# STATISTICA MULTIVARIATA

Corso di laurea in matematica | Anno accademico 2004/05 | Numero 10

## Analisi di gruppi

In un campione di dati statistici sono spesso presenti gruppi che possono essere noti in anticipo o meno. Dell'analisi di queste strutture si occupano soprattutto tre grandi discipline statistiche: l'analisi della varianza, l'analisi delle discriminanti e la teoria del raggruppamento automatico.

Nell'*analisi della varianza* la suddivisione in gruppi è già nota e si studia se e come una o più variabili statistiche differiscano da un gruppo all'altro. Nel caso di una variabile si parla di analisi della varianza univariata, nel caso di più variabili di analisi della varianza multivariata.

Anche nell'*analisi delle discriminanti* la suddivisione in gruppi è nota e si cercano funzioni discriminanti con cui distinguere i gruppi. Assumiamo quindi di avere un insieme di pazienti $A \subset \mathbb{R}_m$ e una partizione $A = S \dot\cup M$ in sani e malati. Allora cerchiamo una funzione $f : \mathbb{R}_m \longrightarrow R$, detta *funzione discriminante*, tale che gli insiemi degli individui con test positivo risp. negativo corrispondano il più possibile ad $M$ ed $S$. Spesso l'insieme dei positivi è definito come $P := (f > 0)$ e quindi l'insieme dei negativi come $N := (f \leq 0)$. È importante che nelle applicazioni dell'analisi delle discriminanti in statistica medica in genere si vorrebbe successivamente applicare lo stesso criterio $f$ a individui che non fanno parte di $A$ per poter valutare se siano affetti da quella malattia.

Nella terza delle tre discipline, la teoria del *raggruppamento automatico* (nella letteratura inglese nota come *cluster analysis*), non è ancora nota la suddivisione in gruppi e l'obiettivo è proprio un tale raggruppamento. Ci occuperemo di questo compito in questa parte finale del corso.

## Raggruppamento automatico

Questo campo della statistica si occupa della costruzione di raggruppamenti (in inglese *cluster* significa grappolo, gruppetto) da un insieme di dati ed è particolarmente adatto per l'uso degli algoritmi genetici, sia perché mutazioni e incroci sono definibili in modo molto naturale, sia perché nella cluster analysis viene utilizzata una molteplicità di criteri di ottimalità per le partizioni che negli approcci tradizionali richiedono ogni volta algoritmi di ottimizzazione diversi e spesso computazionalmente difficili e quindi non applicabili per insiemi grandi (e spesso anche solo medi) di dati, mentre, come abbiamo già osservato, gli algoritmi genetici non dipendono dalle proprietà matematiche delle funzioni utilizzate e hanno una complessità che cresce solo in modo lineare con il numero dei dati. Siccome l'algoritmo non dipende dalla funzione di ottimalità scelta, anche se ci limiteremo probabilmente all'uso del cosiddetto criterio della varianza, lo stesso algoritmo può essere usato per un criterio di ottimalità qualsiasi. Nella letteratura è descritta una grande varietà di misure di somiglianza o di diversità, tra le quali in un'applicazione concreta si può scegliere per definire l'ottimalità delle partizioni, ma il modo in cui viene usato l'algoritmo genetico è sempre uguale.

È per esempio piuttosto difficile trovare algoritmi tradizionali per il caso che l'omogeneità e la diversità dei gruppi non siano descritte mediante misure di somiglianza o diversità tra gli individui ma direttamente da misure per i gruppi, mentre ciò non causa problemi per l'algoritmo genetico.

