Cartes auto-organisatrices de Kohonen

Description et classification automatique

Ricco RAKOTOMALALA
Université Lumière Lyon 2

PLAN

1. Cartes de Kohonen (SOM) – Principe
2. Algorithme d'apprentissage
3. Visualisations des données
4. Déploiement – Traitement d'un individu supplémentaire
5. Logiciels – Exemple d'analyse (R, Tanagra)
6. Classification (clustering) à partir de SOM
7. Extension à l'apprentissage supervisé
8. Bilan
9. Bibliographie

Les cartes de Kohonen

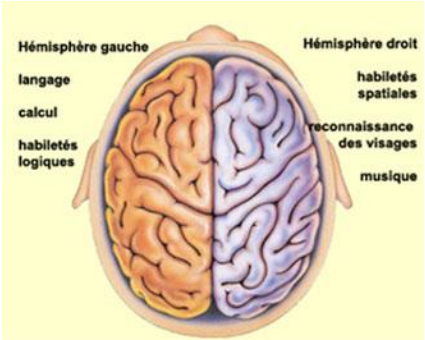
Self-Organizing Maps (**Kohonen, 1984**)

Carte auto-organisatrice

Carte auto-adaptative, carte topologique – Self-organizing map (SOM)

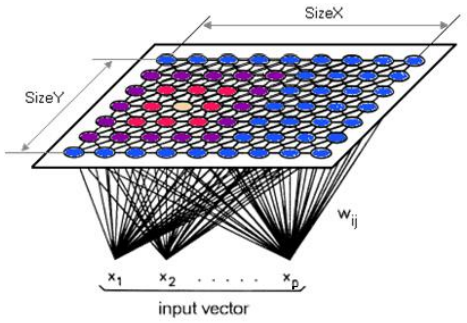
Métaphore biologique

Notre cerveau est subdivisé en zones spécialisées, elles répondent spécifiquement à certains stimuli c.-à-d. des stimuli de même nature activent une région du cerveau particulière.



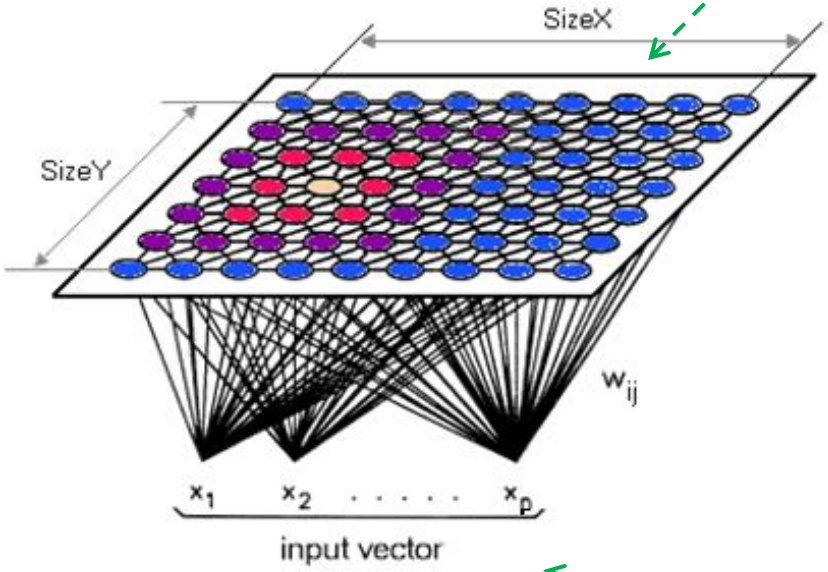
Carte de Kohonen

Transposer l'idée à un système d'apprentissage non-supervisé compétitif où l'espace d'entrée est « mappée » dans un espace réduit (souvent rectangulaire) avec un principe fort : des individus similaires dans l'espace initial seront projetés dans le même neurone ou tout du moins dans deux neurones proches dans l'espace de sortie (préservation des proximités).



Sert à la fois pour la réduction de la dimensionnalité, la visualisation et la classification automatique (clustering, apprentissage non supervisé).

SOM - Architecture



• A chaque neurone (**nœud**) va correspondre un ensemble d'observations des données initiales.

• A chaque neurone est associé un vecteur de poids à p valeurs (**codebook,** « profil type » du neurone).

• Les positions des neurones dans la carte sont importantes c.-à-d. (1) deux neurones voisins présentent des codebook similaires ; (2) un ensemble de neurones contigus correspondent à un profil particulier dans les données.

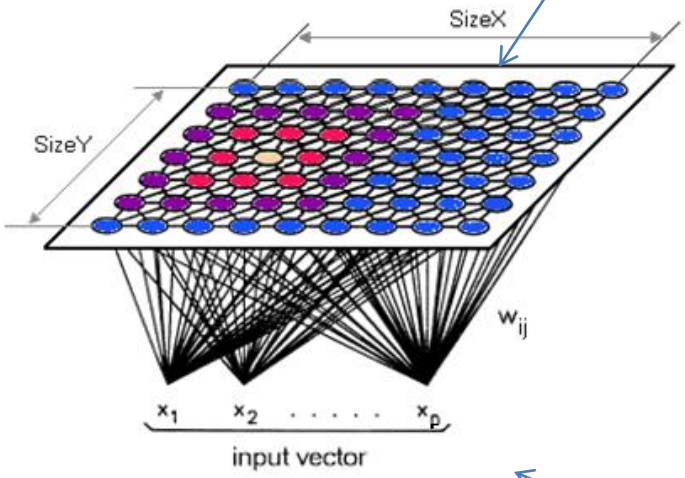
Les liens entre les couches d'entrée et de sorties indiquent les correspondances entre les vecteurs d'entrées et de sortie.

Espace d'entrée, description des données dans l'espace initial à p variables (vecteur à p valeurs).

SOM – Un exemple (1)

Modele	CYL	PUISS	LONG	POIDS	VMAX	RPOIDPUIS
Toyota Corolla	1166	55	399	815	140	14.82
Lada-1300	1294	68	404	955	140	14.04
Citroen GS Club	1222	59	412	930	151	15.76
Renault 16 TL	1565	55	424	1010	140	18.36
Moyenne	**1311.8**	**59.3**	**409.8**	**927.5**	**142.8**	**15.7**

Un neurone de « petits » véhicules (4), peu rapides, peu puissantes avec un rapport poids-puissance élevé (peu sportives).



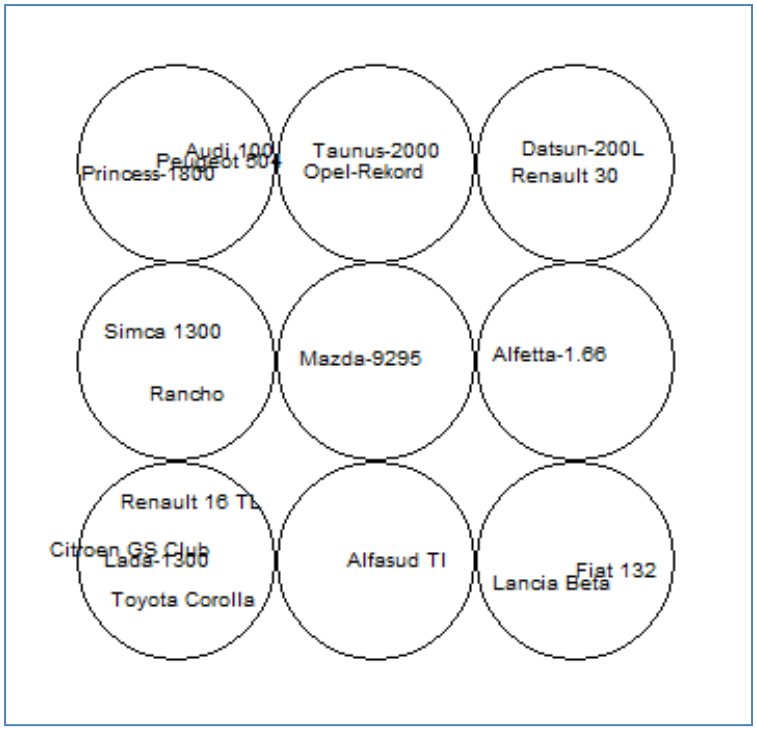
Modele	CYL	PUISS	LONG	POIDS	VMAX	RPOIDPUIS
Alfasud TI	1350	79	393	870	165	11.01
Audi 100	1588	85	468	1110	160	13.06
Simca 1300	1294	68	424	1050	152	15.44
Citroen GS Club	1222	59	412	930	151	15.76
Fiat 132	1585	98	439	1105	165	11.28
Lancia Beta	1297	82	429	1080	160	13.17
Peugeot 504	1796	79	449	1160	154	14.68
Renault 16 TL	1565	55	424	1010	140	18.36
Renault 30	2664	128	452	1320	180	10.31
Toyota Corolla	1166	55	399	815	140	14.82
Alfetta-1.66	1570	109	428	1060	175	9.72
Princess-1800	1798	82	445	1160	158	14.15
Datsun-200L	1998	115	469	1370	160	11.91
Taunus-2000	1993	98	438	1080	167	11.02
Rancho	1442	80	431	1129	144	14.11
Mazda-9295	1769	83	440	1095	165	13.19
Opel-Rekord	1979	100	459	1120	173	11.20
Lada-1300	1294	68	404	955	140	14.04
Moyenne	**1631.7**	**84.6**	**433.5**	**1078.8**	**158.3**	**13.2**

Base des « voitures ».

