04-gRafici

Ottavia M. Epifania, Ph.D

Lezione di Dottorato @Università Cattolica del Sacro Cuore (MI)

8-9 Giugno 2023

Table of contents

- Grafici base
- Grid graphics & `ggplot2`

Entrambi:

- High level functions $\rightarrow$ le funzioni che producono effettivamente il grafico
- Low level functions $\rightarrow$ Le funzioni che lo rendono più "bello"

Table of Contents

High level functions

```
plot()          # scatter plot, specialized plot methods
boxplot()
hist()          # histogram
qqnorm()        # quantile-quantile plot
barplot()
pie()           # pie chart
pairs()         # scatter plot matrix
persp()         # 3d plot
contour()       # contour plot
coplot()        # conditional plot
interaction.plot()
```

`demo(graphics)` vi fornisce un tour guidato dei grafici

Low level functions

```
points()  # Aggiunge punti al grafico
lines()   # Aggiunge linee  al grafico
rect()
polygon()
abline() # aggiunge una riga con intercetta a e pendenza b
arrows() # aggiunge barre d'errore
text()   # aggiunge testo nel plot
mtext()  # aggiunge testo nei margini
axis()   # personalizza gli assi
box()    # box attorno al grafico
legend() # cambia parametri della legenda
```

Plot layout

Ogni plot è composto da due regioni:

- Plotting region (dove effettivamente sta il plot)
- La regione dei margini (contiene i margini e le varie etichette degli assi)

Plot layout

Ogni plot è composto da due regioni:

- Plotting region (dove effettivamente sta il plot)
- La regione dei margini (contiene i margini e le varie etichette degli assi)

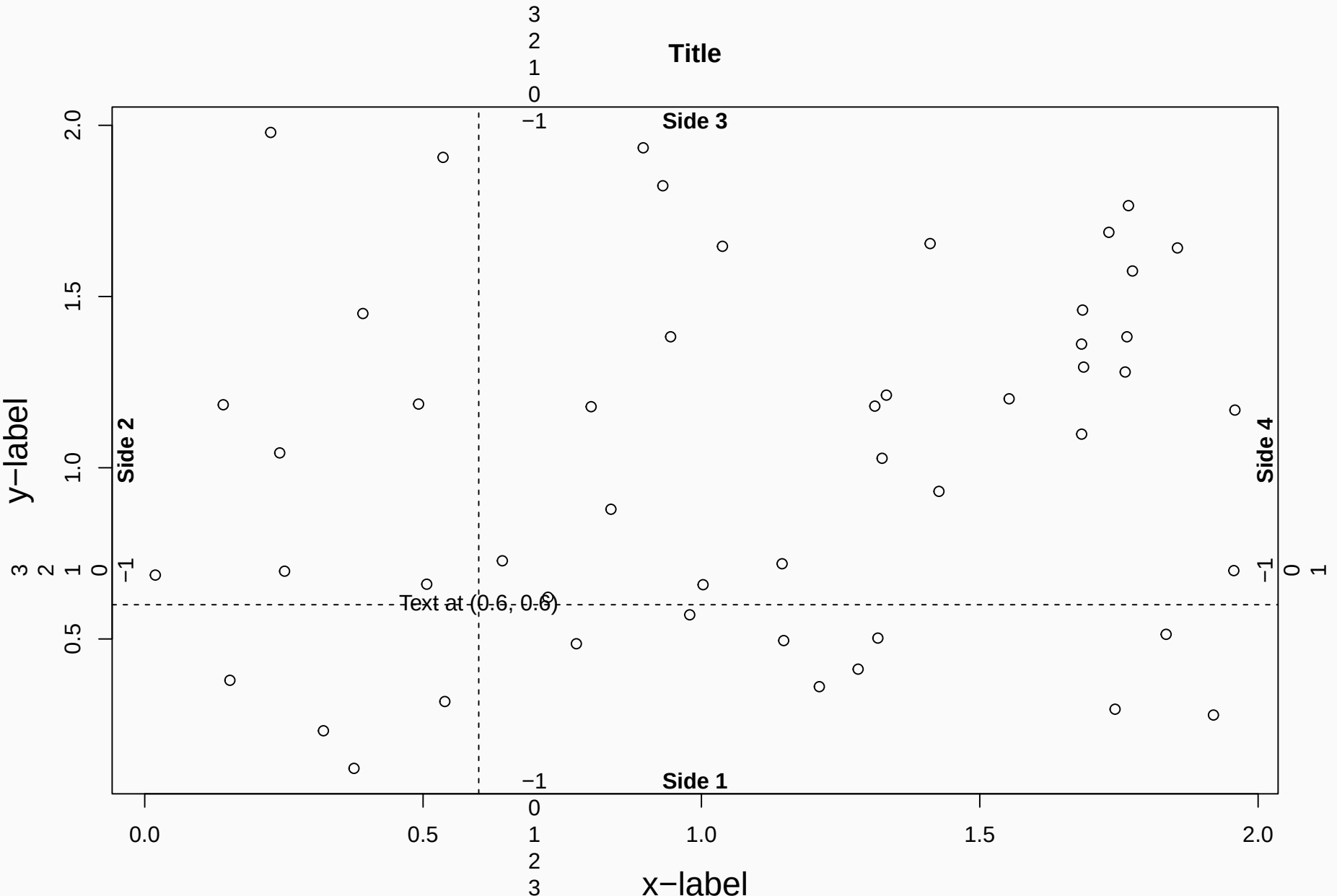
Uno scatter plot:

```
x <- runif(50, 0, 2) # 50 numeri random
y <- runif(50, 0, 2) # da una distr. uniforme
plot(x, y, main="Titolo",
     sub="Sottotitolo", xlab="x-label",
     ylab="y-label") # ecco il plot
```

