UNIVERSITÀ DEGLI STUDI DI SALERNO

Dipartimento di informatica

Progetto di Statistica e Analisi dei Dati

Anno Accademico 2018/2019

Rosanna Coccaro
Matr. 0512103059

Aspetti della vita quotidiana: lo sport

Possiamo definire la statistica come la scienza dei fenomeni collettivi.

I metodi statistici, infatti, trovano applicazione ovunque occorra investigare **fenomeni di massa** che possano essere in qualche modo quantificati e che siano caratterizzati da variabilità. La statistica definisce le procedure per misurare e rappresentare i fenomeni collettivi e studia i metodi per stimare i valori caratteristici di una popolazione a partire da un campione.

Prendiamo in considerazione il database che raccoglie i dati relativi ad aspetti della vita quotidiana ed in particolare l'attitudine allo sport. Il dataset colleziona dati raccolti per 100 persone con le stesse caratteristiche: ovvero, si tratta di un campione composto da persone di 3 anni e più che svolgono o no pratica sportiva. Per ciascuna regione italiana è indicata la percentuale di persone, relativamente all'anno 2016, che: praticano sport in modo continuativo, praticano sport in modo saltuario, praticano solo qualche attività fisica oppure non praticano sport né attività fisica.

	praticano sport			non praticano sport, né attività fisica
	in modo continuativo	in modo saltuario	solo qualche attività fisica	
Piemonte	26,9	11,5	28,6	32,6
Valle d'Aosta	28,5	14,3	25,2	31,8
Liguria	24,9	7,8	28,6	38,3
Lombardia	30,5	10,7	27,9	30,6
Trentino Alto Adige	36,2	16,7	31,5	15,5
Veneto	29,5	14,4	29,7	26,1
Friuli-Venezia Giulia	27,6	11,9	34	26,2
Emilia-Romagna	31,1	10,8	26	31,9
Toscana	25,9	9,8	31	33,1
Umbria	24,6	8,1	28,6	38,7
Marche	27,6	8	31,3	32,9
Lazio	28,5	7,7	22,8	40,6
Abruzzo	23	11,2	25,3	40,3
Molise	19,3	5,8	22,1	52,5
Campania	13,9	6,1	22,7	56,9
Puglia	20,8	8,5	19,7	50,6
Basilicata	19	7,1	23,2	50,4
Calabria	16,5	7,7	21,9	53,4
Sicilia	16,5	7,5	17	58,4
Sardegna	26,1	10,7	28,6	34,7

Prendere in analisi tale dataset ha come obiettivo:

- Individuare regioni che hanno il più alto tasso di persone che praticano sport
- Individuare regioni che hanno il più alto tasso di persone che non praticano sport
- Individuare relazioni esistenti tra le diverse modalità che può assumere il nostro campione
- Individuare la distribuzione delle persone che praticano o non praticano attività sportiva nelle varie regioni
- Individuare eventuali valori anomali e valori fuori dal range accettabile
- Selezionare cluster in cui inserire le regioni più simili o meno distanti tra loro.

Introduzione all'analisi dei dati

Prima di poter cominciare la nostra analisi, inizializziamo la matrice sulla quale andremo ad operare nel seguito, utilizzando l'ambiente integrato R.
Per prima cosa, per ciascuna delle colonne della tabella creiamo un vettore ad essa corrispondente.

```
in_modo_continuativo <- c(26.9, 28.5, 24.9, 30.5, 36.2, 29.5, 27.6, 31.1,25.9, 24.6,
                          27.6, 28.5, 23, 19.3, 13.9, 20.8, 19, 16.5, 16.5, 26.1)

in_modo_saltuario <- c(11.5, 14.3, 7.8, 10.7, 16.7, 14.4, 11.9, 10.8, 9.8, 8.1,
                       8, 7.7, 11.2, 5.8, 6.1, 8.5, 7.1, 7.7, 7.5, 10.7)

solo_qualche_attivita <- c(28.6, 25.2, 28.6, 27.9, 31.5, 29.7, 34, 26, 31, 28.6,
                           31.3, 22.8, 25.3, 22.1, 22.7, 19.7, 23.2, 21.9, 17, 28.6)

non_praticano_sport <- c(32.6, 31.8, 38.3, 30.6, 15.5, 26.1, 26.2, 31.9, 33.1, 38.7,
                         32.9, 40.6, 40.3, 52.5, 56.9, 50.6, 50.4, 53.4,58.4, 34.7)
```

Costruiamo poi, in R, la matrice dei dati attraverso il comando *cbind* che permette di creare opportune matrici componendo vettori di uguale lunghezza e matrici delle stesse dimensioni. Tale funzione usa i vettori per creare le colonne. Con il comando *rownames* invece possiamo definire un vettore contenente i nomi da assegnare alle righe della matrice, al contrario, la funzione *colnames* permette di ridefinire i nomi delle colonne della matrice di dati.

```
matriceDati <- cbind(in_modo_continuativo, in_modo_saltuario,
                     solo_qualche_attivita, non_praticano_sport)
rownames(matriceDati) <- c("Piemonte", "Valle d'Aosta", "Liguria",
                           "Lombardia", "Trentino Alto Adige", "Veneto",
                           "Friuli Venezia Giulia", "Emilia Romagna",
                           "Toscana", "Umbria", "Marche", "Lazio", "Abruzzo",
                           "Molise", "Campania", "Puglia", "Basilicata",
                           "Calabria", "Sicilia", "Sardegna")
colnames(matriceDati)<-c("in modo continuativo", "in modo saltuario",
                         "solo qualche attività", "non praticano sport")

matriceDati
```

La matrice che otteniamo è riportata nella figura seguente:

```
> matriceDati
      in modo continuativo in modo saltuario solo qualche attività non praticano sport
Piem                 26.9               11.5                 28.6                32.6
VdA                  28.5               14.3                 25.2                31.8
Lig                  24.9                7.8                 28.6                38.3
Lomb                 30.5               10.7                 27.9                30.6
TAA                  36.2               16.7                 31.5                15.5
Ven                  29.5               14.4                 29.7                26.1
FVG                  27.6               11.9                 34.0                26.2
ER                   31.1               10.8                 26.0                31.9
Tos                  25.9                9.8                 31.0                33.1
Umb                  24.6                8.1                 28.6                38.7
Mar                  27.6                8.0                 31.3                32.9
Lazio                28.5                7.7                 22.8                40.6
Abr                  23.0               11.2                 25.3                40.3
Mol                  19.3                5.8                 22.1                52.5
Camp                 13.9                6.1                 22.7                56.9
Pu                   20.8                8.5                 19.7                50.6
Bas                  19.0                7.1                 23.2                50.4
Cal                  16.5                7.7                 21.9                53.4
Sic                  16.5                7.5                 17.0                58.4
Sar                  26.1               10.7                 28.6                34.7
```

Sport, attività fisica e sedentarietà: le distribuzioni di frequenza

Nel 2015, sono circa 20 milioni 200 mila le persone di 3 anni e più che praticano, nel tempo libero, uno o più sport, pari al 34,3% della popolazione di tre anni e più, di cui il 24,4% con continuità e il 9,8% saltuariamente. Il 26,5% della popolazione, pur non praticando uno sport, svolge un'attività fisica come fare passeggiate di almeno due km, nuotare, andare in bicicletta o altro (15 milioni 640 mila persone). I sedentari, ovvero coloro che non praticano alcuno sport o attività fisica nel tempo libero, sono oltre 23 milioni e 50 mila, pari al **39,1%** della popolazione.
Siamo interessati a comprendere come questi dati variano dal 2015 al 2016, andando ad analizzare cosa accade precisamente in ciascuna regione italiana.
Nell'analisi si può privilegiare la dimensione delle unità statistiche o dei caratteri. È necessario osservare che vi sono diverse tipologie di caratteri, e ciò comporta delle conseguenze, in particolare, sul modo in cui essi vengono misurati.
Quindi, prima di cominciare la nostra analisi, è necessario specificare che in *statistica descrittiva* esistono tre tipi diversi di variabili:

- **Variabili qualitative**: i caratteri qualitativi sono quelli che assumono modalità non numeriche: ad esempio, possiamo considerare il sesso ed il titolo di studio delle persone appartenenti al campione.
  Il carattere "sesso" è di tipo sconnesso: tra i valori che può assumere si può stabilire solo la relazione di uguaglianza/disuguaglianza.
  Il carattere "titolo di studio", invece, è suscettibile di ordinamento.
- **Variabili quantitative:** i caratteri quantitativi hanno modalità che risultano essere numeri. Possiamo ad esempio considerare la variabile "voto di un esame" che può assumere valori interi nell'intervallo [18-30].
- **Variabili ordinative:** che presuppongono un ordinamento e possono assumere valori numerici o rappresentare qualità.

La semplice elencazione dei valori dei caratteri delle unità statistiche non consente di sintetizzare l'informazione presente, di individuare dei sottogruppi, di confrontare agevolmente questi dati con altre situazioni.

Per questo motivo, utilizziamo le *distribuzioni di frequenza* che rivestono un ruolo fondamentale **nell'analisi della distribuzione** dei dati riguardanti un determinato fenomeno.

Distribuzioni di frequenza semplice

Per quanto riguarda la distribuzione di frequenza semplice, consideriamo una variabile X e indichiamo con $z_1, z_2, \ldots, z_k$ le modalità distinte da essa assunte (i valori che X può assumere).
Consideriamo poi un campione $(x_1, x_2, \ldots, x_n)$ costituito da n osservazioni di X. Se indichiamo con n_i il numero di volte in cui ciascuna modalità z_i è presente nel campione, ossia **la frequenza assoluta** con cui essa appare nel campione, l'insieme $\{(z_i, n_i), i = 1, 2, \ldots, k\}$ si chiama distribuzione di frequenza.
Se non esistono dati mancanti, la somma delle frequenze assolute è sempre uguale alla numerosità del campione, ossia $n = n_1 + n_2 + \ldots + n_k$.
È possibile anche calcolare le **frequenze relative** come:

$$f_i = \frac{n_i}{n} \quad (i = 1, 2, \ldots, k)$$

Se non esistono dati mancanti si ha: $f_1 + f_2 + \ldots + f_k = 1$.
Per una variabile qualitativa in R si utilizza la funzione table(X) per calcolare le frequenze assolute, mentre la funzione table(X)/lenght(X) dove X rappresenta il campione di ampiezza n, per calcolare le frequenze relative.
Nel nostro dataset, è immediato notare che tutti i dati a nostra disposizione sono numerici e analizzando ciascuna colonna della tabella, possiamo apprendere che nessuna di esse può assumere valori classificabili all'interno di una ben precisa modalità della caratteristica presa in esame. A tal proposito, è possibile per le variabili quantitative raccogliere le informazioni in classi e poi calcolare le frequenze con cui gli elementi del campione cadono nelle diverse classi.
Una prima analisi per il nostro insieme di dati, quindi, potrebbe consistere nell'estrapolare informazioni *circa il numero di regioni che hanno la percentuale di persone che praticano o no sport all'interno di un certo range*.
Per fare questo si utilizza la funzione *cut()* che permette di raggruppare i dati relativi ad un vettore in intervalli, elencando nel parametro breaks gli estremi degli intervalli che sono aperti a sinistra e chiusi a destra. Se desideriamo ottenere intervalli chiusi a sinistra e aperti a destra occorre specificare in *cut()* l'opzione aggiuntiva *right = FALSE*.

```
> classi <- c(0, 12, 24, 36, 48, 60)
```

In questo modo definiamo quali sono le classi di riferimento per ricavarci le distribuzioni di frequenza dalla matrice dei dati. Le classi prese in considerazione sono:

(0,12] (12,24] (24,36] (36,48] (48,60]

A questo punto applichiamo la funzione *table()* per poter calcolare la frequenza assoluta delle classi per ciascun vettore che compone la nostra matrice dei dati.

- Percentuale di persone che praticano sport *in modo continuativo*

```
> table(cut(in_modo_continuativo, classi))

 (0,12] (12,24] (24,36] (36,48] (48,60]
      0       7      12       1       0
```

Da questa prima analisi risulta che solo in una regione si ha il tasso più alto di persone che praticano sport assiduamente. Guardando la tabella, ci rendiamo conto che tale regione è il Trentino Alto Adige con una percentuale di persone che frequentano sport pari al 36,2%.

- Percentuale di persone che praticano sport in modo saltuario

```
> table(cut(in_modo_saltuario, classi))

 (0,12] (12,24] (24,36] (36,48] (48,60]
     17       3       0       0       0
```

È immediato notare che in 17 regioni italiane, la percentuale di persone che praticano sport in modo saltuario è compresa all'interno dell'intervallo (0,12]. In generale possiamo affermare già da questa prima analisi che in Italia la percentuale di persone che praticano sport in maniera saltuaria è abbastanza bassa.

- Percentuale di persone che praticano solo qualche attività fisica

```
> table(cut(solo_qualche_attivita, classi))

 (0,12] (12,24] (24,36] (36,48] (48,60]
      0       7      13       0       0
```

In questo caso, invece la maggior parte delle regioni italiane hanno una percentuale di persone che praticano solo qualche attività fisica compresa all'interno dell'intervallo (24,36].

- Percentuale di persone che non praticano sport né attività fisica

```
> table(cut(non_praticano_sport, classi))

 (0,12] (12,24] (24,36] (36,48] (48,60]
      0       1       9       4       6
```

Da questi risultati possiamo notare che solo una regione ha una percentuale bassa di persone che non praticano attività fisica. Dalla tabella ricaviamo che la regione in cui si ha questa situazione è il Trentino Alto Adige, cosa che avevamo già appreso come conseguenza del risultato dell'analisi effettuata sul vettore *"in_modo_continuativo"*. In ben nove regioni, invece, la percentuale delle persone che non praticano sport è compresa tra il 24% e il 26%, mentre solo quattro hanno un tasso più elevato compreso tra il 36% e il 48%. In altre 6, la situazione sembra più grave, in quanto la percentuale di sedentarietà è maggiore del 48%: ciò vuol dire che la metà o più della metà della popolazione, in queste regioni, risulta non praticare sport nel tempo libero.

Per calcolare le frequenze relative, invece, usiamo la funzione *table()/lenght()*. *Vediamo come applicare tale funzione al nostro dataset:*

- Vediamo cosa accade per il vettore "in_modo_continuativo"

```
> table(cut(in_modo_continuativo, classi))/length(in_modo_continuativo)

 (0,12] (12,24] (24,36] (36,48] (48,60]
   0.00    0.35    0.60    0.05    0.00
```

Ciò significa che nel 60% delle regioni italiane, le persone praticano sport in maniera assidua all'interno dell'intervallo (24,36]. Solo nel 5% delle regioni la percentuale delle persone che praticano sport assiduamente è compresa tra il 36% e il 48%.

- Quello che accade, invece, per il vettore "in_modo_saltuario" è che l'85% delle regioni italiane ha una percentuale di persone che praticano sport saltuariamente compresa nell'intervallo (0,12].

```
> table(cut(in_modo_saltuario, classi))/length(in_modo_saltuario)

 (0,12] (12,24] (24,36] (36,48] (48,60]
   0.85    0.15    0.00    0.00    0.00
```

- Un'altra conclusione importante è relativa al vettore "solo_qualche_attività":

```
> table(cut(solo_qualche_attivita, classi))/length(solo_qualche_attivita)

 (0,12] (12,24] (24,36] (36,48] (48,60]
   0.00    0.35    0.65    0.00    0.00
```

Possiamo affermare, infatti, che nella maggior parte delle regioni, la percentuale delle persone che praticano solo qualche attività fisica è nel range (24,36].

- Infine, possiamo affermare che nel 30% delle regioni italiane, vi è la percentuale più alta di persone che non praticano sport.

```
> table(cut(non_praticano_sport, classi))/length(non_praticano_sport)

 (0,12] (12,24] (24,36] (36,48] (48,60]
   0.00    0.05    0.45    0.20    0.30
```

Ovviamente, le considerazioni fatte dopo il calcolo delle frequenze relative, sono solo una conferma ai risultati dell'analisi fatta sulle frequenze assolute.

Le rappresentazioni grafiche

In questa sezione vedremo come analizzare la nostra tabella di dati attraverso la realizzazione di diverse rappresentazioni grafiche.
Come già affermato in precedenza, occorre tener presente la distinzione tra variabili qualitative e quantitative, in quanto caratteri qualitativi richiedono grafici che molto spesso per i caratteri quantitativi non hanno alcuna necessità di essere considerati.
Infatti, per una variabile qualitativa possiamo considerare l'idea di realizzare un *grafico a* bastoncini utilizzando il comando *plot(table(x))* dove X rappresenta il vettore contenente le varie modalità assunte dalla variabile qualitativa presa in considerazione.
Nel nostro caso però, non ha senso considerare dei grafici a bastoncini per visualizzare le frequenze associate a ciascuna modalità di pratica sportiva, poiché ovviamente siccome stiamo parlando di percentuali, nessuna regione avrà una percentuale esattamente uguale ad un'altra. Per questo non considereremo questo tipo di grafici, ma preferiamo invece far riferimento a grafici a barra e grafici a torta.

Grafici a barre

Disponiamo sull'asse orizzontale ed in modo equispaziato le modalità assunte da X e sull'asse verticale riportiamo le frequenze assolute o le frequenze relative. Tracciamo dei rettangoli centrati sulle modalità z_i tutti della stessa base e con altezza pari alle frequenze (assolute o relative), ottenendo un *grafico (o diagramma) a barre*.
In R si ottiene un grafico a barre utilizzando *barplot(table(x))*.
A questo punto, procediamo con la nostra analisi incentrata sui grafici a barre e consideriamo i vettori "in_modo_continuativo", "in_modo_saltuario", "solo_qualche_attivita" e "non_praticano_sport", che abbiamo già utilizzato nella fase precedente.
Per ciascuno di questi vettori, andiamo a realizzare il nostro grafico a barre.
Quello che si ottiene è un grafico in cui:

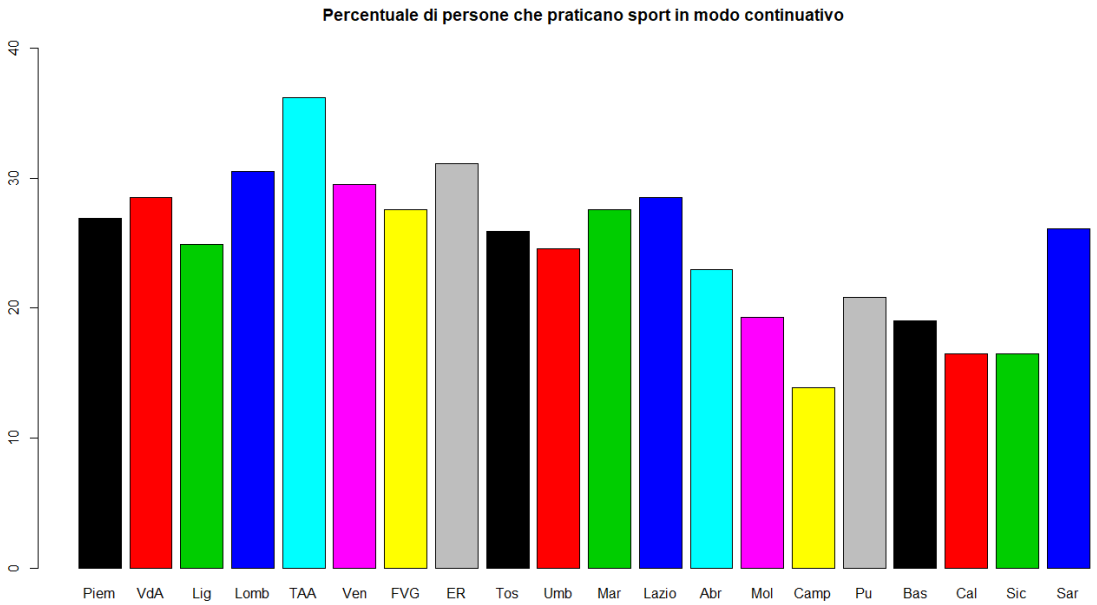
- sull'asse delle x sono poste le 20 regioni
- sull'asse delle y sono poste le percentuali di persone che praticano o no sport in ciascuna regione.

Ovviamente tale analisi non ha molto senso, ma ci permette di dedurre, tramite una visione d'insieme, le regioni che hanno la percentuale più alta di persone che fanno sport assiduamente e le regioni che hanno la percentuale minima di persone che non praticano attività sportiva.

- Per il vettore "in_modo_continuativo si ha:

```
barplot(matriceDati[,1], col = 1:20, ylim = c(0,40),
        main = "Percentuale di persone che praticano
        sport in modo continuativo")
```

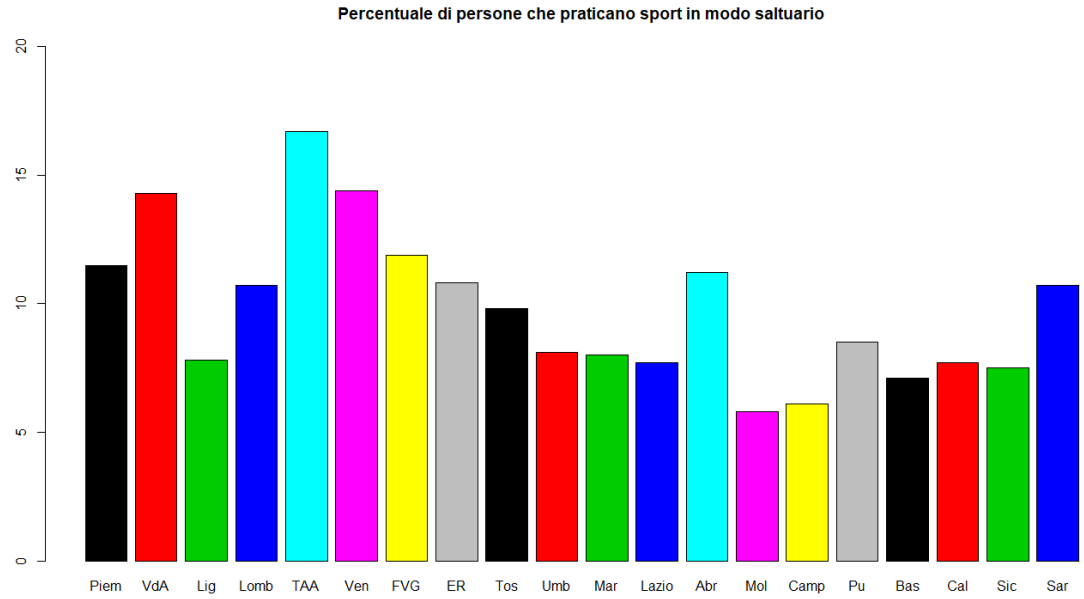
  in cui abbiamo utilizzato il parametro *col* per assegnare a ciascun rettangolo un colore differente.



- Per il vettore "in_modo_saltuario" si ha:

```
barplot(matriceDati[,2], col = 1:20, ylim = c(0,20),
        main = "Percentuale di persone che praticano
        sport in modo saltuario")
```

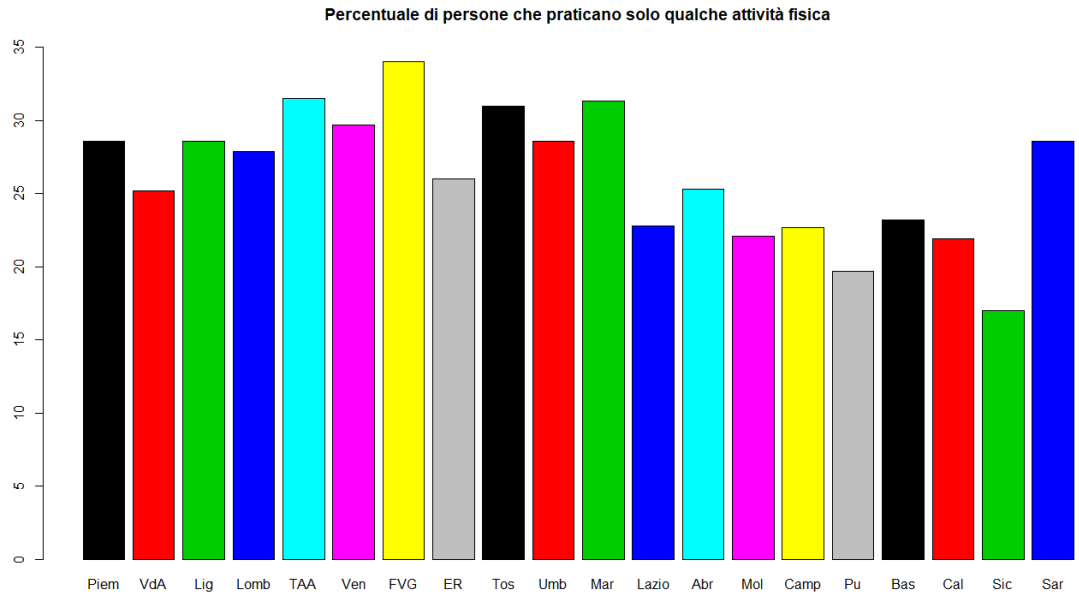
ed il grafico realizzato da R è il seguente:



- Per il vettore “solo_qualche_attivita”:

```
barplot(matriceDati[,3], col = 1:20, ylim = c(0,35),
        main = "Percentuale di persone che praticano
        solo qualche attività fisica")
```

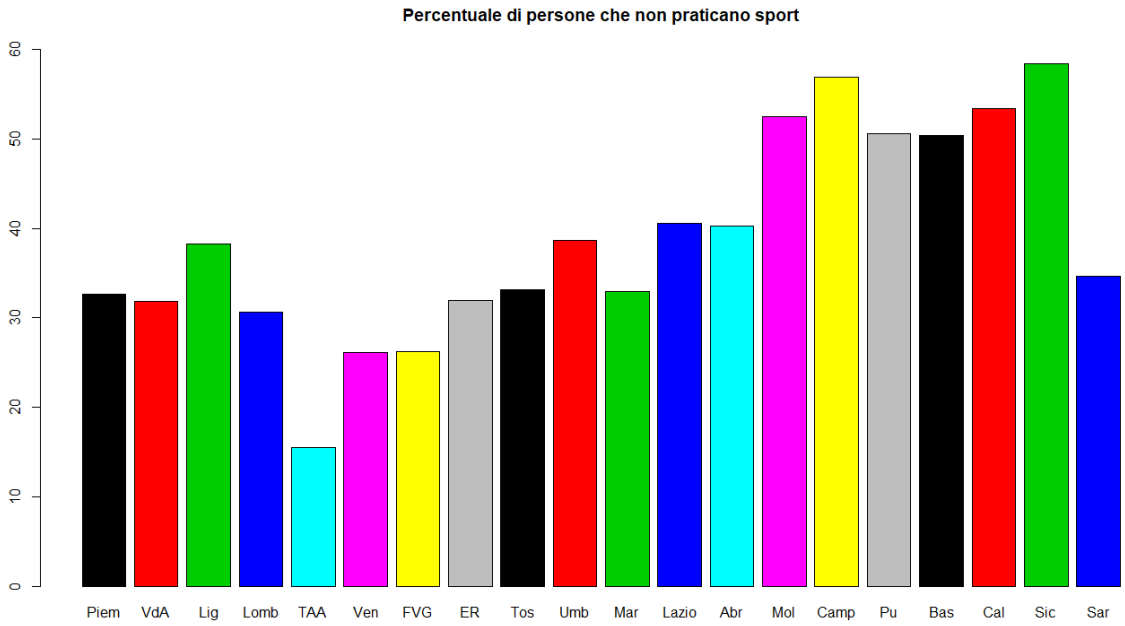
ed il grafico a barre che ne deriva è riportato sotto:



- Infine, è stato considerato il grafico per il vettore “non_praticano_sport”:

```
barplot(matriceDati[,4], col = 1:20, ylim = c(0,60),
        main = "Percentuale di persone che non praticano
        sport")
```

L'esecuzione del comando barplot() in R, ha prodotto il seguente grafico:



Nelle righe di codice R sopra riportate, la matrice dei dati viene utilizzata attraverso l'espressione *matriceDati[,i]* per indicare che, come vettore, viene considerata la colonna i-esima della matrice.

Grafici a torta

Dalla matrice estrapolata dal sito dell'ISTAT, è possibile ricavare i dati generalizzati per l'intera nazione. Infatti se consideriamo la nuova tabella con all'interno la riga che raccoglie i dati relativamente alla pratica sportiva in Italia e non nelle singole regioni italiale:

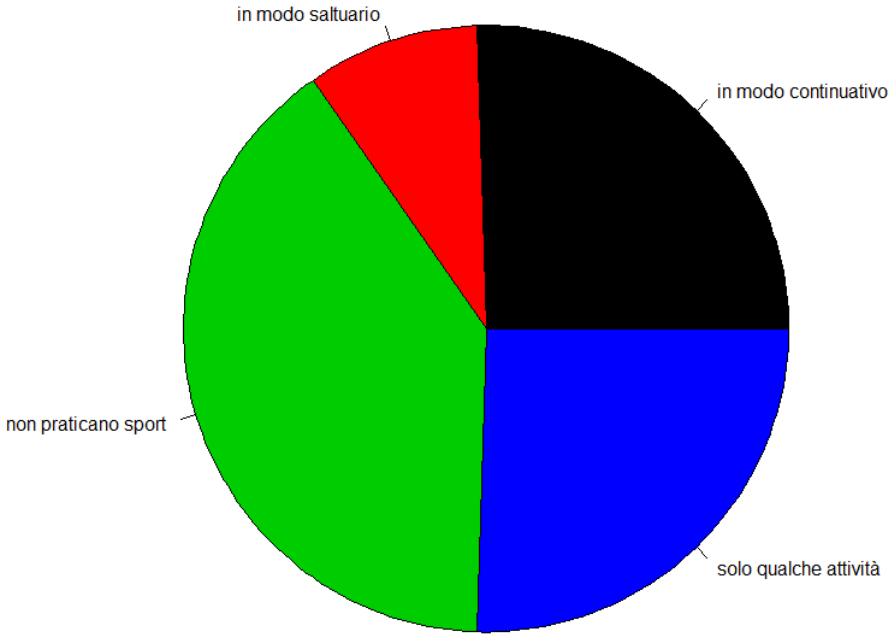
	praticano sport			non praticano sport, né attività fisica
	in modo continuativo	in modo saltuario	solo qualche attività fisica	
Italia	25,1	9,7	25,7	39,2
Piemonte	26,9	11,5	28,6	32,6
Valle d'Aosta	28,5	14,3	25,2	31,8
Liguria	24,9	7,8	28,6	38,3
Lombardia	30,5	10,7	27,9	30,6
Trentino Alto Adige	36,2	16,7	31,5	15,5
Veneto	29,5	14,4	29,7	26,1
Friuli-Venezia Giulia	27,6	11,9	34	26,2
Emilia-Romagna	31,1	10,8	26	31,9
Toscana	25,9	9,8	31	33,1
Umbria	24,6	8,1	28,6	38,7
Marche	27,6	8	31,3	32,9
Lazio	28,5	7,7	22,8	40,6
Abruzzo	23	11,2	25,3	40,3
Molise	19,3	5,8	22,1	52,5
Campania	13,9	6,1	22,7	56,9
Puglia	20,8	8,5	19,7	50,6
Basilicata	19	7,1	23,2	50,4
Calabria	16,5	7,7	21,9	53,4
Sicilia	16,5	7,5	17	58,4
Sardegna	26,1	10,7	28,6	34,7

Possiamo realizzare un *diagramma a torta* che ci permette di riassumere la situazione italiana. È quindi possibile stabilire, per l'anno 2016, se in Italia la maggior parte delle persone risulta non frequentare sport assiduamente o al contrario, se l'Italia può vantarsi di avere una popolazione di sportivi.

In R, i diagrammi a torta ci consentono di attribuire ciascuna modalità della variabile in esame ad un settore circolare di un cerchio, la cui ampiezza è proporzionale alle frequenze. Il sistema R realizza il diagramma a torta con i diversi settori colorati differentemente utilizzando il comando *pie(table())*. Il seguente codice

```
italia <- c(rep("in modo continuativo", 25.1),
            rep("in modo saltuario", 9.7),
            rep("solo qualche attività", 25.7),
            rep("non praticano sport", 39.2))
pie(table(italia), col = 1:4)
```

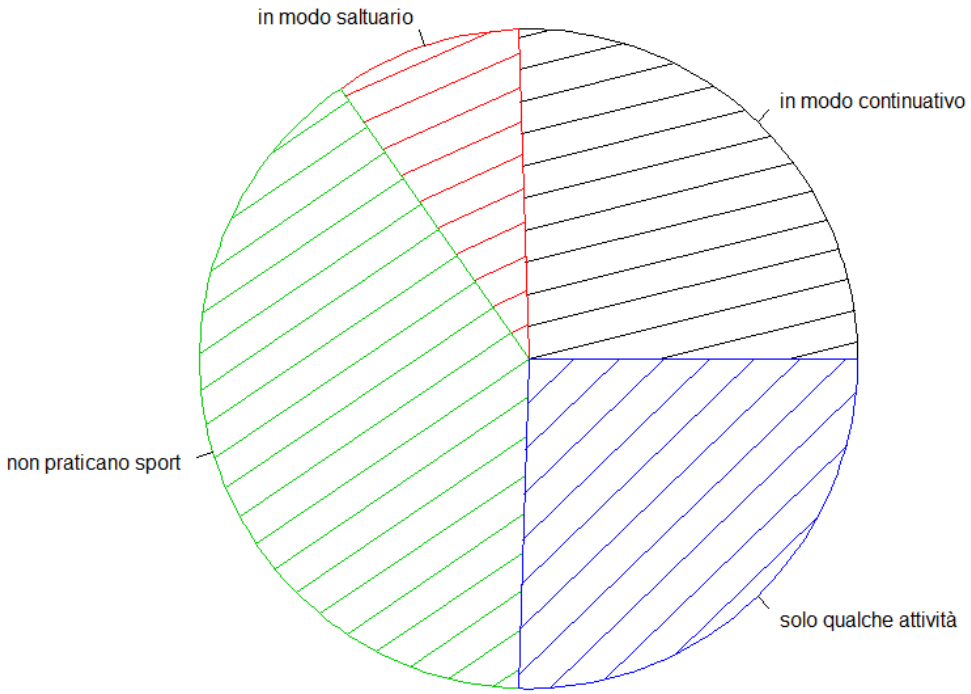
ha prodotto il diagramma a torta riportato di seguito:



Si può anche scegliere un tratteggio particolare da utilizzare nei diagrammi a torta per tratteggiare differentemente i diversi settori utilizzando i comandi *density* e *angle*. Riferendosi al precedente esempio il seguente codice

```
pie(table(italia), density = 4, angle = 4+10*(1:8), col = 1:4)
```

si ottiene il grafico riportato di seguito.



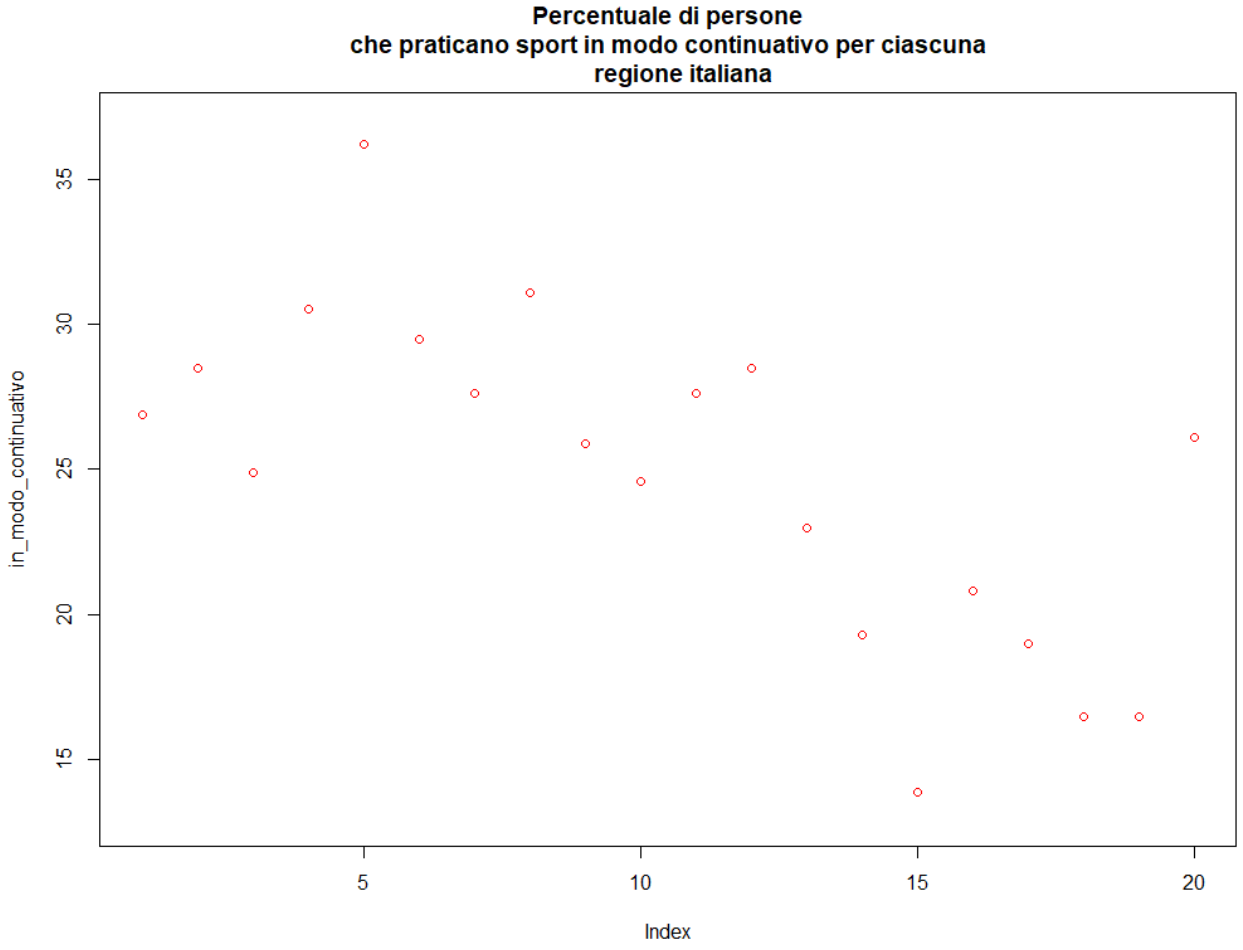
Dai grafici a torta ottenuti, si può evincere che in Italia è maggiore il numero di persone che non praticano sport, mentre minore è il numero di persone che praticano sport in modo saltuario.

La funzione plot()

Consideriamo ora la *variabile quantitativa* "in_modo_continuativo" e i valori numerici assunti dalle varie regioni inseriti in questo vettore precedentemente.
In questo caso la funzione *plot()* illustra l'andamento dei valori assunti dalla modalità di praticare sport presa in considerazione rispetto alle relative regioni. Tale comando, quindi, non fornisce informazioni significative circa la distribuzione di frequenza. Ad esempio, il seguente codice

```
plot(in_modo_continuativo, main="Percentuale di persone
che praticano sport in modo continuativo per ciascuna
     regione italiana", ylim = c(13,37), col = "red")
```

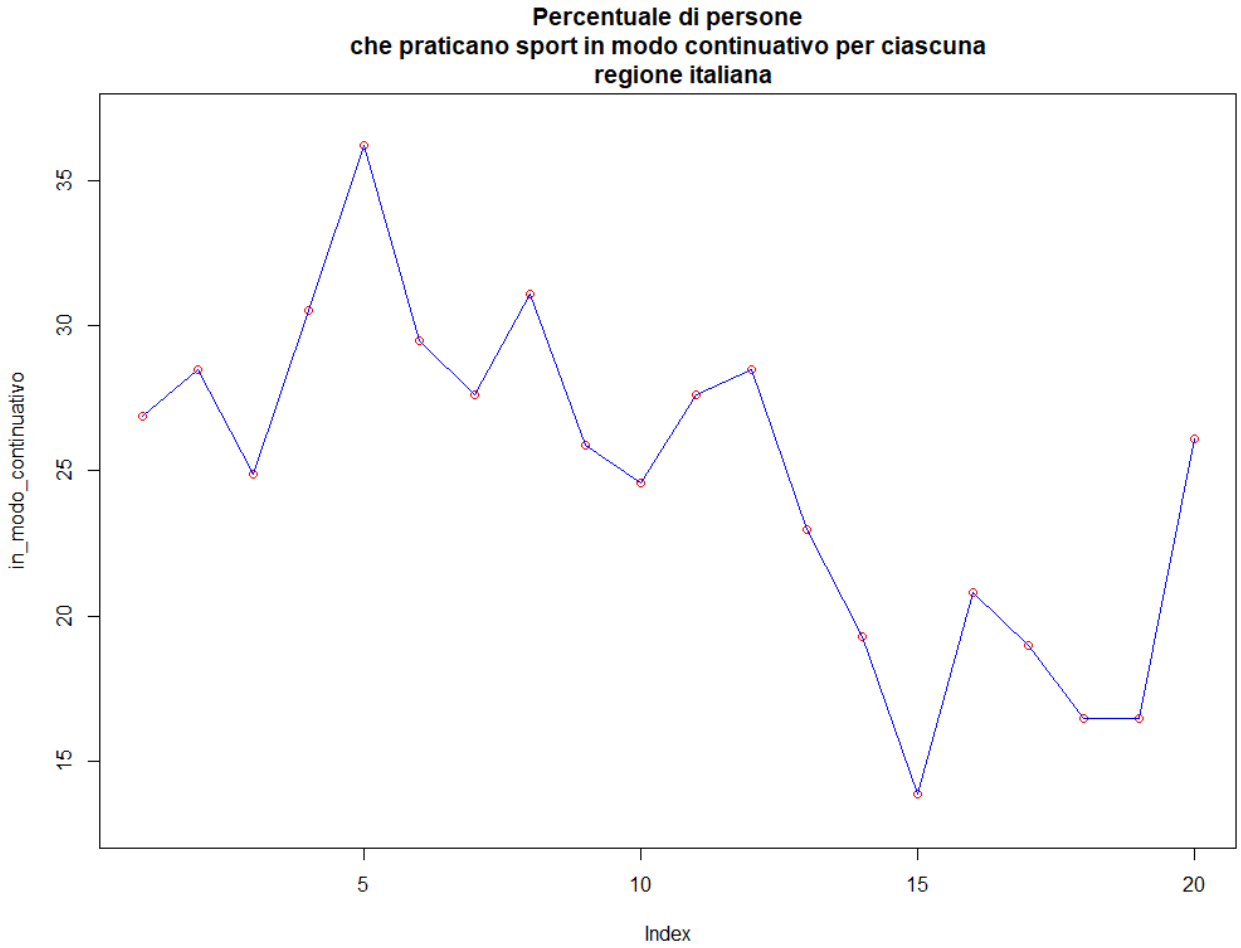
ha prodotto il grafico in figura, che illustra la percentuale di persone che praticano sport in modo continuativo per ognuna delle 20 regioni individuate dalle loro posizioni nel vettore.