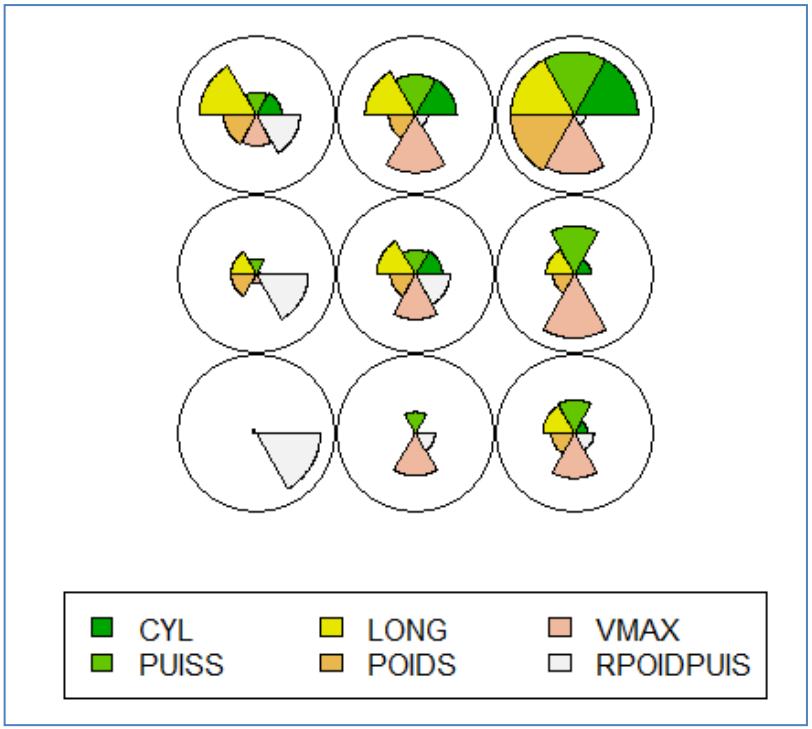
p = 6 variables.

SOM – Un exemple (2)

Une carte **rectangulaire** avec (3 x 3) neurones



Mapping plot

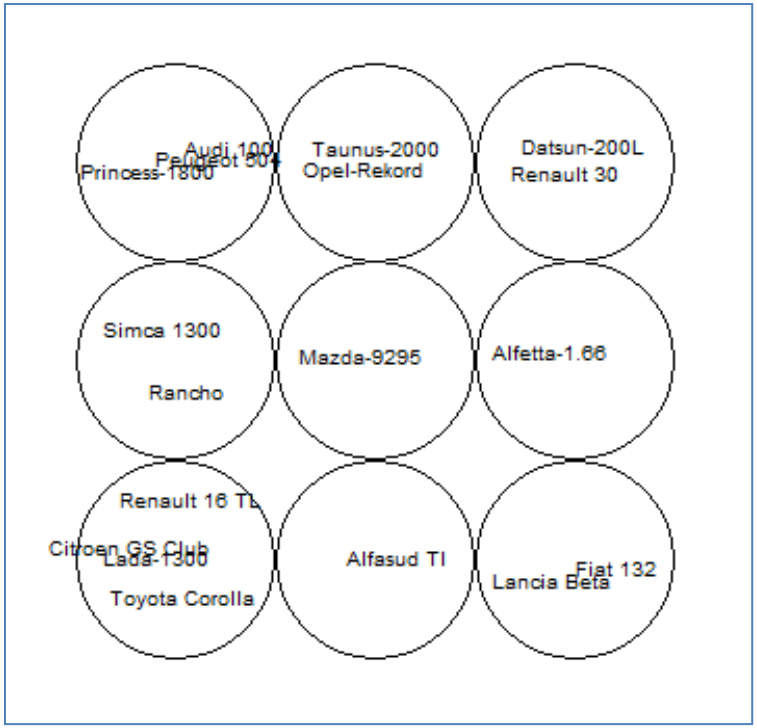


Codebooks plot

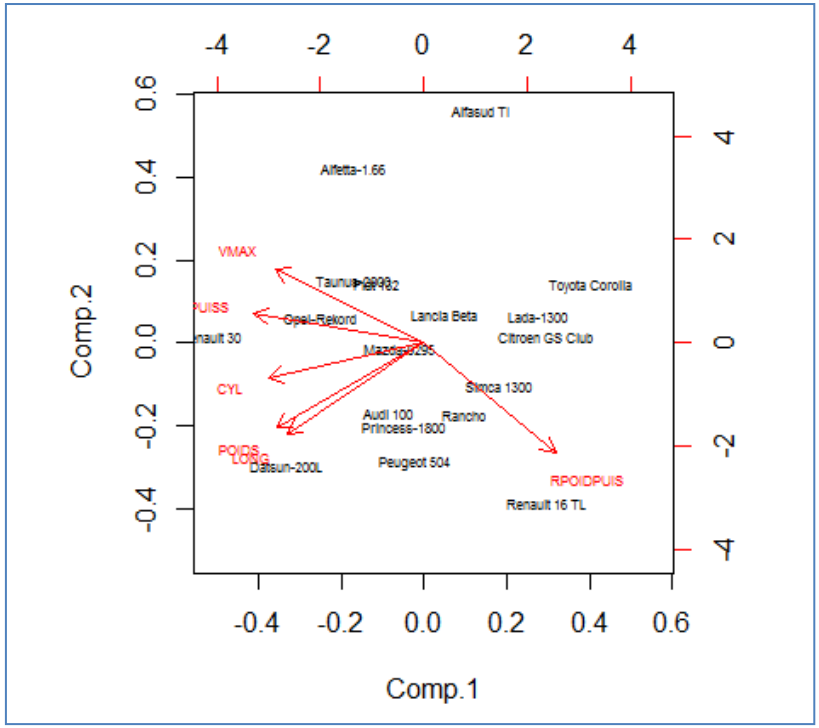
Nous disposons à la fois d'un outil de visualisation (les proximités entre les neurones ont un sens) et de typologie (il y a au moins un premier pre-clustering).

SOM et ACP (Analyse en composantes principales)

L'ACP est une technique de visualisation et de réduction de la dimensionnalité très populaire.



Mapping plot



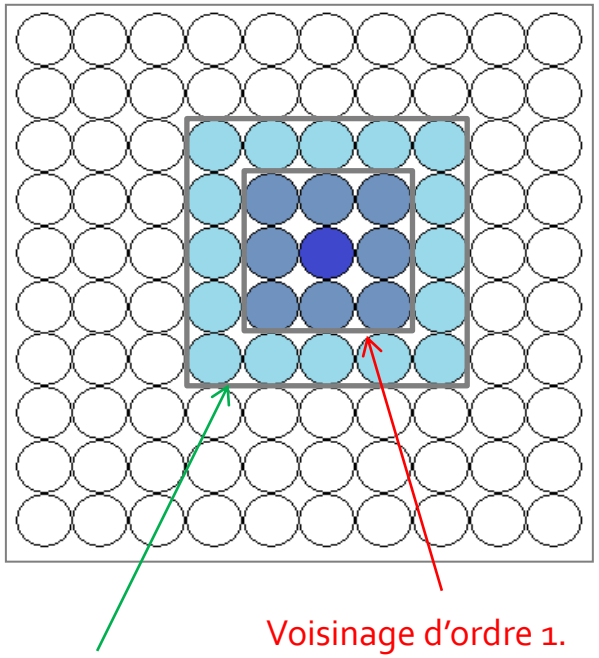
Biplot

On voit à peu près les mêmes proximités. Mais il y a une contrainte de linéarité dans l'ACP (les composantes sont des combinaisons linéaires des variables initiales) qui n'existe pas dans SOM. Cette contrainte, ainsi que l'orthogonalité entre les axes, peut être un handicap dans le traitement des problèmes non linéaires (cf. exemple sur la page en anglais de Wikipédia). SOM est forcément en 2D (très souvent), ACP non.

