Distanze e dissimilarità

Introduzione

Consideriamo un insieme di caratteristiche $x_1, x_2, \ldots, x_p$ misurate su n osservazioni. I metodi cosiddetti di apprendimento supervisionato (ad esempio, regressione e classificazione) consentono di ottenere una previsione relativa ad una variabile risposta; in questo contesto siamo invece interessati a estrarre informazioni sulla struttura dei dati:

- Esiste un modo informativo per visualizzare dati multivariati?
- Si possono individuare sottogruppi di unità che siano simili rispetto a quelle in gruppi diversi?

Per rispondere a domande come queste vengono utilizzate tecniche di apprendimento non supervisionato che includono l'analisi delle componenti principali e il clustering. Il problema fondamentale del clustering può essere formulato come segue:

Dato un insieme di p variabili rilevate su n unità, si vogliono raggruppare le unità in modo tale che le unità appartenenti a ciascun gruppo siano il più possibili omogenee.

Tale problema può essere affrontato utilizzando un'ampia varietà di metodi, a seconda dell'approccio specifico utilizzato per definire la somiglianza tra due punti dati, dell'algoritmo adottato per derivare i gruppi (ad esempio, metodi gerarchici e non gerarchici) e del tipo di dati specifico (alcuni tecniche sono applicabili solo a dati continui, alcuni altri metodi possono essere utilizzati su dati qualitativi o dati misti).

Un esempio con dati simulati

L'esempio seguente utilizza dati simulati costituiti dal numero medio (mensile) di pasti consumati fuori casa (ad es., al ristorante) e dalla spesa per i pasti consumati a casa (annuale) per un insieme di 100 famiglie. Per prima cosa leggiamo i dati dal file `food_spending.csv`, che contiene le colonne `FreqAway` e `Home`, quindi otteniamo il grafico a dispersione dei dati o *scatterplot*:

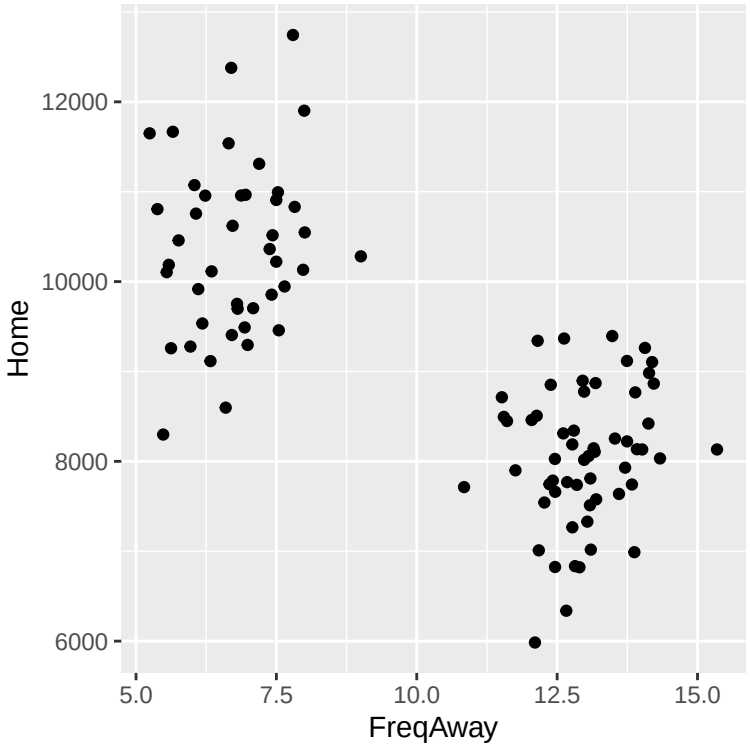
```
FoodS<-read.csv("food_spending.csv", header=TRUE)
head(FoodS)
```

```
##    FreqAway      Home
## 1  6.951280 10965.524
## 2  7.415451  9854.342
## 3 12.462907  6824.662
## 4 14.136932  8982.304
## 5  5.969204  9276.950
## 6  7.976161 10131.322
```

```
dim(FoodS)
```

```
## [1] 100   2
```

```
library(ggplot2)
ggplot(FoodS, aes(FreqAway, Home)) +
  geom_point()
```



Il grafico evidenzia due gruppi ben separati, il gruppo con una spesa bassa per i pasti consumati a casa e un'elevata frequenza di pasti fuori casa, e il gruppo di famiglie con una spesa elevata per i pasti consumati fuori casa e una bassa spesa per i pasti consumati a casa.

In un problema non supervisionato miriamo a identificare cluster distinti sulla base delle variabili osservate su un insieme di unità. I gruppi risultanti sono tali che i membri dello stesso gruppo sono omogenei rispetto alle variabili coinvolte, mentre le osservazioni in gruppi diversi sono piuttosto diverse tra loro. Il primo passo è quantificare, per due unità qualsiasi, il grado di 'somiglianza'.