Elenchiamo alcuni campi di applicazione del raggruppamento automatico: classificazione di specie in botanica e zoologia (*tassonomia numerica*) o di aree agricole o biogeografiche, classificazione di specie virali o batteriche, definizione di gruppi di persone con comportamento (istruzione, attitudini, ambizioni, livello di vita, professione) simile in studi sociologici o psicologici, creazione di gruppi di dati omogenei nell'elaborazione dei dati (per banche dati o grandi biblioteche), elaborazione di immagini (ad esempio messa in evidenza di formazioni patologiche in radiografie mediche), individuazione di gruppi di pazienti con forme diverse di una malattia o riguardo alla risposta a un tipo di trattamento, classificazione di malattie in base a sintomi e test di laboratorio, studi linguistici, raggruppamento di regioni (province, comuni) relativamente a caratteristiche economiche (o livello generale di vita o qualità dei servizi sanitari), individuazione di gruppi di località con frequenza simile per quanto riguarda una determinata malattia, reperti archeologici o paleontologici o mineralogici (descritti ad esempio mediante la loro composizione chimica) o antropologici, dati criminalistici (impronte digitali, caratteristiche genetiche, forme di criminalità e loro distribuzione geografica o temporale), confronto tra molecole organiche, classificazione di scuole pittoriche, indagini di mercato (in cui si cerca di individuare gruppi omogenei di consumenti), raggruppamenti dei clienti di un assicurazione in gruppi per definire il prezzo delle polizze, classificazione di strumenti di lavoro o di prodotti nell'industria oppure dei posti di lavoro in una grande azienda, confronto del costo della vita nei paesi europei, divisione dei componenti di un computer in gruppi per poterli disporre in modo da minimizzare la lunghezza di cavi e circuiti.

In queste applicazioni, che si differenziano fortemente per la quantità degli oggetti da classificare (poche decine nel caso di oggetti archeologici, milioni di pixel nell'elaborazione di immagini) e per la natura dei dati, spesso non è facile scegliere un criterio di ottimalità robusto (cambi di scala possono ad esempio influenzare l'esito della classificazione, quando si usano distanze euclidee) e superare la spesso notevole complessità computazionale.

In questo numero



## Il criterio della varianza

$A$ sia un sottoinsieme finito di $\mathbb{R}_m$.
Per un sottoinsieme non vuoto $\alpha$ di $A$ denotiamo con

$$\overline{\alpha} := \frac{1}{|\alpha|} \sum_{x \in \alpha} x$$

il baricentro di $\alpha$. Poniamo inoltre

$$\Delta\alpha := \sum_{x \in \alpha} |x - \overline{\alpha}|^2$$

Per una partizione $P$ di $A$ sia infine

$$g(P) := \sum_{\alpha \in P} \Delta\alpha$$

Questa è la funzione da minimizzare quando si usa *il criterio della varianza*.

Più precisamente si fissa il numero $k$ delle classi della partizione; la partizione ottimale è quella partizione $P$ di $A$ con $k$ classi per cui $g(P)$ assume il minimo; il minimo esiste certamente, perché $A$ è un insieme finito e quindi anche il numero delle partizioni di $A$ è finito, benché molto grande.

In generale, nel raggruppamento automatico si vorrebbe da un lato che ogni classe della partizione sia il più possibile omogenea e quindi le distanze tra gli elementi di una stessa classe siano piccole, dall'altro che le classi siano il più separate tra di loro. Il criterio della varianza soddisfa, come si può dimostrare, allo stesso tempo entrambe queste richieste. Esso è, per dati che hanno una rappresentazione naturale nel $\mathbb{R}_m$, il criterio di ottimalità più usato, benché non esente da limitazioni (cfr. pagina 37); bisogna in ogni caso come sempre scalare in modo appropriato le variabili, utilizzando ad esempio una delle tecniche di standardizzazione che conosciamo.

---

*„In alcuni campi di ricerca si può pertanto ritenere che la fase di classificazione sia il momento essenziale del procedimento scientifico ...“* (Rizzi, 72)

---

*„Do not assume that clustering methods are the best way to discover interesting groupings in the data; in our experience the visualization methods are often far more effective.“* (Venables/Ripley, 316)

## Il numero delle partizioni

Quante sono le partizioni di un insieme finito? Denotiamo con $S(n,k)$ il numero delle partizioni di un insieme con $n$ elementi in $k$ classi. I numeri della forma $S(n,k)$ sono detti *numeri di Stirling di seconda specie*.

**Lemma 1.** *Per $n,k \geq 1$ vale*

$$S(n,k) = S(n-1,k-1) + k \cdot S(n-1,k)$$

Dimostrazione. Una partizione di $\{1,\dots,n\}$ può contenere $\{n\}$ come elemento (in tal caso $n$ è equivalente solo a se stesso) oppure no.