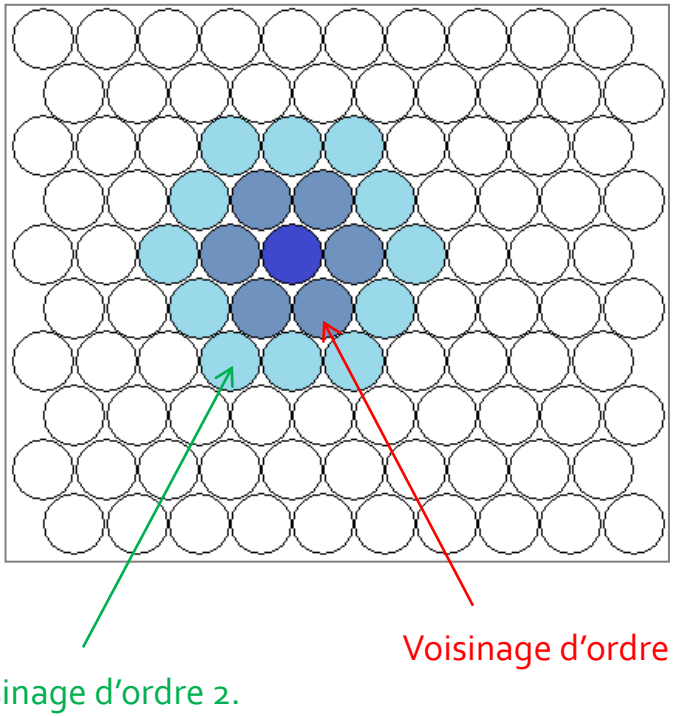
SOM
Architecture et notion de voisinage

La notion de voisinage est primordiale dans SOM, notamment pour la mise à jour des poids et leurs propagations durant le processus d'apprentissage.

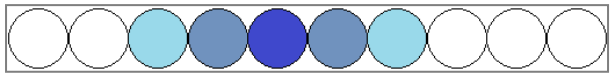
Carte rectangulaire – Voisinage rectangulaire



Carte hexagonale – Voisinage circulaire



Remarque : une carte unidimensionnelle (vecteur) est possible



SOM – Algorithme d'apprentissage

Initialisation, compétition, coopération, adaptation

SOM - Algorithme

Entrée : Réduire les données pour éviter les problèmes d'échelle.

Entrée : un tableau de données, taille et forme de la carte
Sortie : carte topologique avec les poids

1. Initialisation aléatoire des poids de nœuds
2. Pour l'ensemble de la base
   1. Sélectionner un individu aléatoirement
   2. Chercher le nœud qui lui est le plus proche (nœud gagnant)
   3. Les poids de ce nœud est mis à jour
   4. Les poids des nœuds voisins également, mais dans une moindre mesure
3. Répéter 2 pour t = 1 à Tmax itérations, en réduisant au fur et à mesure la taille du voisinage et l'amplitude de la correction

(1) Phase d'initialisation

(2) Il faut faire passer tous les individus de la base. Un individu peut repasser plusieurs fois au fil des itérations (epochs).

(3) Phase de compétition. Il faut définir une mesure de distance entre le codebook des nœuds et la description des individus.

(4) Mise à jour des nœuds. Apprentissage.

(5) Phase de coopération. C'est ce qui assure la similitude de profils entre nœuds contigus. La taille du voisinage à considérer va être réduit au fur et à mesure.
Remarque : si on ne tient pas compte du voisinage, on a l'algorithme des k-means.

(6) Adaptation. Au début, aller vite vers la solution ; à la fin, éviter les oscillations pour mieux converger.

SOM – Détails de l'algorithme (1)

σ_0, ε_0 et Tmax sont des paramètres de l'algorithme

Entrée : un tableau de données, taille et forme de la carte
Sortie : carte topologique avec les poids

1. Initialisation aléatoire des poids de nœuds
2. Pour l'ensemble de la base
   a. Sélectionner un individu aléatoirement
   b. Chercher le nœud qui lui est le plus proche (nœud gagnant)
   c. Les poids de ce nœud est mis à jour
   d. Les poids des nœuds voisins également, mais dans une moindre mesure
3. Répéter 2 pour Tmax itérations (epochs) en réduisant graduellement la taille du voisinage « h », de même pour l'intensité de la mise à jour « ε ».

Règle de mise à jour des poids pour un nœud j, sachant que j* est le nœud gagnant

$$w_{t+1}(j) = w_t(j) + \varepsilon_t \times h_t(j, j^*) \times \left(w_t(j) - x\right)$$

(a) h() est une fonction de voisinage. Son amplitude diminue au fil des itérations

$$h_t(j, j^*) = \exp\left(-\frac{d^2(j, j^*)}{2\sigma^2(t)}\right)$$

Où

- d(j,j*) est la distance entre les nœuds j et j* sur la carte
- $\sigma(t) = \sigma_0 \exp\left(-\frac{t}{\text{Tmax}}\right)$

(b) ε est le pas d'apprentissage. Sa valeur diminue au fil des itérations

$$\varepsilon_t = \varepsilon_0 \exp\left(-\frac{t}{\text{Tmax}}\right)$$

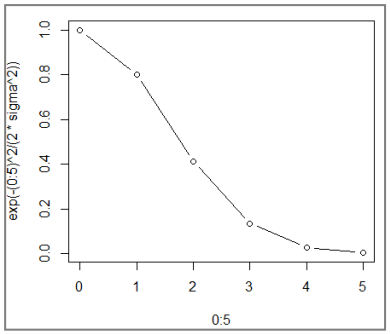
Les implémentations diffèrent d'un logiciel à l'autre, mais les idées directrices sont là.
Réduction graduelle : de la taille du voisinage à considérer, du pas d'apprentissage.

SOM – Détails de l'algorithme (2)

$\sigma_0 = 1.5$, Tmax = 20

Rôle de l'amplitude du voisinage d(j,j*) = 0, …, 5 (**t = 0**)

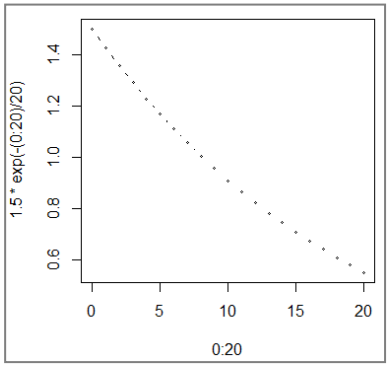
$$h_0(j, j^*) = \exp\left(-\frac{d^2(j, j^*)}{2\sigma^2(0)}\right)$$



Voisins d'ordre 1 : poids mis à jour avec un facteur 0.8 ; d'ordre 2 : 0.41 ; etc.