Calcolo della matrice di dissimilarità o distanza

In `R`, la funzione `dist()` consente di calcolare la distanza tra due punti (ad esempio, la distanza euclidea tra due vettori); il tipo di distanza da calcolare viene specificato con `method`, che può essere "euclidean" (default), "maximum", "manhattan", "canberra", "binary" o "minkowski":

```
# distance matrix via dist function
D.<- dist(FoodS) # Euclidean distance
str(D.)
```

```
##  'dist' num [1:4950] 1111 4141 1983 1689 834 ...
##  - attr(*, "Size")= int 100
##  - attr(*, "Diag")= logi FALSE
##  - attr(*, "Upper")= logi FALSE
##  - attr(*, "method")= chr "euclidean"
##  - attr(*, "call")= language dist(x = FoodS)
```

```
head(D.)
```

```
## [1] 1111.1824 4140.8662 1983.2338 1688.5747  834.2036 1707.2239
```

```
# coercion to matrix and
#selection of 5 rows/columns
```

```
Dmat<-as.matrix(D.)
Dmat[1:5, 1:5]
```

```
##          1         2        3         4         5
## 1    0.000 1111.1824 4140.866 1983.2338 1688.5747
## 2 1111.182    0.0000 3029.684  872.0644  577.3939
## 3 4140.866 3029.6844    0.000 2157.6424 2452.2967
## 4 1983.234  872.0644 2157.642    0.0000  294.7596
## 5 1688.575  577.3939 2452.297  294.7596    0.0000
```

```
range(Dmat)
```

```
## [1]    0.000 6759.434
```

```
#View(Dmat)
```

In alternativa, il pacchetto `cluster` può essere utile per ottenere una matrice di dissimilarità attraverso la funzione `daisy()` ed ha il vantaggio di avere tra le possibili opzioni la scelta dell'indice di Gower per variabili miste.

```
library(cluster)  # load cluster library
```

Per conoscere meglio la funzione `daisy()` si ricorra alla documentazione dell'Help:

```
 ?daisy
```

Cominciamo calcolando la matrice delle distanze euclidee per il dataset `FoodS` (opzione di default per `metric`)

```
dist1<-daisy(FoodS)
```

Otteniamo le prime cinque righe e colonne della matrice delle distanze (si noti che la funzione `as.matrix` è necessaria per trasformare l'output di `daisy` che restituisce un oggetto della classe *dissimilarity*)

```
as.matrix(dist1)[1:5, 1:5]
```

```
##          1         2        3         4         5
## 1    0.000 1111.1824 4140.866 1983.2338 1688.5747
## 2 1111.182    0.0000 3029.684  872.0644  577.3939
## 3 4140.866 3029.6844    0.000 2157.6424 2452.2967
## 4 1983.234  872.0644 2157.642    0.0000  294.7596
## 5 1688.575  577.3939 2452.297  294.7596    0.0000
```

Talvolta, quando le variabili presentano scale di grandezza diverse o variabilità molto eterogenea, è opportuno uniformare il contributo di ciascuna variabile trasformando preliminarmente i dati. Spesso, ad esempio, le variabili vengono centrate e divise per la loro deviazione standard, oppure si normalizza ciascuna variabile in base al range della stessa.

La funzione `daisy()` consente di centrare le variabili e riscalarle attraverso l'argomento `stand` (si veda l'Help per maggiori dettagli sulla trasformazione applicata ai dati originali):

```
dist2<-daisy(FoodS, stand=TRUE)
```

Dopo aver trasformato il risultato in una matrice, possiamo stampare le prime righe/colonne

```
dist2m<-as.matrix(dist2)
dist2m[1:5, 1:5]
```

```
##           1         2        3        4         5
## 1 0.0000000 0.9536446 3.945398 2.891414 1.4663512
## 2 0.9536446 0.0000000 3.053032 2.321756 0.6808883
## 3 3.9453977 3.0530320 0.000000 1.908494 2.9730049
```

```
## 4 2.8914145 2.3217564 1.908494 0.000000 2.6862840
## 5 1.4663512 0.6808883 2.973005 2.686284 0.0000000
```

Esercizio 1 *Si ottenga la matrice di dissimilarità per* `FoodS` *utilizzando la distanza di Manhattan nella funzione* `dist()`*. Si confronti il risultato con la matrice ottenuta mediante la standardizzazione preventiva dei dati effettuata con la funzione* `scale()`*.*

Per calcolare le dissimilarità a partire da variabili di tipo misto è opportuno ricorrere all'indice di **Gower**. Nella funzione `daisy` occorre specificare `metric = "gower"`.