Aggiunge del testo al plot

```
text(0.6, 0.6, "Testo @ (0.6, 0.6)")
abline(h=.6, v=.6, lty=2) # linee h. e v.
```

Margins region



Modificare il layout dei plot

Vanno creati dei pannelli

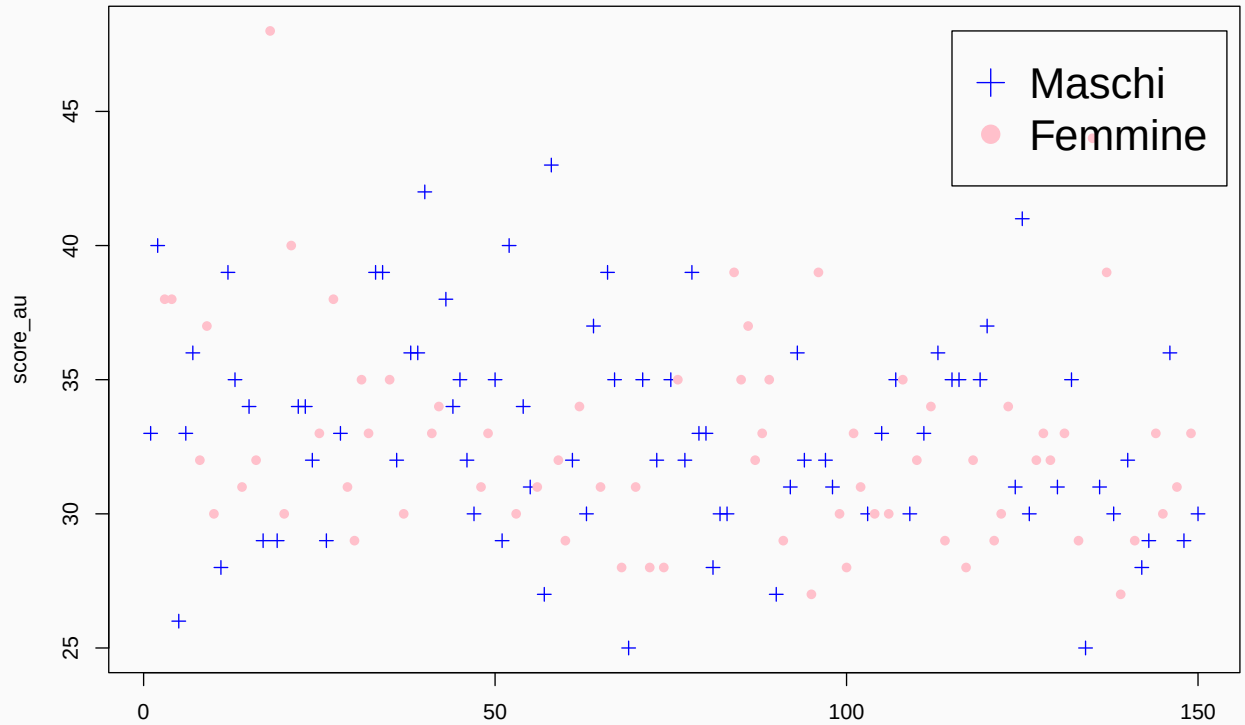
```
par(mfrow=c(nrighe, ncolonne)) # pannelli vengono riempiti per riga
par(mfcol=c(nrighe, ncolonne)) # pannelli vengono riempiti per colo
```

plot()

```
plot(x) # solo una variabile
plot(x, y) # due variabile (scatter plot)
plot(y ~ x) # due variabile, y in funzione di x
```