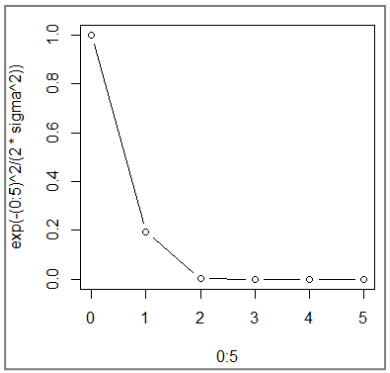
Décroissement de la prise en compte du voisinage au fil du temps (t = 0, …, 20)

$$\sigma(t) = \sigma_0 \exp\left(-\frac{t}{\text{Tmax}}\right)$$



Rôle de l'amplitude du voisinage d(j,j*) = 0, …, 5 (**t = 20**)

$$h_{20}(j, j^*) = \exp\left(-\frac{d^2(j, j^*)}{2\sigma^2(20)}\right)$$

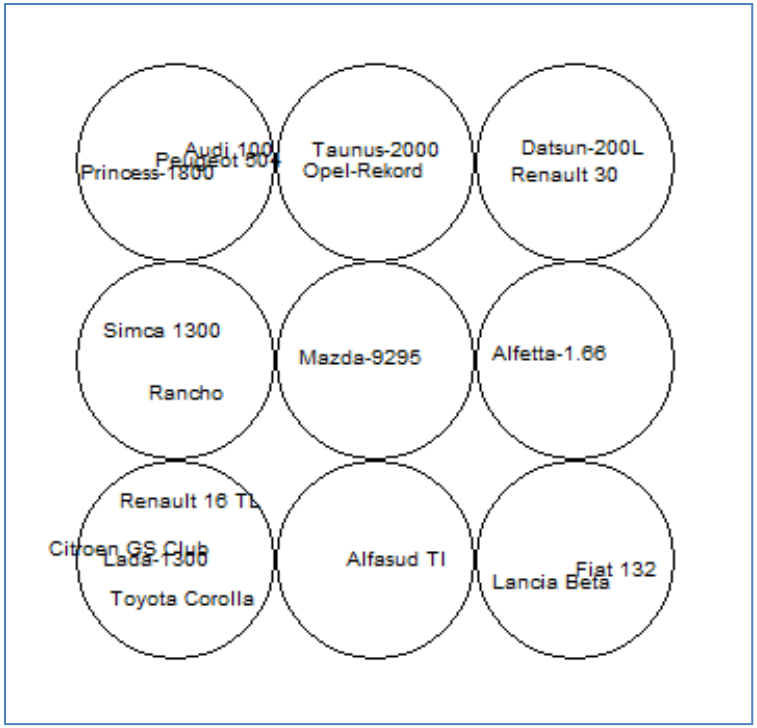


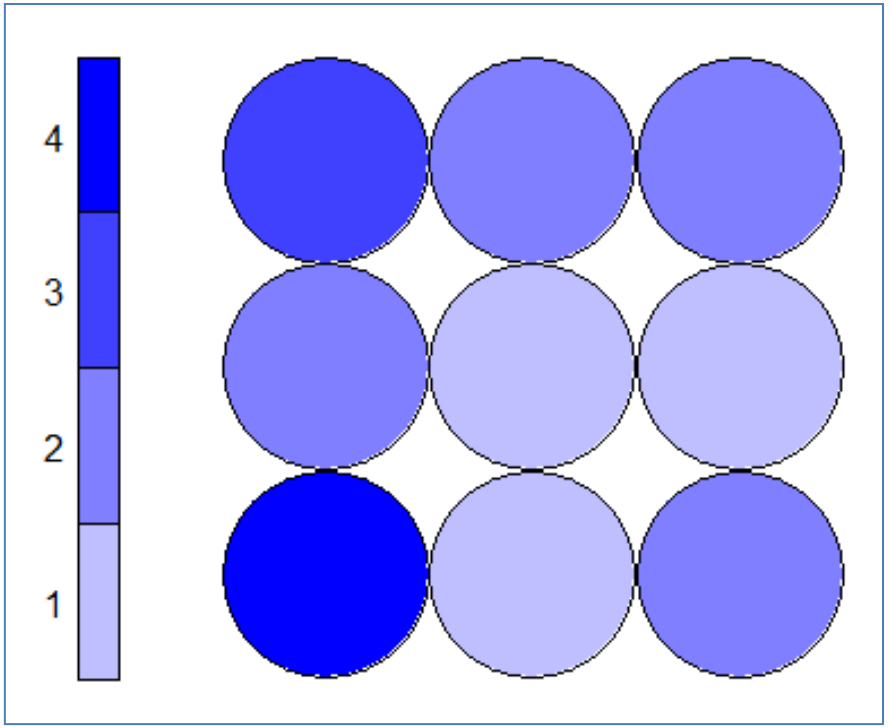
Le rayonnement sur les voisins s'amenuise au fil des itérations. Pour t = 20, seul le voisinage d'ordre 1 est impacté par la mise à jour des poids.

Visualisations

SOM propose des scénarios de visualisation des données très intéressantes

Visualisation – Effectifs, liste des individus



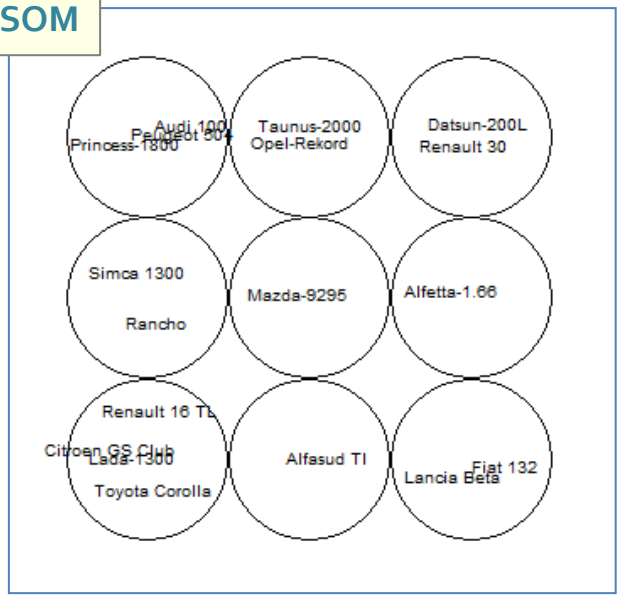


Etiquette des individus, impraticable dès que les effectifs augmentent.

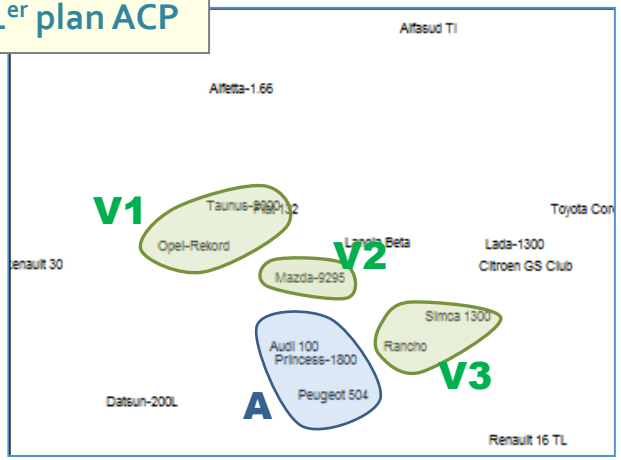
Permet d'identifier les zones à forte densité d'individus. Intéressant sur les grandes bases de données.

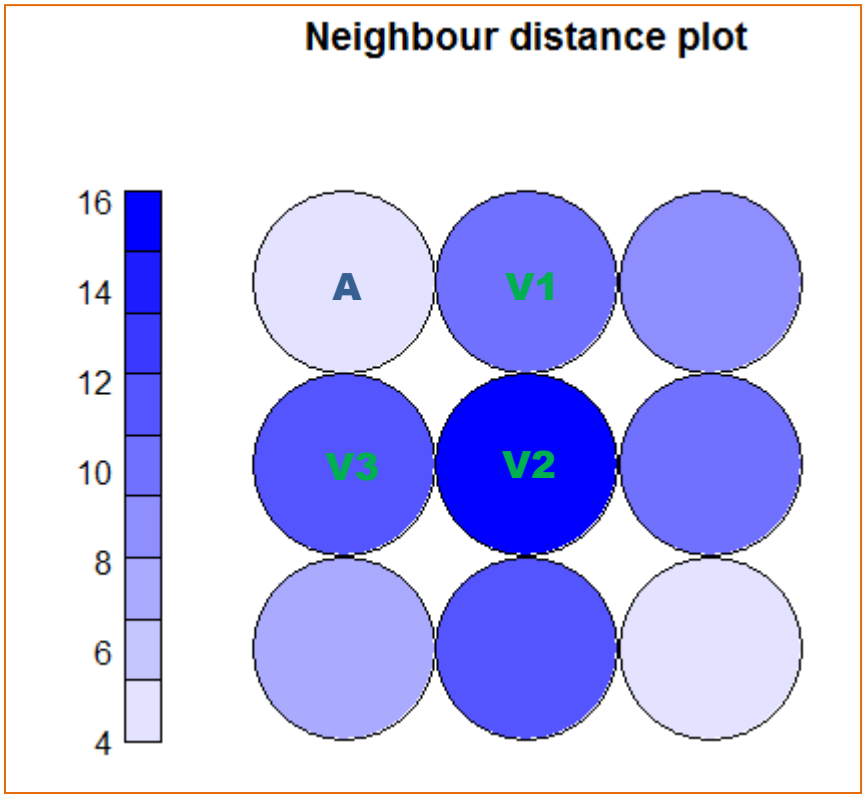
Visualisation – Distance au voisinage (U-matrix)