Consideriamo un dataset riferito ad un insieme di clienti di una assicurazione per i quali si dispone delle seguenti caratteristiche del sinistro:

- "litig": presenza di un contenzioso (no=0, sì=1)
- "soft_injury": presenza di lesione dei tessuti molli (no=0, sì=1)
- "emergency_tr": presenza di un trattamento di emergenza (no=0, sì=1)
- "NumTreat": numero di cure mediche

Le prime tre variabili sono binarie e la quarta è numerica.

Importiamo i dati e codifichiamo le variabili categoriali come fattori:

```
d<-read.csv(file="claims.csv", header = TRUE)
head(d)
```

```
##   litig soft_injury emergency_tr NumTreat
## 1     0           1            0        2
## 2     0           0            1        1
## 3     0           0            0        2
## 4     0           0            0        4
## 5     0           1            1        5
## 6     0           1            1        3
```

```
d$litig<-as.factor(d$litig)
d$soft_injury<-as.factor(d$soft_injury)
d$emergency_tr<-as.factor(d$emergency_tr)

str(d)
```

```
## 'data.frame':    1000 obs. of  4 variables:
##  $ litig       : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 2 ...
##  $ soft_injury : Factor w/ 2 levels "0","1": 2 1 1 1 2 2 1 1 1 2 ...
##  $ emergency_tr: Factor w/ 2 levels "0","1": 1 2 1 1 2 2 2 1 2 2 ...
##  $ NumTreat    : int  2 1 2 4 5 3 4 3 1 8 ...
```

Esercizio 2 *Descrivere le variabili del dataset utilizzando rappresentazioni grafiche e indici statistici appropriati.*

Per calcolare la dissimilarità tra le unità mediante l'indice di Gower si ricorre ad una media ponderata dei contributi di ciascuna variabile, dove

- il contributo di una variabile nominale o binaria alla dissimilarità è 0 se entrambi i valori sono uguali, 1 altrimenti;
- il contributo delle altre variabili è la differenza assoluta di entrambi i valori, divisa per il *range* di quella variabile

Si noti che è possibile assegnare un peso per ciascuna variabile invece di 1 nella formula originale di Gower.

Limitandoci alle variabili binarie si ottiene la matrice di dissimilarità come segue

```r
diss<-as.matrix(daisy(d[,-4], metric= "gower"))
diss[1:5, 1:5]
```

```
##           1         2         3         4         5
## 1 0.0000000 0.6666667 0.3333333 0.3333333 0.3333333
## 2 0.6666667 0.0000000 0.3333333 0.3333333 0.3333333
## 3 0.3333333 0.3333333 0.0000000 0.0000000 0.6666667
## 4 0.3333333 0.3333333 0.0000000 0.0000000 0.6666667
## 5 0.3333333 0.3333333 0.6666667 0.6666667 0.0000000
```

Considerando tutte le variabili si ottiene

```r
diss<-as.matrix(daisy(d, metric= "gower"))
diss[1:5, 1:5]
```

```
##           1         2          3          4         5
## 1 0.0000000 0.5178571 0.25000000 0.28571429 0.3035714
## 2 0.5178571 0.0000000 0.26785714 0.30357143 0.3214286
## 3 0.2500000 0.2678571 0.00000000 0.03571429 0.5535714
## 4 0.2857143 0.3035714 0.03571429 0.00000000 0.5178571
## 5 0.3035714 0.3214286 0.55357143 0.51785714 0.0000000
```

```r
range(diss)
```

```
## [1] 0 1
```

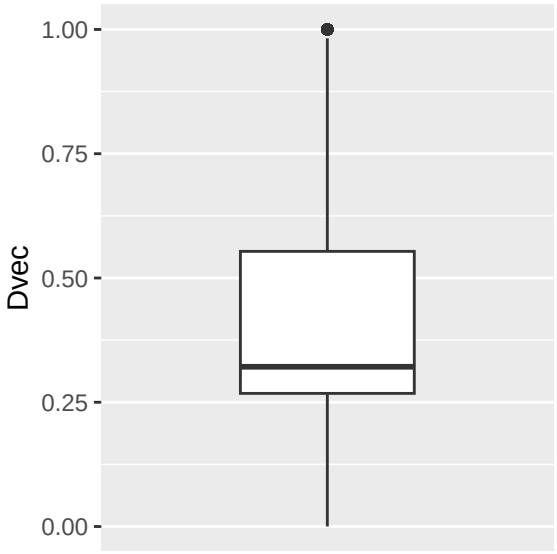
```r
dim(diss)
```

```
## [1] 1000 1000
```

```r
d[as.vector(which(diss == max(diss), arr.ind = TRUE)[1,]), ] # two units with max dissim
```

```
##     litig soft_injury emergency_tr NumTreat
## 656     1           1            1       14
## 69      0           0            0        0
```

```r
Dvec<-as.vector(as.dist(diss))
ggplot() +
  geom_boxplot(aes(y = Dvec)) +
  scale_x_discrete( )
```


1.00
0.75
0.50
0.25
0.00
Dvec