Il numero delle partizioni di $\{1,\dots,n\}$ in $k$ classi di cui una coincide con $\{n\}$ è evidentemente uguale al numero delle partizioni di $\{1,\dots,n-1\}$ in $k-1$ classi, cioè uguale a $S(n-1,k-1)$.

Se una partizione di $\{1,\dots,n\}$ con $k$ classi non contiene $\{n\}$ come elemento, essa si ottiene da una partizione di $\{1,\dots,n-1\}$ in $k$ classi, aggiungendo $n$ ad una delle $k$ classi. Per fare questo abbiamo $k$ possibilità.

---

Dalla definizione otteniamo direttamente le seguenti relazioni (per la prima si osservi che l'insieme vuoto $\emptyset$ può essere considerato in modo banale come partizione di $\emptyset$).

$$S(0,0) = 1.$$
$$S(0,k) = 0 \text{ per } k \geq 1.$$
$$S(n,0) = 0 \text{ per } n \geq 1$$

Possiamo così scrivere un programma in R per il calcolo ricorsivo di $S(n,k)$:

```
M.stirling2 = function (n,k)
{if (n==0) if (k==0) 1 else 0
else if (k==0) 0 else
Recall(n-1,k-1)+k*Recall(n-1,k)}
```

I numeri di Stirling di seconda specie crescono fortemente:

$$S(5,2) = 15$$
$$S(10,2) = 511$$
$$S(10,3) = 9330$$
$$S(20,5) = 749206090500$$
$$S(50,4) = 52818655359845226611906445312$$

## Calcolo della funzione $g$

Rappresentiamo in primo luogo il sottoinsieme $A$ mediante la matrice dei dati $X \in \mathbb{R}^n_m$; più precisamente $A$ è l'insieme delle righe di $X$. Denotiamo con $k$ il numero delle classi. Una partizione è rappresentata da un vettore $P \in \{1,\dots,k\}^n$. Una riga $X^i$ appartiene alla $a$-esima classe $\alpha_a$ se $P^i$ è uguale ad $a$.

Per ogni $a \in \{1,\dots,k\}$ dobbiamo calcolare il baricentro $\overline{\alpha_a}$; otteniamo così una matrice $B \in \mathbb{R}^k_m$ con $B^a = \overline{\alpha_a}$, almeno se la $a$-esima classe non è vuota, perché altrimenti il baricentro $\overline{\alpha_a}$ non è ben definito. D'altra parte però gli indici $a$ con $\alpha_a = \emptyset$ non entrano veramente nel calcolo di $g$, come risulta da

$$g(P) = \sum_{\alpha \in P} \sum_{x \in \alpha} |x - \overline{\alpha}|^2$$

o dalla formula equivalente

$$g(P) = \sum_{i=1}^{n} |X^i - B^{P^i}|^2$$

Infatti, se $\alpha_a = \emptyset$, $P^i$ sarà sempre $\neq a$. Qui possiamo utilizzare a nostro favore il fatto che R permette di creare matrici numeriche in cui appaiono i valori `Inf` e `NaN`, per cui possiamo creare una matrice $B$ che contiene anche questi valori come elementi.

Definiamo prima una funzione che per ogni vettore $v \in \{1,\dots,k\}^n$ calcola la frequenza con cui appaiono i suoi elementi:

```
S.conta = function (v,k)
{u=rep(0,k)
for (a in v) u[a]=u[a]+1
u}

# Esempio:
v=c(1,2,4,4,1,1,4,2,5)
print(v)
# 1 2 4 4 1 1 4 2 5

u=S.conta(v,5)
print(u)
# 3 2 0 3 1
```

Adesso calcoliamo la matrice dei baricentri. Nella penultima riga appare l'espressione `B[a,]/cont[a]` che in R però è ammissibile anche quando il denominatore si annulla. Infatti, quando `cont[a]` è uguale a zero, anche `B[a,]` è uguale a zero, e `0/0` in R diventa `NaN`, valore che, come abbiamo detto, può far parte dei coefficienti di una matrice.

```
Sra.baricentri = function (X,P,k)
{m=ncol(X); n=nrow(X)
cont=S.conta(P,k)
B=Mm(rep(0,m*k),righe=k)
for (i in 1:n)
{a=P[i]; B[a,]=B[a,]+X[i,]}
for (a in 1:k) B[a,]=B[a,]/cont[a]
B}
```