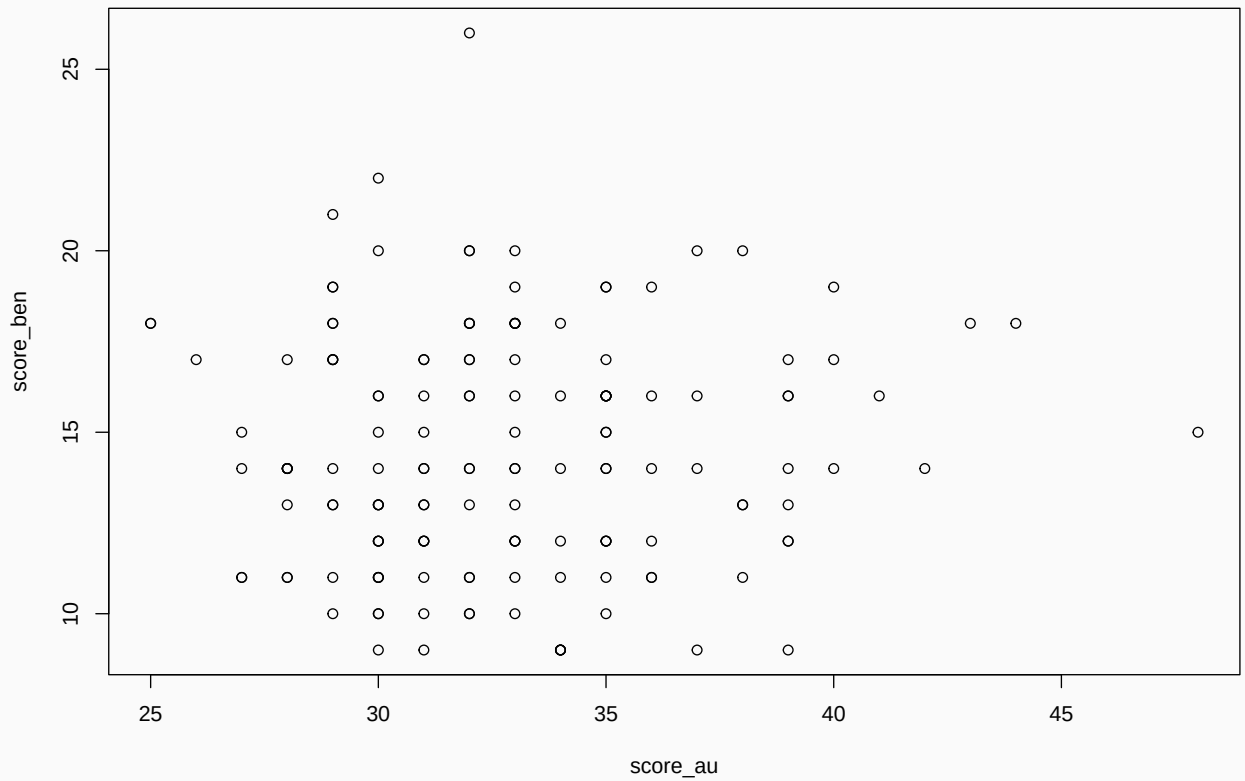
Esempi: `plot(x)`

```
with(benessere,
     plot(score_au,
          col = ifelse(genere == 1, "blue", "pink"),
          pch = ifelse(genere == 1, 3, 16)))
legend(x = 115, y = 48,
       c("Maschi", "Femmine"), pch = c(3, 16),
       col =c("blue", "pink"), cex = 2)
```