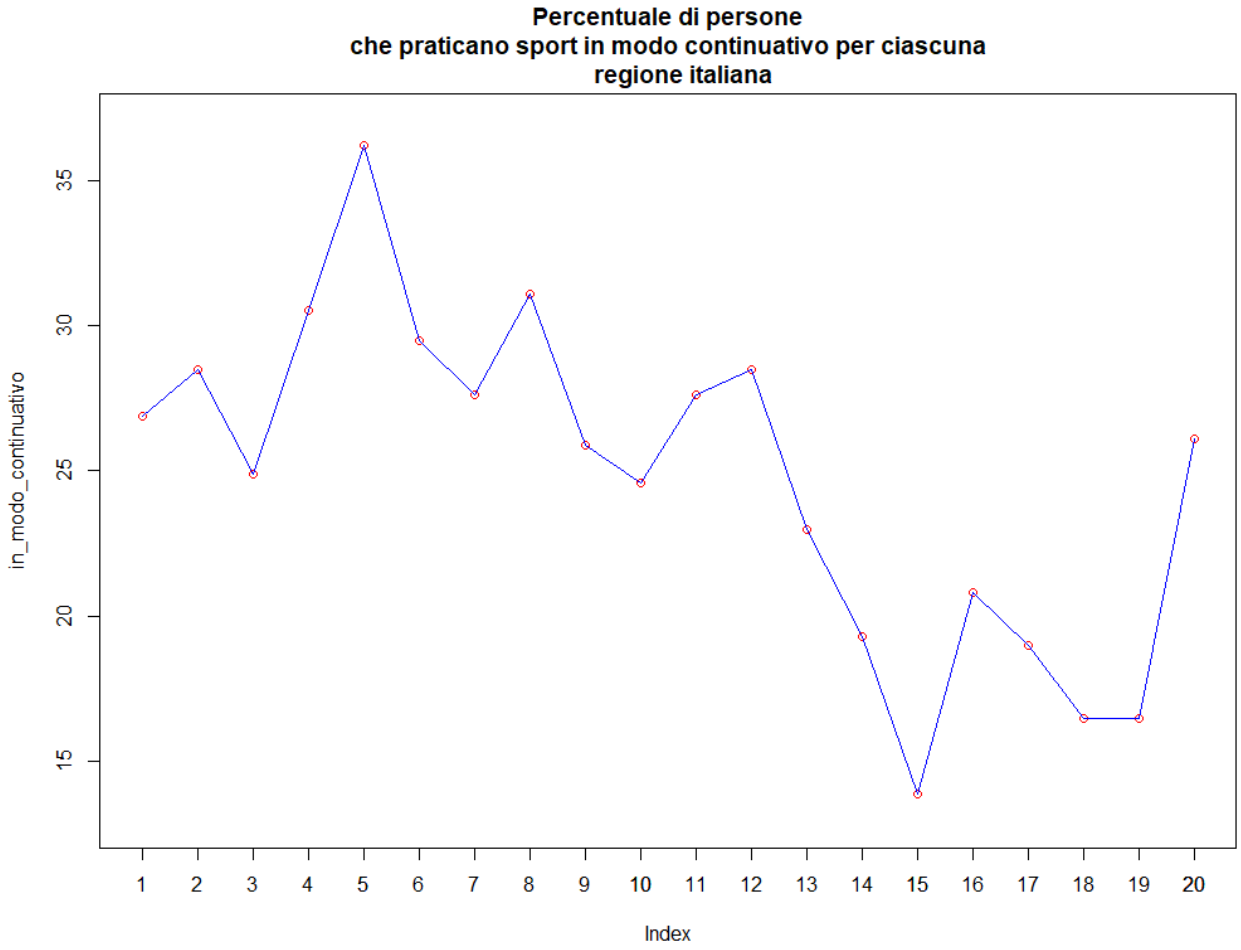
Vogliamo ora connettere mediante linee i punti in figura sopra riportata, creando una serie storica delle percentuali delle persone che praticano sport assiduamente. A tal fine, utilizziamo il comando di basso livello *lines()*

```
lines(in_modo_continuativo, col ="blue")
axis(1, 1:20)
box()
```

che produce il grafico in figura:



Dopo aver utilizzato il comando lines(), abbiamo ritenuto opportuno utilizzare anche il comando *axis(1, 1:20)* che si occupa di disegnare l'asse orizzontale in basso.
Mendiante axis() è possibile costruire il grafico che segue:



Lo stesso procedimento è stato ripetuto per tutti gli altri vettori che compongono la nostra matrice dei dati, ma come abbiamo già detto, il comando plot() non fornisce informazioni significative sulla distribuzione dei dati presi in considerazione.
Allora ci proponiamo di considerare gli istogrammi.

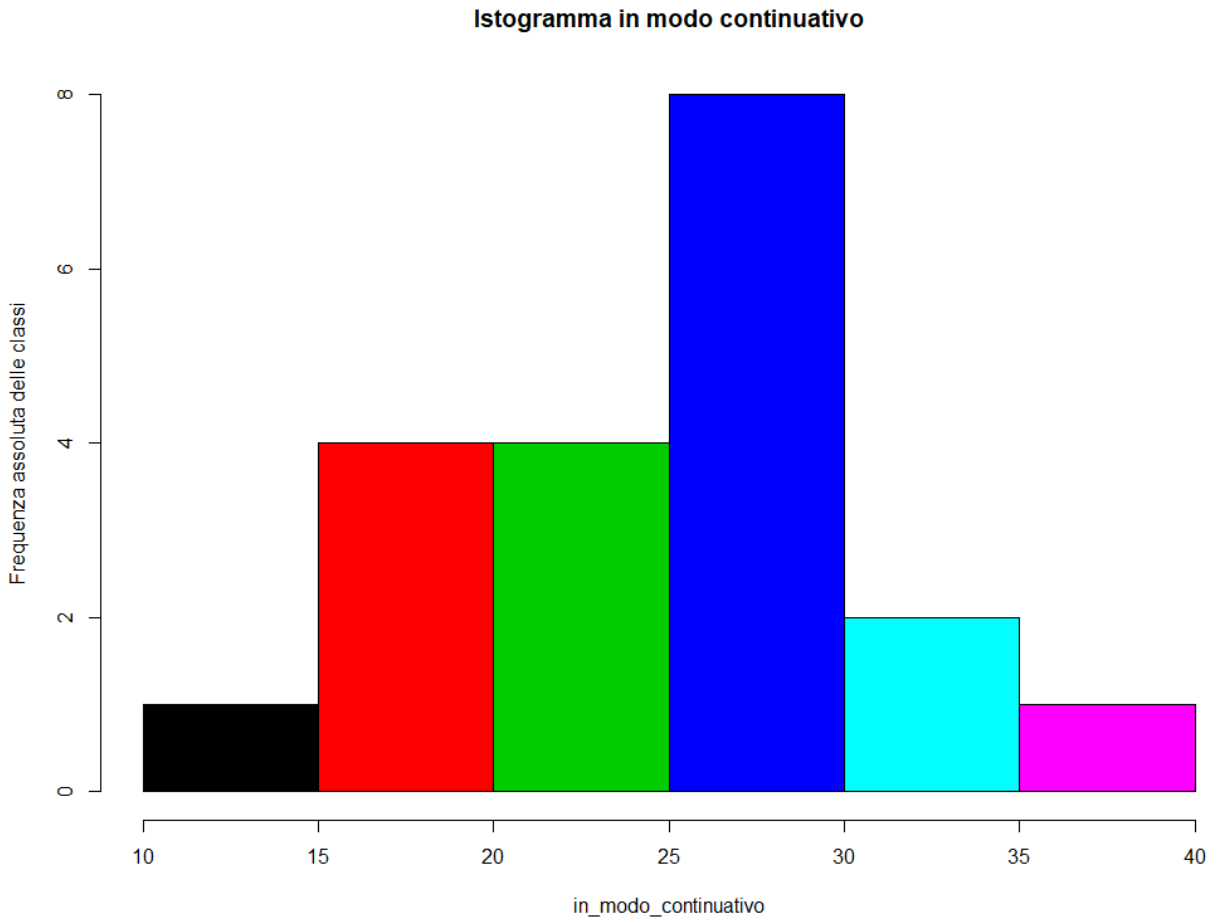
Istogrammi

Per rappresentare correttamente la distribuzione di frequenza dei nostri dati numerici è necessario considerare delle classi, come già fatto in precedenza.
Infatti, gli istogrammi, che si utilizzano per variabili quantitative, sono una particolare rappresentazione grafica di una distribuzione di frequenza in classi.
Consideriamo, quindi, i nostri campioni di dati (i vettori che compongono la matrice dei dati) che suddividiamo in classi; ogni osservazione deve cadere in una ed una sola classe (o intervallo).
Gli istogrammi sono quindi una particolare rappresentazione grafica ottenuta mediante rettangoli adiacenti aventi per basi segmenti i cui estremi corrispondono agli estremi delle classi. L'asse delle ascisse di un istogramma è quindi sempre dotato di un'unità di misura.
Fissate le basi, le altezze debbono essere tali che l'area di ogni rettangolo risultante sia uguale alla frequenza (relativa o assoluta) della classe stessa.
Se si utilizzano le frequenze assolute delle classi, l'area di ogni rettangolo è uguale alla frequenza assoluta della classe e l'area totale dei rettangoli è uguale all'ampiezza del campione. Quindi, l'area del rettangolo i-esimo è uguale a $n_i = b_i \times h_i$, dove n_i è il numero di

valori che cadono nella classe i-esima (frequenza assoluta della classe i-esima), b_i è la base e h_i è l'altezza della classe i-esima.
La funzione che realizza in R un istogramma è *hist()*. È possibile lasciare scegliere ad R il numero di classi da utilizzare, oppure suggerire ad R il numero di classi da utilizzare mediante il parametro *breaks*. Per realizzare un istogramma in base alle frequenze assolute, occorre impostare nella funzione hist() il parametro *freq* = *TRUE* (di default).
A questo punto, consideriamo il vettore "in_modo_continuativo" e realizziamo l'istogramma relativo alle frequenze assolute per tale vettore. Le seguenti linee di codice

```
freq_ass_cont <- hist(in_modo_continuativo, freq = TRUE,
                      main = "Istogramma in modo continuativo",
                      ylab="Frequenza assoluta delle classi", col=1:6)
```

producono l'istogramma rappresentato in figura



in cui il numero di classi è stato scelto automaticamente da R. Abbiamo utilizzato il parametro *main* per definire il titolo del grafico. Si nota che l'area totale è uguale all'ampiezza del campione.
La funzione hist() è in grado di generare oltre al grafico, anche una serie di informazioni sulla sua natura che possono essere salvate in un variabile di tipo list. Queste informazioni possono essere visualizzate utilizzando la funzione *str()* applicata alla variabile generata dalla funzione hist().
Riferendoci all'istogramma sopra riportato, scriviamo le seguenti linee di codice

```
> str(freq_ass_cont)
List of 6
 $ breaks  : int [1:7] 10 15 20 25 30 35 40
 $ counts  : int [1:6] 1 4 4 8 2 1
 $ density : num [1:6] 0.01 0.04 0.04 0.08 0.02 0.01
 $ mids    : num [1:6] 12.5 17.5 22.5 27.5 32.5 37.5
 $ xname   : chr "in_modo_continuativo"
 $ equidist: logi TRUE
 - attr(*, "class")= chr "histogram"
```

Come si vede la funzione str() fornisce i punti di suddivisione in classi (breaks), le frequenze assolute delle classi (counts), la densità delle classi (density) e i punti centrali delle classi (mids) come mostra il seguente codice:

```
> freq_ass_cont$breaks
[1] 10 15 20 25 30 35 40
> freq_ass_cont$counts
[1] 1 4 4 8 2 1
> freq_ass_cont$density
[1] 0.01 0.04 0.04 0.08 0.02 0.01
> freq_ass_cont$mids
[1] 12.5 17.5 22.5 27.5 32.5 37.5
```

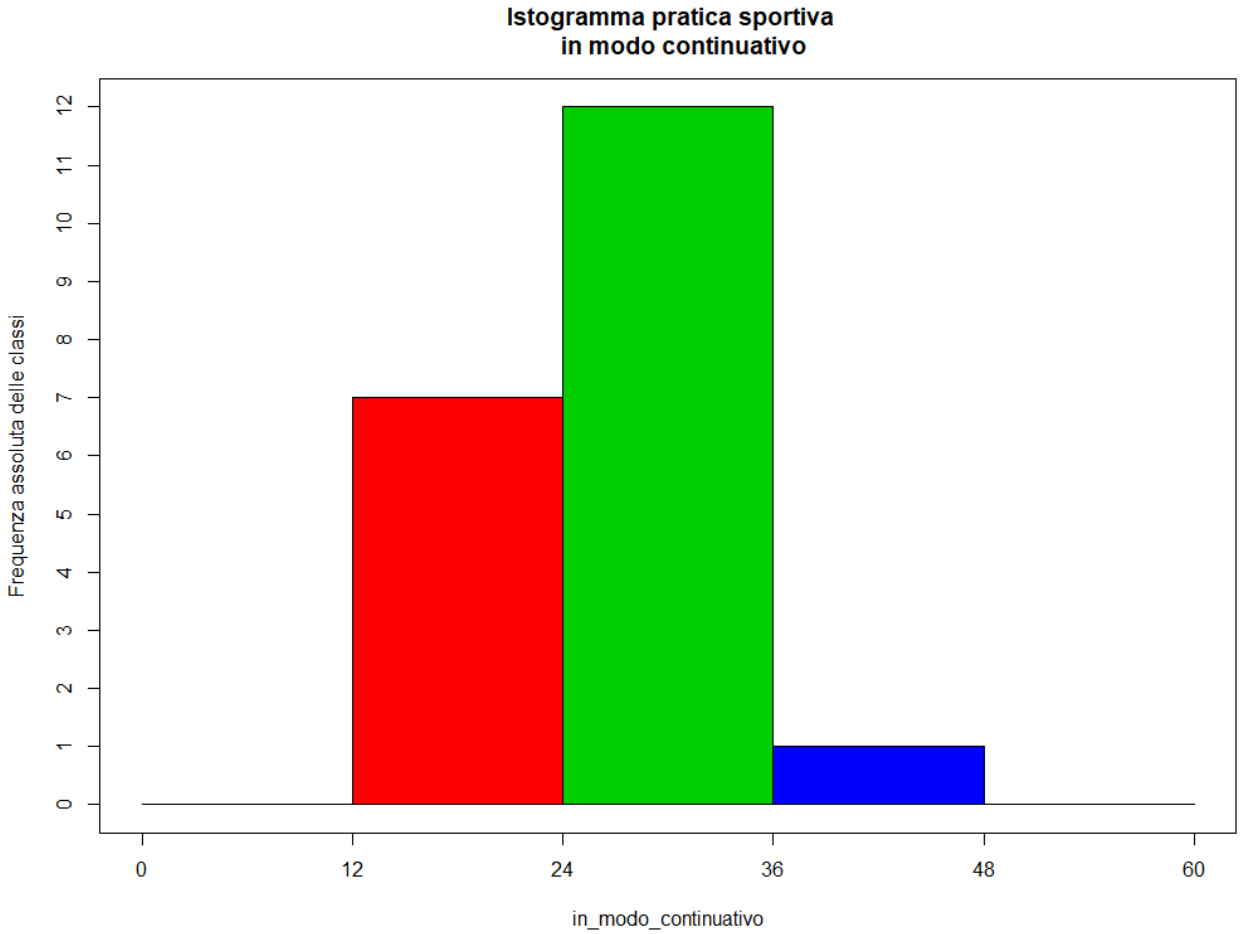
La suddivisione in classi scelta automaticamente da R è la seguente: (10,15] (15,20] (20,25] (25,30] (30,35] (35,40]. Infatti, dei 20 valori, 1 cade nella prima classe, 4 nella seconda classe, 4 nella terza classe, 8 nella quarta classe, 2 nella quinta classe e soltanto 1 nella sesta classe (frequenza assoluta delle classi).
Le frequenze relative associate alle sei classi possono essere ottenute moltiplicando gli elementi del vettore freq_ass_cont$density per 5 (ampiezza effettiva di ogni classe) e sono:

```
> freq_rel <- freq_ass_cont$density *5
> freq_rel
[1] 0.05 0.20 0.20 0.40 0.10 0.05
> sum(freq_rel)
[1] 1
```

Si noti che la somma delle frequenze relative associate alle classi è unitaria.
Supponiamo ora di fissare le classi dell'istogramma con breaks, considerando seguenti linee di codice

```
> classi <- c(0, 12, 24, 36, 48, 60)
hist(in_modo_continuativo, freq =TRUE ,
     main=" Istogramma pratica sportiva
     in modo continuativo", breaks = classi,
     ylab="Frequenza assoluta delle classi", col=1:5,
     axes= F)
axis(1, classi)
axis(2, 0:12)
box()
```

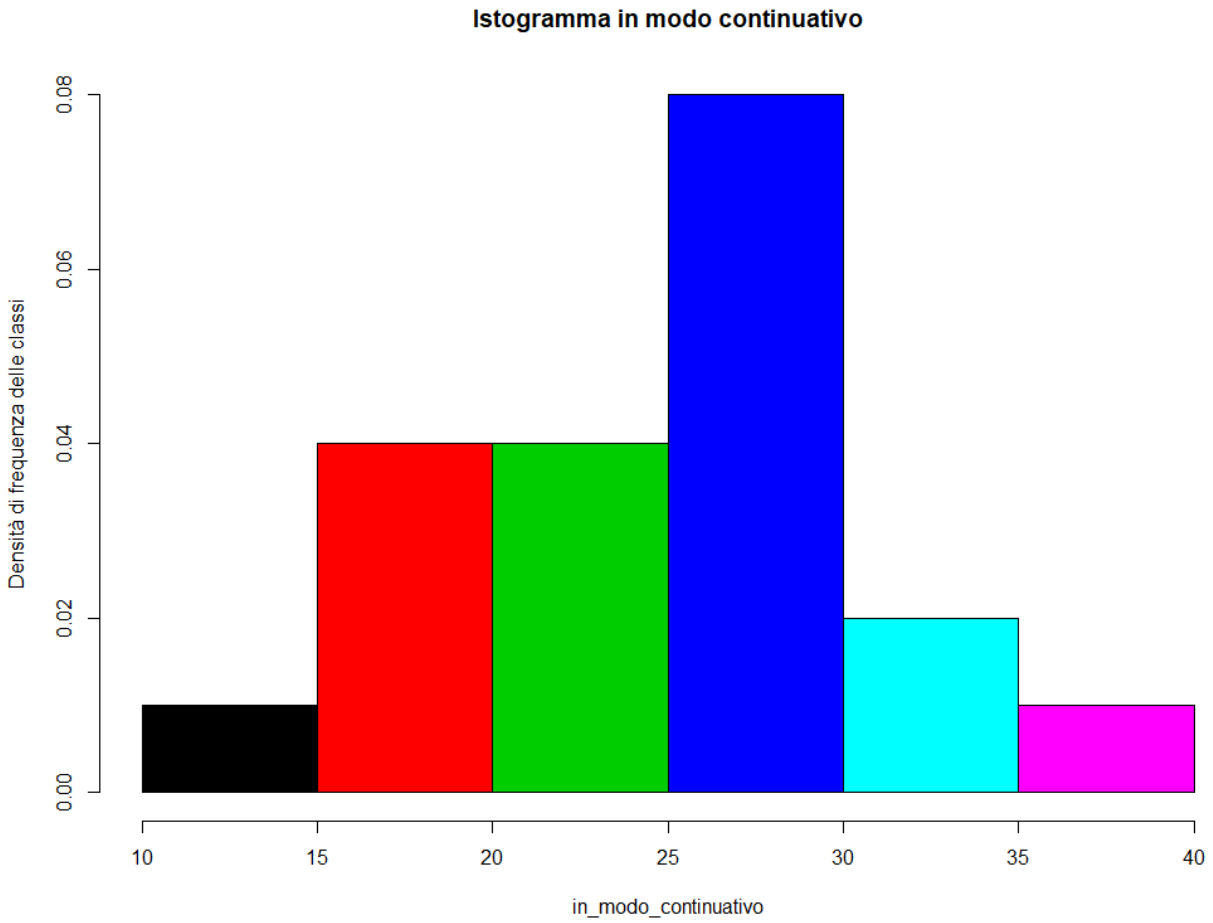
che produce invece l'istogramma visualizzato in figura. Gli intervalli unitari sono aperti a sinistra e chiusi a destra, ossia del tipo (k, k + 1] dove k = 0, 12, 24, 36, 48, 60. Si nota nuovamente che l'area totale è uguale all'ampiezza del campione.



Se si utilizzano le frequenze relative delle classi l'area di ogni rettangolo è uguale alla frequenza relativa della classe stessa e l'area totale è uguale all'unità.
Quindi, l'area del rettangolo i-esimo è uguale a $f_i = n_i/n = b_i \times h_i$, dove f_i è la frequenza relativa dei valori che della classe i-esima, b_i è l'ampiezza e h_i è l'altezza della classe i-esima. In questo caso l'altezza di ogni rettangolo esprime la densità di frequenza associata alla classe considerata.
Per realizzare un istogramma in base alle frequenze relative, occorre impostare nella funzione hist() il parametro *freq = FALSE*. Consideriamo, quindi, ancora una volta il vettore "in_modo_continuativo" e scriviamo le seguenti linee di codice

```
freq_rel_cont<- hist(in_modo_continuativo, freq = FALSE,
                     main = "Istogramma in modo continuativo",
                     ylab="Densità di frequenza delle classi", col=1:6)
```

produce l'istogramma rappresentato in figura in base alle densità di frequenza delle classi. In tal caso il numero di classi è stato scelto automaticamente da R e l'area totale `e unitaria.



Anche in questo caso la funzione hist() restituisce un elemento di tipo lista con gli stessi attributi del caso in cui viene considerata la frequenza assoluta.
Se consideriamo il codice seguente:

```
> str(freq_rel_cont)
List of 6
 $ breaks  : int [1:7] 10 15 20 25 30 35 40
 $ counts  : int [1:6] 1 4 4 8 2 1
 $ density : num [1:6] 0.01 0.04 0.04 0.08 0.02 0.01
 $ mids    : num [1:6] 12.5 17.5 22.5 27.5 32.5 37.5
 $ xname   : chr "in_modo_continuativo"
 $ equidist: logi TRUE
 - attr(*, "class")= chr "histogram"
> freq_rel_cont$breaks
[1] 10 15 20 25 30 35 40
> freq_rel_cont$counts
[1] 1 4 4 8 2 1
> freq_rel_cont$density
[1] 0.01 0.04 0.04 0.08 0.02 0.01
> freq_rel_cont$mids
[1] 12.5 17.5 22.5 27.5 32.5 37.5
> frequenza_relativa <- freq_rel_cont$density * 5
> frequenza_relativa
[1] 0.05 0.20 0.20 0.40 0.10 0.05
> sum(frequenza_relativa)
[1] 1
```

si nota che freq_rel_cont$density fornisce le altezze dei rettangoli associate alle 6 classi.
Abbiamo ripetuto poi il procedimento per gli altri vettori: "in_modo_saltuario", "solo_qualche_attivita" e "non_praticano_sport".

Boxplot

Consideriamo il campione "in_modo_continuativo". Se ordiniamo i valori del campione in ordine crescente possiamo affermare che: si chiama primo quartile, e si indica con Q_1, il valore per il quale il 25% dei dati sono alla sua sinistra e il restante 75% alla sua destra. Analogamente si chiama terzo quartile, e si indica con Q_3, il valore per il quale il 75% dei dati sono alla sua sinistra e il restante 25% alla sua destra. Il secondo quartile Q_2, ossia il valore per il quale 50% dei dati sono alla sua sinistra e il restante 50% è alla sua destra è detto mediana. Q_0 e Q_4 forniscono il minimo e il massimo dei valori del campione.
In R i quartili si calcolano tramite la funzione *quantile(nomeVettore)* e la funzione *summary(nomeVettore)* permette di determinare i valori precisi del minimo, del massimo, della media, della mediana, del primo e del terzo quartile.
Ad esempio, riferendoci al vettore "in_modo_continuativo" risulta:

```
> quantile(in_modo_continuativo)
    0%    25%    50%    75%   100%
13.900 20.425 26.000 28.500 36.200
```

da cui si deduce che $Q_0 = 13.9$, $Q_1 = 20.425$, $Q_2 = 26$, $Q_3 = 28.5$, $Q_4 = 36.2$. Inoltre, si ha

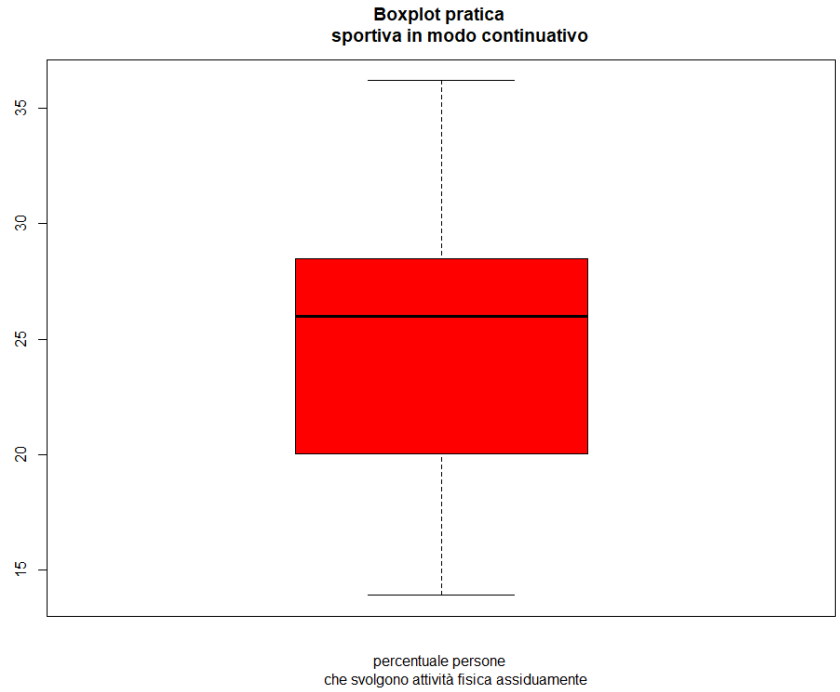
```
> summary(in_modo_continuativo)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  13.90   20.43   26.00   24.84   28.50   36.20
```

A questo punto possiamo definire cosa s'intende per *boxplot*. Il boxplot, detto anche *scatola con baffi*, è il disegno di una scatola i cui estremi sono Q_1 e Q_3, tagliata da una linea orizzontale in corrispondenza di Q_2, ossia della mediana. In basso e in alto sono presenti altre due linee orizzontali, dette i *baffi*. Il baffo inferiore corrisponde al *valore più piccolo tra le osservazioni* che risulta maggiore o uguale di $a=Q_1-1.5\cdot(Q_3-Q_1)$, mentre il baffo superiore corrisponde al *valore più grande delle osservazioni* che risulta minore o uguale a $b=Q_3+1.5\cdot(Q_3-Q_1)$. La distanza tra il primo e il terzo quartile è detta intervallo interquartile o *scarto interquartile*. Quindi, se tutti i dati rientrano nell'intervallo (a,b), i baffi sono posti in corrispondenza del minimo valore e del massimo valore del campione. Gli eventuali valori al di fuori dell'intervallo (a,b) sono visualizzati nel grafico sotto forma di punti, detti *valori anomali* o *outlier* (valori anomali perché si discostano dagli altri valori all'interno del campione). Questi valori infatti costituiscono una "anomalia" rispetto alla maggior parte dei valori osservati e pertanto è necessario identificarli per poterne analizzare le caratteristiche e le eventuali cause che li hanno determinati. Il boxplot viene utilizzato per illustrare alcune caratteristiche di una distribuzione di frequenza: la centralità, la forma, la dispersione e la presenza di eventuali valori anomali, detti "outlier". La centralità è espressa dalla mediana.
La forma simmetrica o asimmetrica può essere dedotta esaminando le distanze del primo e del terzo quartile dalla linea mediana. Infatti, se $Q_3 - Q_2$ è più o meno simile alla distanza $Q_2 - Q_1$ allora possiamo dire che la mediana si trova più o meno al centro e quindi esiste la simmetria dei dati.
I baffi, superiore e inferiore, forniscono informazioni sulla dispersione e sulla forma della distribuzione ed anche sulle code della distribuzione. Infatti, la dispersione è deducibile esaminando le distanze del baffo superiore da Q3 e del baffo inferiore da Q1.

In R un boxplot è ottenuto tramite la funzione *boxplot(nomeVettore)*. La figura può essere disegnata sia in senso verticale che orizzontale sulla base del parametro logico *horizontal* che ovviamente è FALSE (per default) nel caso verticale.
Nel nostro caso, prendiamo in esame il vettore "in_modo_continuativo" e costruiamo il boxplot ad esso corrispondente attraverso le seguenti linee di codice

```
boxplot(in_modo_continuativo, main="Boxplot pratica
        sportiva in modo continuativo", xlab = "percentuale persone
        che svolgono attività fisica assiduamente", col =  "red")
```

che producono il boxplot nelle figura riportata di seguito.

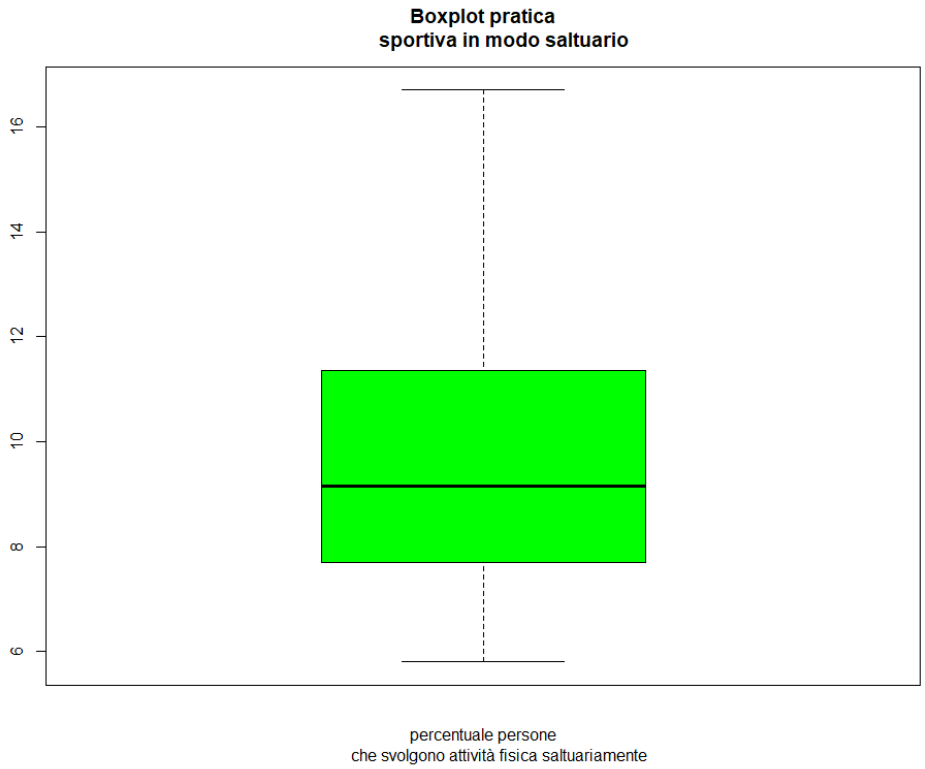


Si nota che gli estremi della scatola sono Q1 = 20.43 e Q3 = 28.5; il boxplot è tagliato da una linea orizzontale in corrispondenza di Q2 = 26. Il baffo inferiore corrisponde al valore più piccolo tra le osservazioni che risulta maggiore o uguale di $Q1 - 1.5 \cdot (Q3 - Q1) = 8.33$, ossia 13.9, mentre il baffo superiore corrisponde al valore più grande delle osservazioni che risulta minore o uguale a $Q3 + 1.5 \cdot (Q3 - Q1) = 40.6$, ossia 36.2. Poiché tutti i valori cadono nell'intervallo [8.33, 40.6] i baffi sono posti in corrispondenza del minimo 13.9 e del massimo 36.2. Possiamo anche affermare che tra i dati non esiste simmetria in quanto la differenza $Q_3 - Q_2$ è più piccola rispetto alla distanza $Q_2 - Q_1$. Visivamente questa deduzione è immediata in quanto la mediana non è posta al centro della scatola.
Mostriamo ora i boxplot per tutti gli altri vettori che compongono la nostra matrice dei dati. In generale, di seguito riportiamo il codice R ed il boxplot che ne consegue.

- Per il vettore "in_modo_saltuario" si ha:

```
> quantile(in_modo_saltuario)
    0%    25%    50%    75%   100%
 5.800  7.700  9.150 11.275 16.700
> summary(in_modo_saltuario)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  5.800   7.700   9.150   9.815  11.275  16.700
> boxplot(in_modo_saltuario, main="Boxplot pratica
+         sportiva in modo saltuario", xlab = "percentuale persone
+         che svolgono attività fisica saltuariamente", col =  "green")
```

Il codice seguente produce il boxplot nella figura successiva

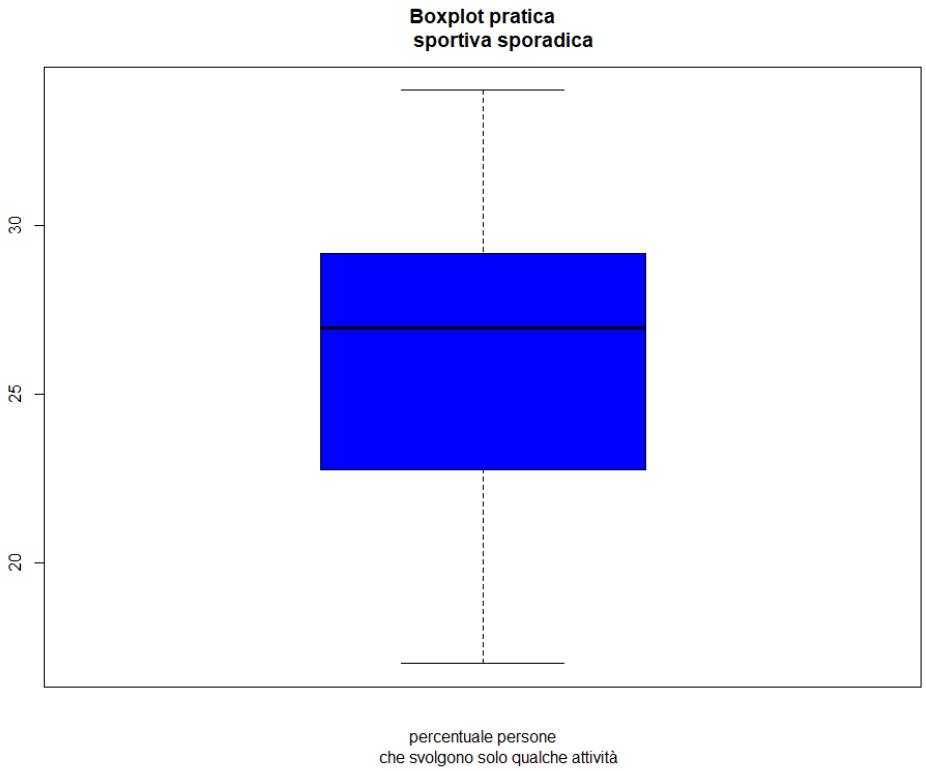


Dal boxplot in figura è evidente che la distanza tra il terzo quartile e il baffo superiore è abbastanza grande, ciò vuol dire che maggiore è la dispersione dei dati.

- Per il vettore “solo_qualche_attivita”, consideriamo le linee di codice seguenti

```
> quantile(solo_qualche_attivita)
    0%    25%    50%    75%   100%
17.000 22.775 26.950 28.875 34.000
> summary(solo_qualche_attivita)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  17.00   22.77   26.95   26.29   28.88   34.00
> boxplot(solo_qualche_attivita, main="Boxplot pratica
+         sportiva sporadica", xlab = "percentuale persone
+         che svolgono solo qualche attività", col =  "blue")
```

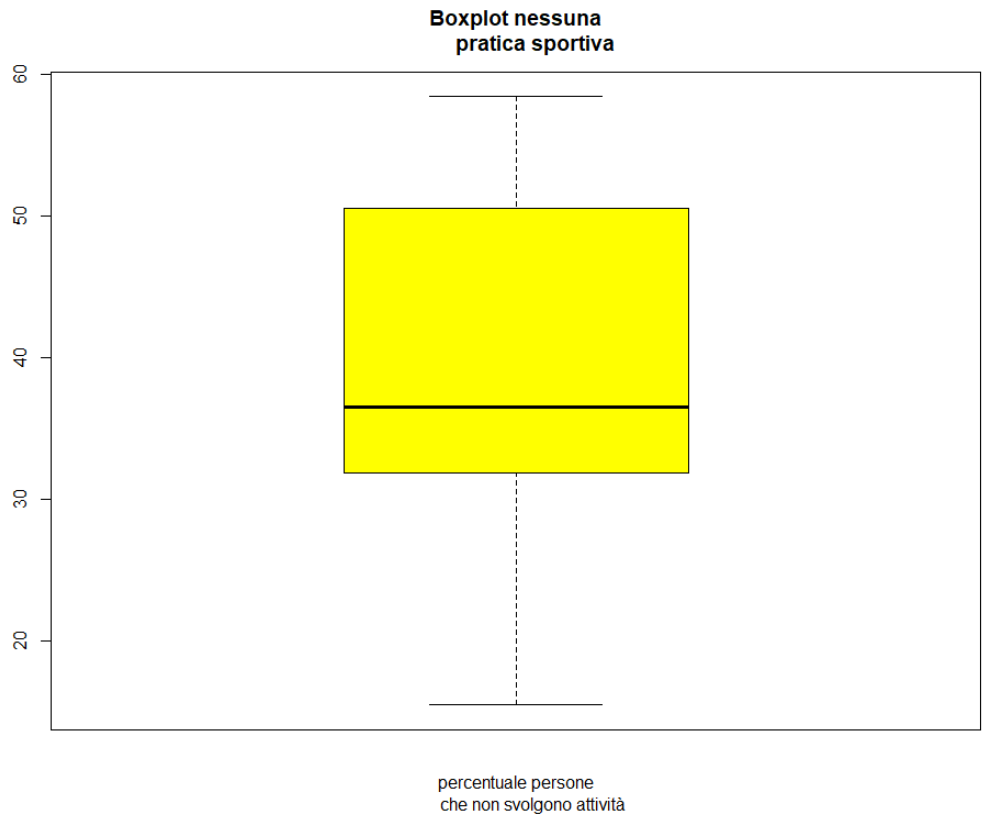
che hanno generato il boxplot in figura



- Infine, consideriamo il vettore “non_praticano_sport”, e scriviamo le seguenti linee di codice

```
> quantile(non_praticano_sport)
    0%    25%    50%    75%   100%
15.500 31.875 36.500 50.450 58.400
> summary(non_praticano_sport)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  15.50   31.88   36.50   38.77   50.45   58.40
> boxplot(non_praticano_sport, main="Boxplot nessuna
+         pratica sportiva ", xlab = "percentuale persone
+         che non svolgono attività", col =  "yellow")
> |
```

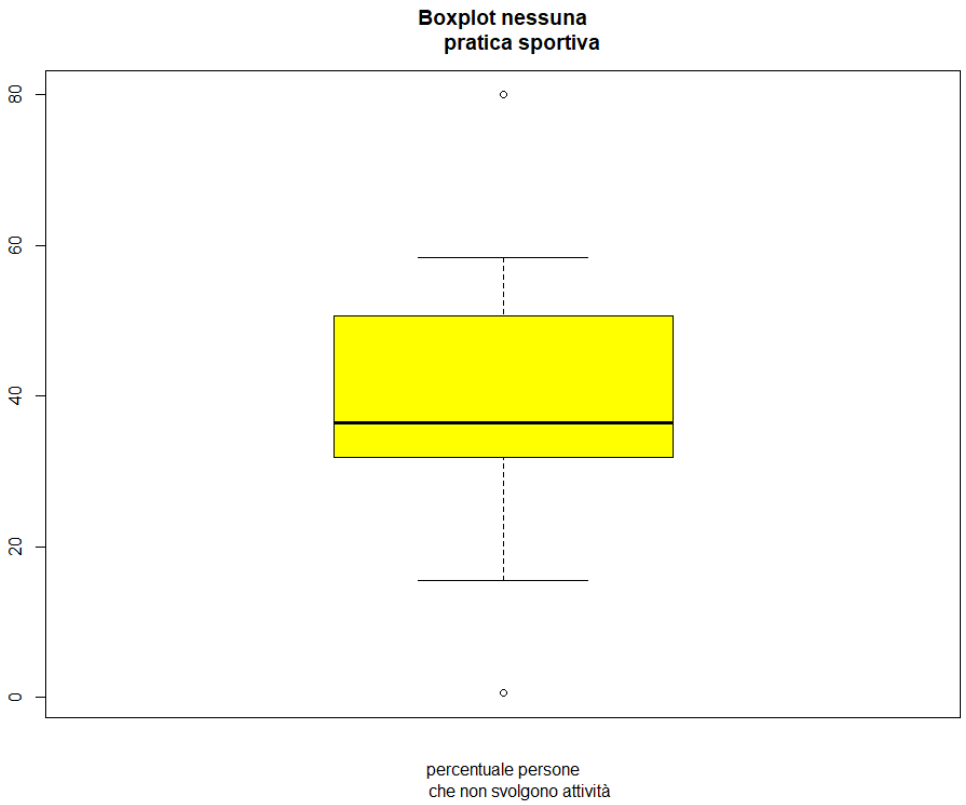
Il risultato è il boxplot in figura sottostante



L'analisi dei quattro boxplot ci porta sicuramente a dire che in nessuno dei vettori esiste simmetria in quanto la mediana non si trova mai al centro della scatola. Inoltre, è possibile affermare che in nessun caso si hanno valori anomali.
A tal proposito consideriamo l'ultimo vettore "non_praticano_sport" ed inseriamo dei valori che siano al di fuori del range $[Q1 - 1.5 \cdot (Q3 - Q1), Q3 + 1.5 \cdot (Q3 - Q1)]$. Se calcoliamo la quantità $Q3 + 1.5 \cdot (Q3 - Q1)$ otteniamo che essa è uguale a 4.025 e quindi scegliamo come valore 0.5. Al contrario, il valore $Q3 + 1.5 \cdot (Q3 - Q1)$ è pari a 78.3 perciò scegliamo il valore 80.
Dalle seguenti linee di codice

```
> non_praticano_sport2 <- c(non_praticano_sport, 0.5, 80)
> quantile(non_praticano_sport2)
    0%    25%    50%    75%   100%
 0.500 31.825 36.500 50.550 80.000
> summary(non_praticano_sport2)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   0.50   31.82   36.50   38.91   50.55   80.00
> boxplot(non_praticano_sport2, main="Boxplot nessuna
+         pratica sportiva ", xlab = "percentuale persone
+         che non svolgono attività", col =  "yellow")
```

è possibile costruire il boxplot in figura contenente dei valori anomali, fuori dal range, rappresentati sottoforma di punti.