SOM



1er plan ACP

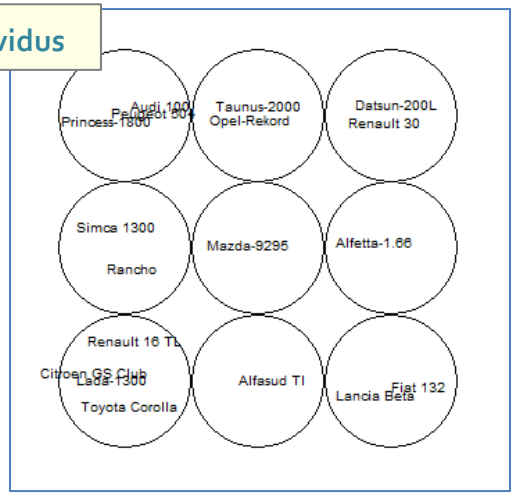




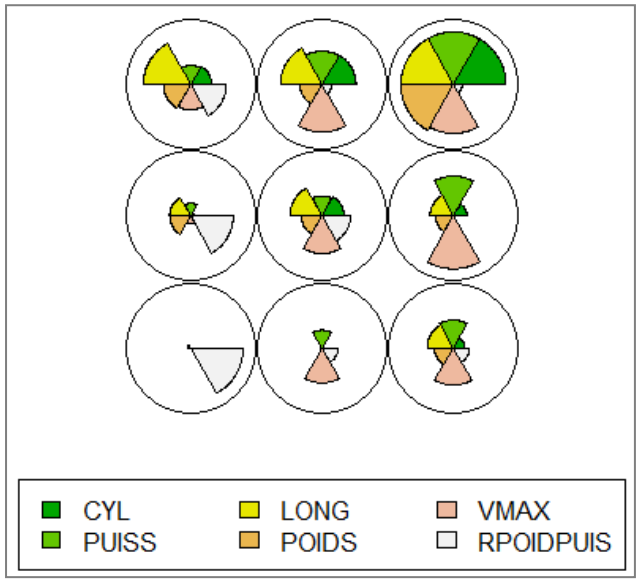
Distance au voisinage immédiat de chaque nœud. Ex. le nœud contenant « Audi 100 », « Peugeot 504 » et « Princess 1800 » est très proche de ses voisins immédiats (V1, V2, et V3)(cf. dans le premier plan factoriel ACP)

Visualisation – Caractérisation par les variables

Objectifs : comprendre ce qui caractérise les régions de la carte topologique



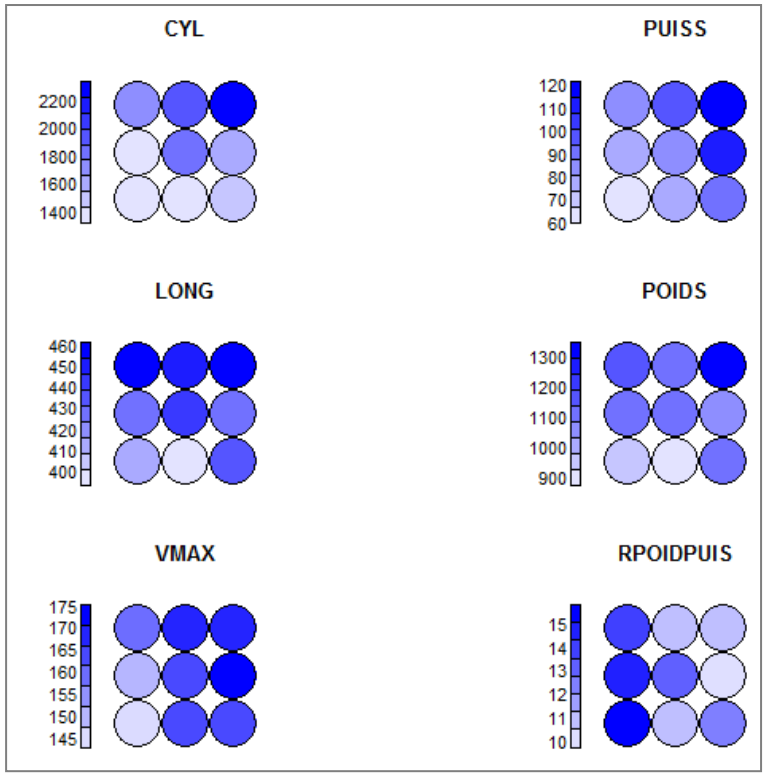
Codebooks



Impraticable dès que le nombre de variables est élevé

Heatmaps

On utilise les moyennes conditionnelles



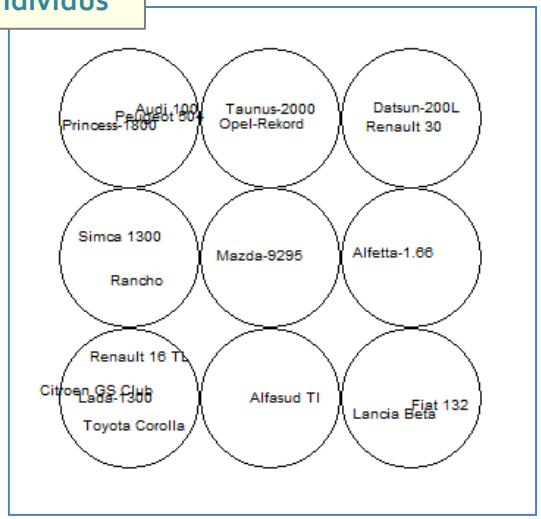
Plus facile à lire, mais la multiplication des graphiques ne facilite pas les choses non plus.

Le rapport de corrélation peut être un indicateur numérique possible.

Visualisation – Caractérisation par les variables

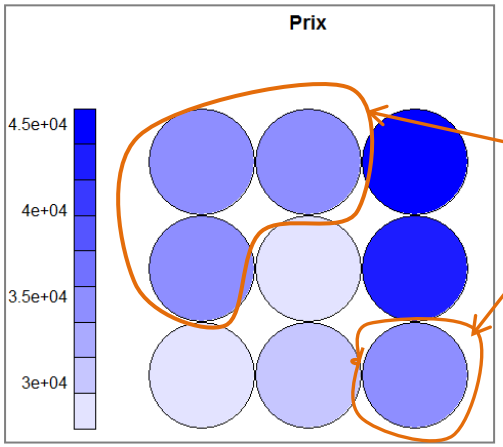
Elle peut s'étendre aux variables illustratives

Individus

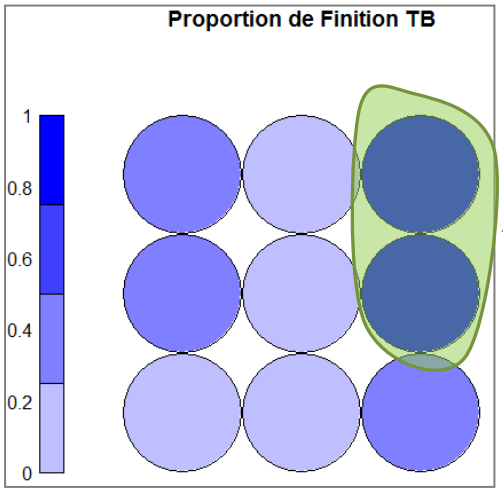


Modele	FINITION	PRIX
Alfasud TI	B	30570
Audi 100	TB	39990
Simca 1300	M	29600
Citroen GS Club	M	28250
Fiat 132	B	34900
Lancia Beta	TB	35480
Peugeot 504	B	32300
Renault 16 TL	B	32000
Renault 30	TB	47700
Toyota Corolla	M	26540
Alfetta-1.66	TB	42395
Princess-1800	B	33990
Datsun-200L	TB	43980
Taunus-2000	B	35010
Rancho	TB	39450
Mazda-9295	M	27900
Opel-Rekord	B	32700
Lada-1300	M	22100

Prix moyens conditionnellement aux noeuds



Même gamme de prix, mais pas pour les mêmes raisons



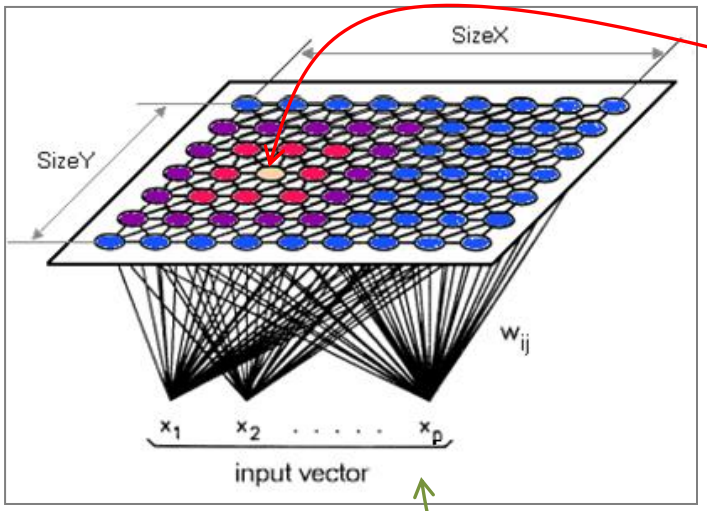
Les voitures les plus coûteuses sont celles qui ont les meilleurs finitions.