A questo punto possiamo definire la funzione per il calcolo di $g$:

```
Sra.g = function (X,P,k)
{n=nrow(X); B=Sra.baricentri(X,P,k)
s=0
for (i in 1:n) {u=X[i,]-B[P[i],]
s=s+Mv.scalare(u,u)}
s}
```

---

Consideriamo la prima figura a pagina 31. Potremmo pensare a due partizioni $P$ e $Q$ a tre classi $\alpha, \beta$ e $\gamma$.

Nella partizione $P$ poniamo

$\alpha$ = {To, Mi, Fi, Bo, Pr, Pd}
$\beta$ = {Ge, Fe, Ve, Ra, Pi}
$\gamma$ = {Bz,Tn,Bl,Vi}

nella partizione $Q$ spostiamo Pisa da $\beta$ a $\gamma$.

Tenendo conto dell'ordine in cui i comuni appaiono nella tabella a pagina 16, $P$ e $Q$ diventano vettori definiti nella tabella seguente:

| | P | Q |
|---|---|---|
| Belluno | 3 | 3 |
| Bologna | 1 | 1 |
| Bolzano | 3 | 3 |
| Ferrara | 2 | 2 |
| Firenze | 1 | 1 |
| Genova | 2 | 2 |
| Milano | 1 | 1 |
| Padova | 1 | 1 |
| Parma | 1 | 1 |
| Pisa | 2 | 3 |
| Ravenna | 2 | 2 |
| Torino | 1 | 1 |
| Trento | 3 | 3 |
| Venezia | 2 | 2 |
| Vicenza | 3 | 3 |

Come standardizzazione usiamo di nuovo la matrice dei ranghi. Quale delle due partizioni è migliore?

```
Db(2)
X=Db.matrice()
XR=Sm.rango(X)

P=c(3,1,3,2,1,2,1,1,1,2,2,1,3,2,3)
Q=c(3,1,3,2,1,2,1,1,1,3,2,1,3,2,3)

gp=Sra.g(P,XR,3)
gq=Sra.g(Q,XR,3)

print(gp)
# 2.141156

print(gq)
# 2.63733
```

La partizione $P$ è quindi migliore. Con la matrice non standardizzata invece risulterebbe leggermente migliore la seconda partizione:

```
gp=Sra.g(P,X,3)
gq=Sra.g(Q,X,3)

print(gp)
# 1484101

print(gq)
# 1448306
```

## Il programma principale

Presentiamo adesso un programma completo in R che contiene le funzioni per il raggruppamento automatico mediante un algoritmo genetico. Il programma è piuttosto semplice e segue l'algoritmo di base dell'ottimizzazione genetica visto a pagina 32. Benché molto più lento di un programma analogo in C, è sufficiente per trattare i nostri 15 comuni.

La funzione principale `Sra` è interattiva, permettendo all'utente di impostare durante l'esecuzione l'intervallo di tempo `dt` che intercorre tra le visualizzazioni del risultato ottimale raggiunto.

```
Sra = function (X,k)
{n=nrow(X)
MP=Mm(numeric(n*40),col=40)
MP=Sra.nuovi(MP,k,1,40)
dt=10; t=0; repeat
{t=t+1; R=Sra.rendimenti(MP,X,k)
Ord=order(R); MP=MP[,Ord]
MP=Sra.nuovi(MP,k,31,40)
MP=Sra.mutazioni(MP,k,R,X)
MP=Sra.incroci(MP,k,R,X)
if (t%%dt==0)
{dt=Sra.visualizza(t,dt,
  R[Ord[1]],MP[,1])
if (dt==0) break}}}
```

Si noti l'introduzione del vettore `R` dei rendimenti. Per le visualizzazioni usiamo

```
Sra.visualizza = function (t,dt,rend,P)
{P=paste(P,collapse=' ')
cat('\n',rend,': ',P,'   dopo ',t,
   ' generazioni\n',sep='')
a=readline('Vuoi continuare? ')
if (a=='n') 0
else {v=as(a,'numeric')
if (!is.na(v)) v else dt}}
```

Battendo semplicemente invio, il programma continua; con `'n'` si ferma, mentre se inseriamo un numero, questo viene usato come nuovo valore della variabile `dt` che indica l'intervallo tra due visualizzazioni.