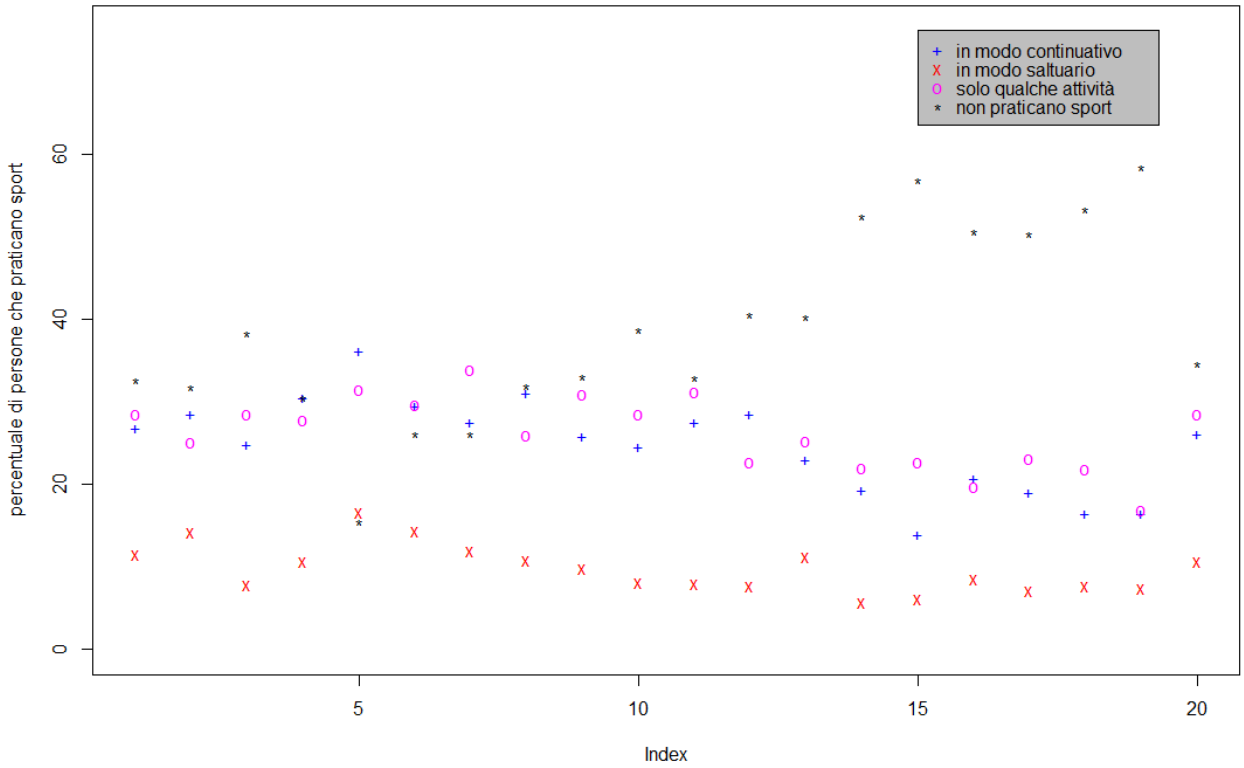


Confrontare variabili

Per confrontare differenti variabili che descrivono insiemi di dati numerici di uno stesso fenomeno quantitativo, è possibile costruire un grafico che contiene i diversi boxplot delle distribuzioni associate alle diverse variabili.
Prima però di procedere in questo modo, possiamo rappresentare in uno stesso grafico la percentuale di persone a seconda della modalità con cui praticano o non praticano sport, associando simboli diversi a ciascun vettore, ossia: "+" per indicare il vettore "in_modo_continuativo", "x" per indicare il vettore "in_modo_saltuario", "o" per indicare il vettore "solo_qualche_attività" e infine utilizziamo il simbolo "*" per indicare il vettore "non_praticano_sport". A tal fine, le seguenti linee di codice,

```
plot(in_modo_continuativo ,pch ="+",ylim=c(0,75)
     ,ylab="percentuale di persone che praticano sport",col="blue")
points(in_modo_saltuario,pch ="x",col ="red ")
points(solo_qualche_attivita, pch ="o",col ="magenta")
points(non_praticano_sport, pch ="*",col ="black")
legend (15,75, c("in modo continuativo","in modo saltuario",
                 "solo qualche attività","non praticano sport"),
        pch=c("+","x","o","*"),col =c("blue","red ","magenta", "black"),
        bg="gray ",cex =1)
```

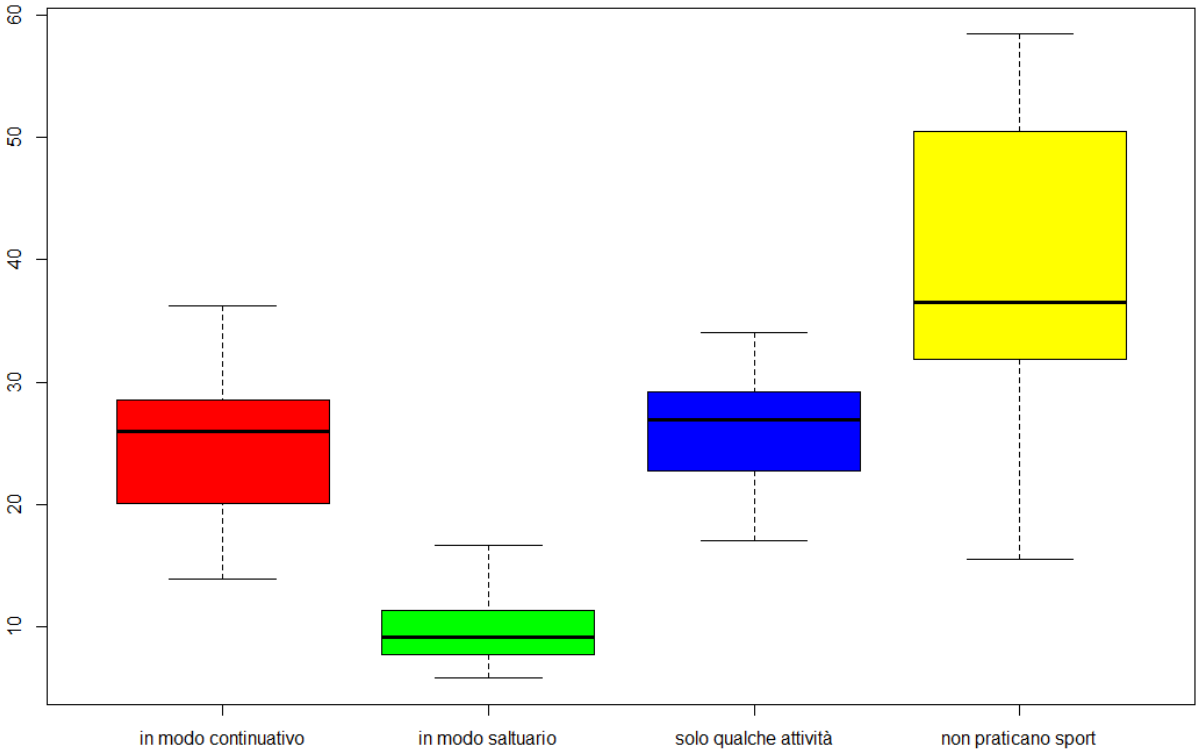
producono il grafico illustrato in figura.



La funzione *points()* permette di aggiungere punti al grafico precedentemente rappresentato, il parametro *pch* individua il tipo di carattere da utilizzare, il parametro *bg* indica quale colore di sfondo utilizzare e il parametro *cex* indica la grandezza del testo e dei simboli generati.
Realizziamo ora un grafico che permetta di confrontare i boxplot delle distribuzioni dei quattro vettori che compongono la nostra matrice dei dati. A tal fine, le seguenti linee di codice

```
boxplot(in_modo_continuativo, in_modo_saltuario, solo_qualche_attivita,
        non_praticano_sport, names =c("in modo continuativo","in modo saltuario",
                                      "solo qualche attività", "non praticano sport"),
        col =c("red","green", "blue", "yellow"))
```

produce il grafico illustrato in figura che consiste in quattro boxplot per il vettore “in_modo_continuativo”, “in_modo_saltuario”, “solo_qualche_attività” e “non_praticano_sport”.



Se utilizziamo la funzione summary() possiamo visualizzare le principali misure statistiche:

```
> summary(in_modo_continuativo)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  13.90   20.43   26.00   24.84   28.50   36.20
> summary(in_modo_saltuario)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  5.800   7.700   9.150   9.815  11.275  16.700
> summary(solo_qualche_attivita)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  17.00   22.77   26.95   26.29   28.88   34.00
> summary(non_praticano_sport)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  15.50   31.88   36.50   38.77   50.45   58.40
>
```

dalle quali è possibile dedurre che per ciascuna regione la percentuale di persone praticano sport in modo continuativo non va oltre 36.20%. Al contrario, la percentuale di persone che non praticano sport, può toccare picchi elevatissimi come il 58.40%. Dalla tabella, possiamo evincere che la regione che ha il maggior numero di persone che risultano rifiutare in generale la pratica sportiva è la Sicilia.
Inoltre, è evidente che l'ultimo vettore presenta una maggiore variabilità tra il minimo e il massimo.

Scatterplot

Le relazioni tra variabili quantitative possono essere rappresentate graficamente mediante diagrammi di dispersione chiamati anche scatterplot in cui ogni coppia di osservazioni viene

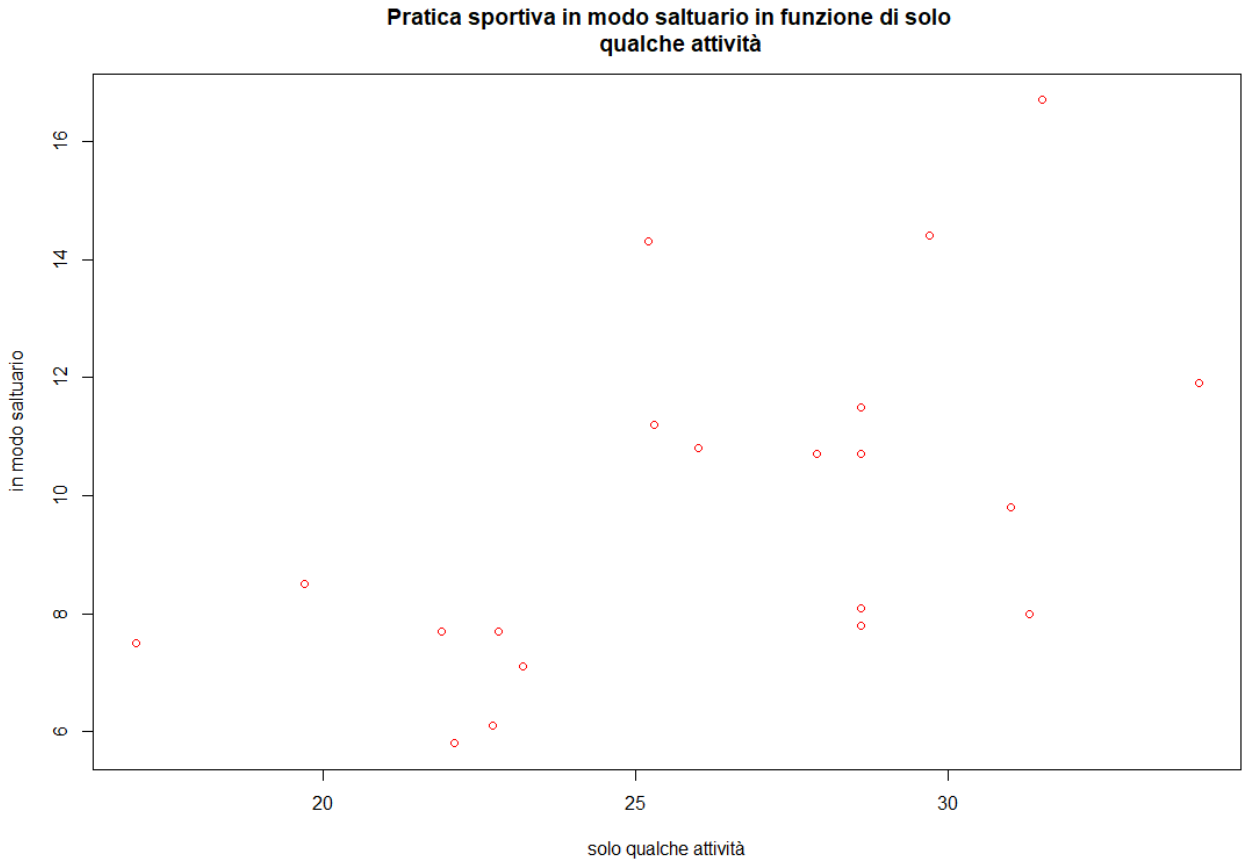
rappresentata sotto forma di un punto in un piano euclideo. Dopo aver scelto la variabile da porre sulle ascisse (variabile indipendente) e la variabile da porre sulle ordinate (variabile dipendente), si disegnano dei punti in corrispondenza delle coppie. Il risultato finale è una nuvola di punti che può essere ottenuto con la funzione *plot(x, y)*. Il grafico che si ottiene mira ad evidenziare se le coppie di punti presentano qualche forma di regolarità. Inoltre, il grafico di dispersione tende ad evidenziare se esiste una relazione tra le variabili e di che tipo è tale relazione (lineare, quadratica, . . .).
Supponiamo di voler analizzare due colonne di dati presenti nella tabella iniziale che riguardano le percentuali di persone che praticano solo qualche attività e le percentuali di persone che praticano sport in maniera saltuaria.
Le seguenti linee di codice

```
dataFrame <- data.frame(in_modo_continuativo, in_modo_saltuario,
                        solo_qualche_attivita, non_praticano_sport)
```

definiscono un data frame contenente le quattro modalità di pratica sportiva delle 20 regioni italiane.
Adesso, realizziamo lo scatterplot considerando la variabile “solo_qualche_attività” come variabile indipendente, e la variabile “in_modo_saltuario” come variabile dipendente, disegnando 20 punti in uno spazio euclideo attraverso le seguenti linee di codice:

```
plot(dataFrame$solo_qualche_attivita, dataFrame$in_modo_saltuario,
     main =" Pratica sportiva in modo saltuario in funzione di solo
     qualche attività", ylab="in modo saltuario", xlab="solo qualche attività",
     col ="red ")
```

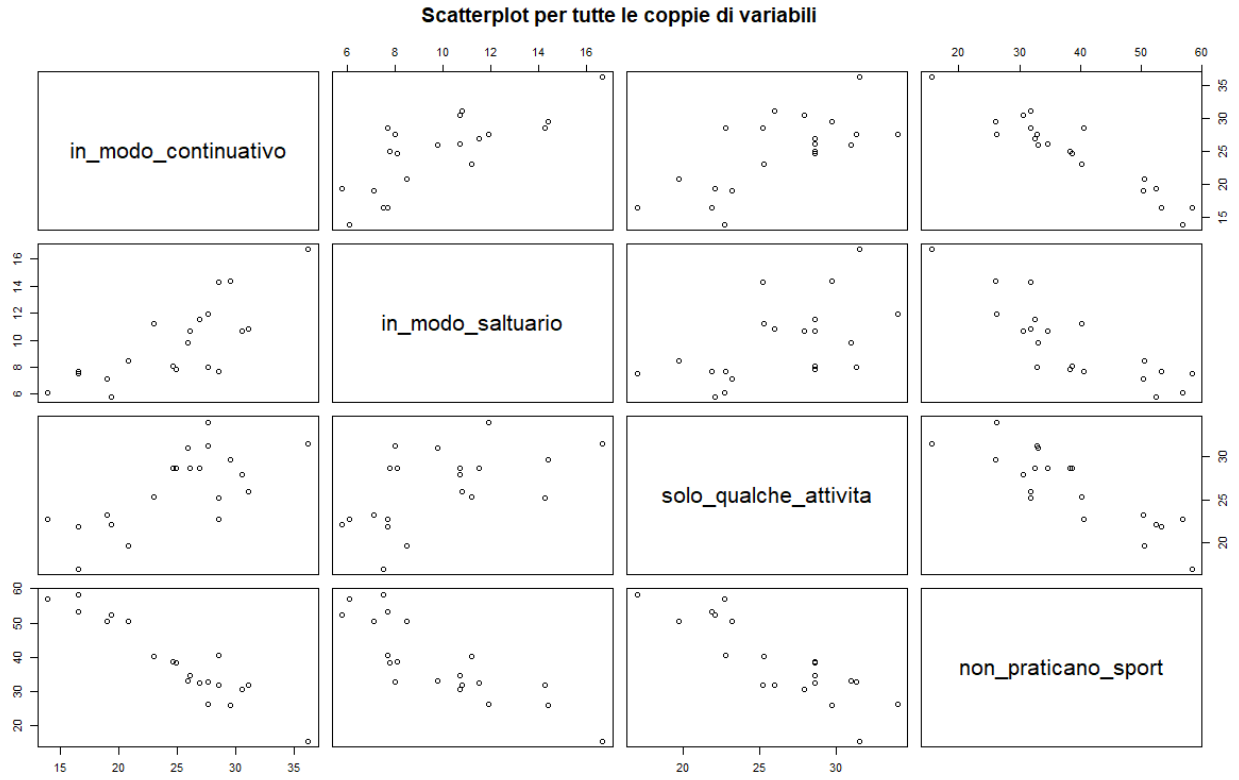
che producono il grafico nella figura sottostante.



Possiamo poi graficare tutte le relazioni possibili tra le variabili all'interno di un dataframe (o di una matrice). Tutto ciò è possibile utilizzando la funzione pairs() che permette di visualizzare in un unico grafico tutti gli scatterplot ottenuti mettendo in relazione tutte le possibili coppie di variabili.
Infatti, le seguenti linee di codice:

```
pairs(dataFrame, main="Scatterplot per tutte le coppie di variabili")
```

produce il grafico visualizzato in figura.



Le diverse immagini illustrano le nuvole di punti che si ottengono prendendo in considerazione tutte le differenti coppie di variabili.
Notiamo che una qualche regolarità è presente, quasi per tutte le coppie di variabili.
In generale, è possibile, però, affermare che i punti sembrano essere maggiormente sparsi nel caso "solo qualche attività-in modo saltuario".
Nel seguito ci soffermeremo nel valutare se esiste una relazione tra le variabili "non_praticano_sport" e "in_modo_continuativo" e in caso positivo studieremo che tipo di dipendenza esiste tra di esse.
Studieremo la regressione per questa coppia di variabili quando tratteremo nelle prossime pagine la statistica bivariata.

Statistica descrittiva univariata

Nel caso in cui avessimo a disposizione variabili aleatorie, avremmo sicuramente a che fare con una funzione di distribuzione che la descrive. In tal caso, infatti, potremmo utilizzare il valor medio e la varianza della variabile aleatoria come valori di sintesi.

Ma noi ovviamente, abbiamo a disposizione un insieme di dati numerici. Quindi, per i fenomeni numerici, è utile definire la *funzione di distribuzione empirica*. Precisiamo che non si parla di funzione di distribuzione teorica perché non abbiamo a che fare con variabili aleatorie.
La funzione di distribuzione può essere di due tipi.

Funzione di distribuzione empirica discreta.

La funzione di distribuzione empirica discreta che viene utilizzata quando il nostro campione di dati risulta avere k valori distinti ordinati in ordine crescente.
Andiamo poi a considerare le frequenze relative $f_i = n_i/n$ e le frequenze relative cumulate $F_i = f_1 + f_2 + \ldots + f_i$ con (i = 1,2,...,k) dove la generica F_i rappresenta la proporzione dei dati del campione minori o uguali di z_i.
Una funzione di distribuzione empirica discreta risulta:

$$F(x) = \frac{\#\{x_i \leq x, i = 1, 2, \ldots, n\}}{n} = \begin{cases} 0, & x < z_1 \\ F_1, & z_1 \leq x < z_2 \\ \ldots & \\ F_i, & z_i \leq x < z_{i+1} \\ \ldots & \\ 1, & x \geq z_k \end{cases}$$

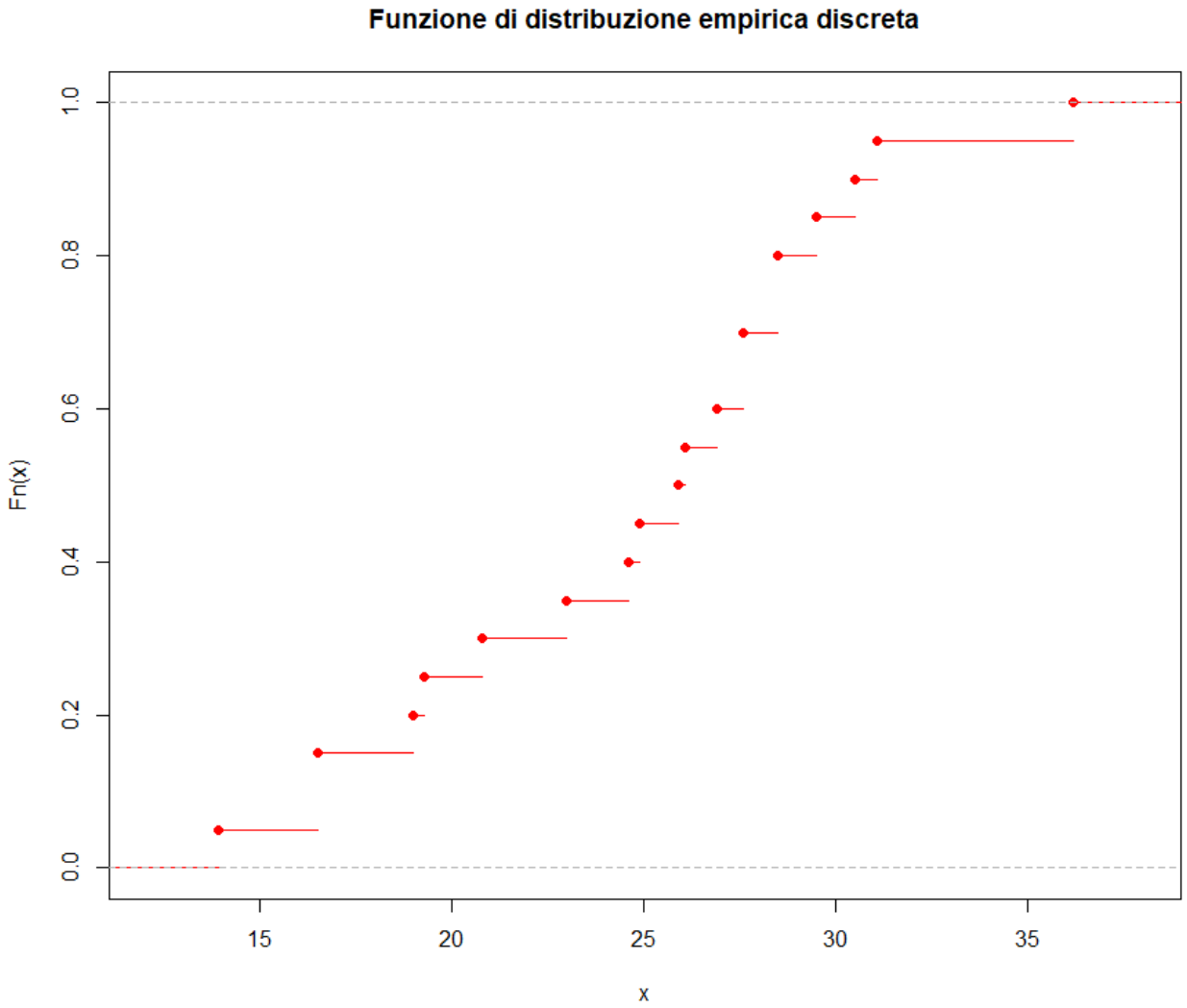
dove # è la cardinalità dell'insieme, le z_i rappresentano i possibili valori assunti dal campione, le x_i gli effettivi valori del campione e le F_i la proporzione dei dati del campione minori o uguali di z_i.
La funzione di distribuzione empirica F(x) è definita per ogni x reale ed è una funzione a gradini in cui ogni gradino indica quale proporzione di dati presenta un valore minore o uguale di quello indicato sull'asse delle ascisse.
La funzione di distribuzione empirica F(x) gode delle seguenti proprietà:

- è una funzione non decrescente;
- la funzione assume il valore a sinistra in corrispondenza ad ogni punto di salto;
- la funzione vale 0 per ogni valore minore dell'osservazione minima e vale 1 per ogni valore maggiore o uguale dell'osservazione massima.

In R, è messa a disposizione la funzione *ecdf()* (empirical cumulative distribution function) permette di disegnare il grafico della funzione di distribuzione empirica per variabili quantitative discrete e fornisce per ciascun x reale il valore di tale funzione.
Per le informazioni contenute nella nostra matrice, una funzione di distribuzione empirica discreta non risulta di particolare interesse ai fini della nostra trattazione perché i dati risultano assumere tutti valori distinti. Però possiamo considerare il vettore "in_modo_continuativo" e rappresentarne la funzione di distribuzione empirica discreta a titolo di esempio.
Le seguenti linee di codice:

```
plot(ecdf(in_modo_continuativo), main = "Funzione di distribuzione empirica discreta",
          verticals = FALSE, col = "red")
```

permettono di disegnare il grafico in figura sottostante.



Se si utilizza il parametro verticals = TRUE i segmenti sono uniti da segmenti verticali.
Se inoltre desideriamo conoscere il valore della funzione di distribuzione empirica discreta nel punto x = 15 occorre scrivere:

```
> ecdf(in_modo_continuativo) (15)
[1] 0.05
```

che, osservando la figura, corrisponde a 0.05.

Funzione di distribuzione empirica continua

La funzione di distribuzione empirica continua risulta essere necessaria nel caso in cui i nostri dati vengono raccolti in più classi. Infatti considerando che il nostro campione di dati non assume valori distinti $z_1, z_2, \dots, z_k$, possiamo raggruppare le informazioni contenute all'interno della tabella in k distinte classi.
Ancora una volta occorre calcolare le frequenze relative cumulate delle varie classi. In tal caso, la funzione di distribuzione empirica continua con suddivisione in classi risulta essere:

$$F(x) = \begin{cases} 0, & x < z_1 \\ \dots & \\ F_i, & x = z_i \\ \dfrac{F_{i+1} - F_i}{z_{i+1} - z_i}\, x + \dfrac{z_{i+1} F_i - z_i F_{i+1}}{z_{i+1} - z_i}, & z_i < x < z_{i+1} \\ F_{i+1}, & x = z_{i+1} \\ \dots & \\ 1, & x \geq z_{k+1} \end{cases}$$

Si nota che F(x) = 0 per $x < z_1$, F(x) = 1 per $x \geq z_{k+1}$, mentre se $z_i < x < z_{i+1}$ la funzione di distribuzione empirica continua coincide con il segmento che passa per i punti (z_i, F_i) e (z_{i+1}, F_{i+1}), ossia

$$\frac{y - F_i}{x - z_i} = \frac{F_{i+1} - F_i}{z_{i+1} - z_i}$$

Nel nostro specifico caso, introduciamo le seguenti classi [0,10) [10,15) [15,20) [20,25) [25,30) [30,35) [35,40) [40,45) [45,50) [50,55) [55,60).
La classe [0,10) rappresenta una classe fittizia che ci permette di poter mostrare il caso in cui la funzione di distribuzione vale 0, allo stesso modo, le ultime cinque classi vengono utilizzate per poter mostrare il caso in cui la funzione di distribuzione vale 1.
Le seguenti linee di codice

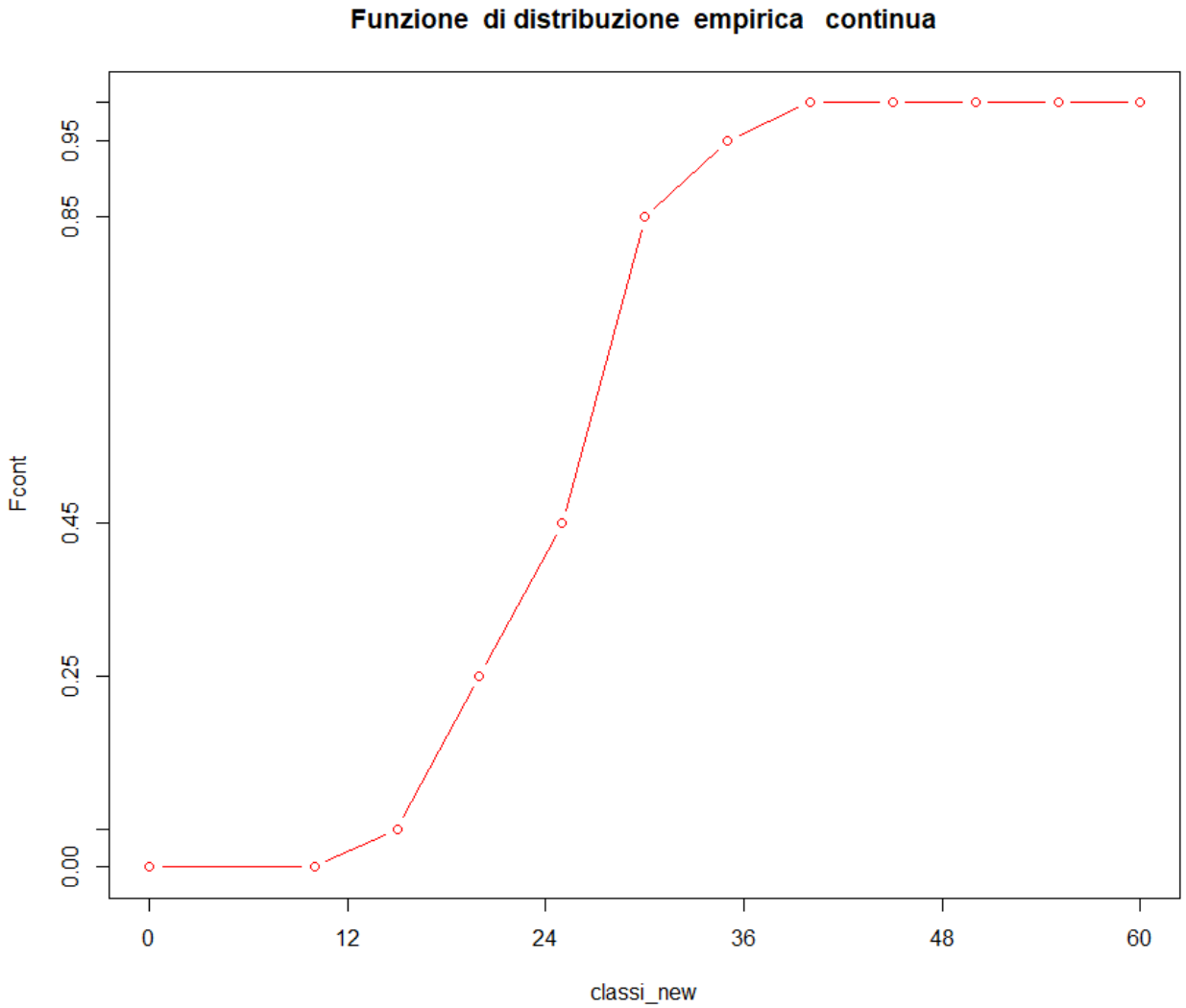
```
classi_new <- c(0,10,15,20,25,30,35,40,45,50,55,60)
Fcont<-cumsum (table (cut (in_modo_continuativo , breaks=classi_new,
                           right=FALSE )))/length (in_modo_continuativo)
Fcont
```

```
[0,10) [10,15) [15,20) [20,25) [25,30) [30,35) [35,40) [40,45) [45,50) [50,55) [55,60)
  0.00    0.05    0.25    0.45    0.85    0.95    1.00    1.00    1.00    1.00    1.00
```

permettono di visualizzare le frequenze relative cumulate associate alle classi scelte.
Vogliamo ora scrivere delle linee di codice in R che consentono di ottenere il grafico della funzione di distribuzione empirica continua. Infatti le seguenti linee di codice

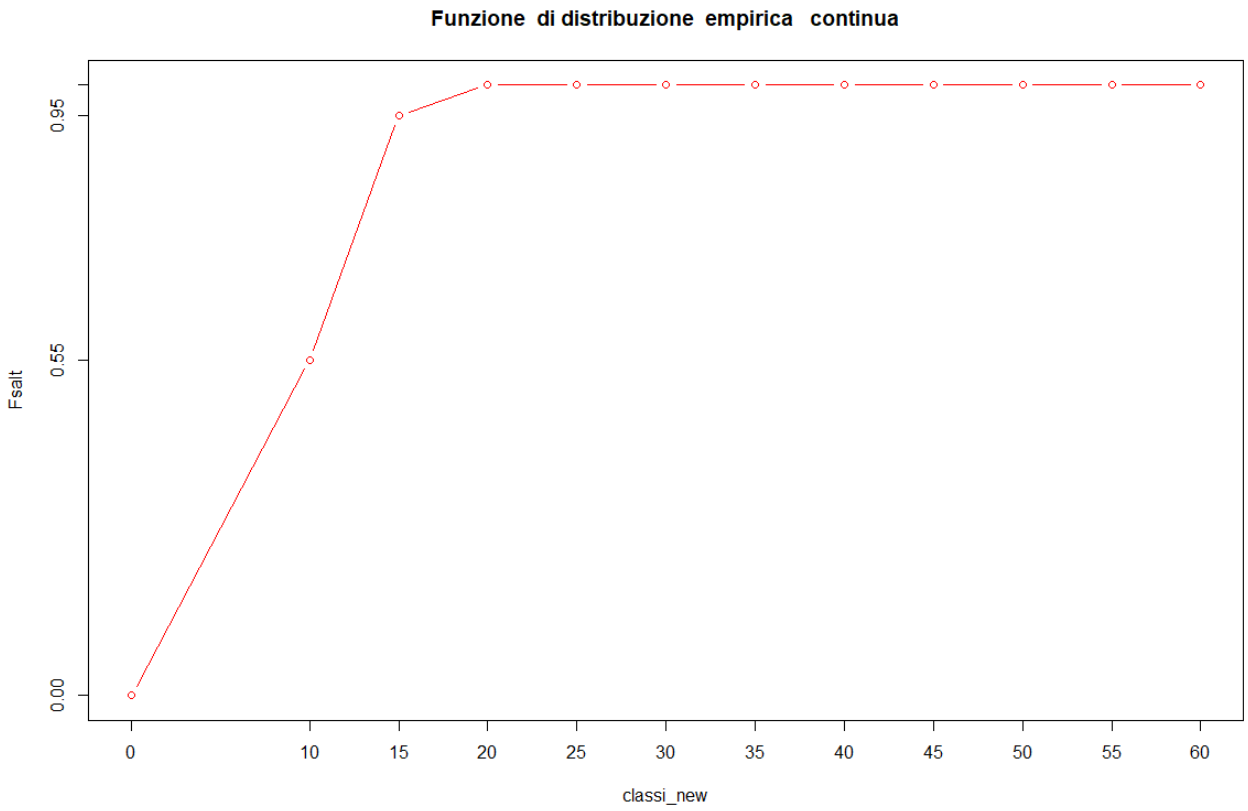
```
Fcont<-c(0,Fcont)
plot(classi_new, Fcont, type = "b", axes = FALSE,
     main= " Funzione  di distribuzione  empirica   continua ", col ="red ")
axis(1, classi )
axis(2, format(Fcont , digits = 2))
box()
```

producono il grafico illustrato in figura.



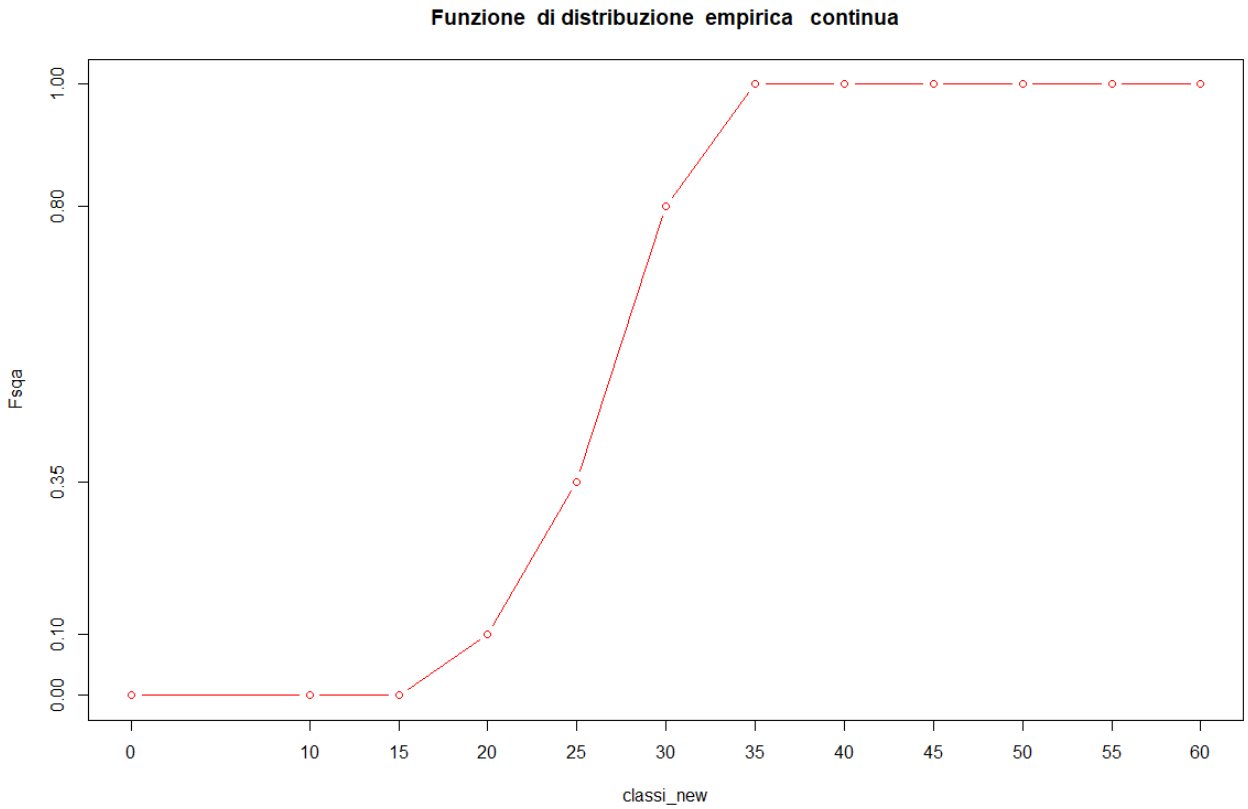
L'utilizzo della funzione di concatenazione c(0, Fcont) permette di aggiungere uno zero all'inizio del vettore delle frequenze relative cumulative. La funzione plot() permette di rappresentare ordinatamente le coppie di punti (classi_new, Fcont) contenuti nei vettori classi_new e Fcont. Nella funzione plot() si è utilizzata l'opzione type = "b" che consente di congiungere i punti successivi del grafico mediante linee continue ai cui estremi sono presenti dei cerchietti, mentre l'opzione axes = FALSE consente di non tracciare gli assi. Infine mediante l'istruzione axis(1, classi_new) si disegna l'asse orizzontale in basso, mediante l'istruzione axis(2, format(Fcont, digits = 2)) si ottiene l'asse verticale di sinistra con una formattazione opportuna dei numeri. Successivamente mediante l'istruzione box() si racchiude il grafico in un rettangolo.
È possibile notare che in tal caso esiste una concentrazione massima sulla prime classi in quanti si raggiunge con maggiore velocità l'unità.
Lo stesso procedimento è stato ripetuto per tutti gli altri vettori che compongono la matrice dei dati e di seguito ne riportiamo le funzioni di distribuzioni empiriche continue.

- Per il vettore "in_modo_saltuario", il grafico in figura

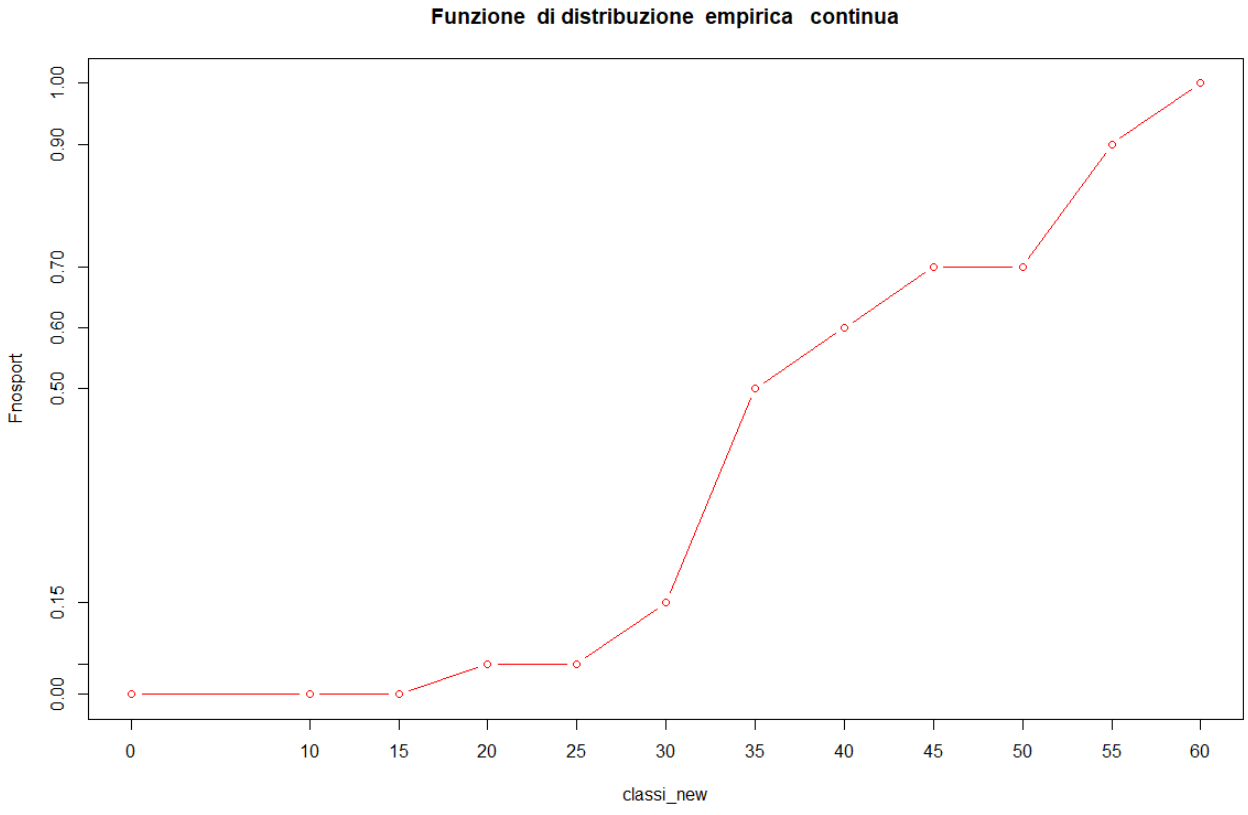


mostra che si raggiunge velocemente l'unità: ciò vuol dire che i nostri dati sono maggiormente concentrati nelle prime due classi.

- Per il vettore "solo_qualche_attività", si ha la situazione descritta in figura sottostante.