Traitement d'un individu supplémentaire

Affectation au nœud le plus proche

Traitement d'un individu supplémentaire

Affectation d'un individu supplémentaire à un neurone de sortie. Ce calcul prendra tout son sens lorsqu'on utilisera SOM dans le clustering.



Identifier le neurone de sortie qui est le plus proche (neurone vainqueur) au sens de la distance utilisée (ex. distance euclidienne aux codebooks)

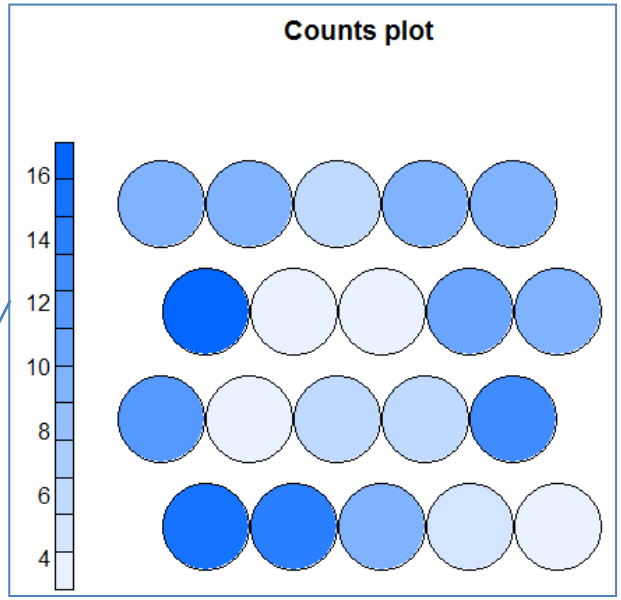
Présenter l'individu à la couche d'entrée, avec éventuellement les préparations adéquates (centrage / réduction)

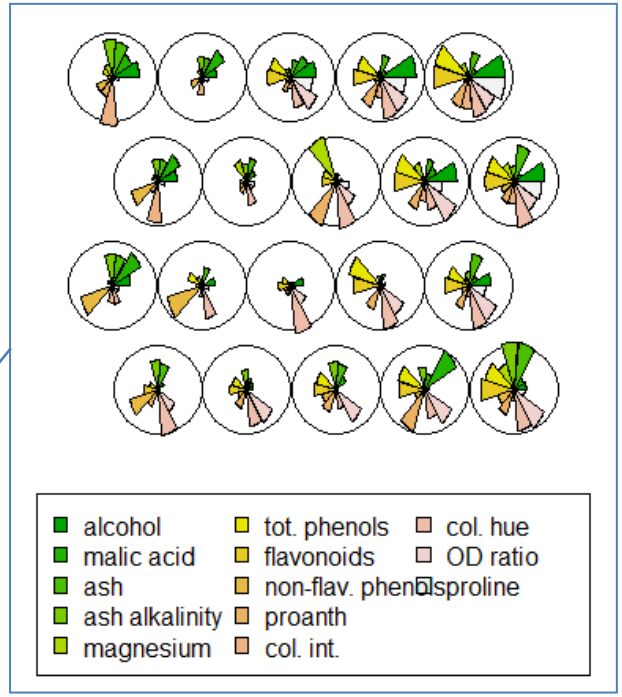
Logiciels – Exemple d'analyse

R (package Kohonen, Tanagra)

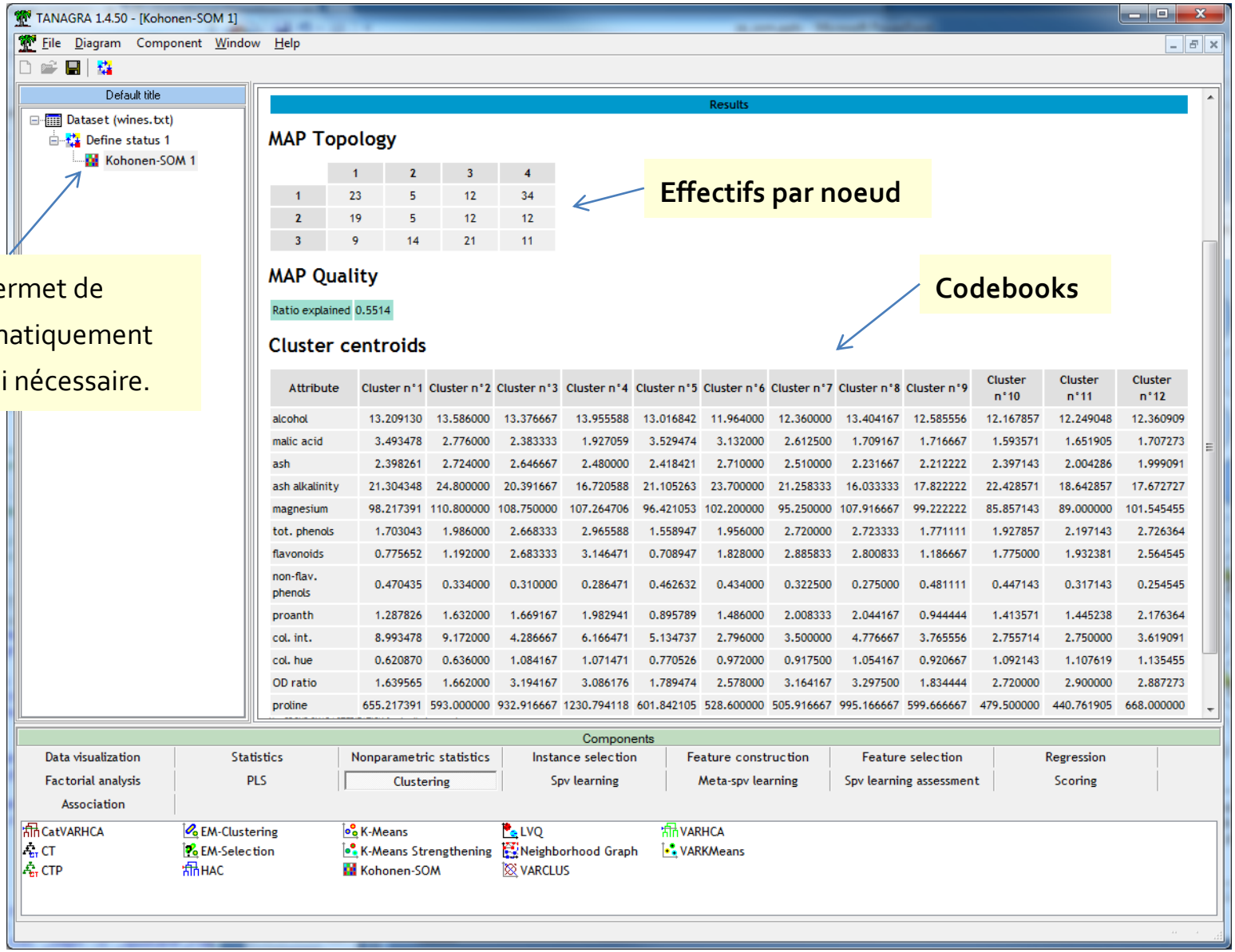
R – Package « kohonen »

```
#package kohonen
library(kohonen)

#données wines intégrées au package (n = 177, p = 13)
data(wines)
print(summary(wines))

#Z - données centrées réduites (important)
Z <- scale(wines,center=T,scale=T)

#apprentissage - grille hexagonale
grille <- som(Z,grid=somgrid(5,4,"hexagonal"))

#degradé de bleu pour les couleurs des noeuds
degrade.bleu <- function(n){
  return(rgb(0,0.4,1,alpha=seq(1/n,1,1/n)))
}

#nombre d'observations dans les cellules
plot(grille,type="count",palette.name=degrade.bleu)

#profil des cellules - codebook
plot(grille,type="codes",codeRendering = "segments")
```