## L'algoritmo genetico

La creazione di una nuova *matrice di partizioni* (`MP`), che contiene 40 colonne ciascuna delle quali rappresenta una partizione, avviene con

```
Sra.nuovi = function (MP,k,a,b)
{n=nrow(MP); for (j in a:b)
MP[,j]=Snc.interi(n,1,k)
MP}
```

il calcolo dei rendimenti con

```
Sra.rendimenti = function (MP,X,k)
apply(MP,2,Sra.g,X,k)
```

Qui viene usata la funzione `Sra.g` definita a pagina 35.

---

Per le mutazioni usiamo

```
Sra.muta = function (P,k,R)
{n=length(P); p=runif(1,0,0.5)
for (i in 1:n) if (runif(1)<p)
P[i]=Snc.interi(1,1,k)
P}
```

e

```
Sra.mutazioni = function (MP,k,R,X)
{for (j in 1:40)
{P=Sra.muta(MP[,j],k,R)
if (Sra.g(P,X,k)<R[j]) MP[,j]=P}
MP}
```

per gli incroci

```
Sra.incroci = function (MP,k,R,X)
{n=nrow(MP)
for (j in seq(1,40,2))
{P1=MP[,j]; P2=MP[,j+1]
p=runif(1,0,0.5)
for (i in 1:n) if (runif(1)<p)
{a=P1[i]; P1[i]=P2[i]; P2[i]=a}
R1=Sra.g(P1,X,k); R2=Sra.g(P2,X,k)
if ((R1<R[j])&&(R2<R[j+1]))
{MP[,j]=P1; MP[,j+1]=P2}}
MP}
```

Vengono incrociate la prima con la seconda partizione, la terza con la quarta, e così via. Nelle mutazioni e negli incroci applichiamo il metodo spartano.

---

## Raggruppamenti dei 15 comuni

Applichiamo il metodo ai 15 comuni. Chiediamo un raggruppamento in 4 classi ed eseguiamo l'algoritmo prima senza standardizzazione con

```
Db(2)
X=Db.matrice()
Sra(X,4)
```

usando, con il comando `Db(2)`, la nostra banca dati. Dopo 200 generazioni otteniamo il risultato

```
410593.9: 1 2 1 3 2 2 4 1 3 1 3 4 1 3 1
```

Proviamo la proiezione su $[0,1]$:

```
Db(2)
X=Db.matrice()
X01=Sm.tra01(X)
Sra(X01,4)
```

Dopo 200 generazioni otteniamo

```
1.124847: 1 2 1 3 2 2 4 2 2 2 3 4 1 3 2
```

Si noti che i rendimenti non sono confrontabili (perché abbiamo usato standardizzazioni diverse) e possono essere usati solo per valutare la bontà del risultato per esecuzioni con la stessa standardizzazione.

Nello stesso modo procediamo per la matrice dei ranghi:

```
Db(2)
X=Db.matrice()
XR=Sm.rango(X)
Sra(XR,4)
```

ottenendo dopo 200 generazioni

```
1.565391: 3 4 3 1 4 1 4 2 4 2 1 4 3 1 2
```

Per vedere concretamente le partizioni riportiamo i risultati in una tabella:

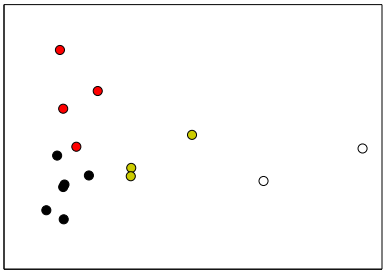
| | X | X01 | XR |
|---|---|---|---|
| Belluno | 1 | 1 | 3 |
| Bologna | 2 | 2 | 4 |
| Bolzano | 1 | 1 | 3 |
| Ferrara | 3 | 3 | 1 |
| Firenze | 2 | 2 | 4 |
| Genova | 2 | 2 | 1 |
| Milano | 4 | 4 | 4 |
| Padova | 1 | 2 | 2 |
| Parma | 3 | 2 | 4 |
| Pisa | 1 | 2 | 2 |
| Ravenna | 3 | 3 | 1 |
| Torino | 4 | 4 | 4 |
| Trento | 1 | 1 | 3 |
| Venezia | 3 | 3 | 1 |
| Vicenza | 1 | 2 | 2 |

Si osservi che i numeri delle partizioni possono essere permutati tra di loro e che perciò il gruppo 1 e il gruppo 2 non sono più simili di quanto lo siano il gruppo 1 e il gruppo 4. Abbiamo così i seguenti gruppi.

Senza standardizzazione:

*Belluno, Bolzano, Padova, Pisa, Trento, Vicenza*
*Bologna, Firenze, Genova*
*Ferrara, Parma, Ravenna, Venezia*
*Milano, Torino*

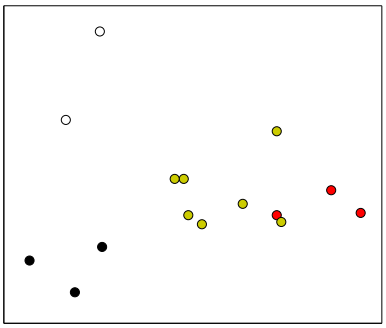
Usando le figure alle pagine 29-31, coloriamo i comuni in colori diversi a seconda della classe nella partizione generata dall'algoritmo di raggruppamento.



Quando confrontiamo i risultati, dobbiamo ricordarci che si tratta di proiezioni 2-dimensionali, mentre il raggruppamento avviene (nel nostro caso) in quattro dimensioni. Questo spiega perché ad esempio nella prossima figura un punto giallo è apparentemente (cioè in due dimensioni) separato dagli altri punti della stessa classe.

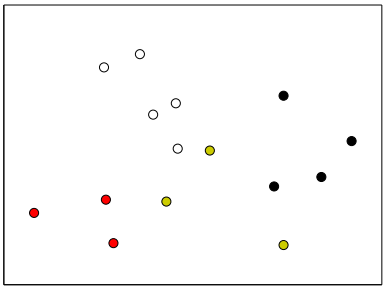
Con proiezione su $[0,1]$:

*Belluno, Bolzano, Trento*
*Bologna, Firenze, Genova, Padova, Parma, Pisa, Vicenza*
*Ferrara, Ravenna, Venezia*
*Milano, Torino*



Con la matrice dei ranghi:

*Ferrara, Genova, Ravenna, Venezia*
*Padova, Pisa, Vicenza*
*Belluno, Bolzano, Trento*
*Bologna, Firenze, Milano, Parma, Torino*



Soprattutto in problemi complicati i risultati di un'ottimizzazione genetica non sono unici e possono differire da un'esecuzione all'altra, anche dopo lo stesso numero di generazioni.

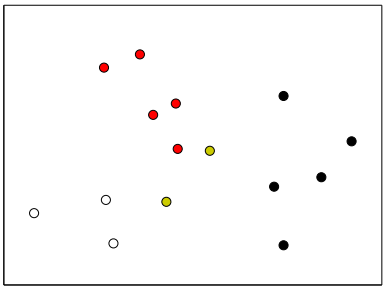
Nell'ultimo esempio (in cui si usa la matrice dei ranghi) può sorprendere che Genova si trovi nello stesso gruppo di Ferrara; perciò proviamo un'altra esecuzione, trovando dopo 400 generazioni

```
1.477381: 4 3 4 1 3 1 3 2 3 1 1 3 4 1 2
```

risultato che rimane uguale anche dopo 800 generazioni e quindi probabilmente è ottimale; esso corrisponde alla partizione

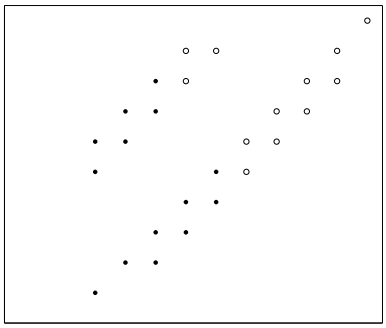
*Ferrara, Genova, Pisa, Ravenna, Venezia*
*Padova, Vicenza*
*Bologna, Firenze, Milano, Parma, Torino*
*Belluno, Bolzano, Trento*

Genova è rimasta nel gruppo di Ferrara, a cui si è aggiunta Pisa.