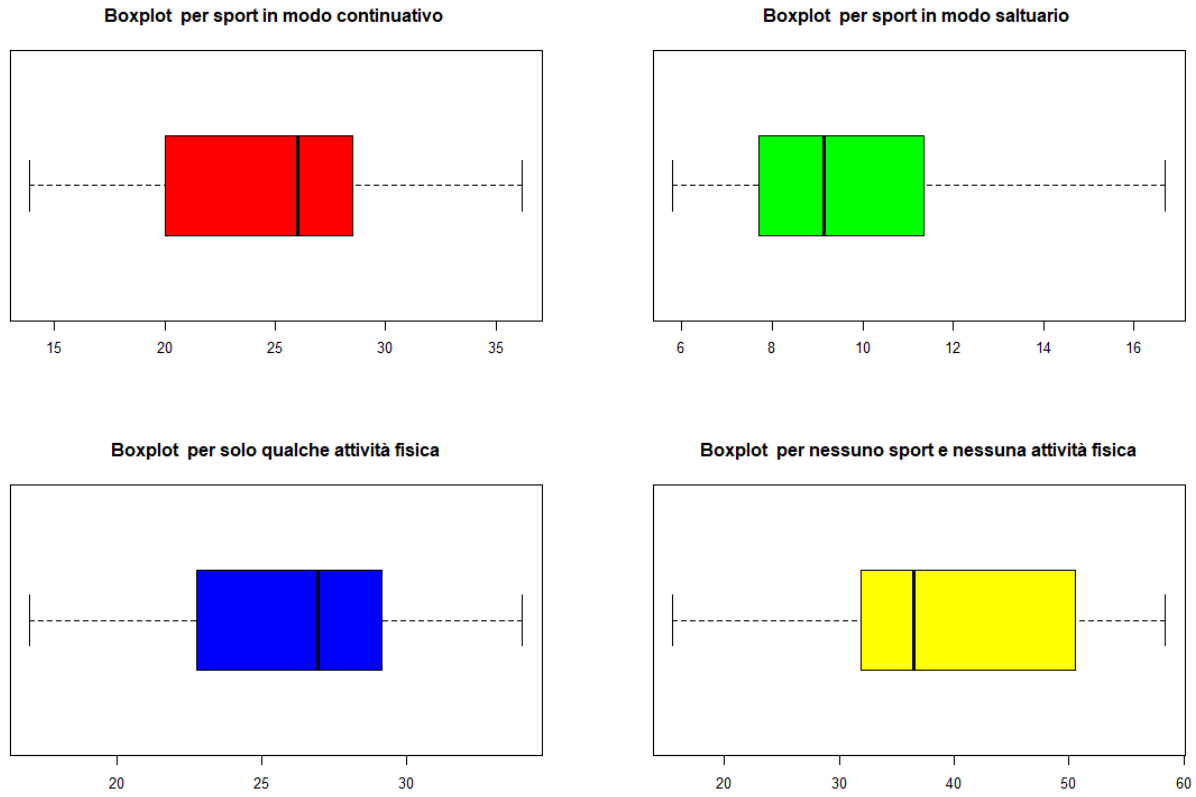
- Per il vettore “non_praticano_sport” la situazione è totalmente differente rispetto alle altre.



La figura mostra che i dati sono concentrati più o meno equamente rispetto agli altri vettori.

Indici di sintesi

Il boxplot, come già detto, è un grafico relativo a caratteri quantitativi che descrive le caratteristiche salienti della distribuzione di frequenza di un dato campione numerico. Per rappresentare una distribuzione in modo sintetico, il box plot è un'ottima possibilità: con poche informazioni, si riesce a comprendere la sua forma, simmetrica o asimmetrica che sia. Prendiamo in esame la figura sottostante.



Notiamo che i quattro box plot relativi ai quattro vettori che compongono la nostra matrice dei dati evidenziano efficacemente l'asimmetria della distribuzione dei caratteri descritti.
Gli indici che introdurremo nel seguito servono a misurare quantitativamente alcune delle caratteristiche osservate nei grafici delle distribuzioni di frequenza e nei box plot.

Indici di posizione

Indici di posizione centrali

Fino ad ora abbiamo visto come raccogliere e rappresentare i nostri dati. Ora è il momento di entrare nel mondo dell'analisi statistica. Abbiamo già detto che la statistica è la scienza che raccoglie e analizza i dati. Ma come si analizzano i dati? Quali sono le informazioni che ci interessano? Con i dati possiamo fare molte cose, in particolare sui nostri dati calcoleremo la media campionaria, la mediana e la moda detti anche **indici di posizione centrale** poiché descrivono attorno a quali valori è centrato l'insieme dei dati.

Media, mediana e moda

Dato un campione numerico di ampiezza o numerosità pari a n x_1, x_2, ..., x_n la media campionaria è definita come la media aritmetica di questi valori.
Si definisce *media campionaria* e si denota con x, la quantità:

$$\overline{x} = \frac{1}{n} \sum_{i=1}^{n} x_i$$

Si noti che la media campionare gode della *proprietà di linearità*. Inoltre, la media campionaria è una media pesata dei valori distinti infatti essa può essere riscritta in termini di frequenze relative.

$$\overline{x} = \frac{1}{n} \sum_{i=1}^{n} x_i = \frac{1}{n} \sum_{i=1}^{k} z_i \, n_i = \sum_{i=1}^{k} \frac{n_i}{n} \, z_i = \sum_{i=1}^{k} f_i \, z_i$$

Per ogni valore x_i si definisce lo *scarto dalla media campionaria* la quantità

$$s_i = x_i - \bar{x}$$

che indica il grado di scostamento del singolo valore x_i dalla media campionaria $\bar{x}$. È possibile però notare che la somma algebrica degli scarti dalla media campionaria non assume un particolare significato in quanto è sempre nulla.
La **media campionaria** utilizza i dati ed è quindi **influenzata** in maniera sensibile da **valori** eccezionalmente **alti** o **bassi** (valori anomali).

Una seconda statistica che indica la centralità di un insieme di dati è la *mediana campionaria*. Dato un insieme di dati numerici di ampiezza n, ordinato in ordine crescente dal minore al maggiore, si definisce la mediana campionaria il valore che si trova in posizione n+1/2 per n dispari. È invece definita come la media aritmetica dei valori in posizione n/2 e n/2+1 per n pari. Questa definizione di mediana campionaria bipartisce le osservazioni in due gruppi di uguale numerosità, in maniera tale che *lo stesso numero di valori cada sia a sinistra che a destra della mediana stessa*.
La mediana campionaria dipende solo da uno o da due valori centrali dei dati e non risente dei valori estremi. Inoltre, l'uso della mediana come indice per descrivere le caratteristiche dei dati ha lo svantaggio di dover prima riordinare i dati in ordine crescente, il che non è richiesto per il calcolo della media.
Il sistema R mette a disposizione le funzioni *mean()* e *median()* per calcolare rispettivamente la media e la mediana di un insieme di dati.
Considerando sempre i nostri quattro vettori che rappresentano le modalità di praticare sport, il seguente codice R ci ha permesso di ricavare la media e la mediana di essi.

```
> mean(in_modo_continuativo)
[1] 24.845
> #per ciascun xi (elemento del campione) possiamo calcolare lo scarto dalla media ad esempio:
> 28.5 - mean(in_modo_continuativo)
[1] 3.655
> mean(in_modo_saltuario)
[1] 9.815
> mean(solo_qualche_attivita)
[1] 26.285
> mean(non_praticano_sport)
[1] 38.775
> median(in_modo_continuativo)
[1] 26
> median(in_modo_saltuario)
[1] 9.15
> median(solo_qualche_attivita)
[1] 26.95
> median(non_praticano_sport)
[1] 36.5
```

Come possiamo notare dal calcolo della media e della mediana per ciascun vettore di dati, tali misure non sono molto distanti tra loro: ad esempio, media e mediana nel caso del vettore "in_modo_saltuario" assumono valori rispettivamente pari a 9.815 e 9.15 evidenziando in tal caso una buona distribuzione dei dati.

La terza statistica utilizzata per descrivere la centralità di una distribuzione di dati è la *moda campionaria*.
La moda campionaria di un insieme di dati, se esiste, è la modalità cui è associata la frequenza (assoluta o relativa) più elevata. Se esistono più modalità con frequenza massima, ciascuna di esse è detto *valore modale*. La moda, quindi, rappresenta il valore prevalente nell'insieme dei dati, ovvero quello che si presenta con maggiore frequenza. Non esiste invece in R una funzione per estrarre la moda da una distribuzione di dati poiché è facilmente ricavabile osservando il grafico delle frequenze assolute. A tal proposito, è stata definita una funzione che calcola i valori modali di un vettore numerico.

```
#funzione per calcolare la moda
function_moda <- function (v){
  y <-table (v)
  z <-which (y== max (y))
  return (c(z))}
```

Tale funzione è stata poi applicata a ciascun vettore all'interno della nostra matrice di dati:

```
> function_moda(in_modo_continuativo)
16.5 27.6 28.5
   2   12   13
> function_moda(in_modo_saltuario)
 7.7 10.7
   5   11
> function_moda(solo_qualche_attivita)
28.6
  12
> function_moda(non_praticano_sport)
15.5 26.1 26.2 30.6 31.8 31.9 32.6 32.9 33.1 34.7 38.3 38.7 40.3 40.6 50.4 50.6 52.5 53.4 56.9 58.4
   1    2    3    4    5    6    7    8    9   10   11   12   13   14   15   16   17   18   19   20
```

Riteniamo che la moda sia un buon indice di sintesi nel caso del primo, del secondo e del terzo vettore in quanto i valori modali si presentano con una frequenza nettamente maggiore rispetto alle frequenze degli altri valori assunti dal campione. Al contrario, invece, la moda campionaria non risulta essere utile nel caso del vettore "non_praticano_sport" in quanto i dati sono numerosi e per la maggior parte diversi tra di loro. Infatti, proprio perché tutte le modalità presentano all'incirca la stessa frequenza, la moda non è un indice di sintesi significativo in quest'ultimo caso.

A questo punto possiamo descrivere la forma della distribuzione di frequenza confrontando la media campionaria e la mediana campionaria ottenute.

	In modo continuativo	In modo saltuario	Solo qualche attività	Non praticano sport
Media campionaria	24.845	9.815	26.285	38.775
Mediana campionaria	26	9.15	26.95	36.5

Se la media e la mediana sono uguali, la distribuzione di frequenza tende ad essere simmetrica; se la media campionaria è sensibilmente maggiore della mediana campionaria, la distribuzione di frequenza è più sbilanciata verso destra; se invece, la media campionaria è sensibilmente minore della mediana campionaria la distribuzione di frequenza è più sbilanciata verso sinistra. Tuttavia questi indici di posizione non tengono conto della variabilità tra i dati, infatti, dati che presentano stessa media e mediana, potrebbero però essere distribuiti in maniera completamente diversa.

Mediana per una distribuzione di frequenza

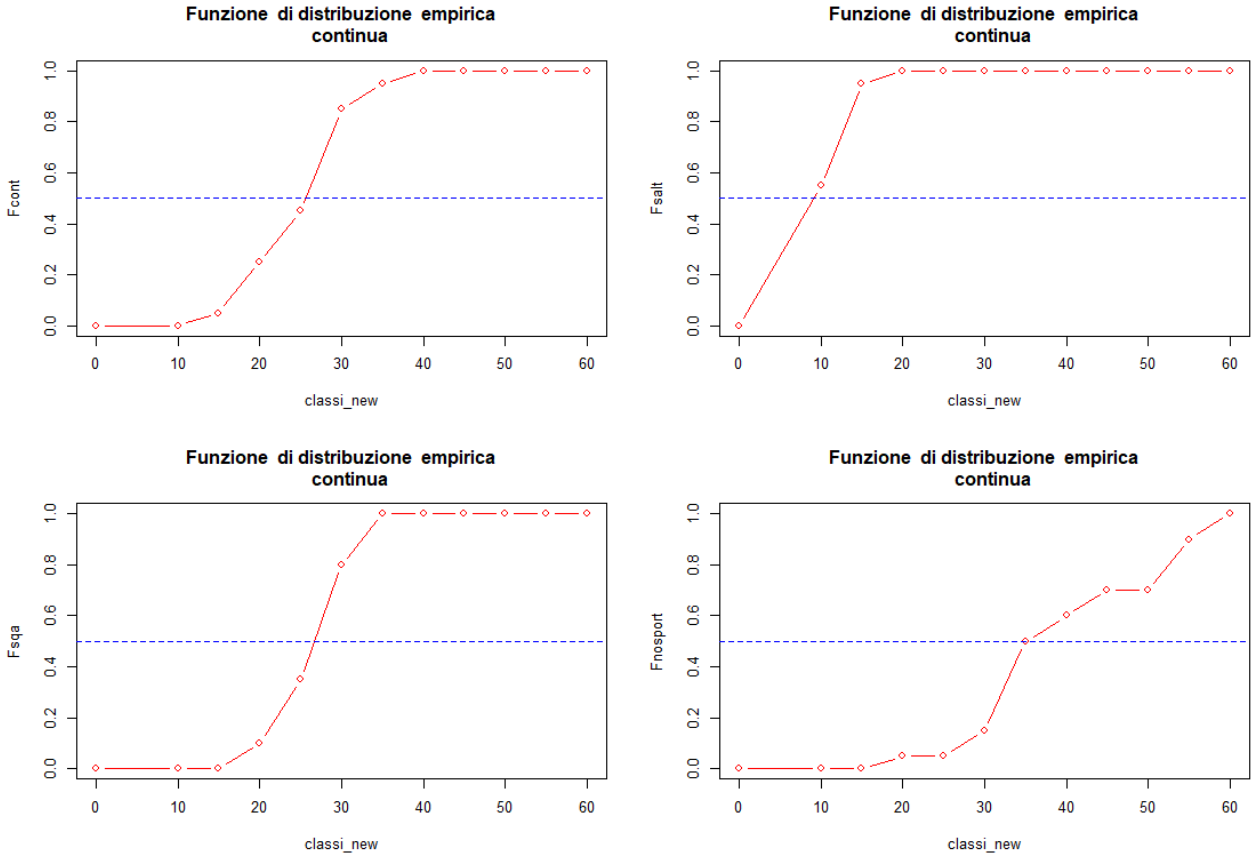
Un modo di procedere diverso per definire la mediana, consiste nel considerare le frequenze relative cumulate. In tal caso, la mediana per una distribuzione di frequenze è definita come la modalità i-esima (i=1,2, … , k) che soddisfa la doppia disuguaglianza:

$$F_{i-1} \leq 0.5, \qquad F_i \geq 0.5$$

Come si evince dalla definizione, la mediana di una distribuzione di frequenza è un valore di sintesi che indica un punto centrale intorno al quale si dispone la distribuzione di frequenza. Per ciascun vettore di dati all'interno della nostra matrice, andiamo ad individuare graficamente la mediana a partire dalla funzione di distribuzione empirica continua. Tracciamo la funzione di distribuzione empirica e sull'asse delle ordinate individuiamo il punto 0.5 e da questo tracciamo una linea orizzontale. Il minimo valore osservato (sulle ascisse) la cui funzione di distribuzione empirica supera 0.5 è proprio la mediana per una distribuzione di frequenze. Le seguenti linee di codice:

```
par(mfrow=c(2,2))
plot(classi_new, Fcont, type = "b", main="Funzione  di distribuzione  empirica
     continua ", col ="red ")
abline(h=0.5, lty=2, col = "blue")

plot(classi_new, Fsalt, type = "b", main="Funzione  di distribuzione  empirica
     continua ", col ="red ")
abline(h=0.5, lty=2, col = "blue")

plot(classi_new, Fsqa, type = "b", main="Funzione  di distribuzione  empirica
     continua ", col ="red ")
abline(h=0.5, lty=2, col = "blue")

plot(classi_new, Fnosport, type = "b", main="Funzione  di distribuzione  empirica
     continua ", col ="red ")
abline(h=0.5, lty=2, col = "blue")
```

permettono di graficare le funzioni di distribuzione empiriche relative ai quattro vettori.



È immediato notare che nel caso dei vettori "in_modo_continuativo" e "solo_qualche_attivita" la funzione di distribuzione empirica continua risulta essere simile infatti:

```
> quantile(in_modo_continuativo, 0.5, type = 1)
 50%
25.9
> quantile(in_modo_saltuario, 0.5, type = 1)
50%
8.5
> quantile(solo_qualche_attivita, 0.5, type = 1)
50%
 26
> quantile(non_praticano_sport, 0.5, type = 1)
 50%
34.7
```

per questi due vettori la mediana risulta essere rispettivamente 25.9 e 26. La funzione *quantile()* in questo caso, ci permette di risalire all'ascissa che corrisponde all'ordinata 0.5 tracciata con *abline()*.

Quantili, percentili, decili e quartili

Oltre la mediana, che è quel valore che divide a metà un insieme di dati ordinati, si possono definire altri indici di posizione, detti quantili, che dividono l'insieme dei dati ordinati in un fissato numero di parti uguali. Possiamo suddividere i dati in 4 parti mediante 3 quantili detti *quartili*, oppure in 10 parti mediante 9 quantili detti *decili* oppure anche in 100 parti mediante 99 quantili detti *percentili*.

I quantili (percentili) sono *indici di posizione non centrali* utilizzati per insiemi numerosi di dati.
Per poter suddividere i dati ordinati in α gruppi, ognuno dei quali contenga (circa) lo stesso numero di osservazioni, possiamo utilizzare 9 differenti tipi di algoritmi. Il tipo di algoritmo impiegato è specificato all'interno della funzione *quantile(v, probs=, type=)* in cui probs rappresenta un vettore di probabilità.
I percentili che vengono maggiormente utilizzati sono il 25-esimo, il 50-esimo e il 75-esimo, detti rispettivamente primo quartile (Q_1), il secondo quartile (Q_2) e il terzo quartile (Q_3).
Se si omette probs vengono di default calcolati i quartili con type = j e la funzione restituisce il minimo, il massimo e i tre quartili Q1, Q2 e Q3. Se si omette type e si scrive vengono di default calcolati i percentili specificati nel vettore delle probabilità
R di default utilizza l'algoritmo di tipo 7 per il calcolo dei quantili.
Se consideriamo un campione di ampiezza n ordinato in ordine crescente, e consideriamo P_k il percentile di interesse. Calcoliamo l'indice h e otteniamo che:

- **L'algoritmo di tipo 2** calcola i percentili come:
  - **(v[h] + v[h+1])/2** se np è un intero;
  - **v[h*]** (dove h* rappresenta il primo intero successivo ad h) se np non è un intero.
- **L'algoritmo di tipo 7** calcola i percentili come:
  - **v[h*] + (h – h*) (v[h* + 1] – v[h*]** in cui h* rappresenta il più grande intero minore o uguale ad h.
- **L'algoritmo di tipo 1,** invece, calcola il quantile di ordine p come la modalità i-esima che soddisfa la doppia disuguaglianza:
  - $\mathbf{F_{i-1} < p}$, $\mathbf{F_i \leq p}$.

Si può facilmente mostrare che per p = 0.5 e k = 50, l'algoritmo di tipo 2 e l'algoritmo di tipo 7 conducono allo stesso risultato nel calcolo della mediana mentre i valori per il primo e il terzo quartile potrebbero essere differenti.
Consideriamo il vettore "in_modo_continuativo", calcoliamo la mediana con entrambi gli algoritmi:

```
> quantile(in_modo_continuativo, probs = 0.5, type = 7)
50%
 26
> quantile(in_modo_continuativo, probs = 0.5, type=2)
50%
 26
```

Se invece consideriamo sempre per lo stesso vettore, i quartili calcolati con l'algoritmo di tipo 7 e l'algoritmo di tipo 2, otteniamo la situazione seguente:

```
> quantile(in_modo_continuativo)
    0%    25%    50%    75%   100%
13.900 20.425 26.000 28.500 36.200
> quantile(in_modo_continuativo, type=2)
   0%   25%   50%   75%  100%
13.90 20.05 26.00 28.50 36.20
```

Dalla quale è possibile notare immediatamente che il calcolo dei quartili coincide per il secondo e per il terzo mentre, il primo quartile risulta essere pari a 20.425 con l'algoritmo di tipo 7 e 20.05 con l'algoritmo di tipo 2.

Varianza, deviazione standard e coefficiente di variazione

gli indici di posizione non tengono conto della variabilità dei dati; infatti ci sono casi in cui esistono distribuzioni di frequenza che sono molto diverse tra loro nonostante abbiano la stessa media campionaria. Indici significativi per misurare la variabilità di una distribuzione di frequenza sono la *varianza campionaria* e la *deviazione standard campionaria*, detta anche *scarto quadratico medio campionario*.
Possiamo definire la varianza campionaria come la quantità:

$$s^2 = \frac{1}{n-1}\sum_{i=1}^{n}(x_i - \overline{x})^2 \qquad (n = 2, 3, \ldots),$$

dove $\bar{x}$ denota la media campionaria dei dati. Inoltre, possiamo definire la deviazione standard campionaria come la radice quadrata della varianza campionaria, ossia:

$$s = \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(x_i - \bar{x})^2} \qquad (n=2,3,\ldots)$$

Si noti che la varianza e la deviazione standard sono tanto più grandi quando più i dati si discostano dalla media. Se il campione fosse costituito da un solo elemento ovviamente non ha senso calcolare la varianza: ha senso solo per campioni di ampiezza almeno pari a 2.
I valori della varianza campionaria s^2 e della deviazione standard campionaria s dipendono dall'unità di misura dei dati. In particolare, la deviazione standard campionaria s misura la dispersione dei dati con la stessa unità di misura dei dati sperimentali e quindi con la stessa unità di misura della media campionaria.
Inoltre, possiamo affermare che la varianza campionaria, a differenza delle media campionaria, non gode della proprietà di linearità: infatti se consideriamo un insieme di dati tale che $y_i = ax_i + b$ si ha che $s^2_y = a^2 s^2_y$. Ciò vuol dire che sommare una costante a ciascuno dei dati non fa cambiare la varianza, al contrario se moltiplichiamo i nostri dati per un fattore costante fa sì che la varianza campionaria risulti moltiplicata per il quadrato di tale fattore.
A questo punto andiamo a valutare di quanto si disperdono i nostri dati rispetto alla media:

```
> var(in_modo_continuativo)
[1] 32.33313
> var(in_modo_saltuario)
[1] 8.621342
> var(solo_qualche_attivita)
[1] 19.6245
> var(non_praticano_sport)
[1] 133.763
```

Per quanto riguarda la deviazione standard campionaria, invece, si ha la situazione seguente:

```
> sd(in_modo_continuativo)
[1] 5.686223
> sd(in_modo_saltuario)
[1] 2.936212
> sd(solo_qualche_attivita)
[1] 4.429955
> sd(non_praticano_sport)
[1] 11.5656
```

La media campionaria e la deviazione standard campionaria sono i due indici di posizione e di dispersione maggiormente utilizzati. Per confrontare le variazioni esistenti tra diversi campioni di dati è utile introdurre il *coefficiente di variazione*. Tale coefficiente è definito

come il rapporto tra la deviazione standard campionaria e il modulo della media campionaria ossia:

$$cv = \frac{s}{|\bar{x}|}$$

Si nota che il coefficiente di variazione è un *numero puro*, ossia è un indice adimensionale che non dipende dall'unità di misura utilizzata, poiché la media campionaria e la deviazione standard campionaria sono espressi in identiche unità di misura. Dalla definizione segue che il coefficiente di variazione è un indice di dispersione che ha senso soltanto per campioni aventi la media campionaria non nulla.
In R definiamo la funzione che calcola il coefficiente di variazione in questo modo:

```
cv <- function (x){
sd(x)/abs ( mean(x))
}
```

Riferendoci ai dati dei vettori "in_modo_continuativo", "in_modo_saltuario", "solo_qualche_attivita" e "non_praticano_sport", si nota che

```
> cv(in_modo_continuativo)
[1] 0.2288679
> cv(in_modo_saltuario)
[1] 0.2991556
> cv(solo_qualche_attivita)
[1] 0.1685355
> cv(non_praticano_sport)
[1] 0.2982746
```

Nella seguente tabella sono riportati la media, la varianza, la deviazione standard e il coefficiente di variazione dei vettori sopra citati.

	In modo continuativo	In modo saltuario	Solo qualche attività	Non praticano sport
Media campionaria	24.845	9.815	26.285	38.775
Varianza campionaria	32.333	8.621	19.625	133.763
Deviazione standard campionaria	5.686	2.936	4.430	11.566
Coefficiente di variazione	0.229	0.299	0.169	0.298

Dalla tabella si evince che il coefficiente di variazione più alto si ottiene per il vettore "in_modo_saltuario" in cui i dati si discostano maggiormente dalla media campionaria. Infatti, era questo il caso in cui i dati variavano in un range abbastanza ampio; ciò si può intuire confrontando i boxplot.

Forma di una distribuzione di frequenza

La media, la mediana e la moda sono utili a comprendere la forma delle distribuzioni di frequenza, nel senso che differenze sostanziali tra questi indici indicano uno sbilanciamento eccessivo della distribuzione di frequenza verso destra o verso sinistra. Abbiamo inoltre

mostrato che anche se la media e la mediana coincidono, le misure di dispersione dei dati possono essere sostanzialmente differenti implicando una differente forma delle distribuzioni di frequenza. Esistono degli indici statistici che permettono di misurare quando una distribuzione di frequenza presenta *simmetria* o *asimmetria* oppure se essa è più o meno *piccata*.
Un indice che permette di misurare la simmetria di una distribuzione di frequenza è la skewness campionaria definita come:

$$\gamma_1 = \frac{m_3}{m_2^{3/2}}$$

In cui m_3 denota il momento centrato di ordine 3. In generale il *momento centrato di ordine j* è definito così:

$$m_j = \frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})^j \qquad \text{(j=1,2,...9)}.$$

Dalla definizione è possibile dedurre che:

- Se $\gamma_1 = 0$, la distribuzione di frequenza è simmetrica;
- Se $\gamma_1 > 0$, la distribuzione di frequenza ha la coda di destra più allungata per *l'asimmetria positiva*;
- Se $\gamma_1 < 0$, la distribuzione di frequenza ha la coda di sinistra più allungata per *l'asimmetria negativa*.

Il calcolo della skewness campionaria può essere così implementato in R:

```
skw <-function (x){
     n<-length (x)
     m2 <-(n -1) *var (x)/n
     m3 <- (sum ( (x- mean(x))^3) )/n
     m3/(m2 ^1.5)
}
```

Riferendoci ai dati dei nostri vettori:

```
> skw(in_modo_continuativo)
[1] -0.2274047
> skw(in_modo_saltuario)
[1] 0.7316883
> skw(solo_qualche_attivita)
[1] -0.2720538
> skw(non_praticano_sport)
[1] 0.1370904
```

da cui è possibile notare che la skeweness non risulta essere mai nulla: da ciò deriva che la distribuzione di frequenza non è mai simmetrica. Invece, la skeweness presenta un'asimmetria negativa per il primo e per il terzo vettore e presenta asimmetria positiva per i rimanenti due vettori.

Un indice che permette di misurare la densità dei dati intorno alla media è la *curtosi campionaria* definita come:

$$\gamma_2 = \beta_2 - 3$$

dove

$$\beta_2 = \frac{m_4}{m_2^2}.$$

Gli indici γ_2 e β_2 permettono di *confrontare la distribuzione di frequenza dei nostri dati con una densità di probabilità*, caratterizzata da $\beta_2 = 3$ e indice di curtosi $\gamma_2 = 0$. Infatti, se risulta:

- $\beta_2 < 3, \gamma_2 < 0$: la distribuzione di frequenza si definisce *platicurtica*, ossia la distribuzione di frequenza è più piatta di una normale;
- $\beta_2 > 3, \gamma_2 > 0$: la distribuzione di frequenza si definisce *leptocurtica*, ossia la distribuzione di frequenza è più piccata di una normale;
- $\beta_2 = 3, \gamma_2 = 0$: la distribuzione di frequenza si definisce *normocurtica*, ossia piatta come una normale.

È immediato notare che anche β_2 è un indice *adimensionale* ossia è indipendente dall'unità di misura dei dati.
Il calcolo della curtosi campionaria può essere implementato in R nel seguente modo:

```
curt <-function (x){
  n <-length (x)
  m2 <-(n -1) *var (x)/n
  m4 <- (sum ( (x-mean(x))^4) )/n
  m4/(m2 ^2) -3
}
```

Riferendoci ai dati contenuti all'interno dei nostri vettori, si nota che:

```
> curt(in_modo_continuativo)
[1] -0.531399
> curt(in_modo_saltuario)
[1] -0.2131893
> curt(solo_qualche_attivita)
[1] -0.6894558
> curt(non_praticano_sport)
[1] -0.7315175
```

la curtosi è negativa per tutti e quattro i vettori dei dati perciò la distribuzione di frequenza è piatta più di una normale, ossia è sempre platicurtica.

Statistica descrittiva bivariata

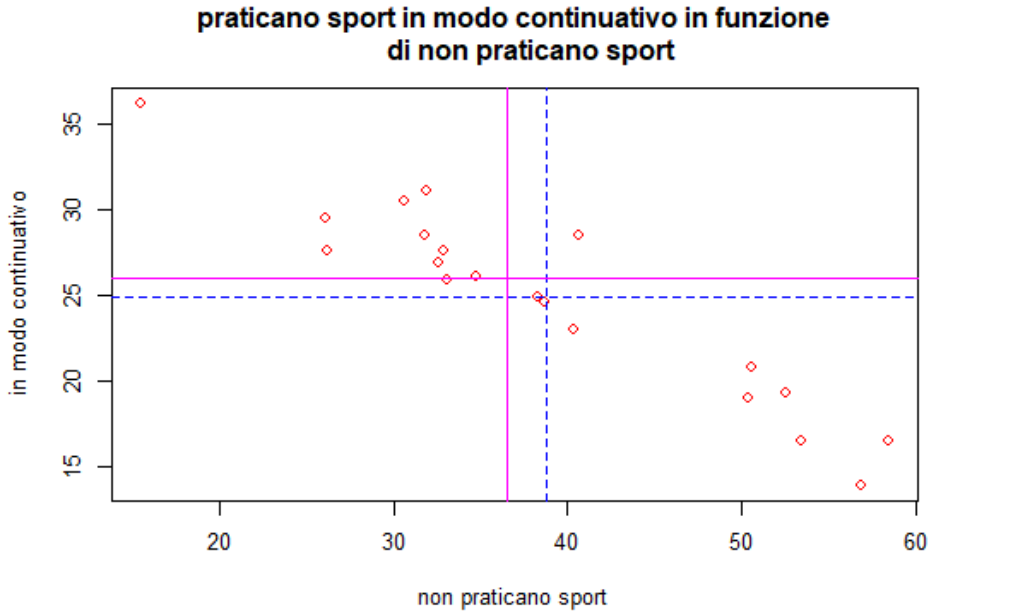
L'idea è quella di poter esaminare le relazioni esistenti tra caratteri quantitativi legati alla stessa unità statistica. Prima di poter impiegare misure di sintesi per misurare la dipendenza lineare tra le variabili, possiamo definire uno scatterplot in cui ogni coppia di osservazioni è rappresentata sotto forma di un punto o un cerchietto in un piano euclideo.
Dopo aver scelto la variabile da porre sulle ascisse (variabile indipendente) e la variabile da porre sulle ordinate (variabile dipendente), si disegnano dei punti in corrispondenza delle coppie (x_i, y_i). Il grafico che si ottiene mira ad evidenziare se le coppie di punti presentano qualche forma di regolarità. Inoltre, il grafico di dispersione mostra se esiste una relazione tra le variabili e di quale tipo è tale relazione.
Consideriamo la percentuale di persone nelle 20 regioni italiane che praticano sport in modo continuativo e non praticano alcuna attività sportiva.
Abbiamo già calcolato gli indici statistici di posizione e di dispersione relativi alle singole variabili che sono riportati nella tabella sottostante:

	In modo continuativo	Non praticano sport
Media campionaria	24.845	38.775
Mediana campionaria	26	36.5
Deviazione standard campionaria	5.686	11.566

Si nota immediatamente che media, mediana e deviazione standard sono maggiori per il secondo vettore. Successivamente abbiamo realizzato lo scatterplot considerando "non_praticano_sport" come variabile indipendente e "in_modo_continuativo" come variabile dipendente. Sullo scatterplot abbiamo anche tracciato delle linee orizzontali e verticali in corrispondenza delle mediane campionarie e delle medie campionarie dei due vettori. Le seguenti linee di codice

producono il diagramma di dispersione in figura sottostante.

```
plot(non_praticano_sport, in_modo_continuativo,
     main ="praticano sport in modo continuativo in funzione
     di non praticano sport", xlab="non praticano sport",
     ylab="in modo continuativo", col = "red")
abline(v=median(non_praticano_sport),lty =1, col =" magenta ")
abline(v=mean(non_praticano_sport),lty =2, col =" blue")
abline(h=median(in_modo_continuativo),lty =1, col =" magenta ")
abline(h=mean(in_modo_continuativo),lty =2, col =" blue")
legend(50,35,c(" Mediana "," Media "), pch =0, col=c("magenta","blue"),
       cex =0.8)
```



Si noti che i dati sembrano posizionati intorno ad una retta discendente e ciò induce a pensare che esista una correlazione lineare negativa tra le due variabili considerate. In

generale, possiamo affermare che il caso peggiore si ha quando i punti all'interno dello spazio euclideo siano sparsi. Se tutti i punti sono vicini ciò indica che esiste una forte relazione tra le variabili.

Covarianza e correlazione campionaria

Abbiamo osservato, tramite lo scatterplot, l'esistenza di una dipendenza tra le due variabili "in_modo_continuativo" e "non_praticano_sport". Abbiamo anche avuto modo di apprendere che la relazione esistente è di tipo lineare. Ma come facciamo ad ottenere una misura quantitativa della correlazione tra le variabili? Utilizziamo un indice che prende il nome di *covarianza campionaria* come:

$$C_{xy} = \frac{1}{n-1}\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})$$

Quindi se le due variabili presentano una forte correlazione, vuol dire che $\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})$ assume un valore o molto positivo o molto negativo. Si divide per n-1 perché in caso x ed y siano uguali, la covarianza risulta essere pari alla varianza campionaria.
In generale:

- Se $C_{xy} > 0$ le variabili sono *correlate positivamente*
- Se $C_{xy} < 0$ le variabili sono *correlate negativamente*
- Se $C_{xy} = 0$ le variabili considerate non risultano essere correlate tra loro, ciò vuol dire che i punti all'interno dello scatterplot sono tutti sparsi.

In R la covarianza è definita attraverso la funzione *cov(x,y)*. Possiamo calcolare la covarianza tra le nostre due variabili considerate attraverso la seguente linea di codice:

```
> cov(non_praticano_sport, in_modo_continuativo)
[1] -62.36145
```

dalla quale è possibile apprendere che la covarianza tra le variabili è minore di zero, ciò vuol dire che i due vettori sono correlati negativamente.

Per ottenere una misura quantitativa della correlazione tra variabili si può anche considerare il *coefficiente di correlazione campionario* così definito:

$$r_{x_y} = \frac{C_{xy}}{s_x s_y}$$

Il coefficiente di correlazione campionario ha lo stesso segno della covarianza. Quando $r_{xy} > 0$ si dice che le variabili sono correlate positivamente, se $r_{xy} < 0$ le variabili sono correlate negativamente e, infine, se $r_{xy} = 0$ le variabili non sono correlate.
Il coefficiente di correlazione campionario gode di alcune proprietà:

- Prima fra tutti, $-1 \leq r_{xy} \leq 1$, infatti si ha:
  - $r_{xy} = 1$ (correlazione perfetta positiva) se tutti i punti sono allineati su una linea retta ascendente;
  - $0 < r_{xy} < 1$ (correlazione positiva) se tutti i punti (x_i, y_i) sono posizionati in una nuvola attorno ad una linea retta interpolante ascendente (in tal caso x_i e y_i tendono ad essere grandi e piccoli insieme);

  - r_{xy} = 0 (nessuna correlazione) se i punti sono completamente dispersi in una nuvola che non presenta alcuna evidente direzione di natura lineare;
  - $-1 < r_{xy} < 0$ (correlazione negativa) se i punti (x_i, y_i) sono posizionati in una nuvola attorno ad una linea retta interpolante discendente (in tal caso x_i è grande y_i è piccolo e viceversa);
  - $r_{xy} = -1$ (correlazione perfetta negativa) tutti i punti sono allineati su una linea retta discendente;
- se esistono due numeri reali a e b, con a > 0, tali che $y_i = a\,x_i + b$ per ogni i = 1, 2, . . . , n, allora $r_{xy} = 1$;
- se esistono due numeri reali a e b, con a < 0, tali che $y_i = a\,x_i + b$ per ogni i = 1, 2, . . . , n, allora $r_{xy} = -1$;
- se esistono quattro numeri reali a, b, c, d e se risulta $z_i = a\,x_i + b$ e $w_i = c\,y_i + d$ per i = 1, 2, . . . , n, allora $r_{zw} = r_{xy}$ se ac > 0 e $r_{zw} = -r_{xy}$ se invece ac < 0.

Le proprietà (2) e (3) mostrano che i valori limite −1 e +1 sono effettivamente raggiunti solo quando tra X e Y sussiste una relazione lineare, ossia quando i punti dello scatterplot giacciono tutti su di una retta. La proprietà (4) afferma che il quadrato del coefficiente di correlazione non cambia se sommiamo costanti o moltiplichiamo per costanti tutti i valori di X e/o di Y . Ciò significa che il *coefficiente di correlazione non dipende dalle unità di misura scelta per rappresentare i dati.*

Occorre ricordare che il coefficiente di correlazione campionario r_{xy} misura la forza del legame di natura lineare esistente tra due variabili quantitative. Eventuali relazioni tra le variabili che assumono una forma curvilinea non possono pertanto essere individuati con tale coefficiente.

Nel linguaggio R la correlazione campionaria fra una coppia di variabili numeriche X e Y può essere immediatamente ottenuta con la funzione *cor(X,Y)*.

Consideriamo ancora una volta il vettore “in_modo_continuativo” in funzione del vettore “non_praticano_sport” e andiamo a calcolarne il coefficiente di correlazione campionario:

```
> cor(non_praticano_sport, in_modo_continuativo)
[1] -0.9482531
```

È immediato notare che il coefficiente di correlazione è uguale a -0.9482531. è minore di zero ed è prossimo all’unità. Ciò indica, come evidenziato dallo scatterplot, che esiste una forte correlazione lineare tra i due vettori presi in considerazione.

Regressione lineare semplice

Gli scatterplot sono dei potenti mezzi per visualizzare le eventuali relazioni che possono intercorrere tra variabili quantitative. Infatti, nel caso appena preso in esame, osservando il grafico si nota che i punti si dispongono attorno ad una linea orientata verso il basso.

Tuttavia per avere un’idea più accurata del fenomeno, è necessario prendere in considerazione altre tecniche statistiche in grado di misurare con maggiore precisione il legame che intercorre tra il vettore “non praticano sport” ed il vettore “in modo continuativo”.

Il modello lineare viene di solito utilizzato per spiegare, descrivere, o anche prevedere un andamento futuro sulla base della relazione che si instaura tra la variabile dipendente, e una o più variabili indipendenti. Nel caso in cui si ha una solo variabile dipendente, l’analisi prende il nome di *regressione semplice*, al contrario invece si parla di *regressione multipla*.

Per poter utilizzare un modello di regressione è fondamentale individuare in primo luogo quali sono le variabili indipendenti e quale è la variabile dipendente.
Noi continuiamo a considerare vettore "in_modo_continuativo" in funzione del vettore "non_praticano_sport" e quindi il primo come variabile dipendente e il secondo come variabile indipendente.
Per questi due vettori abbiamo già calcolato il coefficiente di correlazione che ci ha permesso di comprendere che i nostri punti all'interno dello scatterplot risultano essere disposti intorno ad una linea retta ciò vuol dire che si tratta di regressione lineare semplice. Consideriamo a questo punto l'equazione della retta:

$$Y = \alpha + \beta X$$

dove α rappresenta l'intercetta ovvero l'ordinata del punto d'intersezione con l'asse delle y, mentre β rappresenta il coefficiente angolare ovvero se $\beta > 0$ si ha che la retta è crescente; se $\beta < 0$ allora la retta è decrescente; infine, se $\beta = 0$ la retta risulta essere orizzontale.

Per calcolare i coefficienti di regressione è necessario considerare che la somma Q dei quadrati degli errori

$$Q = \sum_{i=1}^{n} [y_i - (\alpha + \beta x_i)]^2$$

sia minima, dove y_i rappresenta i valori osservati della variabile Y, le x_i sono i valori osservati della variabile X e $(\alpha + \beta x_i)$ rappresentano i valori stimati tramite la retta di regressione.
Derivando Q rispetto ad α e β e ponendo le derivate parziali ottenute pari a zero si ha che:

$$\beta = \frac{s_y}{s_x} r_{xy} \qquad e \qquad \alpha = \bar{y} - \beta \bar{x}$$

Riferendoci al nostro caso specifico, le seguenti linee di codice calcolano il coefficiente angolare β e l'intercetta α:

```
beta <-(sd(in_modo_continuativo)/sd(non_praticano_sport))*cor(non_praticano_sport,
                                                              in_modo_continuativo)
alpha <-mean(in_modo_continuativo)-beta*mean(non_praticano_sport)
> c(alpha,beta)
[1] 42.9222310 -0.4662084
```

Ovviamente, come già sapevamo la nostra retta di regressione risulta essere decrescente in quanto beta < 0. Attraverso alpha invece possiamo affermare che la retta di regressione interseca l'asse delle y nel punto 42.922310, quindi:

$$y = 42.9222310 - 0.4662084\ x\ .$$

Per poter eseguire le analisi di regressione lineare utilizziamo la funzione *lm(y ~x)* (linear model) in cui l'argomento indica che y dipende da x.

Applichiamo tale funzione al nostro caso specifico:

```
> linearmodel <- lm(in_modo_continuativo ~ non_praticano_sport)
> linearmodel

Call:
lm(formula = in_modo_continuativo ~ non_praticano_sport)

Coefficients:
        (Intercept)  non_praticano_sport
            42.9222              -0.4662
```

e scopriamo che la funzione lm() ci restituisce come attribuiti i coefficienti di regressione calcolati in precedenza attraverso la formula che ci eravamo ricavati.