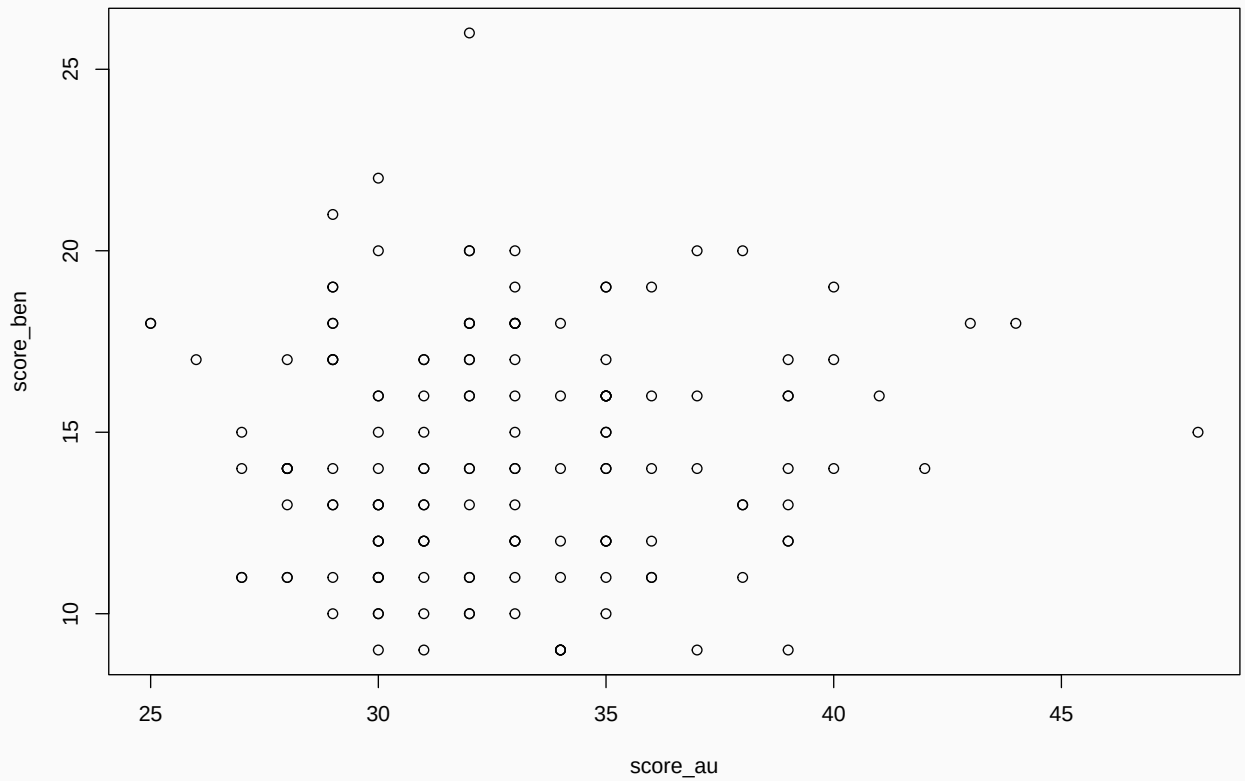
Esempi: `plot(x, y)`

```
with(benessere,
     plot(score_au, score_ben))
```



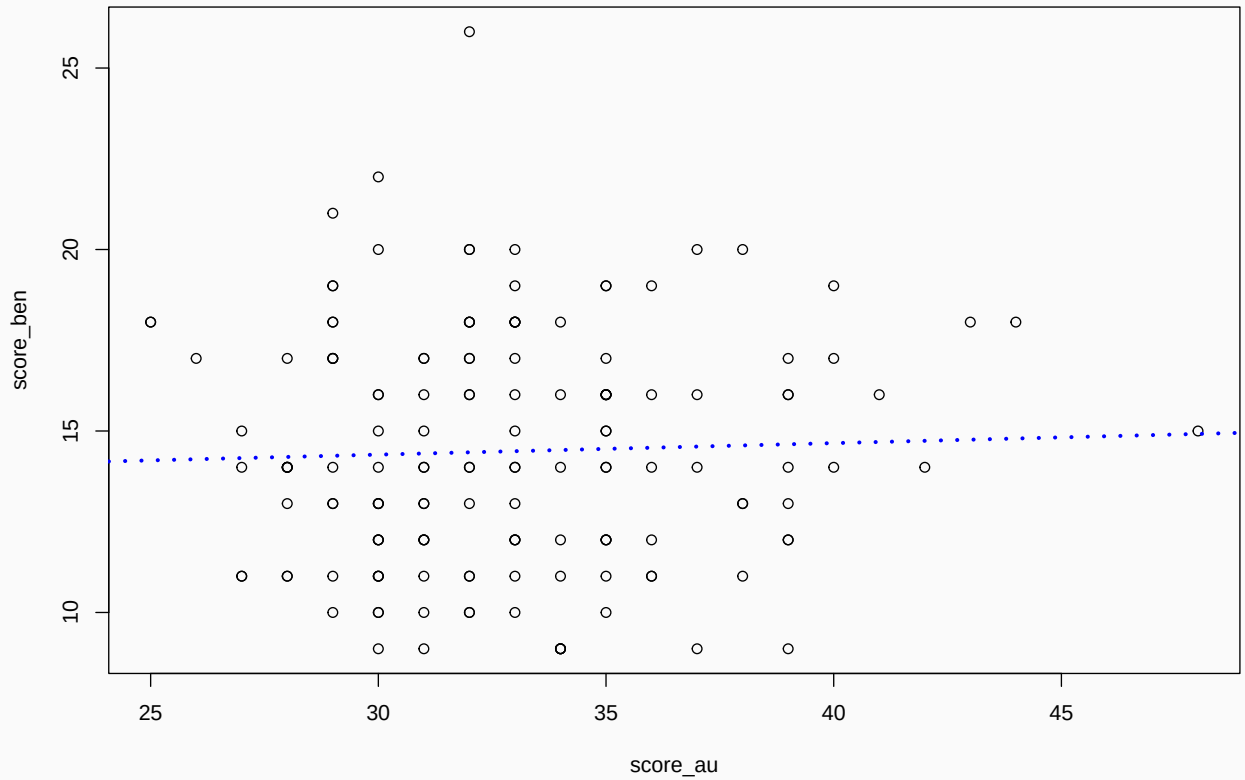
Esempi: `plot(y ~ x)`

```
with(benessere,
     plot(score_ben ~ score_au))
```