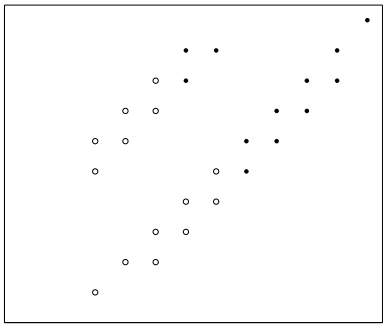


## Il problema dei gruppi sferici

Scopriamo adesso un difetto piuttosto spiacevole dei metodi di raggruppamento o almeno del criterio della varianza, che ne limita in alcune situazioni l'applicazione. Vengono infatti preferiti *gruppi sferici*, anche quando una suddivisione diversa sembrerebbe migliore. Consideriamo l'insieme dei dati nell'ultima figura a pagina 31. Applichiamo il nostro metodo alla matrice non standardizzata con due classi, colorando gli elementi dei due gruppi in modo diverso. Dopo 400 generazioni otteniamo



L'algoritmo ha creato due gruppi approssimativamente sferici invece della più naturale divisione diagonale. Anche una standardizzazione (ad esempio proiezione su $[0,1]$) ovviamente non elimina il problema:



Abbiamo ottenuto esattamente la stessa partizione!

Bisogna allora provare (se ci si accorge del problema, il che non è sempre facile in dimensione maggiore a 2) ad usare un'altra funzione di ottimalità (ad esempio basata sul *criterio del determinante*), ma anche questa può avere limitazioni a sua volta.

## La funzione pam di R

R fornisce il pacchetto `cluster` per le funzioni di raggruppamento automatico. Per ottenere partizioni ottimali si può usare la funzione `pam` che, nella sintassi più semplice, si usa nella forma `pam(tab,k)`, in cui `tab` è una tabella e `k` il numero delle classi.

Dopo `library(cluster)` con

```
Db(2)
tab=Db.tab()
print(pam(tab,4))
```

otteniamo (in un output più complesso) il vettore

```
1 2 1 3 2 2 4 2 3 1 3 4 1 3 1
```

che corrisponde alla partizione

*Belluno, Bolzano, Pisa, Trento, Vicenza*
*Bologna, Firenze, Genova, Padova*
*Ferrara, Parma, Ravenna, Venezia*
*Milano, Torino*

L'esecuzione è molto veloce.

## Suddivisione gerarchica

La teoria dei raggruppameni comprende numerose tecniche e oltre a raggruppamenti tramite partizioni si utilizzano anche *ricoprimenti* (cioè rappresentazioni dell'insieme dei dati come unione di insiemi non necessariamente disgiunti) e *suddivisioni gerarchiche* (spesso rappresentate tramite *dendrogrammi*). Queste ultime sono usate frequentemente nella letteratura statistica applicata, ma spesso in modo non appropriato; è infatti difficile la loro corretta interpretazione. La teoria matematica della classificazione gerarchica si basa sulle *metriche non archimedee* (o *ultrametriche*) ed è esposta nei libri di Diday/ e Bock. Ultrametriche sono note e utilizzate da molto tempo in matematica, soprattutto in alcuni campi dell'algebra e della teoria dei numeri e nella dinamica simbolica.