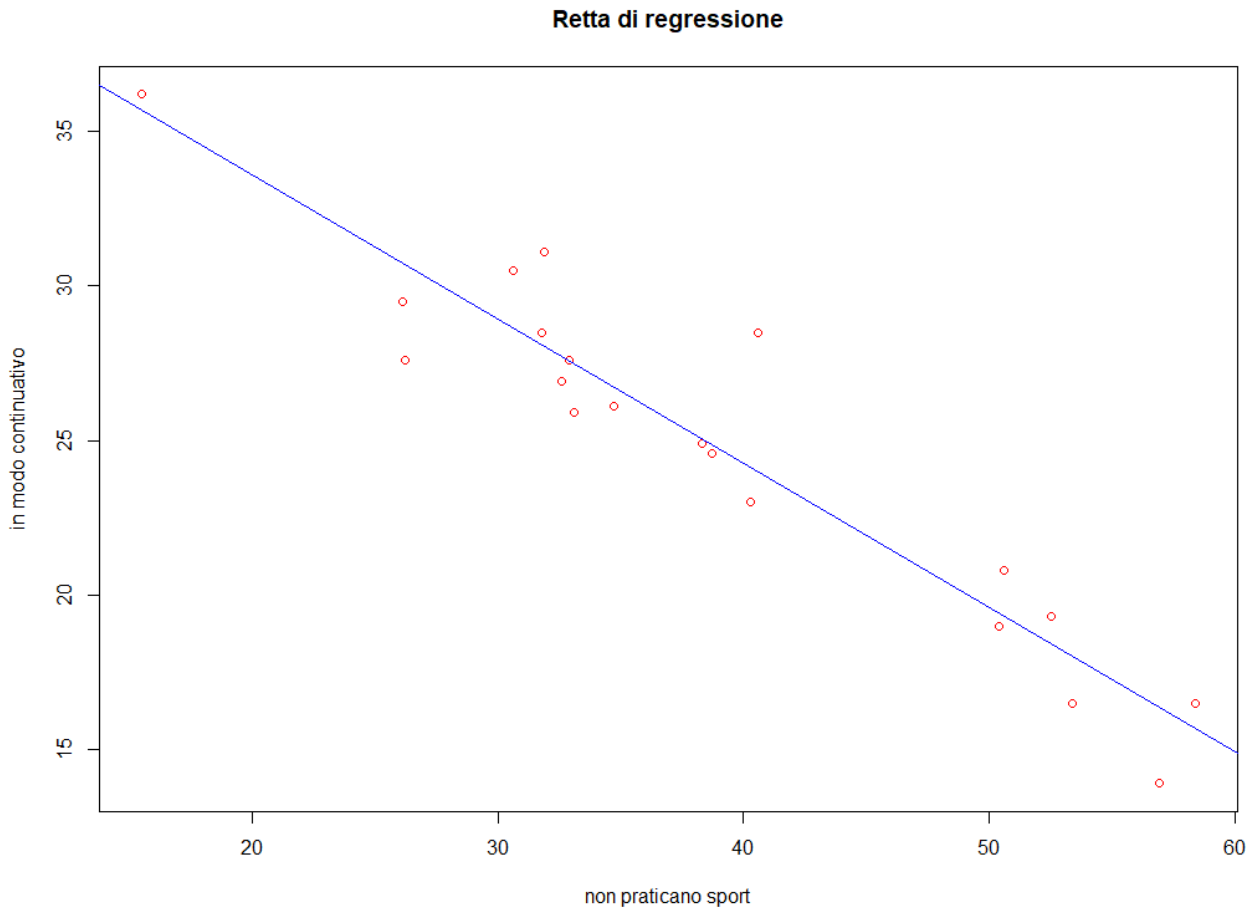
La funzione lm() mette a disposizione anche altri attributi:

```
> attributes(linearmodel)
$`names`
 [1] "coefficients"  "residuals"     "effects"       "rank"          "fitted.values" "assign"
 [7] "qr"            "df.residual"   "xlevels"       "call"          "terms"         "model"

$class
[1] "lm"

> linearmodel$coefficients
       (Intercept) non_praticano_sport
         42.9222310          -0.4662084
```

noi ovviamente, per il momento, ci concentriamo solo sui coefficienti di regressione.
La rappresentazione della retta y = 42.9222310 – 0.4662084 x, può essere aggiunta allo scatterplot facendo uso della funzione *abline(lm(y ~ x))*;

```
plot(non_praticano_sport, in_modo_continuativo, main =" Retta di regressione ",
     xlab="non praticano sport", ylab="in modo continuativo", col="red")
abline(lm(in_modo_continuativo ~ non_praticano_sport), col =" blue")
```

In questo modo il grafico che si ottiene risulta arricchito della linea di regressione, così come mostrato nella figura sottostante.



Residui

Una volta calcolati i coefficienti α e β e disegnata la retta di regressione che interpola la nuvola dei punti nel corrispondente scatterplot, è possibile osservare di quanto si discostano i valori osservati dai valori stimati.
Precedentemente, quando abbiamo parlato della somma Q dei quadrati degli errori, abbiamo già discusso di valori osservati ed abbiamo posto:

$$\hat{y}_i = \alpha + \beta x_i$$

dove $\hat{y}_i$ rappresentano i valori stimati attraverso la retta di regressione.
In generale possiamo affermare che la *media campionaria dei valori stimati è uguale alla media campionaria* $\bar{y}$ delle osservazione.
Se consideriamo invece i *residui* come la differenza tra valori osservati e valori stimati:

$$E_i = y_i - \hat{y}_i$$

Possiamo affermare che la *media dei residui* $\bar{E}$ *è sempre nulla* e come conseguenza si ha che la varianza dei residui $s_E^2 = \frac{1}{n-1}\sum_{i=1}^{n} E_i^2$.
Andiamo ora a calcolare i valori stimati attraverso la funzione *fitted():*

```
> stime <- (fitted(lm(in_modo_continuativo ~ non_praticano_sport), col="brown"))
> stime
       1        2        3        4        5        6        7        8        9       10       11
27.72384 28.09680 25.06645 28.65625 35.69600 30.75419 30.70757 28.05018 27.49073 24.87997 27.58397
      12       13       14       15       16       17       18       19       20
23.99417 24.13403 18.44629 16.39497 19.33209 19.42533 18.02670 15.69566 26.74480
```

ma possiamo ottenerli anche attraverso la seguente linea di codice

```
> linearmodel$fitted.values
       1        2        3        4        5        6        7        8        9       10       11
27.72384 28.09680 25.06645 28.65625 35.69600 30.75419 30.70757 28.05018 27.49073 24.87997 27.58397
      12       13       14       15       16       17       18       19       20
23.99417 24.13403 18.44629 16.39497 19.33209 19.42533 18.02670 15.69566 26.74480
```

Inoltre, le seguenti linee di codice, ci permetto di calcolare i residui attraverso la funzione *resid():*

```
> residui <- resid(lm(in_modo_continuativo ~ non_praticano_sport))
> residui
         1          2          3          4          5          6          7          8
-0.82383691  0.40319636 -0.16644899  1.84374628  0.50399934 -1.25419155 -3.10757071  3.04981720
         9         10         11         12         13         14         15         16
-1.59073271 -0.27996563  0.01602561  4.50583034 -1.13403218  0.85371038 -2.49497263  1.46791441
        17         18         19         20
-0.42532727 -1.52670205  0.80433998 -0.64479926
```

In alternativa, possiamo scrivere

```
> linearmodel$residuals
         1          2          3          4          5          6          7          8
-0.82383691  0.40319636 -0.16644899  1.84374628  0.50399934 -1.25419155 -3.10757071  3.04981720
         9         10         11         12         13         14         15         16
-1.59073271 -0.27996563  0.01602561  4.50583034 -1.13403218  0.85371038 -2.49497263  1.46791441
        17         18         19         20
-0.42532727 -1.52670205  0.80433998 -0.64479926
```

Come mostrato precedentemente, la media dei residui è zero, mentre la mediana, la varianza campionaria e la deviazione standard campionaria dei residui sono:

```
> median(linearmodel$residuals)
[1] -0.2232073
> var(linearmodel$residuals)
[1] 3.259701
> sd(linearmodel$residuals)
[1] 1.805464
```

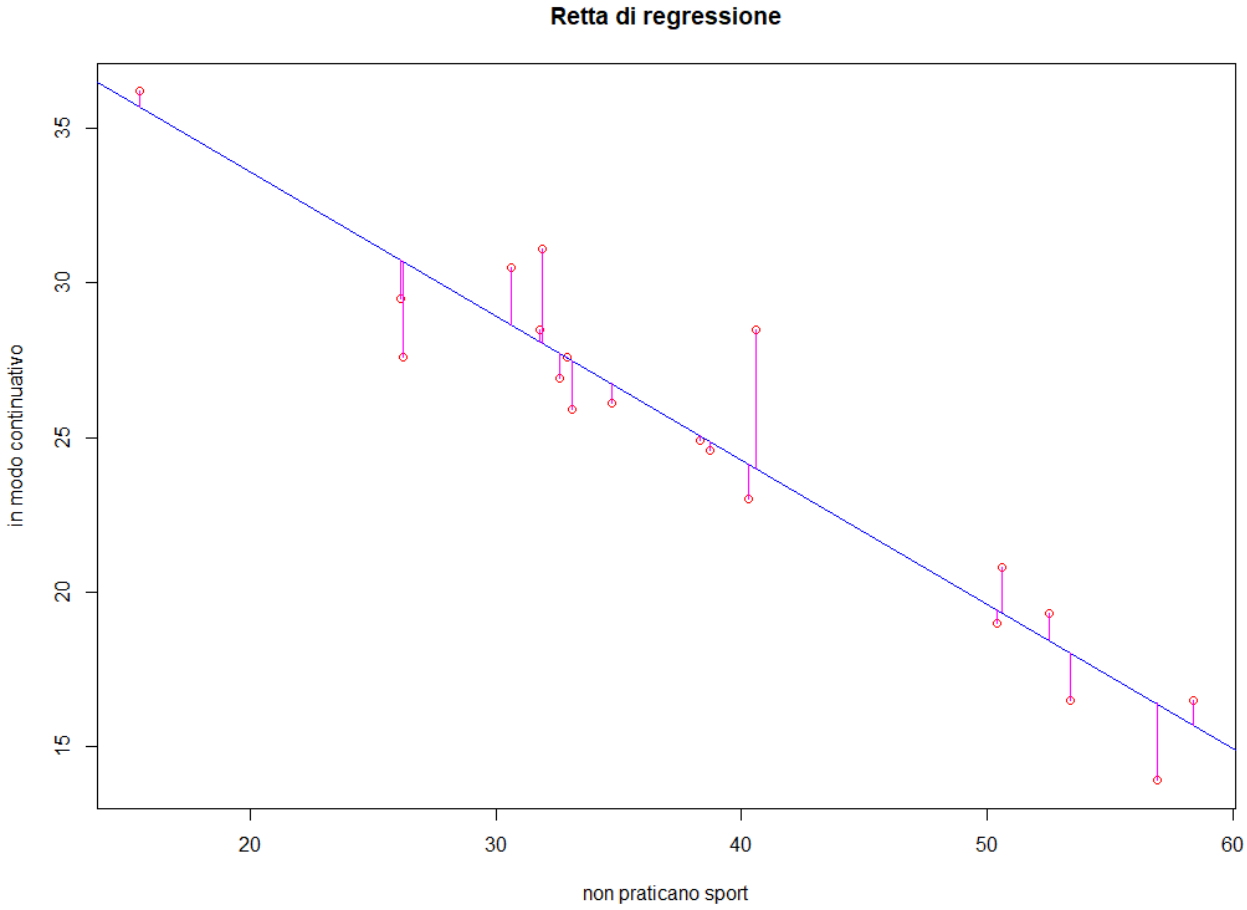
Non si può invece calcolare il coefficiente di variazione in quanto la media campionaria dei residui è nulla.

A questo punto possiamo rappresentare graficamente i residui in tre modi differenti:

1. tracciando dei segmenti verticali che congiungono i valori stimati e i valori osservati.

```
plot(non_praticano_sport, in_modo_continuativo, main =" Retta di regressione ",
     xlab="non praticano sport", ylab="in modo continuativo", col="red")
abline(lm(in_modo_continuativo ~ non_praticano_sport), col =" blue")
stime <- (fitted(lm(in_modo_continuativo ~ non_praticano_sport), col="brown"))
stime
segments(non_praticano_sport, stime, non_praticano_sport, in_modo_continuativo, col ="magenta")
```

   Questi segmenti sono ottenuti sovrapponendo al grafico in figura sottostante, dei segmenti che congiungono i due punti attraverso la funzione *segments()*.

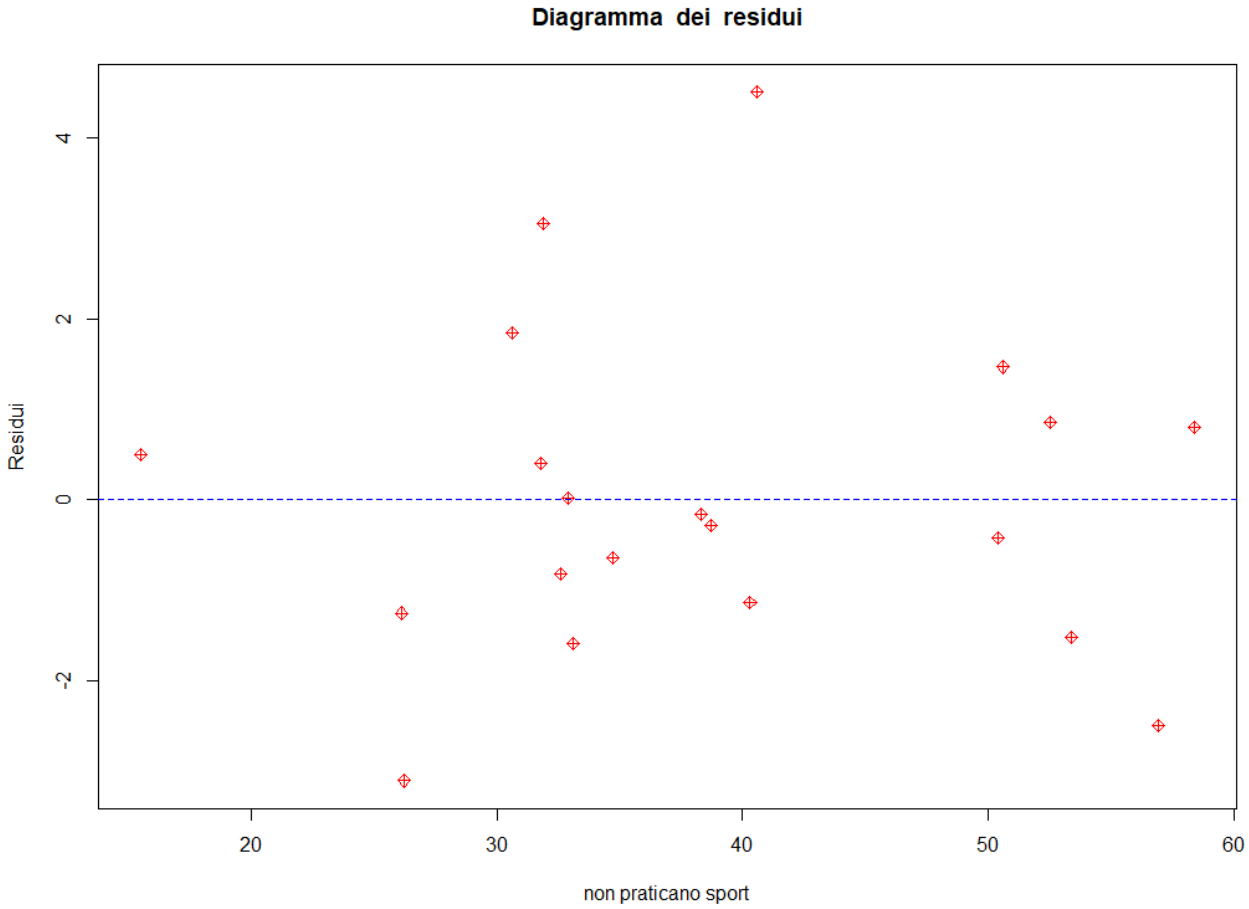


2. rappresentando i valori dei residui rispetto alla variabile indipendente.
   Un esame più accurato del modo con cui la retta di regressione interpola i dati e di come i residui si dispongano intorno alla retta interpolante influenzandone la posizione, può essere ottenuto attraverso il *diagramma dei residui* che è un grafico in cui i valori dei residui sono posti sull'asse delle ordinate e quelli della variabile indipendente sull'asse delle ascisse.

Desideriamo realizzare il diagramma dei residui ponendo i valori dei residui sull'asse delle ordinate e quelli del vettore "non_praticano_sport" sull'asse delle ascisse.
In precedenza avevamo già calcolato i residui e salvati in una variabile, perciò le seguenti linee di codice

```
plot(non_praticano_sport, residui, main="Diagramma  dei  residui",
     xlab="non praticano sport", ylab="Residui",pch =9, col ="red ")
abline(h=0, col ="blue ",lty =2)
```

producono il grafico illustrato in figura sottostante.



I punti indicano dove si collocano i residui rispetto ai valori del vettore "non_praticano_sport". La retta orizzontale è posta nello zero e corrisponde alla media campionaria dei residui che è nulla. Si nota che i punti sono disposti quasi casualmente intorno alla linea orizzontale e non si evidenza nessun comportamento particolare nella distribuzione dei punti.
Il diagramma dei residui aiuta a comprendere quale è l'adattamento della retta di regressione rispetto ai dati, consentendo di identificare quali sono le informazioni che hanno una forte influenza sulla collocazione e direzione della retta di regressione. Occorre notare che la posizione della retta di regressione è fortemente influenzata dalla presenza di eventuali valori anomali che si discostano in modo significativo dagli altri. L'analisi dei residui aiuta ad individuare eventuali punti isolati (valori anomali) dovuti ad errori nella stima. Tali valori possono perturbare significativamente la stima dei parametri di regressione e influenzare l'interpretazione dei residui. Eliminando i valori anomali la varianza campionaria dei residui diminuisce.

3. rappresentando i residui standardizzati rispetto ai valori stimati.

I *residui standardizzati* sono così definiti:

$$E_i^{(s)} = \frac{E_i - \bar{E}}{s_E} = \frac{E_i}{s_E}$$

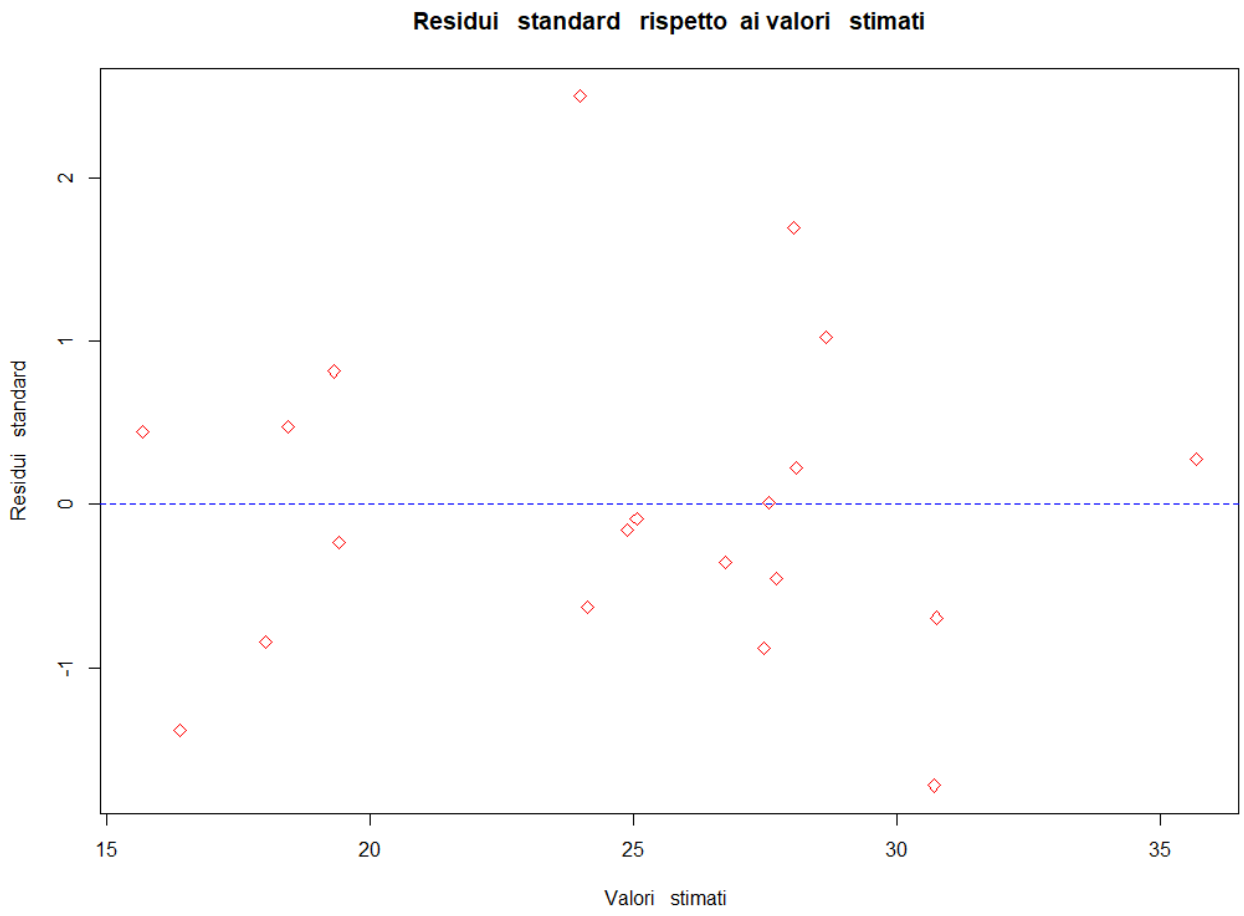
che risultano essere caratterizzati da media nulla e deviazione standard pari a 1.
Per il nostro specifico caso abbiamo:

```
> residuistandard <- residui/sd(residui)
> residuistandard
           1            2            3            4            5            6            7
-0.456302021  0.223320069 -0.092191805  1.021203518  0.279152235 -0.694664360 -1.721203284
           8            9           10           11           12           13           14
 1.689215105 -0.881065827 -0.155065743  0.008876173  2.495663236 -0.628111181  0.472848166
          15           16           17           18           19           20
-1.381901002  0.813039938 -0.235577808 -0.845600898  0.445503174 -0.357137681
```

È poi possibile realizzare un grafico in cui i residui standardizzati vengono disegnati in funzione dei valori stimati mediante la retta di regressione. Il seguente codice

```
plot(stime, residuistandard, main =" Residui   standard   rispetto  ai valori   stimati ",
     xlab="Valori   stimati ",ylab =" Residui   standard ",pch =5, col ="red ")
abline (h=0, col ="blue ",lty =2)
```

produce il grafico in figura sottostante.



I punti indicano la posizione dove si collocano i residui standardizzati rispetto ai valori stimati con la retta di regressione. La retta orizzontale è posizionata nello zero, che corrisponde alla media campionaria dei residui standardizzati. Anche in questo caso i punti sono disposti quasi casualmente attorno alla linea orizzontale e non si evidenzia nessuna tendenza particolare nella distribuzione dei punti.

Coefficiente di determinazione

Poiché si è interessati a vedere quanto la retta si adatta ai dati, l'accento può essere posto su un'altra misura chiamata *coefficiente di determinazione* che è così definito:

$$D^2 = \frac{\frac{1}{n-1}\sum_{i=1}^{n}(\hat{y}_i-\bar{y})^2}{\frac{1}{n-1}\sum_{i=1}^{n}(y_i-\bar{y})^2} = \frac{\sum_{i=1}^{n}(\hat{y}_i-\bar{y})^2}{\sum_{i=1}^{n}(y_i-\bar{y})^2}$$

ovvero rappresenta *rapporto tra la varianza dei valori stimati tramite la retta di regressione e la varianza dei valori osservati.*

È chiaro che il coefficiente di determinazione, nel caso di regressione lineare semplice, equivale al *quadrato del coefficiente di correlazione*. Possiamo affermare che se r_{xy}^2 molto vicino ad 1 indicherà che tutti i punti tenderanno ad allinearsi lungo la retta di regressione, mentre r_{xy}^2 prossimo a 0 esprime una completa incapacità della retta di rappresentare la distribuzione dei dati considerati.

Allo stesso modo, il coefficiente D^2 se è vicino allo 0 indica che tra le due variabili non esiste alcuna correlazione; al contrario se D^2 è vicino ad 1, allora le due variabili sono fortemente correlate tra loro.

Per calcolare tale indice, nel nostro specifico caso possiamo utilizzare il quadrato del coefficiente di correlazione

```
> (cor(non_praticano_sport, in_modo_continuativo))^2
[1] 0.8991839
```

oppure *summary(lm(y~x))$r.square*.

```
> summary(lm(in_modo_continuativo ~ non_praticano_sport))$r.square
[1] 0.8991839
```

Il coefficiente di correlazione calcolato mostra chiaramente la forte correlazione esistente tra le due variabili prese in considerazione. In generale, quindi, possiamo affermare che la retta considerata è adatta a rappresentare la distribuzione dei dati considerati.

Regressione lineare multipla

In molte applicazioni dell'analisi di regressione sono coinvolte situazioni con più di una singola variabile indipendente. È interessante quindi poter valutare la relazione tra la variabile "non_praticano_sport" e tutti gli altri vettori che compongono il nostro dataset ovvero "in_modo_continuativo", "in_modo_saltuario" e "solo_qualche_attività".

Per prima cosa, possiamo utilizzare le funzioni *cov()* e *cor()* applicate al nostro dataframe in modo da poter ottenere due matrici i cui elementi sono le covarianze e le correlazioni tra tutte le possibili coppie di variabili. Le due matrici ottenute sono simmetriche: per la matrice delle covarianze sulla diagonale principale vi sono le varianze, mentre per la matrice delle correlazioni sulla diagonale principale vi è sempre il numero 1. Ciò è possibile notarlo dalle seguenti linee di codice:

```
> #regressione lineare multipla
> cov(dataFrame)
                    in_modo_continuativo in_modo_saltuario solo_qualche_attivita non_praticano_sport
in_modo_continuativo             32.33313         12.815079             17.685974          -62.36145
in_modo_saltuario                12.81508          8.621342              7.158132          -28.40592
solo_qualche_attivita            17.68597          7.158132             19.624500          -44.04934
non_praticano_sport             -62.36145        -28.405921            -44.049342          133.76303
> cor(dataFrame)
                    in_modo_continuativo in_modo_saltuario solo_qualche_attivita non_praticano_sport
in_modo_continuativo            1.0000000         0.7675558             0.7021110          -0.9482531
in_modo_saltuario               0.7675558         1.0000000             0.5503170          -0.8364758
solo_qualche_attivita           0.7021110         0.5503170             1.0000000          -0.8597498
non_praticano_sport            -0.9482531        -0.8364758            -0.8597498           1.0000000
```

Si nota che esiste una forte correlazione lineare (negativa) tra il vettore "in_modo_continuativo" e "non_praticano_sport"; anche tra il vettore "non_praticano_sport e "solo_qualche_attività" esiste una correlazione lineare (negativa) ma meno forte rispetto alla prima relazione individuata già in precedenza. È inoltre, immediato notare che per quanto riguarda il vettore "non_praticano_sport" tutte le eventuali relazioni con le altre variabili possono essere rappresentate attraverso una retta discendente in quanto il coefficiente di correlazione è sempre minore di 0.
Volendo valutare le relazioni esistenti tra il vettore "non_praticano_sport" e tutti gli altri vettori presenti nella nostra matrice di dati, possiamo utilizzare il modello di *regressione lineare multipla* esprimibile attraverso l'equazione:

$$Y = \alpha + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_p Xp$$

in cui α è l'intercetta, β_1, β_2, … , β_p sono i *regressori*. In particolare ciascun β_i rappresenta l'inclinazione di Y rispetto alla variabile X_i.
Anche nel caso di regressione lineare multipla, per calcolare l'intercetta e i regressori, occorre considerare il metodo dei minimi quadrati e minimizzare la quantità:

$$Q = \sum_{i=1}^{n} \left[y_i - (\alpha + \beta_1 x_{i,1} + \beta_2 x_{i,2} + \ldots \beta_p x_{i,p}) \right]^2,$$

Il problema è quindi determinare i coefficienti in modo tale che Q sia minima e per fare ciò calcoliamo la derivata rispetto ad $\alpha, \beta_1, \beta_2, \ldots, \beta_p$ e imponiamo che sia uguale a 0.
Da ciò deriva che:

$$\alpha = \bar{y} - (\beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_p xp)$$

Per eseguire l'analisi di regressione lineare multivariata si usa la funzione *lm(y ~x₁ + x₂ + . . . x_p)*. L'argomento passato alla funzione indica che y è la variabile dipendente da x_1, x_2, …, x_p.
Riferendoci al nostro specifico caso:

```
> multipleLinearModel <- lm(non_praticano_sport ~ in_modo_continuativo + in_modo_saltuario + solo_qualche_attivita)
> attributes(multipleLinearModel)
$`names`
 [1] "coefficients"  "residuals"     "effects"       "rank"          "fitted.values" "assign"        "qr"
 [8] "df.residual"   "xlevels"       "call"          "terms"         "model"

$class
[1] "lm"

> multipleLinearModel$coefficients
          (Intercept)  in_modo_continuativo     in_modo_saltuario solo_qualche_attivita
           99.1516758            -0.9940880            -1.0003196            -0.9838472
```

Come nel caso della regressione lineare semplice, la funzione lm() restituisce una lista di attributi tra cui i coefficienti. È immediato notare che tutti e tre i regressori hanno segno negativo, ciò implica che hanno un effetto negativo sulla percentuale di persone che non praticano sport.

Residui

Una volta calcolati i valori dei coefficienti possiamo calcolarci i valori residui, ovvero gli scostamenti dei valori osservati dai valori stimati. Possiamo indicare i valori stimati con:

$$\hat{y}_i = \alpha + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_p x_p$$

Anche in questo caso *la media campionaria dei valori stimati coincide con la media campionaria dei valori osservati*.
I residui sono così calcolati:

$$E_i = y_i - \hat{y}_i$$

La *media campionaria dei residui* $\bar{E}$ nulla, ossia in media gli scostamenti positivi e negativi, si compensano.
La varianza campionaria dei residui è anche in questo caso: $s_E^2 = \frac{1}{n-1} \sum_{i=1}^{n} E_i^2$.
Nel nostro caso specifico, per calcolare i valori stimati:

```
> stimemult <- fitted(lm(non_praticano_sport ~ in_modo_continuativo + in_modo_saltuario + solo_qualche_attivita))
> stimemult
       1        2        3        4        5        6        7        8        9       10       11       12       13
32.76900 31.72265 38.45836 30.67924 15.46917 26.20122 26.36024 31.85206 33.10240 38.45649 32.91787 40.68599 40.19274
      14       15       16       17       18       19       20
52.42090 56.89857 50.59014 50.33648 53.50051 58.52143 34.36453
```

I residui invece, sono ottenuti attraverso la funzione *resid()*.
Infatti, le seguenti linee di codice:

```
> residuimult <- resid(lm(non_praticano_sport ~ in_modo_continuativo + in_modo_saltuario + solo_qualche_attivita))
> residuimult
           1            2            3            4            5            6            7            8            9
-0.169004439  0.077350850 -0.158362984 -0.079236473  0.030832535 -0.101216915 -0.160240248  0.047938632 -0.002402540
          10           11           12           13           14           15           16           17           18
 0.243506512 -0.017874161 -0.085991892  0.107260904  0.079098480  0.001427640  0.009860189  0.063519496 -0.100509985
          19           20
-0.121425085  0.335469486
```

determinano i residui.

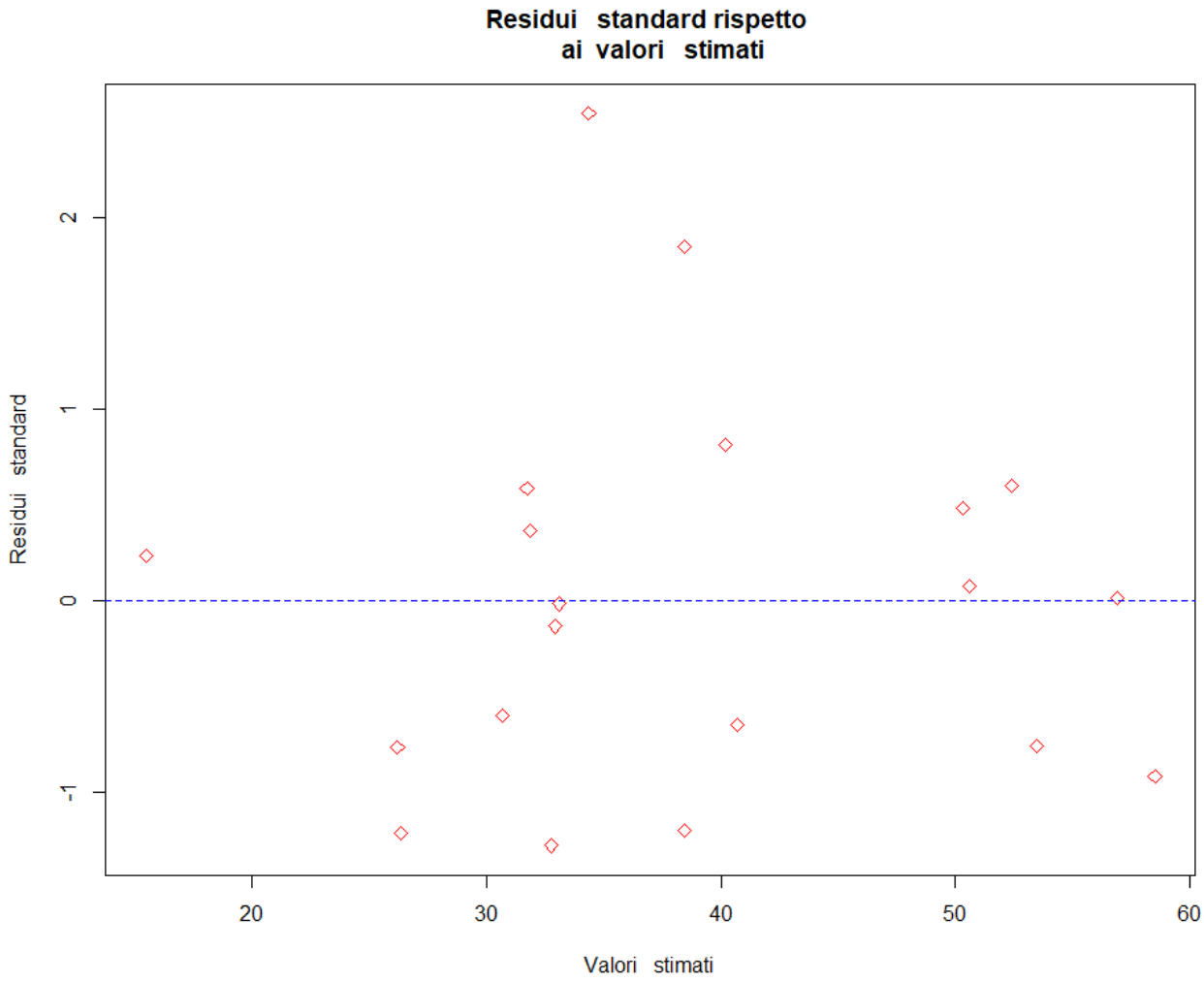
Anche nel caso multivariato è interessante calcolare i *residui standardizzati*

$$E_i^{(s)} = \frac{E_i - \bar{E}}{s_E} = \frac{E_i}{s_E}$$

che risultano essere caratterizzati da media nulla e varianza unitaria.
A questo punto possiamo considerare un grafico in cui i residui standardizzati vengono disegnati in funzione dei valori stimati. Il seguente codice

```
residuimultstandard <-residuimult/sd(residuimult)
plot(stimemult, residuimultstandard, main=" Residui   standard rispetto
     ai  valori   stimati ", xlab=" Valori   stimati ",
     ylab=" Residui   standard ",pch =5, col ="red ")
abline(h=0, col =" blue",lty =2)
```

produce il grafico in figura sottostante.



I punti indicano la posizione dove si collocano i residui standardizzati rispetto ai valori stimati con la retta di regressione. La retta orizzontale è posizionata nello zero, che corrisponde alla media campionaria dei residui standardizzati. Anche in questo caso i punti sono disposti quasi casualmente attorno alla linea orizzontale e non si evidenzia nessuna tendenza particolare nella distribuzione dei punti.

Coefficiente di determinazione

Anche in questo caso il coefficiente di determinazione è definito come il rapporto tra la varianza dei valori stimati tramite la funzione di regressione e la varianza dei valori osservati della variabile dipendente:

$$D^2 = \frac{\frac{1}{n-1}\sum_{i=1}^{n}(\hat{y}_i-\bar{y})^2}{\frac{1}{n-1}\sum_{i=1}^{n}(y_i-\bar{y})^2} = \frac{\sum_{i=1}^{n}(\hat{y}_i-\bar{y})^2}{\sum_{i=1}^{n}(y_i-\bar{y})^2}$$

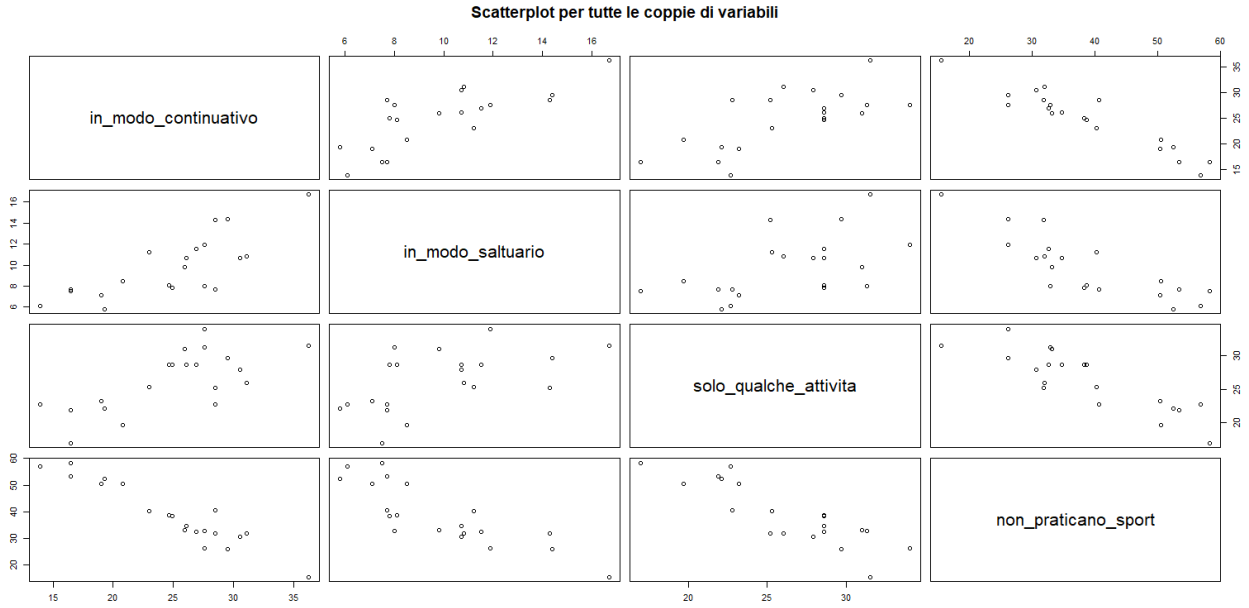
L'indice D^2 risulta essere compreso tra 0 e 1: come nel caso della regressione lineare sempre, se tale indice è più vicino allo 0 non vi è alcuna relazione tra le variabili, in caso contrario, il modello di regressione multipla utilizzato spiega perfettamente i dati.
Nel nostro specifico caso, utilizziamo la funzione *summary(lm(y~x1 + x2 + . . . xp))$r.square*:

```
> summary(lm(non_praticano_sport ~ in_modo_continuativo +
+              in_modo_saltuario + solo_qualche_attivita))$r.square
[1] 0.9998696
```

Il coefficiente di determinazione è quindi 0.9998696, ossia il modello di regressione multipla utilizzato può spiegare significativamente i dati.

Regressione non lineare

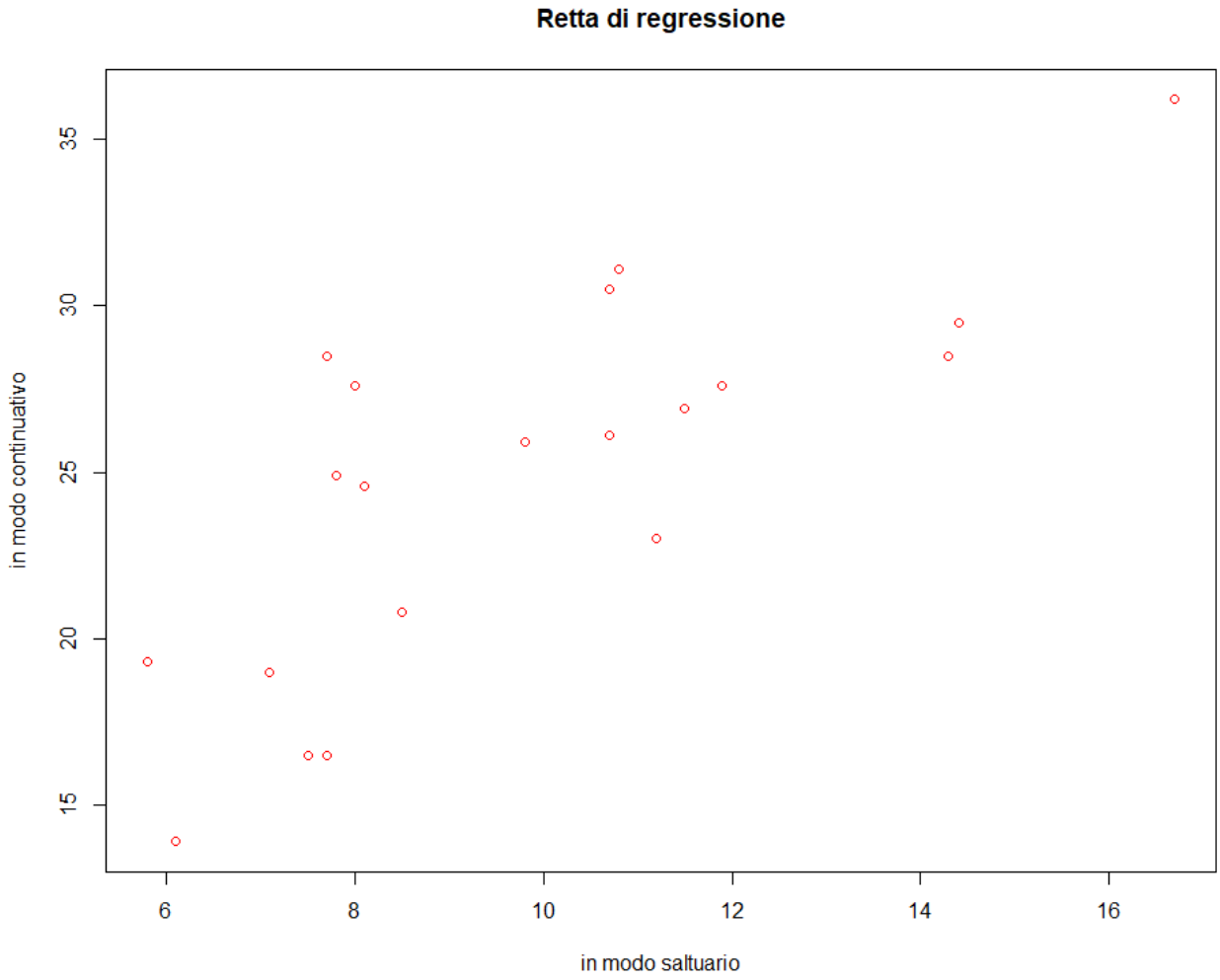
Molto spesso può capitare che la retta non la soluzione migliore per approssimare i nostri dati. Si tratta di quei casi in cui il coefficiente di determinazione risulta essere vicino allo 0. Allora, possiamo ricorrere *alla regressione non lineare*.



Se consideriamo il grafico in figura, è possibile notare che quasi tutte le coppie considerate presentano relazioni lineari e quindi esprimibili attraverso una retta.
Ma andiamo ad esaminare nel dettaglio la coppia "in_modo_saltuaro" – "in_modo_continuativo".
Le seguenti linee di codice:

```
plot(in_modo_saltuario, in_modo_continuativo, main =" Retta di regressione ",
     xlab="in modo saltuario", ylab="in modo continuativo", col="red" )
```

forniscono lo scatterplot in figura.