Tanagra – Composant « Kohonen – SOM »



	1	2	3	4
1	23	5	12	34
2	19	5	12	12
3	9	14	21	11

Attribute	Cluster n°1	Cluster n°2	Cluster n°3	Cluster n°4	Cluster n°5	Cluster n°6	Cluster n°7	Cluster n°8	Cluster n°9	Cluster n°10	Cluster n°11	Cluster n°12
alcohol	13.209130	13.586000	13.376667	13.955588	13.016842	11.964000	12.360000	13.404167	12.585556	12.167857	12.249048	12.360909
malic acid	3.493478	2.776000	2.383333	1.927059	3.529474	3.132000	2.612500	1.709167	1.716667	1.593571	1.651905	1.707273
ash	2.398261	2.724000	2.646667	2.480000	2.418421	2.710000	2.510000	2.231667	2.212222	2.397143	2.004286	1.999091
ash alkalinity	21.304348	24.800000	20.391667	16.720588	21.105263	23.700000	21.258333	16.033333	17.822222	22.428571	18.642857	17.672727
magnesium	98.217391	110.800000	108.750000	107.264706	96.421053	102.200000	95.250000	107.916667	99.222222	85.857143	89.000000	101.545455
tot. phenols	1.703043	1.986000	2.668333	2.965588	1.558947	1.956000	2.720000	2.723333	1.771111	1.927857	2.197143	2.726364
flavonoids	0.775652	1.192000	2.683333	3.146471	0.708947	1.828000	2.885833	2.800833	1.186667	1.775000	1.932381	2.564545
non-flav. phenols	0.470435	0.334000	0.310000	0.286471	0.462632	0.434000	0.322500	0.275000	0.481111	0.447143	0.317143	0.254545
proanth	1.287826	1.632000	1.669167	1.982941	0.895789	1.486000	2.008333	2.044167	0.944444	1.413571	1.445238	2.176364
col. int.	8.993478	9.172000	4.286667	6.166471	5.134737	2.796000	3.500000	4.776667	3.765556	2.755714	2.750000	3.619091
col. hue	0.620870	0.636000	1.084167	1.071471	0.770526	0.972000	0.917500	1.054167	0.920667	1.092143	1.107619	1.135455
OD ratio	1.639565	1.662000	3.194167	3.086176	1.789474	2.578000	3.164167	3.297500	1.834444	2.720000	2.900000	2.887273
proline	655.217391	593.000000	932.916667	1230.794118	601.842105	528.600000	505.916667	995.166667	599.666667	479.500000	440.761905	668.000000

Une option permet de réduire automatiquement les variables si nécessaire.

Clustering à partir de SOM

Classification Mixte - Traitement des grands ensembles de données

Classification automatique

Typologie, apprentissage non-supervisé, clustering

X (tous quantitatifs) - Pas de Y à prédire

Modele	Prix	Cylindree	Puissance	Poids	Consommation	Groupe
Daihatsu Cuore	11600	846	32	650	5.7	
Suzuki Swift 1.0 GLS	12490	993	39	790	5.8	
Fiat Panda Mambo L	10450	899	29	730	6.1	
VW Polo 1.4 60	17140	1390	44	955	6.5	
Opel Corsa 1.2i Eco	14825	1195	33	895	6.8	
Subaru Vivio 4WD	13730	658	32	740	6.8	
Toyota Corolla	19490	1331	55	1010	7.1	
Opel Astra 1.6i 16V	25000	1597	74	1080	7.4	
Peugeot 306 XS 108	22350	1761	74	1100	9	
Renault Safrane 2.2. V	36600	2165	101	1500	11.7	
Seat Ibiza 2.0 GTI	22500	1983	85	1075	9.5	
VW Golt 2.0 GTI	31580	1984	85	1155	9.5	
Citroen ZX Volcane	28750	1998	89	1140	8.8	
Fiat Tempra 1.6 Liberty	22600	1580	65	1080	9.3	
Fort Escort 1.4i PT	20300	1390	54	1110	8.6	
Honda Civic Joker 1.4	19900	1396	66	1140	7.7	
Volvo 850 2.5	39800	2435	106	1370	10.8	
Ford Fiesta 1.2 Zetec	19740	1242	55	940	6.6	
Hyundai Sonata 3000	38990	2972	107	1400	11.7	
Lancia K 3.0 LS	50800	2958	150	1550	11.9	
Mazda Hachtback V	36200	2497	122	1330	10.8	
Mitsubishi Galant	31990	1998	66	1300	7.6	
Opel Omega 2.5i V6	47700	2496	125	1670	11.3	
Peugeot 806 2.0	36950	1998	89	1560	10.8	
Nissan Primera 2.0	26950	1997	92	1240	9.2	
Seat Alhambra 2.0	36400	1984	85	1635	11.6	
Toyota Previa salon	50900	2438	97	1800	12.8	
Volvo 960 Kombi aut	49300	2473	125	1570	12.7	

Objectif : identifier des groupes d'observations ayant des caractéristiques similaires (ex. comportement d'achats de clients, caractère « polluant » de véhicules, etc.)

On veut que :

(1) Les individus dans un même groupe se ressemblent le plus possible

(2) Les individus dans des groupes différents se démarquent le plus possible

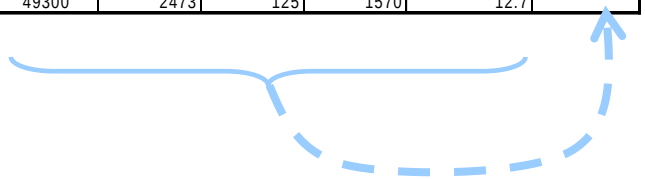
Pourquoi ?

→ Identifier des structures sous-jacentes dans les données

→ Résumer des comportements

→ Affecter de nouveaux individus à des catégories

→ Identifier les cas totalement atypiques



Identifier les catégories (groupes) de voitures « similaires » (c.-à-d. qui se ressemblent)

On peut procéder directement en limitant le nombre de nœuds dans la carte topologique.

Mais rien ne distingue vraiment l'approche de la méthode des k-means dans ce cas.

Classification mixte - Principe

Problème

La CAH nécessite le calcul des distances entre individus pris deux à deux. Il nécessite également l'accès à cette matrice à chaque agrégation. Infaisable sur des grands ensembles de données (en nombre d'observations).

Démarche

S'appuyer sur le pré-regroupement réalisée à l'aide de SOM, démarrer la CAH à partir des pre-clusters. Souvent (*attention, pas toujours*), les nœuds adjacents de la carte topologique appartiendront à la même classe. L'interprétation n'en est que plus facile (interprétation de la carte aide à celle de la classification).

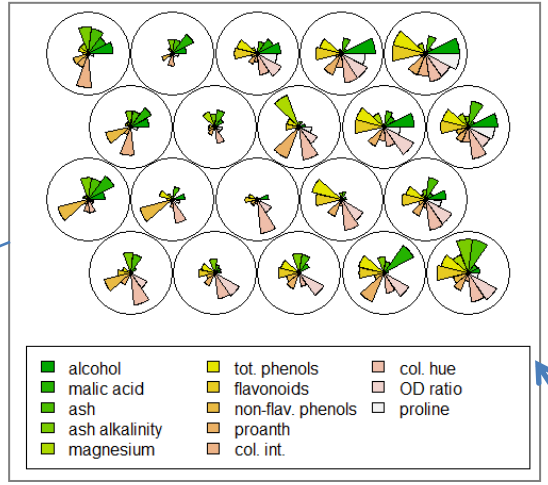
Intérêt

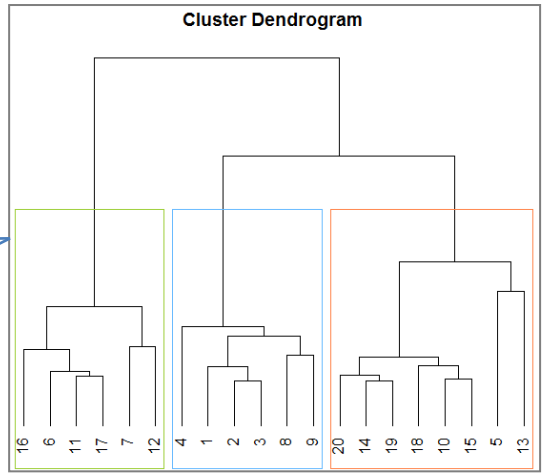
Pouvoir traiter des très grandes bases, tout en bénéficiant des avantages de la CAH (hiérarchie de partitions imbriquées, dendrogramme pour la compréhension et l'identification des classes).