Una metrica $d$ si dice non archimedea, se per ogni numero reale $\varepsilon > 0$ vale la relazione di transitività

$$d(a,b) < \varepsilon \text{ e } d(b,c) < \varepsilon \Longrightarrow d(a,c) < \varepsilon$$

Ciò significa che la relazione

$$a \sim_\varepsilon b \iff d(a,b) < \varepsilon$$

(riflessiva e simmetrica per ogni metrica) è una relazione di equivalenza. Questa condizione, molto naturale nella statistica, non è soddisfatta nella geometria euclidea: se la distanza tra $a$ e $b$ è minore di un metro e lo stesso vale per la distanza tra $b$ e $c$, da ciò non segue che anche la distanza tra $a$ e $c$ sia minore di un metro. Metriche non archimedee non misurano una distanza geometrica, ma comunanze: più proprietà due oggetti hanno in comune, più simili e vicini risultano in un'appropriata metrica non archimedea.

## Il matematico in statistica

Per fare bene il suo lavoro, lo statistico che lavora in un'azienda, nell'amministrazione pubblica o nella ricerca clinica, deve comprendere i compiti che gli vengono posti e deve essere in grado di interagire con i committenti. Nonostante ciò la statistica è di sua natura una disciplina matematica che si basa sul calcolo delle probabilità, una teoria astratta e difficile, e richiede conoscenze tecniche in altri campi della matematica come analisi reale e complessa, analisi armonica, calcolo combinatorio (ad esempio per la pianificazione di esperimenti). Nell'analisi delle componenti principali e nella ricerca di raggruppamenti sarà compito dello statistico scegliere la rappresentazione dei dati e le misure per la somiglianza o diversità di individui e gruppi. In questo corso abbiamo potuto accennare solo ad alcune delle difficoltà concettuali e tecniche che si incontrano.

Nella statistica multivariata in particolare probabilmente molte tecniche sono ancora da scoprire e i metodi più efficienti si baseranno forse su metodi geometrici avanzati, ad esempio della geometria algebrica reale e della teoria delle rappresentazioni di gruppi.

Ci sono tanti campi di applicazione della statistica in medicina, bioinformatica, farmacologia, matematica finanziaria, linguistica, demografia, che uno studente che intraprende questa professione dopo aver acquisito una solida formazione matematica può sperare in un'attività interessante e gratificante.

L'abitudine ai dati e alla loro interpretazione formerà le sue capacità di giudicare situazioni complesse in modo razionale oltre a fornirgli un ricco patrimonio di informazioni, quindi potrà anche aspirare a una carriera amministrativa o manageriale.

Nel suo lavoro giornaliero potrà, nei contatti con ricercatori clinici o amministratori o con l'opinione pubblica utilizzare le proprie conoscenze teoriche per chiarire il significato di risultati di test clinici o di rilievi statistici o per proporre nuovi esperimenti o indagini.

## Bibliografia


15914 **G. Bahrenberg/E. Giese/J. Nipper:** Statistische Methoden in der Geographie II. Borntraeger 2003.

4153 **H. Bock:** Automatische Klassifikation. Vandenhoeck & Ruprecht 1974.

15918 **S. Bolasco:** Analisi multidimensionale dei dati. Carocci 2002.

**R. Cormack:** A review of classification. J. Roy. Stat. Soc. A 134 (1971), 321-367.

17110 **P. Diaconis:** Group representations in probability and statistics. Hayward 1988.

3903 **E. Diday/J. Lemaire/J. Pouget/F. Testu:** Eléments d'analyse de données. Dunod 1982.

16693 **G. Dunn/B. Everitt:** An introduction to mathematical taxonomy. Dover 2004.

15812 **H. Eckey/R. Kosfeld/M. Rengers:** Multivariate Statistik. Gabler 2002.

16238 **L. Fahrmeir/A. Hamerle/G. Tutz:** Multivariate statistische Verfahren. De Gruyter 1996.

16041 **J. Gentle:** Elements of computational statistics. Springer 2002.

4155 **J. Hartung/B. Elpelt:** Multivariate Statistik. Oldenbourg 1986.

15240 **R. Herwig:** Large-scale information theoretic clustering with application to the analysis of genetic fingerprinting data. Logos 2001.

**L. Kaufman/P. Rousseeuw:** Finding groups in data. An introduction to cluster analysis. Wiley 2005.

3784 **A. Rizzi:** Analisi dei dati. NIS 1985.

4154 **H. Späth:** Clusterformation und -analyse. Oldenbourg 1983.

15536 **W. Venables/B. Ripley:** Modern applied statistics with S. Springer 2002.