Osservando il grafico si nota che un modello lineare non approssima efficacemente i nostri dati. Ma per esserne sicuri, calcoliamo il coefficiente di determinazione tramite la regressione lineare:

```
> summary(lm(in_modo_saltuario ~ in_modo_continuativo))$r.square
[1] 0.5891419
```

e scopriamo immediatamente che 0.5891419 è un valore abbastanza basso.
A questo punto, possiamo considerare il modello non lineare

$$Y = \alpha + \beta X + \gamma X^2$$

Per la stima dei parametri α, β e γ si può ricorrere alla regressione multipla

$$Y = \alpha + \beta X_1 + \gamma X_2$$

con regressori X_1=X e X_2=X^2.
Con R è facile stimare i parametri α, β e γ tramite la funzione *lm(y ~ x + I(x ^ 2))* dove I() è un *identificatore di variabile* e viene inserito quando si devono effettuare operazioni matematiche nelle variabili della regressione.
Ma vediamo se l'approssimazione polinomiale è adeguata per stimare i nostri dati. Le seguenti linee di codice

```
lm <- lm(in_modo_continuativo ~ in_modo_saltuario + I(in_modo_saltuario^2))
alpha2 <- lm$coefficients[[1]]
beta2 <- lm$coefficients[[2]]
gamma <- lm$coefficients[[3]]
> c(alpha2, beta2, gamma)
[1]  2.3503127  3.0663858 -0.0727283
```

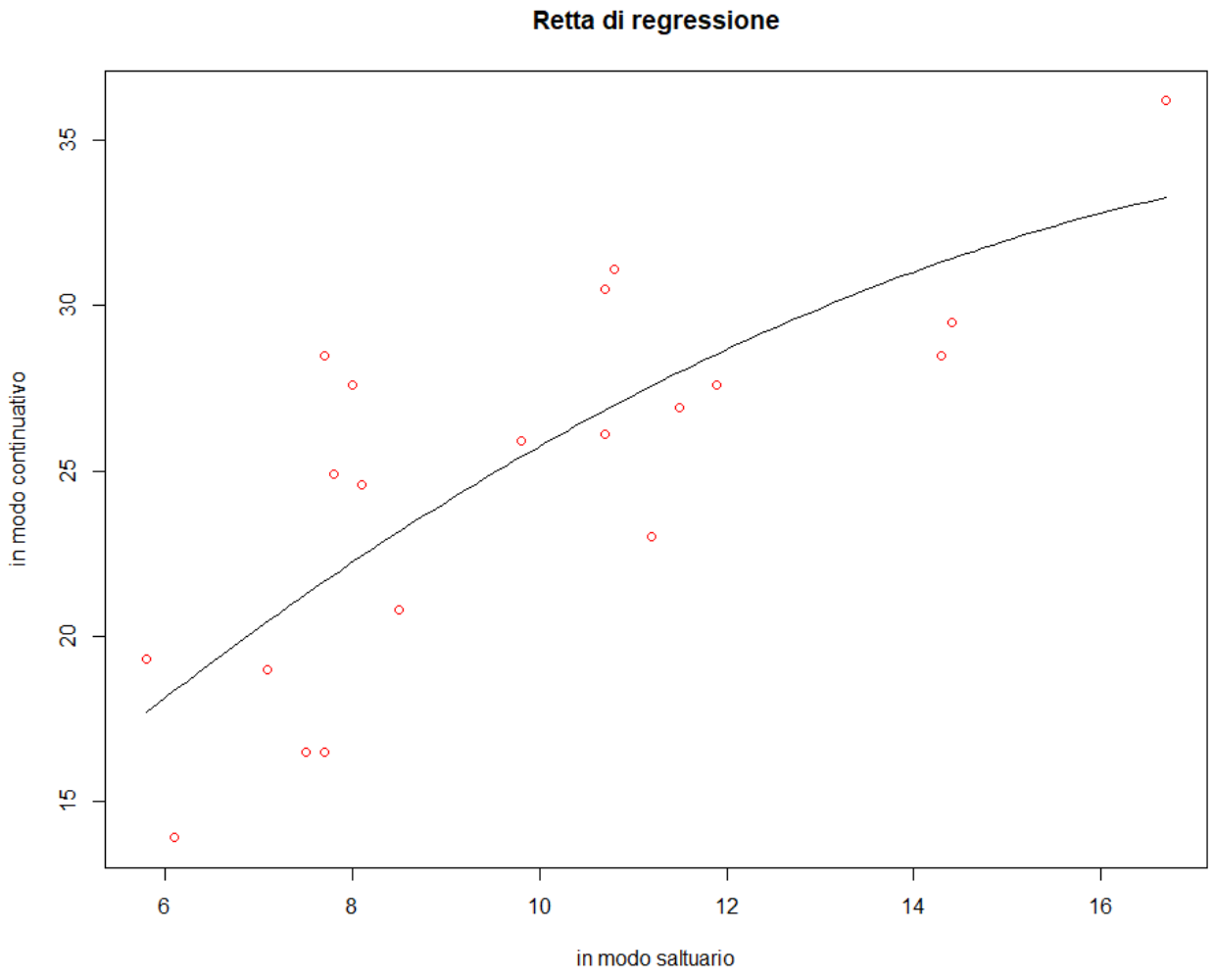
Mostrano che $\alpha = 2.3503127, \beta = 3.0663858$ e $\gamma = -0.0727283$. Il modello polinomiale è quindi $Y = 2.3503127 - 3.0663858X - 0.0727283X^2$.
Determiniamo ora il coefficiente D^2:

```
> summary(lm(in_modo_continuativo ~ in_modo_saltuario + I(in_modo_saltuario^2)))$r.square
[1] 0.6035978
```

È immediato notare che in questo caso $D^2 = 0.6035978$ che risulta essere maggiore di $D^2 = 0.5891419$ nel caso di regressione lineare semplice.
Ciò indica che il modello di regressione polinomiale è più adatto a spiegare i nostri dati rispetto al modello di regressione lineare semplice.
A questo punto, possiamo disegnare la curva stimata sullo scatterplot in figura. La seguente linea di codice

```
curve(alpha2+beta2*x+gamma*x^2, add = TRUE)
```

produce il grafico in figura sottostante.



ANALISI DEI CLUSTER

L'analisi dei cluster è una metodologia che permette di raggruppare in sottoinsiemi, detti *cluster*, entità appartenenti ad un insieme più ampio. I vari metodi attraverso cui si attua l'analisi dei cluster hanno in comune uno scopo generale: *ottenere raggruppamenti in base alla somiglianza in modo che gli elementi di uno stesso gruppo siano tra loro il più possibile simili e gli elementi appartenenti a gruppi distinti siano tra loro il più possibile diversi.*
In questo modo si otterrà un'alta omogeneità all'interno dei cluster mentre, un'alta eterogeneità tra gruppi distinti.
Il problema dell'analisi dei cluster consiste nel determinare m sottoinsiemi, detti cluster, di individui in I, con m intero minore di n, tali che I_i appartenga soltanto ad un unico sottoinsieme. Gli individui che sono assegnati allo stesso cluster sono detti *simili* mentre gli individui che sono assegnati a differenti cluster sono detti *dissimili*.
Consideriamo il nostro dataset costituito da 20 individui rappresentanti le 20 regioni italiane. Vogliamo suddividere le nostre 20 regioni in un certo numero fissato di cluster in modo tale che le regioni all'interno dello stesso cluster siano più simili rispetto a quelle inserite in cluster differenti.
In molti metodi di clustering si raccomanda la standardizzazione di ciascuna variabile (caratteristica osservata) usando la media campionaria e la deviazione standard campionaria in modo da poter far si che tutti i dati all'interno della matrice dei dati siano privi di unità di misura. A questo punto la media delle colonne della matrice è pari a zero, la varianza è unitaria.
Se consideriamo l'elemento x_{ij}, possiamo effettuare uno scalamento:

$$\frac{x_{ij} - \bar{x_j}}{s_j}$$

Sottraendo la media campionaria della caratteristica j-esima e dividendo per la deviazione standard della j-esima caratteristica. In R tutto questo è possibile attraverso la funzione *scale(X, center = TRUE, scale= TRUE).*
Nel caso della nostra matrice dei dati, potremmo anche evitare di effettuare lo scalamento in quanto tutte le caratteristiche prese in considerazione sono prive di unità di misura (sono espresse in percentuali). Ma andiamo comunque a scalare la matrice attraverso la seguente linea di codice:

```
Z <- scale(matriceDati)
```

A questo punto è possibile cominciare la vera e propria analisi dei cluster!
Per risolvere il problema legato al clustering è chiaramente necessario definire i termini di *somiglianza* o *differenza*. A tal proposito possiamo parlare di:

- distanza tra la regione i-esima e la regione j-esima, oppure
- similarità tra regione i-esima e regione j-esima.

Distanza e similarità

le misure metriche di somiglianza si basano per lo più sulla distanza esistente tra i vettori delle caratteristiche. Si parla di funzione distanza se e soltanto se essa soddisfa queste condizioni:

1. $d(X_i,X_j) = 0$ se e solo se $X_i=X_j$;
2. $d(X_i,X_j) \geq 0$ per ogni X_i e X_j;
3. $d(X_i,X_j) = d(X_j,X_i)$ per ogni X_i e X_j;
4. $d(X_i,X_j) \leq d(X_i,X_k) + d(X_k,X_j)$ per ogni X_i, X_j e X_k.

La proprietà (4) è nota come *disuguaglianza triangolare* e afferma che la distanza tra X_i eX_j è sempre minore o uguale della somma delle distanze di ognuno dei due vettori considerati da qualunque altro terzo vettore X_k.
Le distante le possiamo inserire all'interno di una matrice delle distanze simmetrica in cui sulla diagonale principale abbiamo tutti zero.
In R la funzione *dist(X,method = "euclidean", diag = FALSE, upper = FALSE)* restituisce la matrice delle distanze calcolata selezionando la misura di distanza da utilizzare.
Le opzioni per il calcolo della matrice delle distanze sono:

(1) metrica euclidea (euclidean);
(2) metrica del valore assoluto o metrica di Manhattan (manhattan);
(3) metrica del massimo o metrica di Chebycev (maximum);
(4) metrica di Minkowski (minkowski);
(5) distanza di Canberra (canberra);
(6) distanza di Jaccard (binary).

Andiamo ad esaminare ciascuna distanza prendendo in esame le nostre 20 regioni ed i valori che assumo i vettori relativi alle caratteristiche: "in_modo_continuativo", "in_modo_saltuario", "solo_qualche_attivita" e "non_praticano_sport".
La ***distanza euclidea*** è calcolata come:

$$d_2(X_i, X_j) = \left[\sum_{k=1}^{p} (X_{ik} - X_{jk})^2\right]^{1/2}$$

La matrice delle distanze calcolata con la metrica euclidea risente delle unità di misura pertanto è necessario scalare la matrice.
Nel nostro specifico caso, andiamo a calcolare la matrice delle distante con il metodo "euclidean" in questo modo:

```
dist(z, method="euclidean", diag=TRUE, upper=TRUE)
```

Possiamo considerare anche la distanza del valore assoluto (o di Manhattan)

$$d_1(X_i, X_j) = \sum_{k=1}^{p} |X_{ik} - X_{jk}|$$

che nel nostro caso specifico risulta essere calcolata così:

```
dist(z, method="manhattan", diag=TRUE, upper=TRUE)
```

Se invece consideriamo la distanza del massimo così definita:

$$d_\infty(X_i, X_j) = \max_{k=1\ldots p} |X_{ik} - X_{jk}|$$

per applicarla alla nostra matrice di dati consideriamo la seguente linea di codice:

```
dist(z, method="maximum", diag=TRUE, upper=TRUE)
```

Adesso, invece, consideriamo la metrica di Minkowski che include come caso particolare la metrica del valore assoluto, la metrica euclidea e quella di Chebycev. Infatti,

$$d_r(X_i, X_j) = \left[\sum_{k=1}^{p} |x_{ik} - x_{jk}|^r\right]^{1/r}$$

se consideriamo r=1 otteniamo la metrica del valore assoluto; con r pari a due si ha la distanza euclidea e con r = ∞, cioè man mano che cresce r, possiamo ottenere la metrica di Chebycev. Applichiamo la metrica di Minkowski al nostro insieme di dati:

```
dist(z, method="minkowski", diag=TRUE, upper=TRUE)
```

La distanza di canberra invece, può essere applicata solo a dati non negativi. Per questo motivo, al posto della matrice scalata utilizziamo la matrice originale dei dati, in quanto tale metrica non è sensibile all'unità di misura dei vettori delle caratteristiche. Prima di vedere come applicare tale metrica, vediamo come essa è definita:

$$d_c(X_i, X_j) = \sum_{k=1}^{p} \frac{|x_{ik} - x_{jk}|}{|x_{ik} + x_{jk}|}$$

Le seguenti linee di codice

```
dist(matriceDati, method="canberra", diag=TRUE, upper=TRUE)
```

permettono di calcolare la matrice delle distanze secondo la metrica di canberra.

Infine, l'ultima metrica prende il nome di distanza di Jaccard che nel nostro caso non può essere applicata in quanto R mette a disposizione solo la versione binaria di questo tipo di distanza. In generale però la metrica di Jaccard è così definita:

$$d(X_i, X_j) = 1 - \frac{\sum_{k=1}^{p} \min(x_{ik}, x_{jk})}{\sum_{k=1}^{p} \max(x_{ik}, x_{jk})}$$

Misure di similarità

Una funzione è detta di similarità se e solo se soddisfa le seguenti condizioni:

1. $s(X_i,X_j) = 1$ se $X_i=X_j$;
2. $0 \leq s(X_i,X_j) \leq 1$;
3. $s(X_i,X_j) = s(X_j,X_i)$ per ogni X_i e X_j.

La più ovvia differenza tra una misura di similarità e una misura di distanza è che la prima asume valori compresi tra 0 e 1 mentre la seconda assume valori non negativi. In generale però possiamo affermare che la misura di distanza è più forte rispetto alla misura di similarità poiché quest'ultima non gode della proprietà della disuguaglianza triangolare. È importante comunque sottolineare che è sempre possibile trasformare una misura di distanza in una misura di similarità, il contrario però non può mai accadere.
Ad esempio, se d_{ij} è la distanza tra X_i e X_j, allora mediante la trasformazione $s_{ij} = 1/1 + d_{ij}$ (i, j = 1, 2, . . . , n) si ottiene una misura di similarità tra X_i e X_j .

Misura di non omogeneità totale

In questa sede analizziamo il concetto di misura di non omogeneità totale partendo da una matrice delle distanze n x n.

Come già sappiamo è possibile costruire una matrice delle covarianze attraverso la semplice funzione cov(X) in cui sulla diagonale principale compaiono le varianze delle singole caratteristiche prese in considerazione.
A partire dalla matrice delle covarianze è possibile considerare un'altra matrice che prende il nome di "matrice statistica di non omogeneità", così ottenuta:

$$H_I = (n-1)W_x$$

Dove W_x è la matrice delle covarianze. In generale sappiamo che l'elemento h_{rr} nella matrice statistica di non omogeneità è dato da $(n-1)s_r^2$.
Siamo però interessati a calcolare *la traccia della matrice* H_I, ovvero la somma degli elementi sulla diagonale principale. Otteniamo così:

$$trH_I = \sum_{r=1}^{p} h_{rr} = (n-1)\sum_{r=1}^{p} s_r^2$$

La traccia della matrice H_I può essere anche calcolata come:

$$tr\,H_I = \frac{1}{n}\sum_{i=1}^{n}\sum_{j=i}^{n} d_2^2(x_i, x_j)$$

La traccia di non omogeneità totale risulta dipendere da n (ovvero dalla numerosità del campione) ma risulta dipendere anche dalla varianza delle varie caratteristiche (omogeneità interna).

Misure di non omogeneità tra cluster

Abbiamo considerato delle misure di non omogeneità relative all'insieme totale di individui della popolazione. Occorre però considerare anche misure di non omogeneità all'interno dei cluster e delle misure di non omogeneità (o disparità) tra cluster distinti. Al termine del procedimento di classificazione, gli individui appartenenti allo stesso cluster dovrebbero essere il più possibile omogenei tra di loro e il più possibile differenti da quelli appartenenti agli altri cluster individuati.
Essenzialmente andiamo a considerare una misura di non omogeneità interna ai cluster (within) e una misura di non omogeneità tra i cluster (between).
Possiamo considerare:

$$T = W + B$$

In cui T è la misura di non omogeneità totale ed è fissata. Le misure di non omogeneità W e B invece dipendono dalla partizione.
Poiché trT è univocamente determinata per ogni matrice X di cardinalità n x p, fissato il numero m di suddivisioni, i cluster dovrebbero essere individuati in modo da *minimizzare la misura di non omogeneità statistica all'interno dei cluster (within) e massimizzare la misura di non omogeneità statistica tra i gruppi (between).*
Le misure di non omogeneità appena descritte, verranno impiegate nel seguito per mostrare quale suddivisione in cluster risulta essere la migliore.
Per poter ottenere una suddivisione delle nostre 20 regioni in un certo fissato numero di cluster ci possiamo affidare a *metodi gerarchi* e *metodi non gerarchici*.

Metodi non gerarchici

Vogliamo, prima di ogni cosa, analizzare i metodi non gerarchici, in maniera tale da poter decidere il numero di partizioni ottimale per poi continuare la nostra analisi utilizzando il numero di partizioni stabilito con i metodi gerarchici. Nella fase successiva, poi, analizzeremo le differenze riscontrate tra i due diversi metodi di partizionamento degli individui. Come abbiamo già detto, l'idea quella di ottenere un'unica partizione degli n individui di partenza in cluster. I metodi non gerarchici, usualmente, richiedono inizialmente la determinazione di un insieme iniziale di punti di riferimento (ad esempio un insieme iniziale di punti attorno ai quali si addensano i cluster) oppure l'individuazione di una partizione iniziale degli n individui in cluster. Gli algoritmi di tipo non gerarchico procedono, data una prima partizione, a riallocare gli individui nel gruppo con centroide più vicino, fino a che per nessun individui si verifica che sia minima la distanza rispetto al centroide di un gruppo diverso da quello a cui esso appartiene. Il metodo più utilizzato è il *k-means* che consiste nei seguenti passi:

1. fissare a priori il numero m di cluster specificando m punti iniziali (scegliendo alcuni individui o prendendo una configurazione determinata con una tecnica gerarchica) che inducono una prima partizione provvisoria;
2. considerare tutti gli individui e attribuire ciascuno di essi al cluster individuato dal punto di riferimento da cui ha distanza minore;
3. calcolare il centroide di ognuno degli m gruppi così ottenuti. Tali centroidi costituiscono i punti di riferimento per i nuovi cluster;
4. valutare la distanza di ogni unità dal centroide ottenuto al passo precedente.

Ripetiamo la procedura fino ad ottenere una partizione che risulta essere stabile e gli individui non devono essere spostati, oppure possiamo scegliere di fermarci dopo un certo numero massimo di iterazioni. Nel metodo k-means, per garantire la convergenza della procedura iterativa, come misura di distanza tra i vettori delle caratteristiche e i centroidi viene utilizzata la distanza euclidea e, come per il metodo del centroide, si considera
la matrice contenente i *quadrati delle distanze euclidee*.
Siccome inizialmente possiamo scegliere casualmente gli individui da inserire nei cluster, può accadere che la classificazione finale sia influenzata dalla scelta iniziale degli m individui. Infatti, l'algoritmo potrebbe convergere ad un ottimo locale e non globale il che significa che se si inizio con un diverso insieme di individui si può giungere ad una differente partizione finale.
In R per applicare il metodo del k-means si usa proprio la funzione kmeans(X, centers, iter.max=N, start=M), dove X è la matrice dei dati, centers è il numero dei cluster che si vogliono identificare o un vettore di lunghezza pari al numero di cluster contenente un insieme di centroidi iniziali dei cluster, iter.max è il massimo numero di iterazioni permesse (di default iter.max = 10), nstart fornisce il numero di volte in cui ripetere la procedura di scelta casuale dei punti di riferimenti, nel caso in cui centers è il minimo (di default nstart = 1). Si nota che nell'algoritmo *k*-means *non occorre calcolare la matrice iniziale delle distanze* così come invece si richiede nei metodi gerarchici.
Ma vediamo come è stata applicata ed i risultati ottenuti nella nostra analisi specifica.
Andiamo a considerare il caso in cui scegliamo un numero di partizioni pari a 2:

```
> #metodi non gerarchici: kmeans
> km <- kmeans(matriceDati,centers = 2, iter.max = 10, nstart = 10)
> km #visualizza km
K-means clustering with 2 clusters of sizes 14, 6

Cluster means:
  in modo continuativo in modo saltuario solo qualche attività non praticano sport
1             27.92143         10.971429              28.50714            32.37857
2             17.66667          7.116667              21.10000            53.70000

Clustering vector:
 Piem   VdA   Lig  Lomb   TAA   Ven   FVG    ER   Tos   Umb   Mar Lazio   Abr   Mol  Camp    Pu   Bas   Cal
    1     1     1     1     1     1     1     1     1     1     1     1     1     2     2     2     2     2
  Sic   Sar
    2     1

Within cluster sum of squares by cluster:
[1] 930.5650 118.0817
 (between_SS / total_SS =  71.6 %)

Available components:

[1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss" "betweenss"    "size"
[8] "iter"         "ifault"
```

Il metodo k-means individua la seguente partizione in due cluster C_1 = {Piemonte, Valle d'Aosta, Liguria, Lombardia, Trentino Alto Adige, Veneto, Friuli Venezia Giulia, Emilia Romagna, Toscana, Umbria, Marche, Lazio, Abbruzzo, Sardegna} e C_2 = {Molise, Campania, Puglia, Basilicata, Calabria, Sicilia}. I valori 930.5650 e 118.0817 rappresentano le misure di non omogeneità statistiche associate ai due cluster C_1 e C2. La funzione str(km) permette di ottenere una lista di utili informazioni relative all'oggetto km creato con il metodo kmeans() ottenibili considerando: km$cluster, km$centers, km$totss, km$withinss, km$tot.withinss, km$betweenss e km$size. In particolare:

```
> str(km)
List of 9
 $ cluster     : Named int [1:20] 1 1 1 1 1 1 1 1 1 1 ...
  ..- attr(*, "names")= chr [1:20] "Piem" "VdA" "Lig" "Lomb" ...
 $ centers     : num [1:2, 1:4] 27.92 17.67 10.97 7.12 28.51 ...
  ..- attr(*, "dimnames")=List of 2
  .. ..$ : chr [1:2] "1" "2"
  .. ..$ : chr [1:4] "in modo continuativo" "in modo saltuario" "solo qualche attività" "non praticano sport"
 $ totss       : num 3692
 $ withinss    : num [1:2] 931 118
 $ tot.withinss: num 1049
 $ betweenss   : num 2644
 $ size        : int [1:2] 14 6
 $ iter        : int 1
 $ ifault      : int 0
 - attr(*, "class")= chr "kmeans"
> km$cluster #dove sono posizionati gli individui
 Piem   VdA   Lig  Lomb   TAA   Ven   FVG    ER   Tos   Umb   Mar Lazio   Abr   Mol  Camp    Pu   Bas   Cal
    1     1     1     1     1     1     1     1     1     1     1     1     1     2     2     2     2     2
  Sic   Sar
    2     1
> km$centers #centroidi
  in modo continuativo in modo saltuario solo qualche attività non praticano sport
1             27.92143         10.971429              28.50714            32.37857
2             17.66667          7.116667              21.10000            53.70000
> km$totss #misura di non omogeneità totale
[1] 3692.498
> km$withinss #misure di non omogeneità interne ai cluster
[1] 930.5650 118.0817
> km$tot.withinss
[1] 1048.647
> km$betweenss
[1] 2643.851
> km$size
[1] 14  6
```

Si nota che la misura di non omogeneità totale 3692.498, la somma delle misure di non omogeneità totale interne ai cluster è 1048.647 (within) e la misura di non omogeneità tra i cluster è 2643.851 (between). Ma andiamo a vedere se la nostra suddivisione in gruppi risulta essere buona:

```
> km$betweenss/km$totss
[1] 0.7160062
```

Ciò vuol dire che trB/trT è pari a 0.72 il che ci suggerisce di aumentare il numero di cluster in quanti 0.71 sembra essere piccolo. Andiamo quindi a suddividere i nostri individui in 3 cluster:

```
> #metodi non gerarchici: kmeans
> km <- kmeans(matriceDati,centers = 3, iter.max = 10, nstart = 10)
> km #visualizza km
K-means clustering with 3 clusters of sizes 3, 6, 11

Cluster means:
  in modo continuativo in modo saltuario solo qualche attività non praticano sport
1             31.10000          14.333333               31.73333            22.60000
2             17.66667           7.116667               21.10000            53.70000
3             27.05455          10.054545               27.62727            35.04545

Clustering vector:
 Piem   VdA   Lig  Lomb   TAA   Ven   FVG    ER   Tos   Umb   Mar Lazio   Abr   Mol  Camp    Pu   Bas   Cal
    3     3     3     3     1     1     1     3     3     3     3     3     3     2     2     2     2     2
  Sic   Sar
    2     3

Within cluster sum of squares by cluster:
[1] 137.2933 118.0817 306.7036
 (between_SS / total_SS =  84.8 %)

Available components:

[1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss" "betweenss"    "size"
[8] "iter"         "ifault"
> km$cluster #dove sono posizionati gli individui
 Piem   VdA   Lig  Lomb   TAA   Ven   FVG    ER   Tos   Umb   Mar Lazio   Abr   Mol  Camp    Pu   Bas   Cal
    3     3     3     3     1     1     1     3     3     3     3     3     3     2     2     2     2     2
  Sic   Sar
    2     3
> km$centers #centroidi
  in modo continuativo in modo saltuario solo qualche attività non praticano sport
1             31.10000          14.333333               31.73333            22.60000
2             17.66667           7.116667               21.10000            53.70000
3             27.05455          10.054545               27.62727            35.04545
> km$totss #misura di non omogeneità totale
[1] 3692.498
> km$withinss #misure di non omogeneità interne ai cluster
[1] 137.2933 118.0817 306.7036
> km$tot.withinss
[1] 562.0786
> km$betweenss
[1] 3130.419
> km$size
[1]  3  6 11
> km$betweenss/km$totss
[1] 0.8477782
```

In questo caso C_1 = {TAA, Ven, FVG}, C_2= {Mol, Camp, Pu, Bas, Cal, Sic} e C_3 = {Piem, VDA, Lig, Lomb, ER, Tos, Umb, Mar, Lazio, Abr, Sar}.
Quello che si ottiene è che trB/trT è 0.8477782. Da ora in poi quindi considereremo un numero di cluster pari a 3, anche per il metodi gerarchici.

Per poter determinare un altro partizionamento possiamo scegliere i centroidi ad esempio individuati con il metodo gerarchico del centroide:

```
> #scegliamo i centroidi calcolati con un metodo gerarchico
> d <- dist(matriceDati, method = "euclidean")
> d2 <- d^2
> tree <- hclust(d2, method="centroid")
> taglio <- cutree(tree, k=3)
> taglioLista <- list(taglio)
> centroidi <- aggregate(matriceDati, taglioLista, mean)[,-1]
> kmeans(matriceDati, centers=centroidi, iter.max = 10)
K-means clustering with 3 clusters of sizes 11, 3, 6

Cluster means:
  in modo continuativo in modo saltuario solo qualche attività non praticano sport
1             27.05455          10.054545              27.62727            35.04545
2             31.10000          14.333333              31.73333            22.60000
3             17.66667           7.116667              21.10000            53.70000

Clustering vector:
 Piem   VdA   Lig  Lomb   TAA   Ven   FVG    ER   Tos   Umb   Mar Lazio   Abr   Mol  Camp    Pu   Bas   Cal
    1     1     1     1     2     2     2     1     1     1     1     1     1     3     3     3     3     3
  Sic   Sar
    3     1

Within cluster sum of squares by cluster:
[1] 306.7036 137.2933 118.0817
 (between_SS / total_SS =  84.8 %)

Available components:

[1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss" "betweenss"    "size"
[8] "iter"         "ifault"
```

In questo caso, C_1= {Piem, VDA, Lig, Lomb, ER, Tos, Umb, Mar, Lazio, Abr, Sar}, C_2 = {TAA, Ven, FVG} e C_3 = {Mol, Camp, Pu, Bas, Cal, Sic} che è la stessa partizione in gruppi ottenuta scegliendo casualmente i 3 individui da inserire nei primi 3 cluster iniziali.
Anche in questo caso trB/trT = 0.84. Sicuramente possiamo affermare gli individui inseriti nello stesso cluster sono più simili e quindi quasi tutte le regioni del nord sono inserito nello stesso cluster eccetto TAA FVG e Ven e quasi tutte le regioni del Sud sono inserite nello stesso cluster.

Metodi gerarchici

I metodi di clustering gerarchico possono essere *agglomerativi* e divisivi. I metodi gerarchici di tipo agglomerativo partono da una situazione in cui si hanno n cluster distinti ognuno contenente un solo individuo per giungere, attraverso successive unioni dei cluster meno distanti tra loro, ad una situazione in cui si ha un solo cluster che contiene tutti gli individui. L'obiettivo finale dei metodi gerarchici è quello di ottenere una sequenza di partizioni che possono essere rappresentate graficamente mediante una struttura ad albero che prende il nome di *dendrogramma*, nella quale sull'insieme delle ordinate sono riportati i livelli di distanza mentre sull'asse delle ascisse sono riportati i singoli individui. Lo scopo del dendrogramma è proprio quello di fornire una visione d'insieme: infatti, il ricercatore osservando il dendrogramma può stabilire a quale stadio dell'analisi gerarchica fermarsi ottenendo la partiziione dell'insieme totale di individui in cluster.
Molti metodi di analisi gerarchica sono caratterizzati da una struttura comune che si riflette in un algoritmo generale che può essere così esplicitato:

1. a partire dalla matrice X originaria dei dati o dalla matrice scalata, considerare la matrice delle distanze D (o la matrice di similarità S) tra gli individui considerati come singoli cluster contenenti un solo individuo;
2. individuare la coppia di cluster meno distante (o più somigliante) e raggruppare in un unico cluster i due cluster meno distanti. Inoltre, calcolare la distanza (o similarità) di questo cluster, originato dall'agglomerazione, da tutti gli altri gruppi già esistenti;

3. costruire una nuova matrice delle distanze o di similarità che risulterà ridotta di una riga e di una colonna rispetto a quella che la precede;
4. operare sulla nuova matrice a partire dallo step 2 fino a quando non si ottiene una matrice 2 x 2.

L'analisi gerarchica di tipo agglomerativo viene effettuata in R attraverso la funzione *hclust(d,method = "")* dove d rappresenta un oggetto creato tramite la funzone dist() e method seleziona il metodo gerarchico agglomerativo.
Essenzialmente esistono 5 metodi differenti:

1. metodo del legame singolo ("single");
2. metodo del legame completo ("complete");
3. metodo del legame medio ("average");
4. metodo del centroide ("centroid");
5. metodo della mediana ("median").

Analizziamone uno alla volta, portandolo nel nostro caso specifico.

Metodo del legame singolo

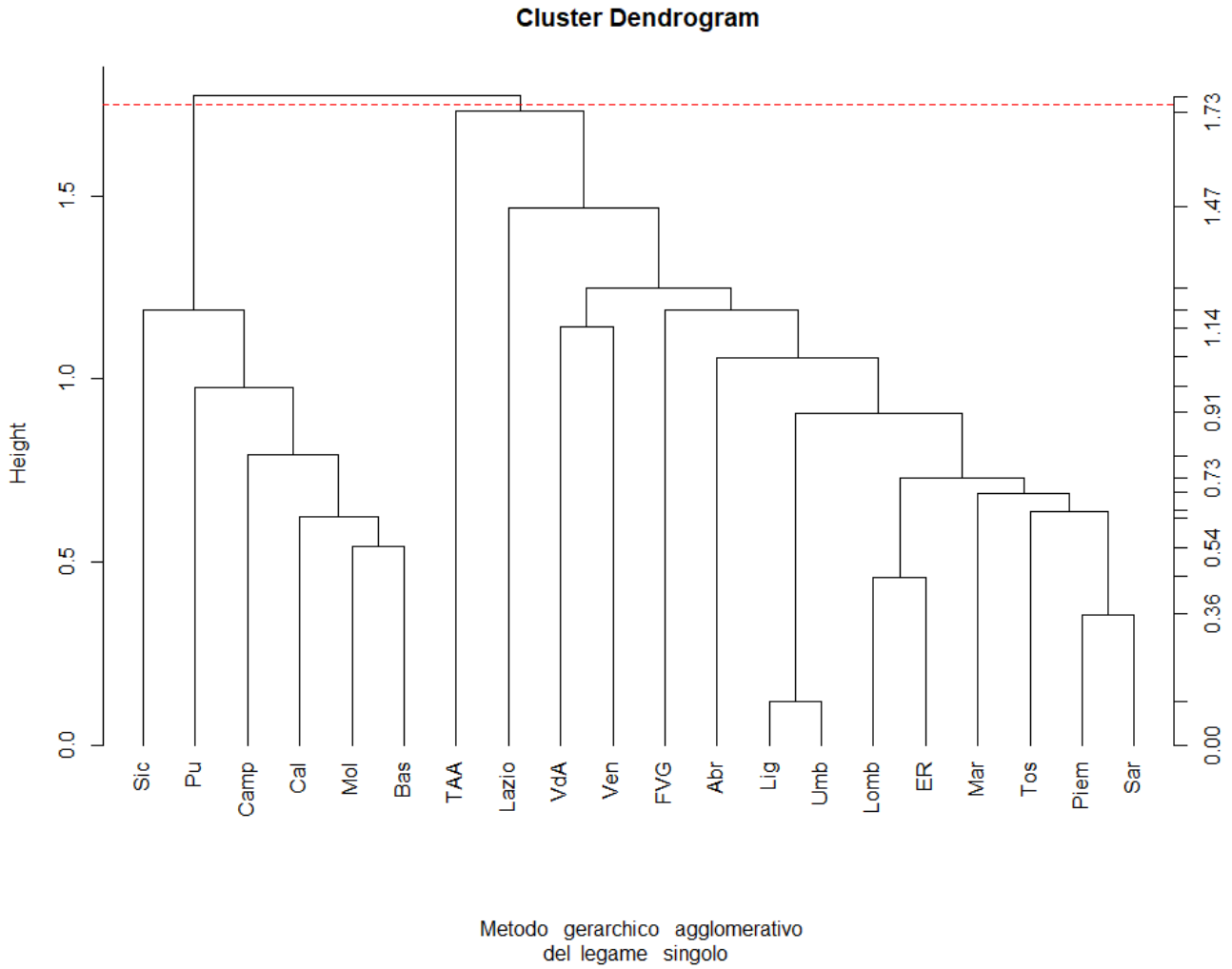
In questo metodo la distanza tra i gruppi G_1 (contenente n_1 individui) e G_2(contenente n_2 individui) è definita come la minima tra tutte le n_1n_2 distanze che si possono calcolare tra ogni individuo di G_1 e ogni individuo di G_2.
Nelle seguenti linee di codice, scaliamo la matrice dei dati, calcoliamo la matrice delle distanze utilizzando la metrica euclidea e applichiamo il metodo gerarchico del legame singolo:

```
#metodo del legame singolo
Z <- scale(matriceDati)
d <- dist(Z, method="euclidean", diag=TRUE, upper=TRUE)
hls <- hclust(d, method="single")
> str(hls) # visualizza informazioni sull ' oggetto cluster
List of 7
 $ merge      : int [1:19, 1:2] -3 -4 -1 -9 -14 3 -18 -16 2 -2 ...
 $ height     : num [1:19] 0.583 2.381 2.385 2.502 2.72 ...
 $ order      : int [1:20] 5 19 15 16 18 14 17 12 13 3 ...
 $ labels     : chr [1:20] "Piem" "VdA" "Lig" "Lomb" ...
 $ method     : chr "single"
 $ call       : language hclust(d = d, method = "single")
 $ dist.method: chr "euclidean"
 - attr(*, "class")= chr "hclust"
```

I risultati di $merge sono stati disposti su due colonne: i numeri con il segno negativo indicano i singoli individui, mentre i numeri positivi indicano i cluster che si formano. Inoltre, $height indica la distanza in cui è avvenuta l'agglomerazione tra i cluster.
Costruiamo ora il dendrogramma utilizzando R. Le seguenti linee di codice

```
plot(hls ,hang =-1, xlab=" Metodo   gerarchico   agglomerativo",
     sub ="del  legame   singolo ")
axis(side=4, at=round(c(0, hls$height ), 2))
abline(h=1.75, lty =2, col ="red ")
```

producono il grafico in figura sottostante.



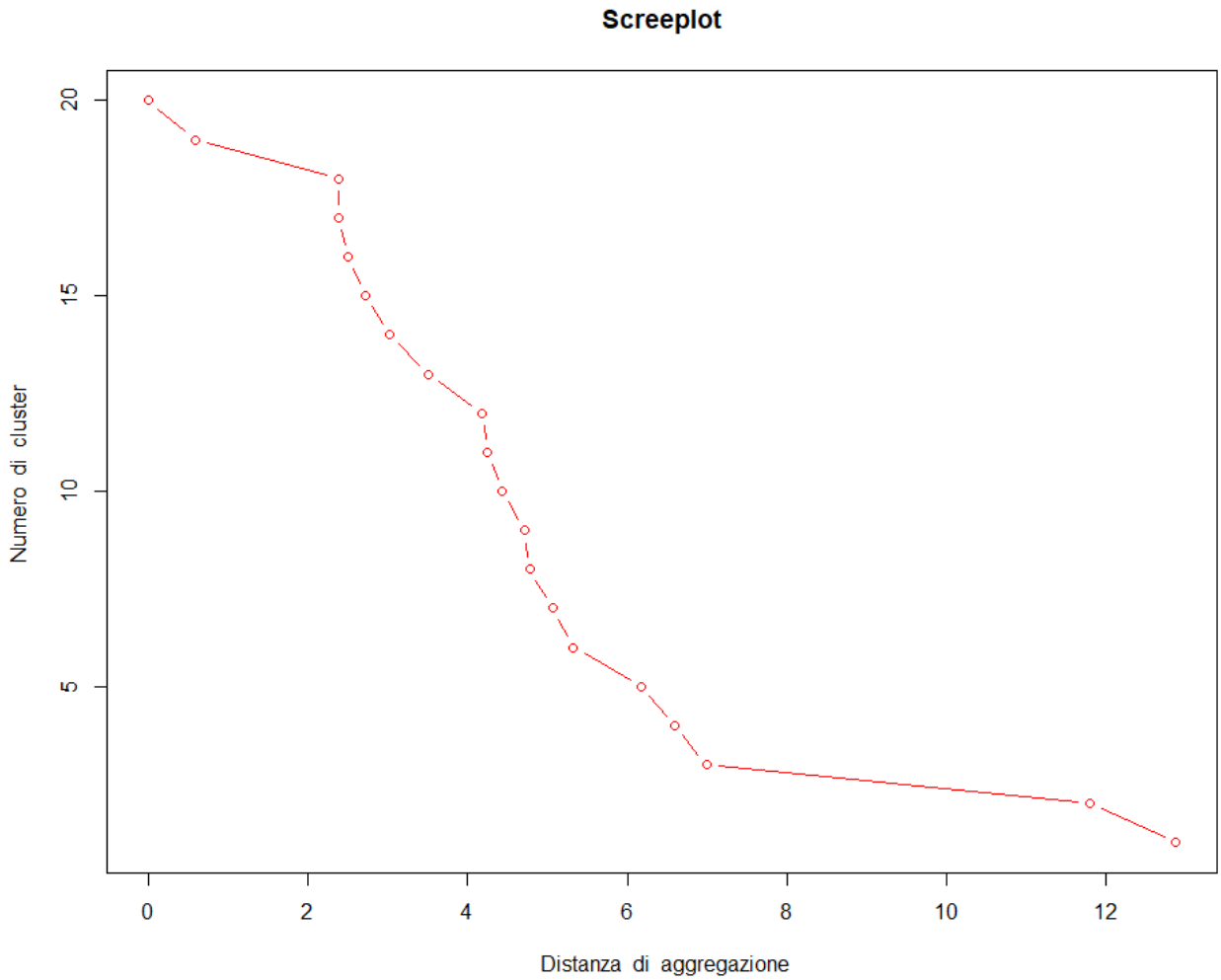
L'istruzione axis(side = 4, at = round(c(0, hls$height), 2)) permette di costruire l'asse delle altezze alla destra del grafico arrotondando i numeri alla seconda cifra decimale.
Per visualizzare il taglio del dendrogramma in corrispondenza di un salto nelle distanze si utilizza la funzione *abline(h = NULL, lty = NULL)*.
Al fine di scegliere una buona partizione del dendrogramma, consideriamo un grafico detto *screeplot* in cui sull'asse delle ordinate sono riportati i numeri di gruppi ottenibili con il metodo gerarchico e sull'asse delle ascisse sono riportate le distanze a cui avvengono le successive aggregazioni tra i gruppi. Possiamo costruire lo screeplot solo per il metodo del legame singolo, completo e medio (gli altri due utilizzano il quadrato delle distanze). È sempre comunque preferibile considerare le misure di non omogeneità statistiche.
In generale, possiamo considerare le quantità:

$$\delta_k = d_{k-1} - d_k \qquad (k = 2, \ldots, n),$$

dove d_k rappresenta il livello di distanza a cui è stata effettuata l'agglomerazione in k gruppi e n è il numero iniziale di individui.
Lo screeplot fornisce una visione di insieme delle altezze a cui sono avvenute le agglomerazioni e si potrebbe scegliere il valore di j per il quale $\delta_j = \max\{\delta_2, \delta_3, \ldots, \delta_n\}$.
nel nostro caso, abbiamo già calcolato la matrice delle distante, applicato il metodo gerarchico del legame singolo e costruito il dendrogramma. Vogliamo ora costruire lo screeplot. Consideriamo la seguente linea di codice

```
plot(c(0, hls$height), seq(20,1), type="b", main="Screeplot",
     xlab=" Distanza  di  aggregazione", ylab="Numero  di  cluster ", col ="red ")
```

La funzione c(0, hls$height) permette di concatenare 0 con il vettore hls$height delle altezze a cui sono avvenute le successive agglomerazioni. La funzione seq(20, 1) permette di costruire il vettore contenente il numero di gruppi da 20 a 1. La precedente linea di codice produce il grafico illustrato in figura sottostante.