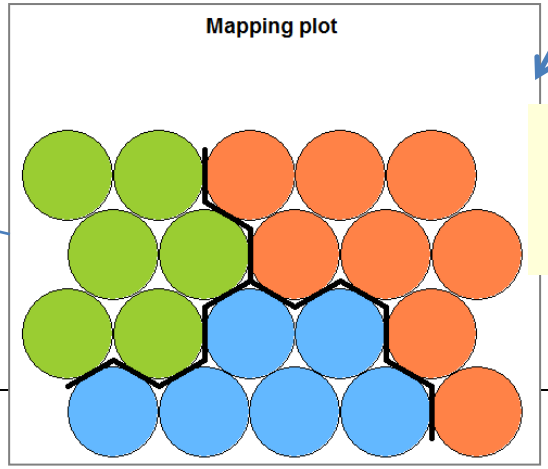
Classification mixte

Un exemple sous R (suite des données « wines »)

```
#profil des cellules - codebook
plot(grille,type="codes",codeRendering = "segments")

#distance entre noeuds en utilisant les codebooks
d <- dist(grille$codes)

#cah - saut maximum
#le poids des noeuds est ignoré ici
cah <- hclust(d,method="ward.D")
plot(cah,hang=-1)

#découpage en 3 classes
groupes <- cutree(cah,k=3)

#matérialisatuion des classes dans le dendrogramme
rect.hclust(cah,k=3,border=c("yellowgreen","steelblue1","sienna1"))

#matérialisation des classes dans la carte topologique
plot(grille,type="mapping",bgcol=c("steelblue1","sienna1","yellowgreen")[groupes])
add.cluster.boundaries(grille,clustering=groupes)
```



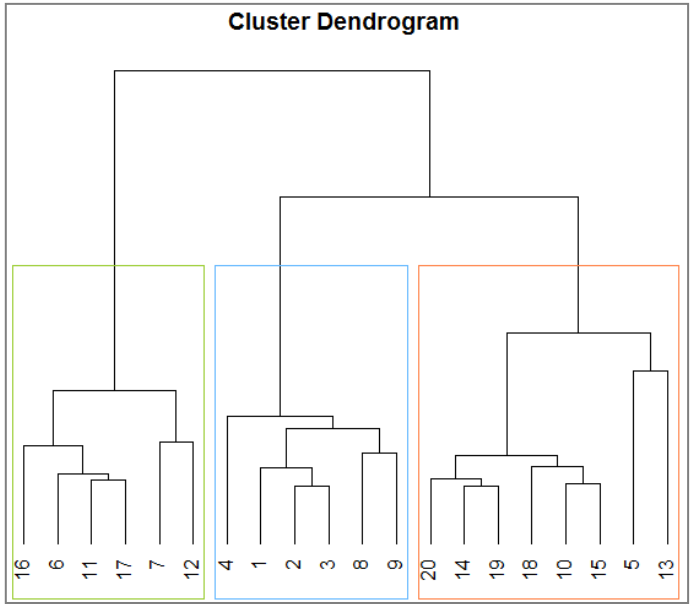


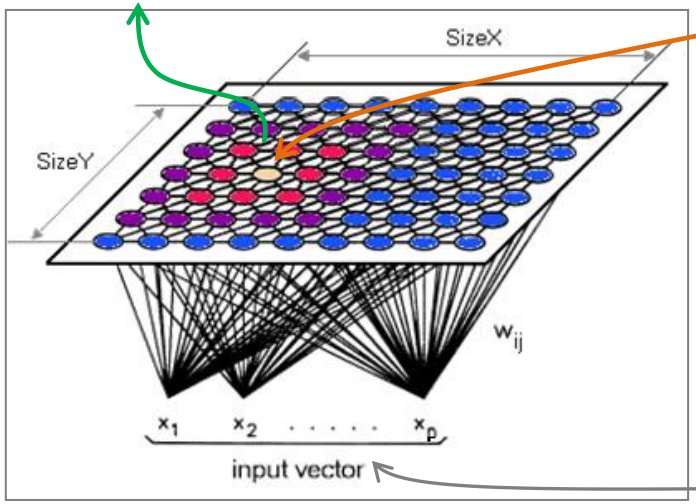


Analyser la correspondance : profils et clusters

Classification mixte - Déploiement

Procéder en deux temps : identifier le nœud de la carte topologique associé à l'individu, puis le cluster associé au nœud.





Identifier le cluster (groupe) associé au neurone de la couche de sortie. Groupe d'appartenance.

Identifier le neurone de sortie qui est le plus proche (neurone vainqueur) au sens de la distance utilisée (ex. distance euclidienne aux codebooks)

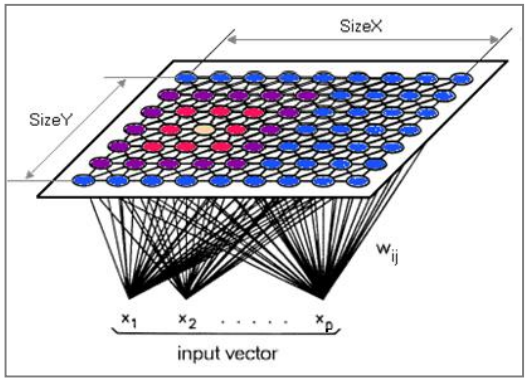
Présenter l'individu à la couche d'entrée, avec éventuellement les préparations adéquates (centrage / réduction)

Extension à l'apprentissage supervisé

Prise en compte d'une variable cible à prédire

$$Y = f(X1, X2, \ldots ; \alpha)$$

Extension de SOM au supervisé



Solution 1. Construire la carte en non-supervisé puis, à l'issue du traitement, calculer la meilleure prédiction sur chaque nœud (modalité la plus fréquente de Y si classement, moyenne de Y si régression).

Solution 2. Rajouter les informations sur la cible dans les codebooks. Calculer D_X, distance aux codebooks définie sur les descripteurs ; et D_Y distance définie sur la cible. Normaliser D_X et D_Y pour équilibrer les influences (c.-à-d. faire varier les distances entre [0..1]), puis définir une distance globale paramétrable

$$D = \alpha.D_X + (1 - \alpha).D_Y$$

On fait varier α *selon l'importance que l'on souhaite accorder à X et Y*

Bilan

Les cartes de Kohonen constituent à la fois une technique de réduction de dimensionnalité, de visualisation et de classification automatique (clustering).

Le couple Kohonen + clustering est particulièrement séduisant.

Avantages

Technique de réduction non linéaire (vs. ACP par ex.).

Nombreuses possibilités de visualisation.

La méthode est simple, facile à expliquer… et à comprendre.

Capacité à traiter des grandes bases (complexité linéaire par rapport au nombre d'observations et de variables).

Inconvénients

Mais… soucis temps de traitements sur les très grandes bases (nécessité de passer plusieurs fois les individus).

Visualisation et interprétation des codebooks devient difficile lorsque le nombre de variables est très élevé.

Bibliographie

Ouvrage de référence

Kohonen T., « Self-Organizing Maps », 3rd Edition, Springer Series in Information Sciences, Vol. 30, 2001.

Supports en ligne et tutoriels

Ahn J.W., Syn S.Y., « Self Organizing Maps », 2005.

Bullinaria J.A., « Self Organizing Maps : Fundamentals », 2004.

Cottrell M., Letrémy P., « Algorithme de Kohonen : classification et analyse exploratoire des données », 2003.

Lynn Shane, « Self Organizing Maps for Customer Segmentation using R », R-bloggers, 2014.

Minsky M., « Kohonen's Self Organizing Features Maps ».

Tutoriel Tanagra, « Les cartes de Kohonen », 2008.

Wikibooks – Data Mining Algorithms in R, « Clustering / Self Organizing Maps (SOM) ».