii proprii ai matricii de covarianță corespunzători celor mai mari M valori proprii însă nu există corespondență directă între numărul unității și numărul de ordine al valorii proprii

Rețele neuronale pt. ACP

Algoritmul Sanger:

Este o variantă a algoritmului Oja care asigură faptul că linia i a matricii W tinde către vectorul propriu corespunzător celei de a i-a valori proprii (in ordine descrescătoare) a matricii de covarianță

Initializari : $w_{ij} := rand(-1,1);\ t := 1$

REPEAT // pt. fiecare exemplu X

$$y_i = \sum_{j=1}^{N} w_{ij} x_j,\ \ i = 1..M$$

Specific algoritm Sanger

$$w_{ij} := w_{ij} + \eta \cdot (x_j - \sum_{k=1}^{i} w_{kj} y_k) y_i,\ i = 1..M,\ j = 1..N$$

$t := t + 1$

UNTIL $t > t_{\max}$