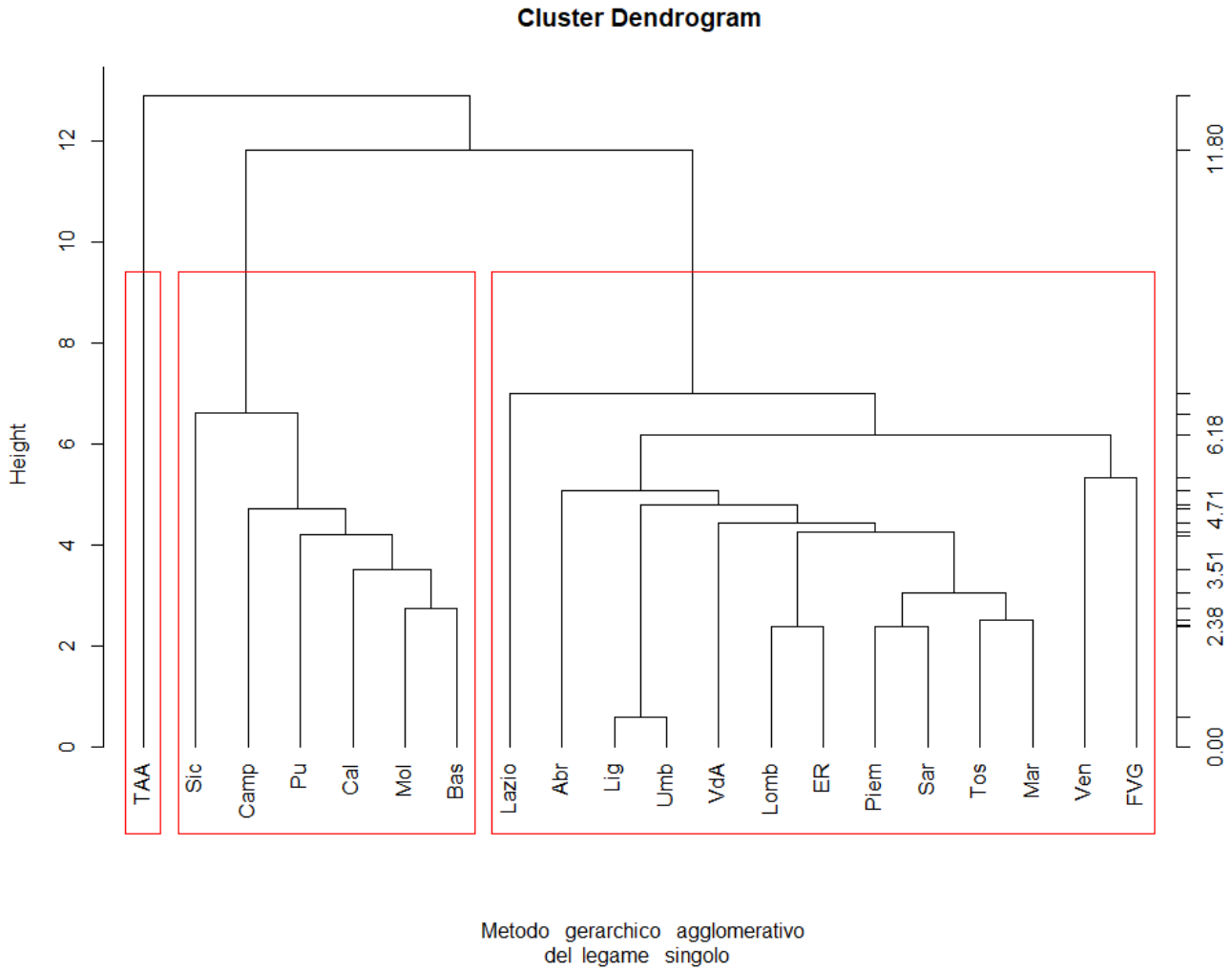
Lo screeplot in Figura ci suggerisce di considerare una suddivisione in due gruppi. Infatti, nel passaggio da due (altezza 2.3811762) a tre gruppi (altezza 2.3853721) si registra un co nsistente incremento della distanza di aggregazione. Inoltre, si ha:

$$\delta_3 = h_2 - h_3 = 2.3811762 - 2.3853721 = -0.0041959$$

è preferibile quindi considerare una suddivisione in tre cluster proprio come ci aveva già suggerito il k-means. Consideriamo la seguente linea di codice:

```
rect.hclust(hls, k=3, border = "red")
```

Ci permette di evidenziare in rosso attraverso dei rettangoli i nostri 3 cluster:



I nostri tre gruppi quindi saranno così composti:
C_1= {TAA}, C_2={Sic, Camp, Pu, Cal, Mol, Bas} e C_3 = {Lazio, Abr, Lig, Umb, VdA, Lomb, ER, Piem, Sar, Tos, Mar, Ven, FVG}.
Considerato un particolare dendrogramma, per ottenere una suddivisione degli individui in cluster, in corrispondenza di un determinato livello di distanza oppure in base ad un numero fissato m di cluster, R utilizza la funzione cutree() che applicata al nostro caso specifico sarà:

```
> cutree(hls, k=3)
 Piem   VdA   Lig  Lomb   TAA   Ven   FVG    ER   Tos   Umb   Mar Lazio   Abr   Mol  Camp    Pu   Bas   Cal
    1     1     1     1     2     1     1     1     1     1     1     1     1     3     3     3     3     3
  Sic   Sar
    3     1
```

In cui hls rappresenta l'oggetto creato tramite la funzione hclust applicata al metodo del legame singolo, e k rappresenta il numero di cluster in cui vogliamo suddividere i nostri 20 individui.
In R è inoltre possibile ricavare le misure di sintesi relative ai singoli cluster ottenuti tagliando il dendrogramma tramite la funzione cutree() e utilizzando la funzione aggregate(). Nel nostro specifico caso:

```
taglios <- cutree(hls, k=3)
tagliolists <- list(taglios)
aggregate(matriceDati, tagliolists, mean)
aggregate(matriceDati, tagliolists, var) #NA
aggregate(matriceDati, tagliolists, sd) #NA
```

in cui tagliolists rappresenta una lista di indici sulla base dei quale le colonne di X vanno aggregate.
Dopo aver effettuato il taglio, siamo interessati a calcolare le misure di non omogeneità statistiche relative all'insieme totale di individui (trT), ai singoli cluster ottenuti effettuando il taglio e alla somma delle loro misure di non omogeneità (trS) e alla misura di non omogeneità tra i cluster (trB):

$$trT = trS + trB,$$

o equivalentemente

$$1 = \frac{tr\,S}{tr\,T} + \frac{tr\,B}{tr\,T}$$

I cluster dovrebbero essere individuati in modo da minimizzare la misura di non omogeneità statistica all'interno dei cluster (within) e massimizzare la misura di non omogeneità statistica tra i gruppi (between).
Quindi, se due differenti metodi gerarchici conducono a due diverse partizioni con lo stesso numero di cluster, occorre scegliere quella partizione con misura di non omogeneità statistica all'interno dei cluster (tr S) più piccola, che corrisponde a maggiore omogeneità interna.
Consideriamo il nostro caso, e consideriamo ancora una volta il metodo del legame singolo che ci ha condotto ad una partizione degli individui in 3 cluster. Per l'insieme totale I si ha:

```
#totale
n <- nrow(matriceDati)
trHI <- (n -1)*sum(apply(matriceDati, 2, var))
> trHI #visualizza la misura di non omogeneità totale
[1] 3692.498
```

La misura di non omogeneità statistica totale è quindi $trH_I = 3692.498$. Calcoliamo ora le misure di non omogeneità statistiche dei tre gruppi:

```
#omogeneità interna legame singolo
d <- dist(matriceDati, method="euclidean", diag=TRUE, upper=TRUE)
hls <- hclust(d, method= "single")
taglios <- cutree(hls, k=3)
nums <- table(taglios) #vettore contenente il numero di individui nei tre cluster
tagliolists <- list(taglios)
agvars <- aggregate(matriceDati, tagliolists, var)[,-1]

#primo cluster
trH1 <- (nums[[1]]-1) * sum(agvars[1,])
trH1 #visualizza la misura di non omogeinità del primo gruppo

#secondo cluster
trH2 <- (nums[[2]]-1) * sum(agvars[2,])
trH2 <- 0

#terzo cluster
trH3 <- (nums[[3]]-1) * sum(agvars[3,])
trH3 #visualizza la misura di non omogeinità del primo gruppo

#misura di non omogeneità statistica interna ai tre gruppi
trS <- trH1 + trH2 + trH3
trS

#misura di non omogeinità statistica tra i cluster
trB <- trHI - trH1 -trH2 - trH3
trB

trB/trHI
```

In questo metodo non occorre definire le matrici dei dati relative ai gruppi e si possono ricavare le misure di non omogeneità statistica dei tre gruppi utilizzando la funzione cutree() e aggregate(). Nelle linee di codice precedenti è stato definito un vettore *nums* che contiene il numero di individui nei tre cluster. Inoltre, la misura di non omogeneità relativa al primo gruppo è calcolata moltiplicando n_1-1 per la somma degli elementi della prima riga della matrice ottenuta con aggregate(matriceDati, tagliolists, var). E così, anche per gli altri due gruppi. Eseguendo le linee di codice precendenti otteniamo:

```
> #omogeneità interna legame singolo
> d <- dist(matriceDati, method="euclidean", diag=TRUE, upper=TRUE)
> hls <- hclust(d, method= "single")
> taglios <- cutree(hls, k=3)
> nums <- table(taglios) #vettore contenente il numero di individui nei tre cluster
> tagliolists <- list(taglios)
> agvars <- aggregate(matriceDati, tagliolists, var)[,-1]
> #primo cluster
> trH1 <- (nums[[1]]-1) * sum(agvars[1,])
> trH1 #visualizza la misura di non omogeinità del primo gruppo
[1] 504.9708
> #secondo cluster
> trH2 <- (nums[[2]]-1) * sum(agvars[2,])
> trH2 <- 0
> #terzo cluster
> trH3 <- (nums[[3]]-1) * sum(agvars[3,])
> trH3 #visualizza la misura di non omogeinità del primo gruppo
[1] 118.0817
> #misura di non omogeneità statistica interna ai tre gruppi
> trS <- trH1 + trH2 + trH3
> trS
[1] 623.0524
> #misura di non omogeinità statistica tra i cluster
> trB <- trHI - trH1 -trH2 - trH3
> trB
[1] 3069.446
> trB/trHI
[1] 0.8312653
```

da cui ricaviamo che la misura di non omogeneità per il primo gruppo è pari a 504.9708, per il secondo gruppo è 0 (perché contiene un solo elemento) e per il terzo gruppo invece risulta essere 118.0817. Quindi, la misura di non omogeneità statistica interna ai tre gruppi risulta essere trS = trH1 + trH2 + trH3 ovvero 623.0524. La misura di non omogeneità statistica tra i cluster si ottiene come la differenza trB = trH_I – trH1 – trH2 – trH3 ovvero 3069.446. Se eseguiamo il calcolo trB/trH_I otteniamo 0.8312653. Ciò indica che la misura di non omogeneità all'interno dei gruppi è quindi piccola rispetto alla misura di non omogeneità tra i cluster.

Metodo del legame completo

In questo metodo la distanza tra i gruppi G_1 (contenente n_1 individui) e G_2 (contenente n_2 individui) è definita come la massima tra tutte le n_1n_2 distanze che si possono calcolare tra ogni individuo di G_1 e ogni individuo di G_2.
Nelle seguenti linee di codice, scaliamo la matrice dei dati, calcoliamo la matrice delle distanze utilizzando la metrica euclidea e applichiamo il metodo gerarchico del legame completo:

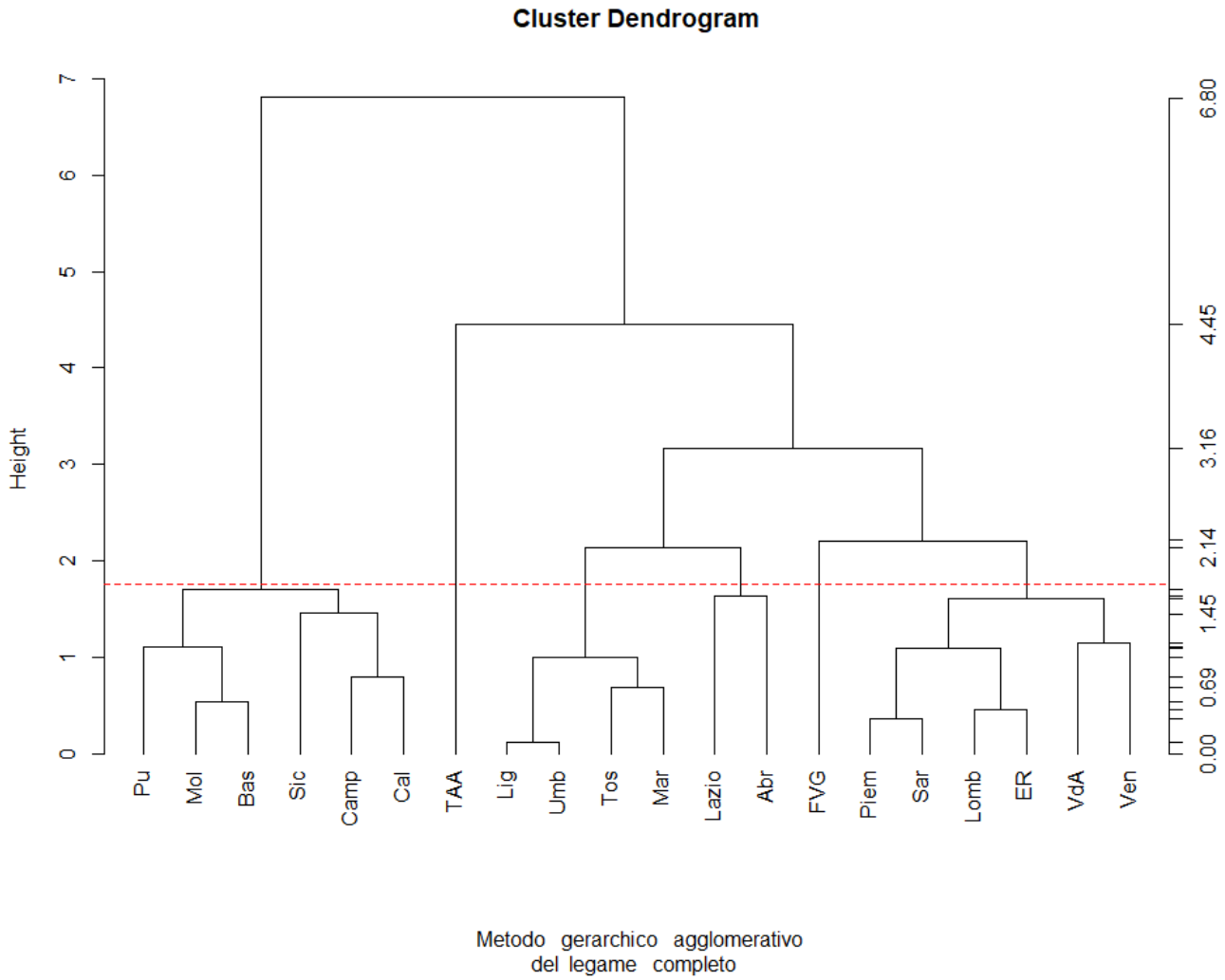
```
#metodo del legame completo
Z <- scale(matriceDati)
d <- dist(Z, method="euclidean", diag=TRUE, upper=TRUE)
hlc <- hclust(d, method="complete")
```

```
> str(hlc) # visualizza informazioni sull'oggetto cluster
List of 7
 $ merge      : int [1:19, 1:2] -3 -1 -4 -14 -9 -15 1 2 -16 -2 ...
 $ height     : num [1:19] 0.12 0.356 0.457 0.542 0.686 ...
 $ order      : int [1:20] 16 14 17 19 15 18 5 3 10 9 ...
 $ labels     : chr [1:20] "Piem" "VdA" "Lig" "Lomb" ...
 $ method     : chr "complete"
 $ call       : language hclust(d = d, method = "complete")
 $ dist.method: chr "euclidean"
 - attr(*, "class")= chr "hclust"
```

I risultati di $merge sono stati disposti su due colonne: i numeri con il segno negativo indicano i singoli individui, mentre i numeri positivi indicano i cluster che si formano. Inoltre, $height indica la distanza in cui è avvenuta l'agglomerazione tra i cluster.
Costruiamo ora il dendrogramma utilizzando R. Le seguenti linee di codice

```
plot(hlc ,hang =-1, xlab=" Metodo   gerarchico   agglomerativo", sub ="del  legame   completo ")
axis(side=4, at=round(c(0, hlc$height ), 2))
abline (h=1.75, lty =2, col ="red")
```

producono il grafico in figura sottostante.

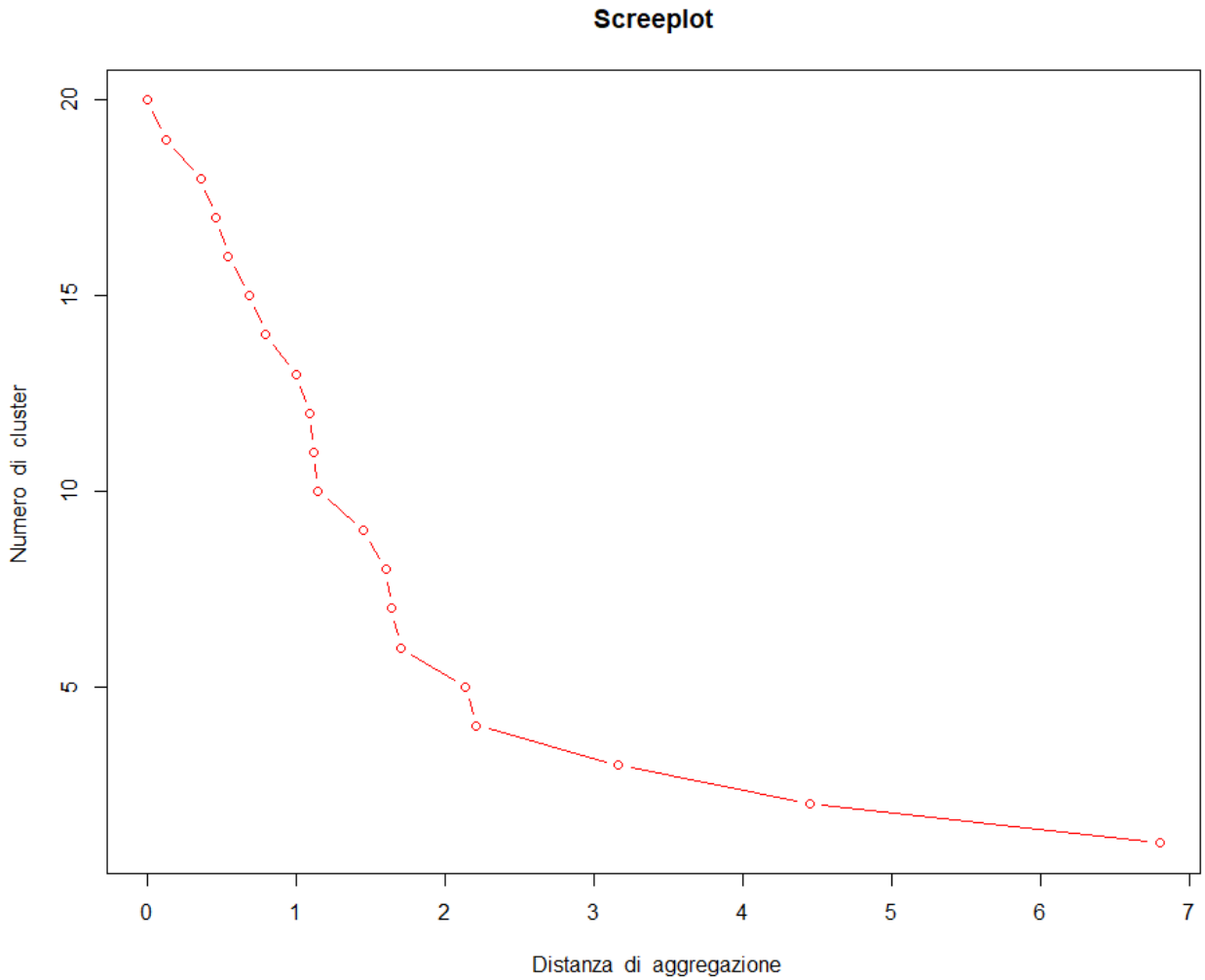


L'istruzione axis(side = 4, at = round(c(0, hlc$height), 2)) permette di costruire l'asse delle altezze alla destra del grafico arrotondando i numeri alla seconda cifra decimale.
Per visualizzare il taglio del dendrogramma in corrispondenza di un salto nelle distanze si utilizza la funzione *abline(h = NULL, lty = NULL)*.

Al fine di scegliere una buona partizione del dendrogramma, consideriamo un grafico detto *screeplot* in cui sull'asse delle ordinate sono riportati i numeri di gruppi ottenibili con il metodo gerarchico e sull'asse delle ascisse sono riportate le distanze a cui avvengono le successive aggregazioni tra i gruppi.
Lo screeplot fornisce una visione di insieme delle altezze a cui sono avvenute le agglomerazioni e si potrebbe scegliere il valore di j per il quale $\delta_j = \max\{\delta_2, \delta_3, \ldots, \delta_n\}$.
nel nostro caso, abbiamo già calcolato la matrice delle distanze, applicato il metodo gerarchico del legame completo e costruito il dendrogramma. Vogliamo ora costruire lo screeplot. Consideriamo la seguente linea di codice

```
plot(c(0, hlc$height), seq(20,1), type="b", main="Screeplot",
     xlab=" Distanza  di  aggregazione", ylab="Numero  di  cluster ", col ="red ")
```

La funzione c(0, hlc$height) permette di concatenare 0 con il vettore hlc$height delle altezze a cui sono avvenute le successive agglomerazioni. La funzione seq(20, 1) permette di costruire il vettore contenente il numero di gruppi da 20 a 1. La precedente linea di codice produce il grafico illustrato in figura sottostante.



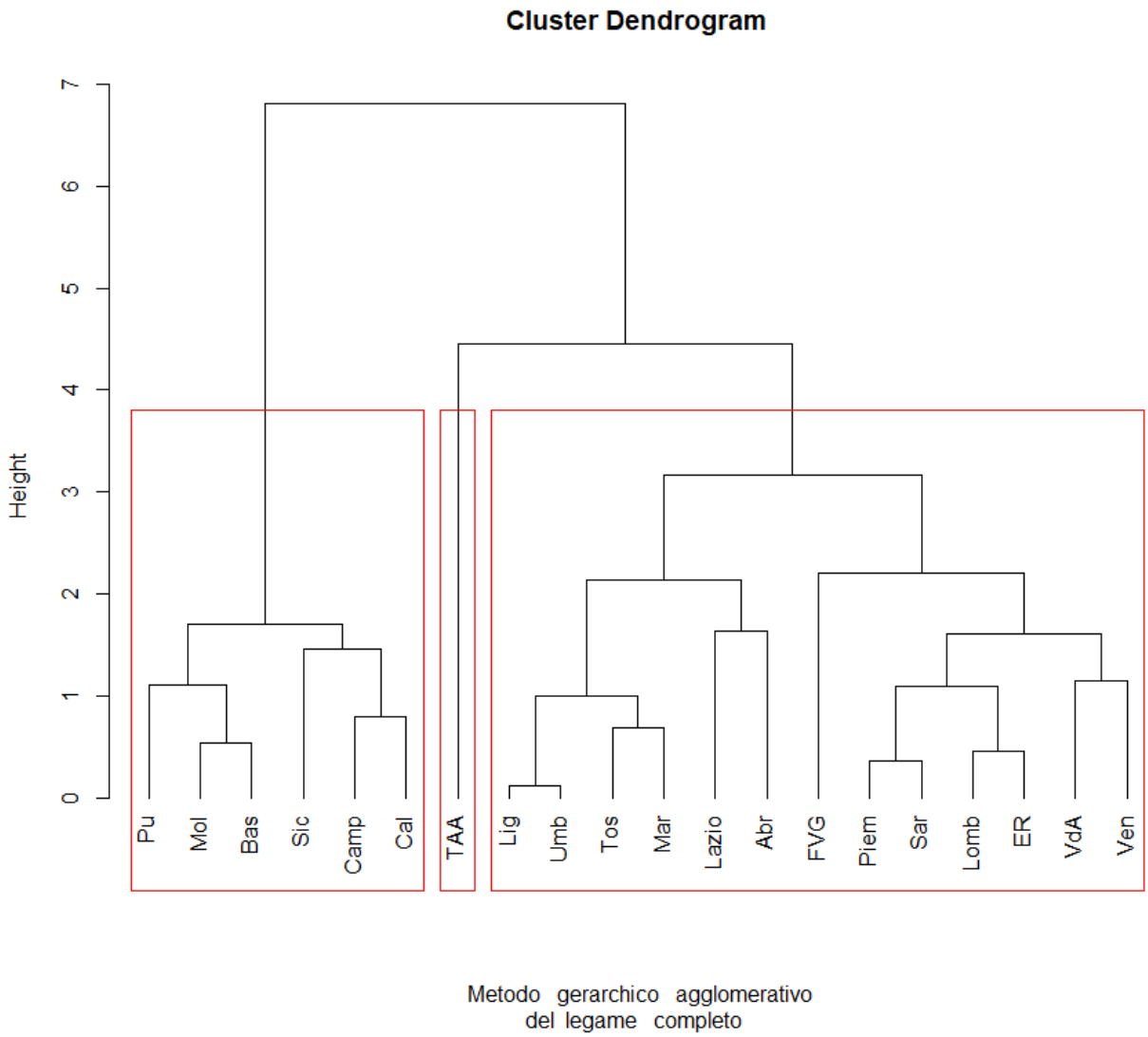
Lo screeplot in Figura ci suggerisce di considerare una suddivisione in due gruppi. Infatti, nel passaggio da uno (altezza 0.1200786) a due gruppi (altezza 0.3563665) si registra un c onsistente incremento della distanza di aggregazione. Inoltre, si ha:

$$\delta_3 = h_1 - h_2 = 0.1200786 - 0.3563665 = -0.2362879$$

è preferibile quindi considerare una suddivisione in due cluster al contrario di come ci aveva suggerito il k-means. Ma per considerare poi il metodo che ci fornisce misura di omogeneità statistica interna ai cluster più bassa, continuiamo a considerare 3 gruppi. Consideriamo la seguente linea di codice:

```
rect.hclust(hlc, k=3, border = "red")
```

Ci permette di evidenziare in rosso attraverso dei rettangoli i nostri 3 cluster:



I nostri tre gruppi quindi saranno così composti:
C_1={Sic, Camp, Pu, Cal, Mol, Bas, Cal}, C_2= {TAA}, e C_3 = {Lazio, Abr, Lig, Umb, VdA, Lomb, ER, Piem, Sar, Tos, Mar, Ven, FVG}. La suddivisione risulta essere uguale alla suddivisione ottenuta con il metodo del legame singolo.
Considerato un particolare dendrogramma, per ottenere una suddivisione degli individui in cluster, in corrispondenza di un determinato livello di distanza oppure in base ad un numero fissato m di cluster, R utilizza la funzione cutree() che applicata al nostro caso specifico sarà:

```
> cutree(hlc, k=3)
 Piem   VdA   Lig  Lomb   TAA   Ven   FVG    ER   Tos   Umb   Mar Lazio   Abr   Mol  Camp    Pu   Bas   Cal   Sic
    1     1     1     1     2     1     1     1     1     1     1     1     1     3     3     3     3     3     3
  Sar
    1
```

In cui hlc rappresenta l'oggetto creato tramite la funzione hclust applicata al metodo del legame completo, e k rappresenta il numero di cluster in cui vogliamo suddividere i nostri 20 individui.
In R è inoltre possibile ricavare le misure di sintesi relative ai singoli cluster ottenuti tagliando il dendrogramma tramite la funzione cutree() e utilizzando la funzione aggregate(). Nel nostro specifico caso:

```
taglioc <- cutree(hlc, k=3)
tagliolistc <- list(taglioc)
aggregate(matriceDati, tagliolistc, mean)
aggregate(matriceDati, tagliolistc, var)
aggregate(matriceDati, tagliolistc, sd)
```

in cui tagliolistc rappresenta una lista di indici sulla base dei quale le colonne di matriceDato vanno aggregate.
Dopo aver effettuato il taglio, siamo interessati a calcolare le misure di non omogeneità statistiche relative all'insieme totale di individui (trT), ai singoli cluster ottenuti effettuando il taglio e alla somma delle loro misure di non omogeneità (trS) e alla misura di non omogeneità tra i cluster (trB):

$$trT = trS + trB,$$

o equivalentemente

$$1 = \frac{tr\,S}{tr\,T} + \frac{tr\,B}{tr\,T}$$

Consideriamo il nostro caso, e consideriamo ancora una volta il metodo del legame completo che ci ha condotto ad una partizione degli individui in 3 cluster. Sappiamo già che le misure di non omogeneità risultano essere uguali a quelle ottenute per il legame singolo. Ma vogliamo comunque provare che sia così. Per l'insieme totale I si ha ovviamente sempre:

```
#totale
n <- nrow(matriceDati)
trHI <- (n -1)*sum(apply(matriceDati, 2, var))
> trHI #visualizza la misura di non omogeneità totale
[1] 3692.498
```

La misura di non omogeneità statistica totale è quindi $trH_I = 3692.498$. Calcoliamo ora le misure di non omogeneità statistiche dei tre gruppi:

```
#omogeneità interna legame completo
d <- dist(matriceDati, method="euclidean", diag=TRUE, upper=TRUE)
hlc <- hclust(d, method= "complete")
taglioc <- cutree(hlc, k=3)
numc <- table(taglioc)
tagliolistc <- list(taglioc)
agvarc <- aggregate(matriceDati, tagliolistc, var)[,-1]

#primo cluster
trH1 <- (numc[[1]]-1) * sum(agvarc[1,])
trH1 #visualizza la misura di non omogeinità del primo gruppo

#secondo cluster
trH2 <- (numc[[2]]-1) * sum(agvarc[2,])
trH2 <- 0

#terzo cluster
trH3 <- (numc[[3]]-1) * sum(agvarc[3,])
trH3

#misura di non omogeneità statistica interna ai tre gruppi
trS <- trH1 + trH2 + trH3
trS

#misura di non omogeinità statistica tra i cluster
trB <- trHI - trH1 -trH2 - trH3
trB

trB/trHI
```

In questo metodo non occorre definire le matrici dei dati relative ai gruppi e si possono ricavare le misure di non omogeneità statistica dei tre gruppi utilizzando la funzione cutree() e aggregate(). Nelle linee di codice precedenti è stato definito un vettore *numc* che contiene il numero di individui nei tre cluster. Inoltre, la misura di non omogeneità relativa al primo gruppo è calcolata moltiplicando n_1-1 per la somma degli elementi della prima riga della matrice ottenuta con aggregate(matriceDati, tagliolistc, var). E così, anche per gli altri due gruppi. Eseguendo le linee di codice precedenti otteniamo:

```
> #omogeneità interna legame completo
> d <- dist(matriceDati, method="euclidean", diag=TRUE, upper=TRUE)
> hlc <- hclust(d, method= "complete")
> taglioc <- cutree(hlc, k=3)
> numc <- table(taglioc)
> tagliolistc <- list(taglioc)
> agvarc <- aggregate(matriceDati, tagliolistc, var)[,-1]
> #primo cluster
> trH1 <- (numc[[1]]-1) * sum(agvarc[1,])
> trH1 #visualizza la misura di non omogeinità del primo gruppo
[1] 504.9708
> #secondo cluster
> trH2 <- (numc[[2]]-1) * sum(agvarc[2,])
> trH2 <- 0
> #terzo cluster
> trH3 <- (numc[[3]]-1) * sum(agvarc[3,])
> trH3
[1] 118.0817
> #misura di non omogeneità statistica interna ai tre gruppi
> trS <- trH1 + trH2 + trH3
> trS
[1] 623.0524
> #misura di non omogeinità statistica tra i cluster
> trB <- trHI - trH1 -trH2 - trH3
> trB
[1] 3069.446
> trB/trHI
[1] 0.8312653
```

da cui ricaviamo che la misura di non omogeneità per il primo gruppo è pari a 504.9708, per il secondo gruppo è 0 (perché contiene un solo elemento) e per il terzo gruppo invece risulta

essere 118.0817. Quindi, la misura di non omogeneità statistica interna ai tre gruppi risulta essere trS = trH1 + trH2 + trH3 ovvero 623.0524. La misura di non omogeneità statistica tra i cluster si ottiene come la differenza trB = trH_I – trH1 – trH2 – trH3 ovvero 3069.446. Se eseguiamo il calcolo trB/trH_I otteniamo 0.8312653. Ciò indica che la misura di non omogeneità all'interno dei gruppi è quindi piccola rispetto alla misura di non omogeneità tra i cluster. Ciò prova che le misure di non omogeneità risultano essere uguali a quelle ottenute attraverso il metodo del legame singolo.

Metodo del legame medio

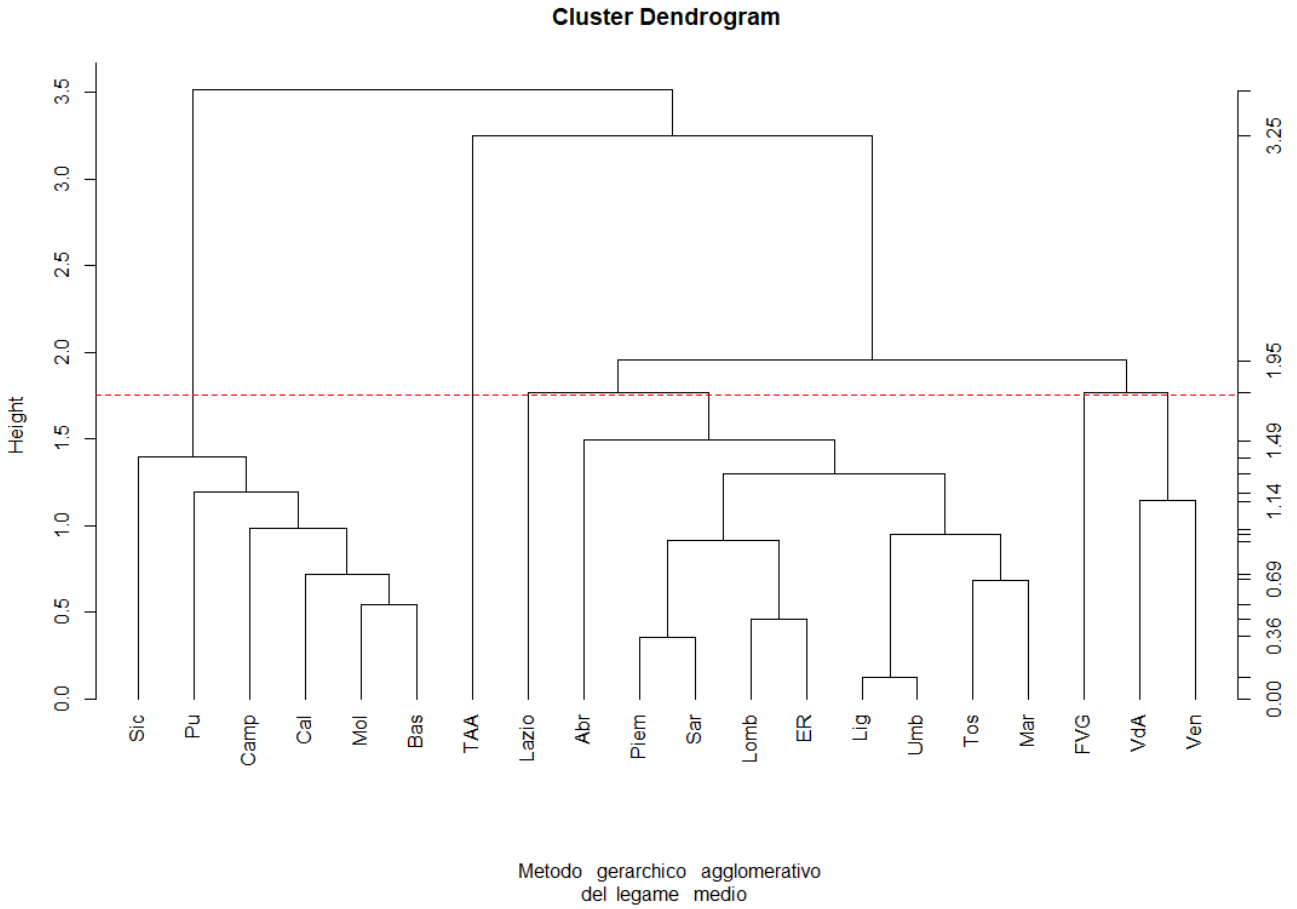
In questo metodo la distanza tra i gruppi G_1 e G_2 è definita come la media aritmetica delle distanze tra tutte le coppie di unità che compongono i due gruppi. Nelle seguenti linee di codice, scaliamo la matrice dei dati, calcoliamo la matrice delle distanze utilizzando la metrica euclidea e applichiamo il metodo gerarchico del legame medio:

```
#metodo del legame medio
Z <- scale(matriceDati)
d <- dist(Z, method="euclidean", diag=TRUE, upper=TRUE)
hlm <- hclust(d, method="average")
> str(hlm) # visualizza informazioni sull ' oggetto cluster
List of 7
 $ merge      : int [1:19, 1:2] -3 -1 -4 -14 -9 -18 2 1 -15 -2 ...
 $ height     : num [1:19] 0.12 0.356 0.457 0.542 0.686 ...
 $ order      : int [1:20] 19 16 15 18 14 17 5 12 13 1 ...
 $ labels     : chr [1:20] "Piem" "VdA" "Lig" "Lomb" ...
 $ method     : chr "average"
 $ call       : language hclust(d = d, method = "average")
 $ dist.method: chr "euclidean"
 - attr(*, "class")= chr "hclust"
```

I risultati di $merge sono stati disposti su due colonne: i numeri con il segno negativo indicano i singoli individui, mentre i numeri positivi indicano i cluster che si formano. Inoltre, $height indica la distanza in cui è avvenuta l'agglomerazione tra i cluster.
Costruiamo ora il dendrogramma utilizzando R. Le seguenti linee di codice

```
plot(hlm ,hang =-1, xlab=" Metodo   gerarchico   agglomerativo", sub ="del  legame   medio ")
axis(side=4, at=round(c(0, hlm$height ), 2))
abline(h=1.75, lty =2, col ="red ")
```

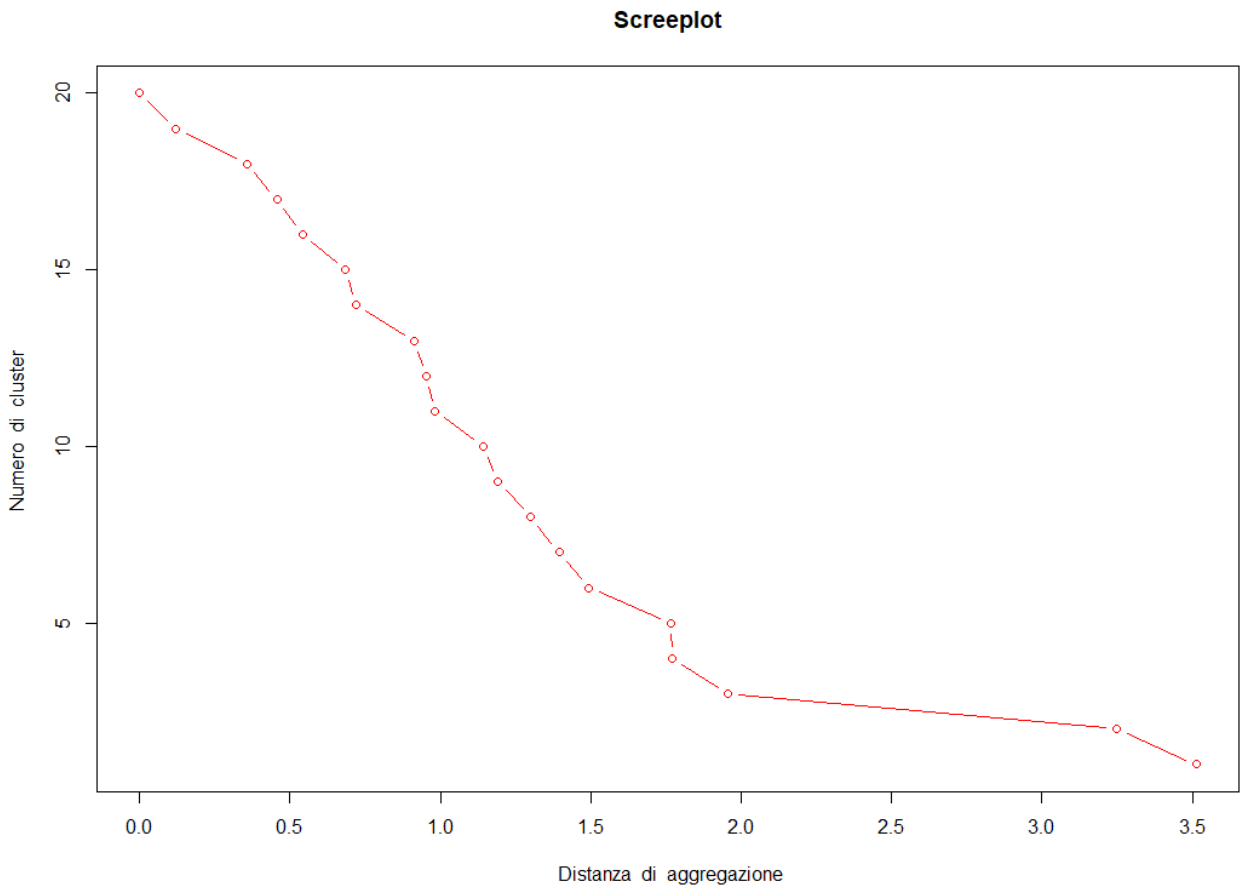
producono il grafico in figura sottostante.



L'istruzione axis(side = 4, at = round(c(0, hlm$height), 2)) permette di costruire l'asse delle altezze alla destra del grafico arrotondando i numeri alla seconda cifra decimale.
Per visualizzare il taglio del dendrogramma in corrispondenza di un salto nelle distanze si utilizza la funzione *abline(h = NULL, lty = NULL)*.
Al fine di scegliere una buona partizione del dendrogramma, consideriamo un grafico detto *screeplot* in cui sull'asse delle ordinate sono riportati i numeri di gruppi ottenibili con il metodo gerarchico e sull'asse delle ascisse sono riportate le distanze a cui avvengono le successive aggregazioni tra i gruppi.
Lo screeplot fornisce una visione di insieme delle altezze a cui sono avvenute le agglomerazioni e si potrebbe scegliere il valore di j per il quale $\delta_j = \max\{\delta_2, \delta_3, \ldots, \delta_n\}$.
nel nostro caso, abbiamo già calcolato la matrice delle distanze, applicato il metodo gerarchico del legame medio e costruito il dendrogramma. Vogliamo ora costruire lo screeplot. Consideriamo la seguente linea di codice

```
plot(c(0, hlm$height), seq(20,1), type="b", main="Screeplot",
     xlab=" Distanza  di  aggregazione", ylab="Numero  di  cluster ", col ="red ")
```

La funzione c(0, hlm$height) permette di concatenare 0 con il vettore hlm$height delle altezze a cui sono avvenute le successive agglomerazioni. La funzione seq(20, 1) permette di costruire il vettore contenente il numero di gruppi da 20 a 1. La precedente linea di codice produce il grafico illustrato in figura sottostante.



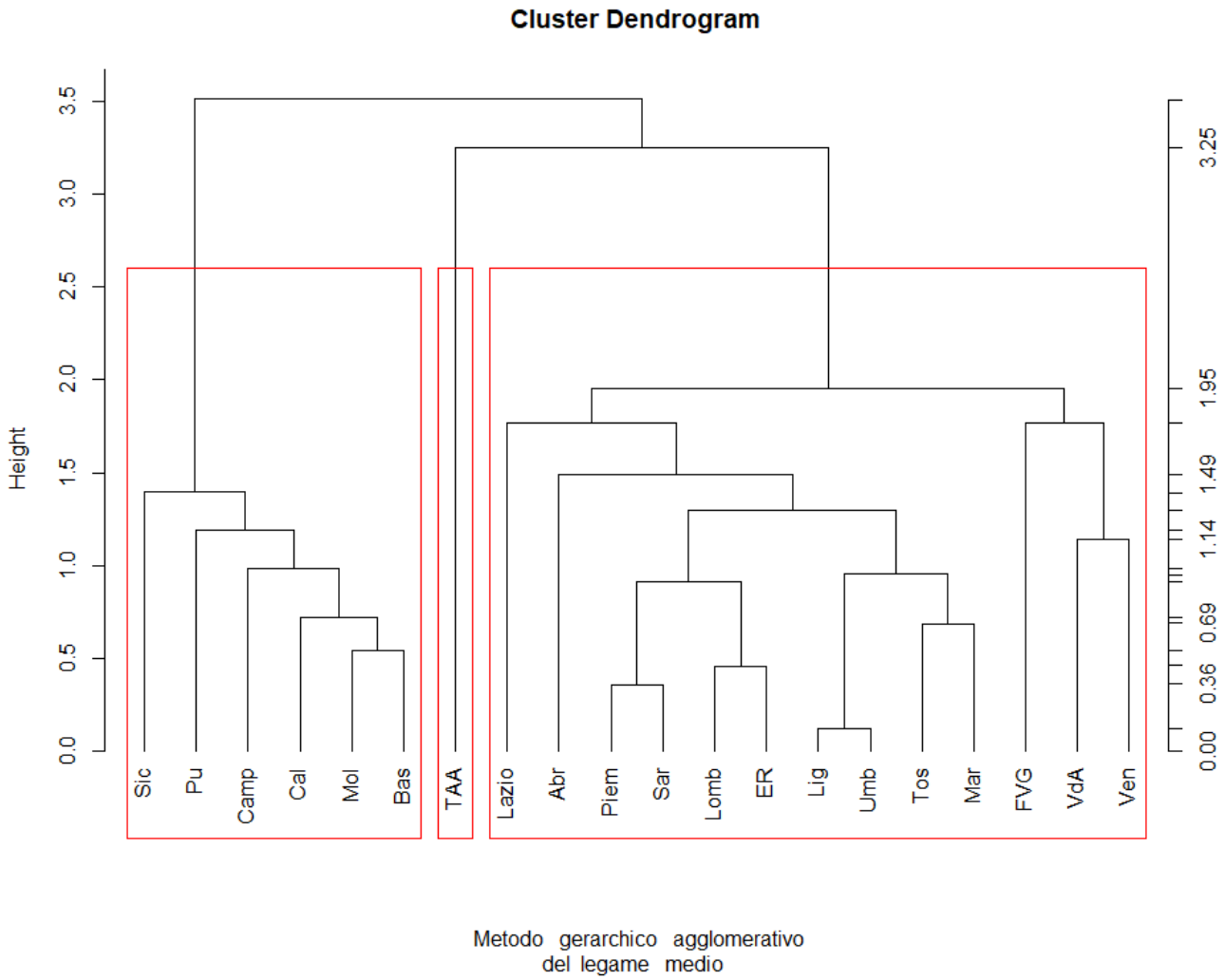
Lo screeplot in Figura ci suggerisce di considerare una suddivisione in tre gruppi. Infatti, n el passaggio da due (altezza 0.3563665) a tre gruppi (altezza 0.4570361) si registra un con sistente incremento della distanza di aggregazione. Inoltre, si ha:

$$\delta_3 = h_1 - h_2 = 0.3563665 - 0.4570361 = -0.1006696$$

è preferibile quindi considerare una suddivisione in tre cluster proprio come ci aveva suggerito il k-means. Consideriamo la seguente linea di codice:

```
rect.hclust(hlm, k=3)
```

Ci permette di evidenziare in rosso attraverso dei rettangoli i nostri 3 cluster:



I nostri tre gruppi quindi saranno così composti:
C_1={Sic, Camp, Pu, Cal, Mol, Bas}, C_2= {TAA}, e C_3 = {Lazio, Abr, Lig, Umb, VdA, Lomb, ER, Piem, Sar, Tos, Mar, Ven, FVG}. La suddivisione risulta essere uguale alla suddivisione ottenuta con gli altri due metodi: l'unica cosa che varia sono le altezze in cui sono avvenute le agglomerazioni.
Considerato un particolare dendrogramma, per ottenere una suddivisione degli individui in cluster, in corrispondenza di un determinato livello di distanza oppure in base ad un numero fissato m di cluster, R utilizza la funzione cutree() che applicata al nostro caso specifico sarà:

```
> cutree(hlm, k=3)
 Piem   VdA   Lig  Lomb   TAA   Ven   FVG    ER   Tos   Umb   Mar
    1     1     1     1     2     1     1     1     1     1     1
Lazio   Abr   Mol  Camp    Pu   Bas   Cal   Sic   Sar
    1     1     3     3     3     3     3     3     1
```

In cui hlm rappresenta l'oggetto creato tramite la funzione hclust applicata al metodo del legame completo, e k rappresenta il numero di cluster in cui vogliamo suddividere i nostri 20 individui.
In R è inoltre possibile ricavare le misure di sintesi relative ai singoli cluster ottenuti tagliando il dendrogramma tramite la funzione cutree() e utilizzando la funzione aggregate(). Nel nostro specifico caso:

```
#per il metodo del legame medio
tagliom <- cutree(hlm, k=3)
tagliolistm <- list(tagliom)
aggregate(matriceDati, tagliolistm, mean)
aggregate(matriceDati, tagliolistm, var)
aggregate(matriceDati, tagliolistm, sd)
```

in cui tagliolistm rappresenta una lista di indici sulla base dei quale le colonne di matriceDato vanno aggregate.
Dopo aver effettuato il taglio, siamo interessati a calcolare le misure di non omogeneità statistiche relative all'insieme totale di individui (trT), ai singoli cluster ottenuti effettuando il taglio e alla somma delle loro misure di non omogeneità (trS) e alla misura di non omogeneità tra i cluster (trB):

$$trT = trS + trB,$$

o equivalentemente

$$1 = \frac{tr\,S}{tr\,T} + \frac{tr\,B}{tr\,T}$$

Consideriamo il nostro caso, e consideriamo ancora una volta il metodo del legame medio che ci ha condotto ad una partizione degli individui in 3 cluster. Sappiamo già che le misure di non omogeneità risultano essere uguali a quelle ottenute per il legame singolo ed il legame completo. Ma vogliamo comunque provare che sia così. Per l'insieme totale I si ha ovviamente sempre:

```
#totale
n <- nrow(matriceDati)
trHI <- (n -1)*sum(apply(matriceDati, 2, var))
> trHI #visualizza la misura di non omogeneità totale
[1] 3692.498
```

La misura di non omogeneità statistica totale è quindi $trH_I = 3692.498$. Calcoliamo ora le misure di non omogeneità statistiche dei tre gruppi:

```
#omogeneità interna legame medio
d <- dist(matriceDati, method="euclidean", diag=TRUE, upper=TRUE)
hlm <- hclust(d, method= "average")
tagliom <- cutree(hlm, k=3)
numm <- table(tagliom)
tagliolistm <- list(tagliom)
agvarm <- aggregate(matriceDati, tagliolistm, var)[,-1]

#primo cluster
trH1 <- (numm[[1]]-1) * sum(agvarm[1,])
trH1 #visualizza la misura di non omogeinità del primo gruppo

#secondo cluster
trH2 <- (numm[[2]]-1) * sum(agvarm[2,])
trH2 <- 0

#terzo cluster
trH3 <- (numm[[3]]-1) * sum(agvarm[3,])
trH3

#misura di non omogeneità statistica interna ai tre gruppi
trS <- trH1 + trH2 + trH3
trS

#misura di non omogeinità statistica tra i cluster
trB <- trHI - trH1 -trH2 - trH3
trB

trB/trHI
```

In questo metodo non occorre definire le matrici dei dati relative ai gruppi e si possono ricavare le misure di non omogeneità statistica dei tre gruppi utilizzando la funzione cutree() e aggregate(). Nelle linee di codice precedenti è stato definito un vettore *numm* che contiene il numero di individui nei tre cluster. Inoltre, la misura di non omogeneità relativa al primo gruppo è calcolata moltiplicando n_1-1 per la somma degli elementi della prima riga della matrice ottenuta con aggregate(matriceDati, tagliolistm, var). E così, anche per gli altri due gruppi. Eseguendo le linee di codice precedenti otteniamo:

```
> #omogeneità interna legame medio
> d <- dist(matriceDati, method="euclidean", diag=TRUE, upper=TRUE)
> hlm <- hclust(d, method= "average")
> tagliom <- cutree(hlm, k=3)
> numm <- table(tagliom)
> tagliolistm <- list(tagliom)
> agvarm <- aggregate(matriceDati, tagliolistm, var)[,-1]
> #primo cluster
> trH1 <- (numm[[1]]-1) * sum(agvarm[1,])
> trH1 #visualizza la misura di non omogeinità del primo gruppo
[1] 504.9708
> #secondo cluster
> trH2 <- (numm[[2]]-1) * sum(agvarm[2,])
> trH2 <- 0
> #terzo cluster
> trH3 <- (numm[[3]]-1) * sum(agvarm[3,])
> trH3
[1] 118.0817
> trS
[1] 623.0524
> #misura di non omogeneità statistica interna ai tre gruppi
> trS <- trH1 + trH2 + trH3
> #misura di non omogeinità statistica tra i cluster
> trB <- trHI - trH1 -trH2 - trH3
> trB
[1] 3069.446
> trB/trHI
[1] 0.8312653
```

da cui ricaviamo che la misura di non omogeneità per il primo gruppo è pari a 504.9708, per il secondo gruppo è 0 (perché contiene un solo elemento) e per il terzo gruppo invece risulta essere 118.0817. Quindi, la misura di non omogeneità statistica interna ai tre gruppi risulta essere trS = trH1 + trH2 + trH3 ovvero 623.0524. La misura di non omogeneità statistica tra i cluster si ottiene come la differenza trB = trH_I – trH1 – trH2 – trH3 ovvero 3069.446. Se eseguiamo il calcolo trB/trH_I otteniamo 0.8312653. Ciò indica che la misura di non omogeneità all'interno dei gruppi è quindi piccola rispetto alla misura di non omogeneità tra i cluster. Ciò prova che le misure di non omogeneità risultano essere uguali a quelle ottenute attraverso il metodo del legame singolo e del legame completo.

Metodo del centroide

In questo metodo la distanza tra il gruppi G_1 e il gruppo G_2 è definita come la distanza tra i centroidi, ossia tra le medie campionarie calcolate sugli individui appartenenti ai due gruppi. Nelle seguenti linee di codice, scaliamo la matrice dei dati, calcoliamo la matrice delle distanze, utilizzando la metrica euclidea, calcoliamo anche la matrice delle distanze al quadrato e applichiamo il metodo gerarchico del centroide:

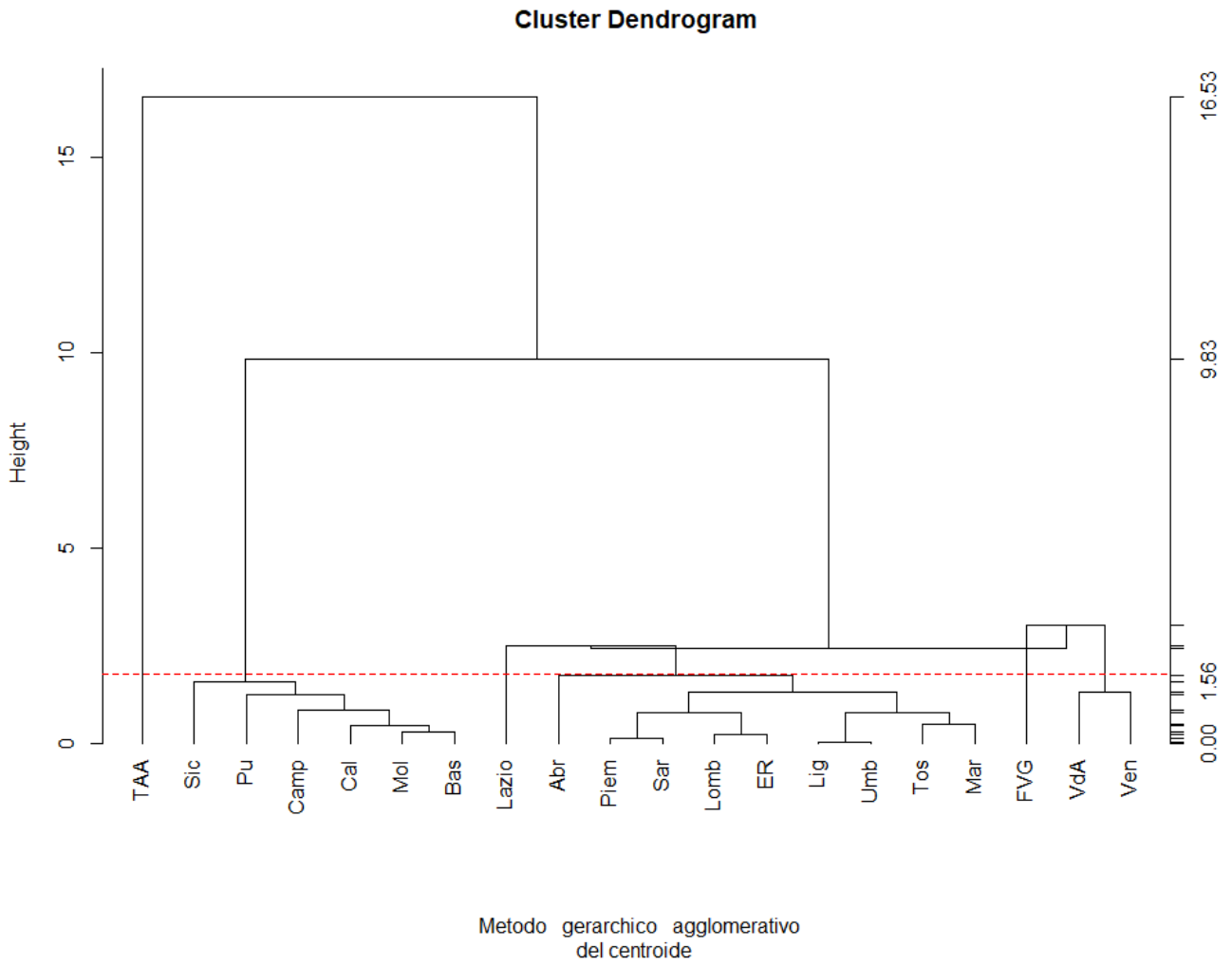
```
#metodo del centroide
Z <- scale(matriceDati)
d <- dist(Z, method="euclidean", diag=TRUE, upper=TRUE)
d2 <- d^2
hlcent <- hclust(d2, method="centroid")
```

```
> str(hlcent) # visualizza informazioni sull ' oggetto cluster
List of 7
 $ merge      : int [1:19, 1:2] -3 -1 -4 -14 -18 -9 2 1 -15 -16 ...
 $ height     : num [1:19] 0.0144 0.127 0.2089 0.2934 0.4555 ...
 $ order      : int [1:20] 5 19 16 15 18 14 17 12 13 1 ...
 $ labels     : chr [1:20] "Piem" "VdA" "Lig" "Lomb" ...
 $ method     : chr "centroid"
 $ call       : language hclust(d = d2, method = "centroid")
 $ dist.method: chr "euclidean"
 - attr(*, "class")= chr "hclust"
```

I risultati di $merge sono stati disposti su due colonne: i numeri con il segno negativo indicano i singoli individui, mentre i numeri positivi indicano i cluster che si formano. Inoltre, $height indica la distanza in cui è avvenuta l'agglomerazione tra i cluster.
Costruiamo ora il dendrogramma utilizzando R. Le seguenti linee di codice

```
plot(hlcent ,hang =-1, xlab=" Metodo   gerarchico   agglomerativo",
     sub ="del centroide ")
axis(side=4, at=round(c(0, hlcent$height ), 2))
abline (h=1.75, lty =2, col ="red ")
```

producono il grafico in figura sottostante.

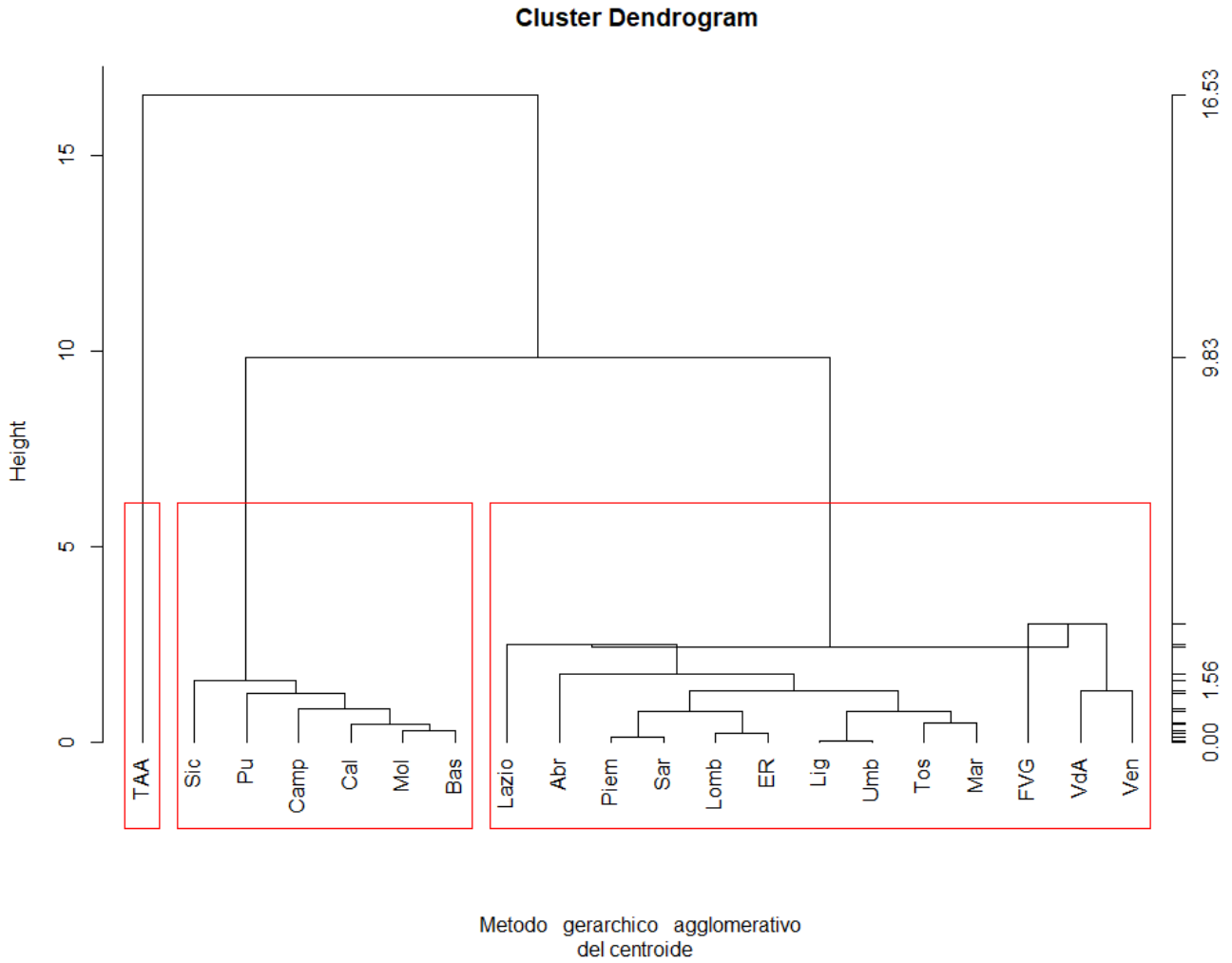


L'istruzione axis(side = 4, at = round(c(0, hlcent$height), 2)) permette di costruire l'asse delle altezze alla destra del grafico arrotondando i numeri alla seconda cifra decimale.
Per visualizzare il taglio del dendrogramma in corrispondenza di un salto nelle distanze si utilizza la funzione *abline(h = NULL, lty = NULL)*.

Anche per questo metodo consideriamo l'ipotesi di voler suddividere i nostri dati in 3 cluster come ci suggerisce il metodo del k-means. Consideriamo la seguente linea di codice:

```
rect.hclust(hlcent, k=3)
```

Ci permette di evidenziare in rosso attraverso dei rettangoli i nostri 3 cluster:



I nostri tre gruppi quindi saranno così composti:
C_1={Sic, Camp, Pu, Cal, Mol, Bas}, C_2= {TAA}, e C_3 = {Lazio, Abr, Lig, Umb, VdA, Lomb, ER, Piem, Sar, Tos, Mar, Ven, FVG}. La suddivisione risulta essere uguale alla suddivisione ottenuta con gli altri due metodi: l'unica cosa che varia sono le altezze in cui sono avvenute le agglomerazioni.
Considerato un particolare dendrogramma, per ottenere una suddivisione degli individui in cluster, in corrispondenza di un determinato livello di distanza oppure in base ad un numero fissato m di cluster, R utilizza la funzione cutree() che applicata al nostro caso specifico sarà:

```
> cutree(hlcent, k=3)
 Piem   VdA   Lig  Lomb   TAA   Ven   FVG    ER   Tos   Umb   Mar Lazio   Abr   Mol  Camp    Pu   Bas   Cal
    1     1     1     1     2     1     1     1     1     1     1     1     1     3     3     3     3     3
  Sic   Sar
    3     1
```

In cui hlcent rappresenta l'oggetto creato tramite la funzione hclust applicata al metodo del centroide, e k rappresenta il numero di cluster in cui vogliamo suddividere i nostri 20 individui.

In R è inoltre possibile ricavare le misure di sintesi relative ai singoli cluster ottenuti tagliando il dendrogramma tramite la funzione cutree() e utilizzando la funzione aggregate(). Nel nostro specifico caso:

```
#per il metodo del centroide
tagliocent <- cutree(hlcent, k=3)
tagliolistcent <- list(tagliocent)
aggregate(matriceDati, tagliolistcent, mean)
aggregate(matriceDati, tagliolistcent, var)
aggregate(matriceDati, tagliolistcent, sd)
```

in cui tagliolistcent rappresenta una lista di indici sulla base dei quale le colonne di matriceDato vanno aggregate.
Dopo aver effettuato il taglio, siamo interessati a calcolare le misure di non omogeneità statistiche relative all'insieme totale di individui (trT), ai singoli cluster ottenuti effettuando il taglio e alla somma delle loro misure di non omogeneità (trS) e alla misura di non omogeneità tra i cluster (trB):

$$trT = trS + trB,$$

o equivalentemente

$$1 = \frac{tr\,S}{tr\,T} + \frac{tr\,B}{tr\,T}$$

Consideriamo il nostro caso, e consideriamo ancora una volta il metodo del legame medio che ci ha condotto ad una partizione degli individui in 3 cluster. Sappiamo già che le misure di non omogeneità risultano essere uguali a quelle ottenute per il legame singolo ed il legame completo. Ma vogliamo comunque provare che sia così. Per l'insieme totale I si ha ovviamente sempre:

```
#totale
n <- nrow(matriceDati)
trHI <- (n -1)*sum(apply(matriceDati, 2, var))
> trHI #visualizza la misura di non omogeneità totale
[1] 3692.498
```

La misura di non omogeneità statistica totale è quindi $trH_I = 3692.498$. Calcoliamo ora le misure di non omogeneità statistiche dei tre gruppi:

```
#omogeneità interna centroide
d <- dist(matriceDati, method="euclidean", diag=TRUE, upper=TRUE)
hlcent <- hclust(d^2, method= "centroid")
tagliocent <- cutree(hlcent, k=3)
numcent <- table(tagliocent)
tagliolistcent <- list(tagliocent)
agvarcent <- aggregate(matriceDati, tagliolistcent, var)[,-1]

#primo cluster
trH1 <- (numcent[[1]]-1) * sum(agvarcent[1,])
trH1 #visualizza la misura di non omogeinità del primo gruppo

#secondo cluster
trH2 <- (numcent[[2]]-1) * sum(agvarcent[2,])
trH2 <- 0

#terzo cluster
trH3 <- (numcent[[3]]-1) * sum(agvarcent[3,])
trH3

#misura di non omogeneità statistica interna ai tre gruppi
trS <- trH1 + trH2 + trH3
trS

#misura di non omogeinità statistica tra i cluster
trB <- trHI - trH1 -trH2 - trH3
trB

trB/trHI
```

In questo metodo non occorre definire le matrici dei dati relative ai gruppi e si possono ricavare le misure di non omogeneità statistica dei tre gruppi utilizzando la funzione cutree() e aggregate(). Nelle linee di codice precedenti è stato definito un vettore *numcent* che contiene il numero di individui nei tre cluster. Inoltre, la misura di non omogeneità relativa al primo gruppo è calcolata moltiplicando n_1-1 per la somma degli elementi della prima riga della matrice ottenuta con aggregate(matriceDati, tagliolistcent, var). E così, anche per gli altri due gruppi. Eseguendo le linee di codice precedenti otteniamo:

```
> #omogeneità interna centroide
> d <- dist(matriceDati, method="euclidean", diag=TRUE, upper=TRUE)
> hlcent <- hclust(d^2, method= "centroid")
> tagliocent <- cutree(hlcent, k=3)
> numcent <- table(tagliocent)
> tagliolistcent <- list(tagliocent)
> agvarcent <- aggregate(matriceDati, tagliolistcent, var)[,-1]
> #primo cluster
> trH1 <- (numcent[[1]]-1) * sum(agvarcent[1,])
> trH1 #visualizza la misura di non omogeinità del primo gruppo
[1] 504.9708
> #secondo cluster
> trH2 <- (numcent[[2]]-1) * sum(agvarcent[2,])
> trH2 <- 0
> #terzo cluster
> trH3 <- (numcent[[3]]-1) * sum(agvarcent[3,])
> trH3
[1] 118.0817
> #misura di non omogeneità statistica interna ai tre gruppi
> trS <- trH1 + trH2 + trH3
> trS
[1] 623.0524
> #misura di non omogeinità statistica tra i cluster
> trB <- trHI - trH1 -trH2 - trH3
> trB
[1] 3069.446
> trB/trHI
[1] 0.8312653
```

da cui ricaviamo che la misura di non omogeneità per il primo gruppo è pari a 504.9708, per il secondo gruppo è 0 (perché contiene un solo elemento) e per il terzo gruppo invece risulta

essere 118.0817. Quindi, la misura di non omogeneità statistica interna ai tre gruppi risulta essere trS = trH1 + trH2 + trH3 ovvero 623.0524. La misura di non omogeneità statistica tra i cluster si ottiene come la differenza trB = trH_I – trH1 – trH2 – trH3 ovvero 3069.446. Se eseguiamo il calcolo trB/trH_I otteniamo 0.8312653. Ciò indica che la misura di non omogeneità all'interno dei gruppi è quindi piccola rispetto alla misura di non omogeneità tra i cluster. Ciò prova che le misure di non omogeneità risultano essere uguali a quelle ottenute attraverso il metodo del legame singolo e del legame completo.

Metodo della mediana

Il metodo della mediana è simile a quello del centroide, con la differenza che la procedura è indipendente dalla numerosità dei cluster. Infatti, quando due gruppi si aggregano, il nuovo centroide è calcolato come la semisomma dei due centroidi precedenti. Nelle seguenti linee di codice, scaliamo la matrice dei dati, calcoliamo la matrice delle distanze utilizzando la metrica euclidea, calcoliamo anche la matrice delle distanze al quadrato e applichiamo il metodo gerarchico della mediana:

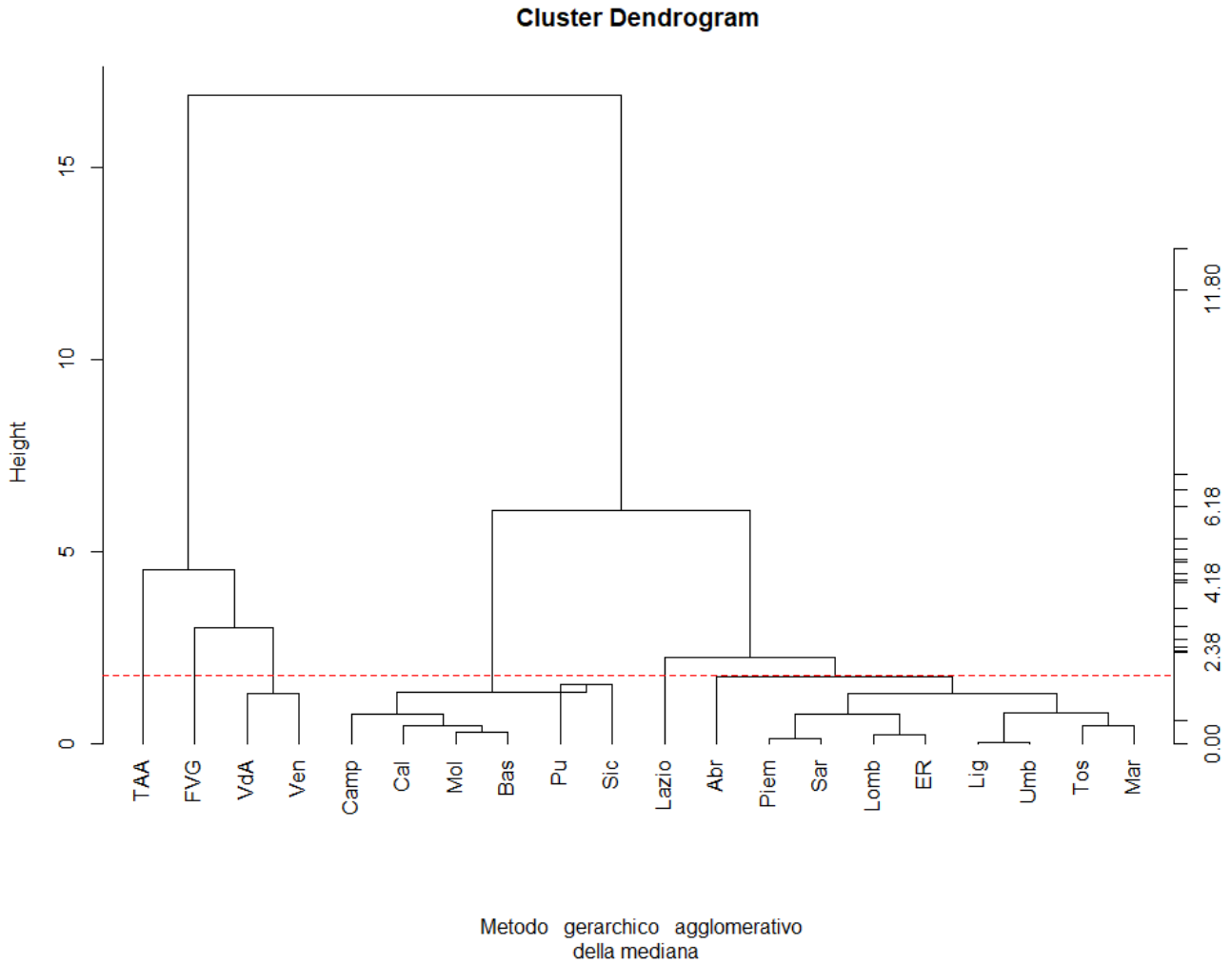
```
#metodo della mediana
Z <- scale(matriceDati)
d <- dist(Z, method="euclidean", diag=TRUE, upper=TRUE)
d2 <- d^2
hlmed <- hclust(d2, method="median")
```

```
> str(hlmed) # visualizza informazioni sull ' oggetto cluster
List of 7
 $ merge      : int [1:19, 1:2] -3 -1 -4 -14 -18 -9 -15 2 1 8 ...
 $ height     : num [1:19] 0.0144 0.127 0.2089 0.2934 0.4555 ...
 $ order      : int [1:20] 5 7 2 6 15 18 14 17 16 19 ...
 $ labels     : chr [1:20] "Piem" "VdA" "Lig" "Lomb" ...
 $ method     : chr "median"
 $ call       : language hclust(d = d2, method = "median")
 $ dist.method: chr "euclidean"
 - attr(*, "class")= chr "hclust"
```

I risultati di $merge sono stati disposti su due colonne: i numeri con il segno negativo indicano i singoli individui, mentre i numeri positivi indicano i cluster che si formano. Inoltre, $height indica la distanza in cui è avvenuta l'agglomerazione tra i cluster. Costruiamo ora il dendrogramma utilizzando R. Le seguenti linee di codice

```
plot(hlmed ,hang =-1, xlab=" Metodo   gerarchico   agglomerativo", sub ="della mediana ")
axis(side=4, at=round(c(0, hls$height ), 2))
abline(h=1.75, lty =2, col ="red ")
```

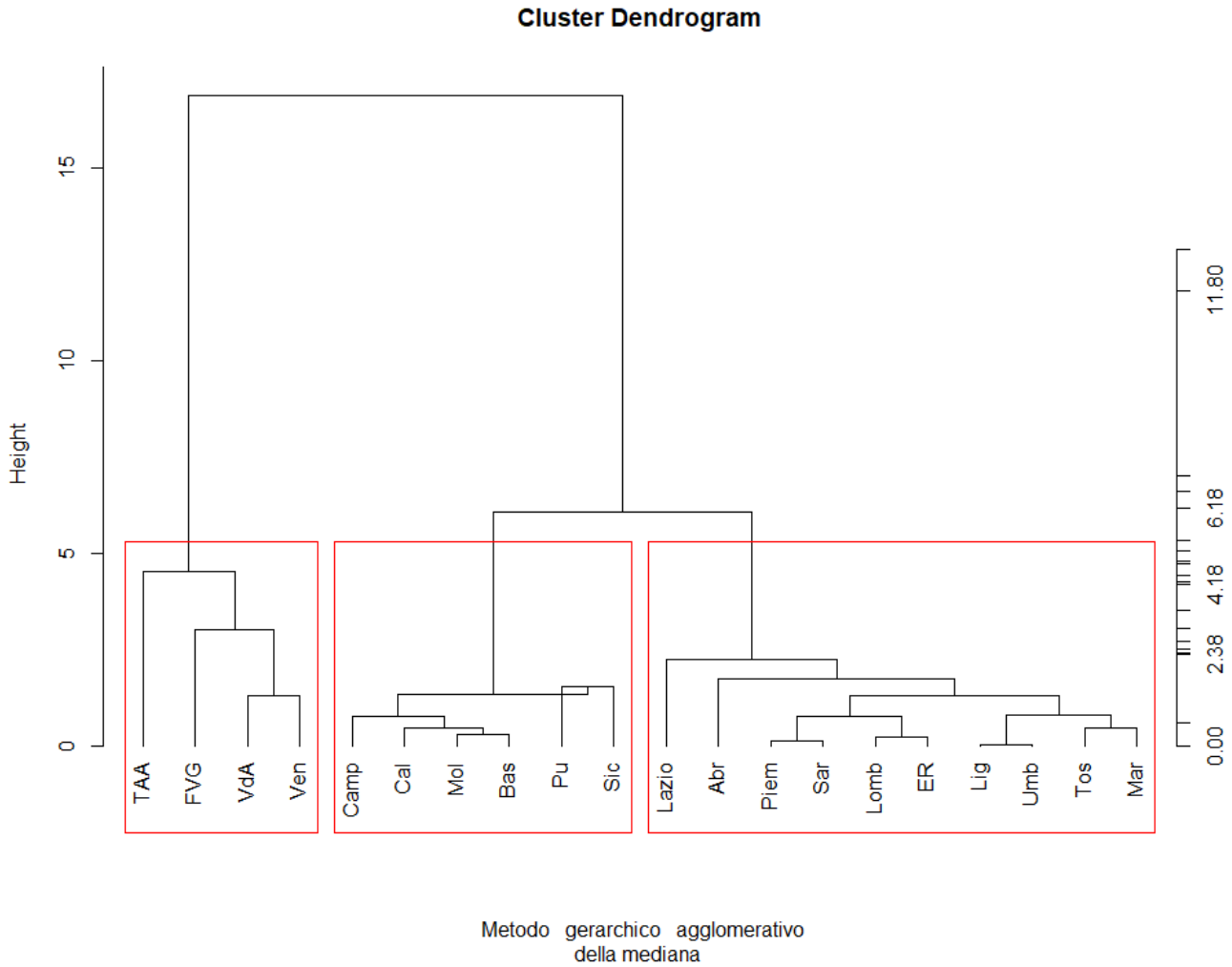
producono il grafico in figura sottostante.



L'istruzione axis(side = 4, at = round(c(0, hlmed$height), 2)) permette di costruire l'asse delle altezze alla destra del grafico arrotondando i numeri alla seconda cifra decimale.
Per visualizzare il taglio del dendrogramma in corrispondenza di un salto nelle distanze si utilizza la funzione *abline(h = NULL, lty = NULL)*.
Anche per questo metodo consideriamo l'ipotesi di voler suddividere i nostri dati in 3 cluster come ci suggerisce il metodo del k-means. Consideriamo la seguente linea di codice:

```
rect.hclust(hlmed, k=3)
```

Ci permette di evidenziare in rosso attraverso dei rettangoli i nostri 3 cluster:



I nostri tre gruppi quindi saranno così composti:
C_1= {TAA, FVG, VdA, Ven}, C_2={Sic, Camp, Pu, Cal, Mol, Bas}, e C_3 = {Lazio, Abr, Lig, Umb, Lomb, ER, Piem, Sar, Tos, Mar}.
Considerato un particolare dendrogramma, per ottenere una suddivisione degli individui in cluster, in corrispondenza di un determinato livello di distanza oppure in base ad un numero fissato m di cluster, R utilizza la funzione cutree() che applicata al nostro caso specifico sarà:

```
> cutree(hlmed, k=3)
 Piem   VdA   Lig  Lomb   TAA   Ven   FVG    ER   Tos   Umb   Mar Lazio   Abr   Mol  Camp    Pu   Bas   Cal
    1     2     1     1     2     2     2     1     1     1     1     1     1     3     3     3     3     3
  Sic   Sar
    3     1
```

In cui hlmed rappresenta l'oggetto creato tramite la funzione hclust applicata al metodo del centroide, e k rappresenta il numero di cluster in cui vogliamo suddividere i nostri 20 individui.
In R è inoltre possibile ricavare le misure di sintesi relative ai singoli cluster ottenuti tagliando il dendrogramma tramite la funzione cutree() e utilizzando la funzione aggregate(). Nel nostro specifico caso:

```
#per il metodo della mediana
tagliomed <- cutree(hlmed, k=2)
tagliolistmed <- list(tagliomed)
aggregate(matriceDati, tagliolistmed, mean)
aggregate(matriceDati, tagliolistmed, var)
aggregate(matriceDati, tagliolistmed, sd)
```

in cui tagliolistmed rappresenta una lista di indici sulla base dei quale le colonne di matriceDati vanno aggregate.
Dopo aver effettuato il taglio, siamo interessati a calcolare le misure di non omogeneità statistiche relative all'insieme totale di individui (trT), ai singoli cluster ottenuti effettuando il taglio e alla somma delle loro misure di non omogeneità (trS) e alla misura di non omogeneità tra i cluster (trB):

$$trT = trS + trB,$$

o equivalentemente

$$1 = \frac{tr\,S}{tr\,T} + \frac{tr\,B}{tr\,T}$$

Consideriamo il nostro caso, e consideriamo ancora una volta il metodo del legame medio che ci ha condotto ad una partizione degli individui in 3 cluster. Sappiamo già che le misure di non omogeneità risultano essere uguali a quelle ottenute per il legame singolo ed il legame completo. Ma vogliamo comunque provare che sia così. Per l'insieme totale I si ha ovviamente sempre:

```
#totale
n <- nrow(matriceDati)
trHI <- (n -1)*sum(apply(matriceDati, 2, var))
> trHI #visualizza la misura di non omogeneità totale
[1] 3692.498
```

La misura di non omogeneità statistica totale è quindi $trH_I = 3692.498$. Calcoliamo ora le misure di non omogeneità statistiche dei tre gruppi:

```
#omogeneità interna mediana
Z <- scale(matriceDati)
d <- dist(Z, method="euclidean", diag=TRUE, upper=TRUE)
d2 <- d^2
hlmed <- hclust(d2, method= "median")
tagliomed <- cutree(hlmed, k=3)
num <- table(tagliomed)
tagliolistmed <- list(tagliomed)
agvarmed <- aggregate(matriceDati, tagliolistmed, var)[,-1]

#primo cluster
trH1 <- (num[[1]]-1) * sum(agvarmed[1,])
trH1 #visualizza la misura di non omogeinità del primo gruppo

#secondo cluster
trH2 <- (num[[2]]-1) * sum(agvarmed[2,])
trH2

#terzo cluster
trH3 <- (num[[3]]-1) * sum(agvarcent[3,])
trH3

#misura di non omogeneità statistica interna ai tre gruppi
trS <- trH1 + trH2 + trH3
trS

#misura di non omogeinità statistica tra i cluster
trB <- trHI - trH1 - trH2 - trH3
trB

trB/trHI
```

In questo metodo non occorre definire le matrici dei dati relative ai gruppi e si possono ricavare le misure di non omogeneità statistica dei tre gruppi utilizzando la funzione cutree() e aggregate(). Nelle linee di codice precedenti è stato definito un vettore *num* che contiene il numero di individui nei tre cluster. Inoltre, la misura di non omogeneità relativa al primo gruppo è calcolata moltiplicando n_1-1 per la somma degli elementi della prima riga della matrice ottenuta con aggregate(matriceDati, tagliolistmed, var). E così, anche per gli altri due gruppi. Eseguendo le linee di codice precedenti otteniamo:

```
> #omogeneità interna mediana
> Z <- scale(matriceDati)
> d <- dist(Z, method="euclidean", diag=TRUE, upper=TRUE)
> d2 <- d^2
> hlmed <- hclust(d2, method= "median")
> tagliomed <- cutree(hlmed, k=3)
> num <- table(tagliomed)
> tagliolistmed <- list(tagliomed)
> agvarmed <- aggregate(matriceDati, tagliolistmed, var)[,-1]
> #primo cluster
> trH1 <- (num[[1]]-1) * sum(agvarmed[1,])
> trH1 #visualizza la misura di non omogeinità del primo gruppo
[1] 266.512
> #secondo cluster
> trH2 <- (num[[2]]-1) * sum(agvarmed[2,])
> trH2
[1] 237.8575
> #terzo cluster
> trH3 <- (num[[3]]-1) * sum(agvarcent[3,])
> trH3
[1] 118.0817
> #misura di non omogeneità statistica interna ai tre gruppi
> trS <- trH1 + trH2 + trH3
> trS
[1] 622.4512
> #misura di non omogeinità statistica tra i cluster
> trB <- trHI - trH1 - trH2 - trH3
> trB
[1] 3070.047
> trB/trHI
[1] 0.8314282
```

da cui ricaviamo che la misura di non omogeneità per il primo gruppo è pari a 266.512, per il secondo gruppo è 237.8575 e per il terzo gruppo invece risulta essere 118.0817. Quindi, la misura di non omogeneità statistica interna ai tre gruppi risulta essere trS = trH1 + trH2 + trH3 ovvero 622.4512. La misura di non omogeneità statistica tra i cluster si ottiene come la differenza $trB = trH_I - trH1 - trH2 - trH3$ ovvero 3070.047. Se eseguiamo il calcolo trB/trH_I otteniamo 0.8314282. Ciò indica che la misura di non omogeneità all'interno dei gruppi è quindi piccola rispetto alla misura di non omogeneità tra i cluster.
A questo punto, andiamo a confrontare la misura di non omogeneità statistica tra i cluster ottenuta tramite l'impiego dei vari metodi e constatiamo che risulta essere più alta nel caso si utilizzi il metodo gerarchico della mediana. Quindi, possiamo concludere dicendo che tra i vari metodi gerarchici, a parità di cluster, risulta essere migliore il metodo della mediana e perciò consideriamo la suddivisione in cluster ottenuta attraverso questo metodo.