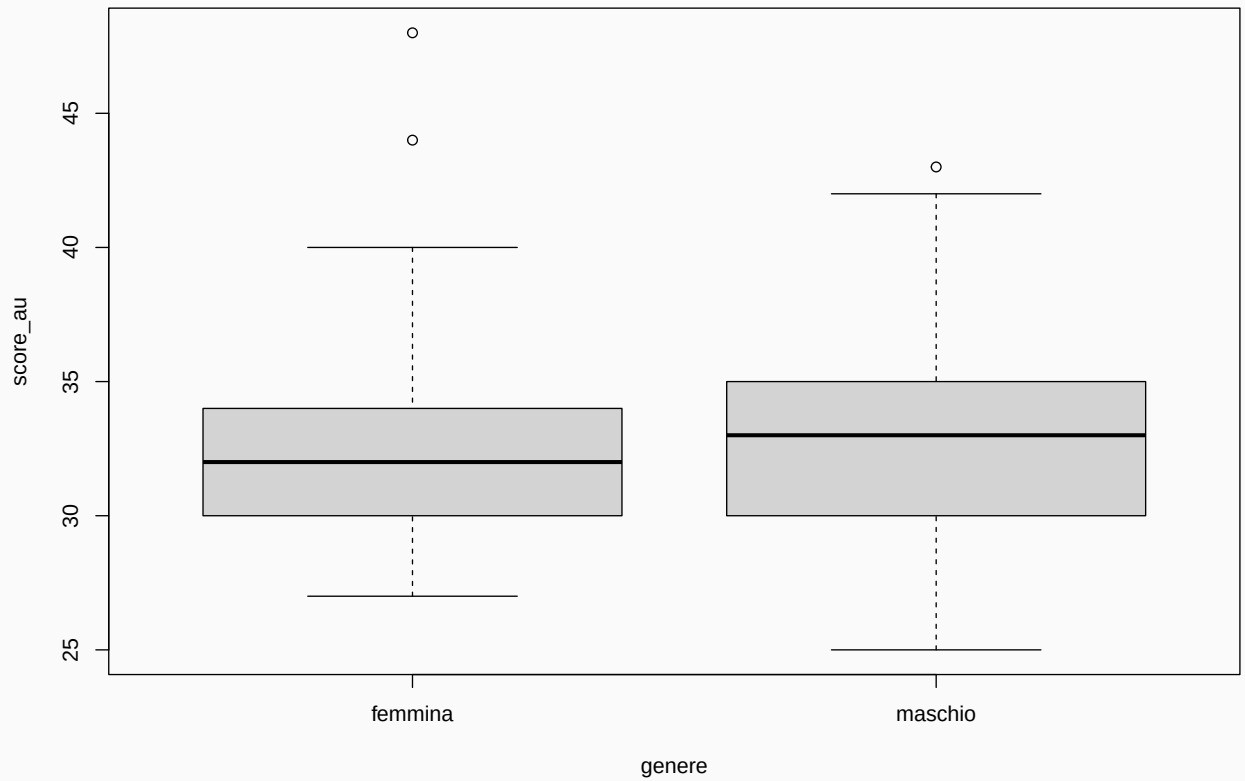
Esempi: `plot(y ~ x)` con retta di regressione

```
with(benessere,
     plot(score_ben ~ score_au))
abline(lm(score_ben ~ score_au, data = benessere),
          col = "blue", lty = 3, lwd = 3)
```



Esempi: `plot(y ~ x)` con x categoriale

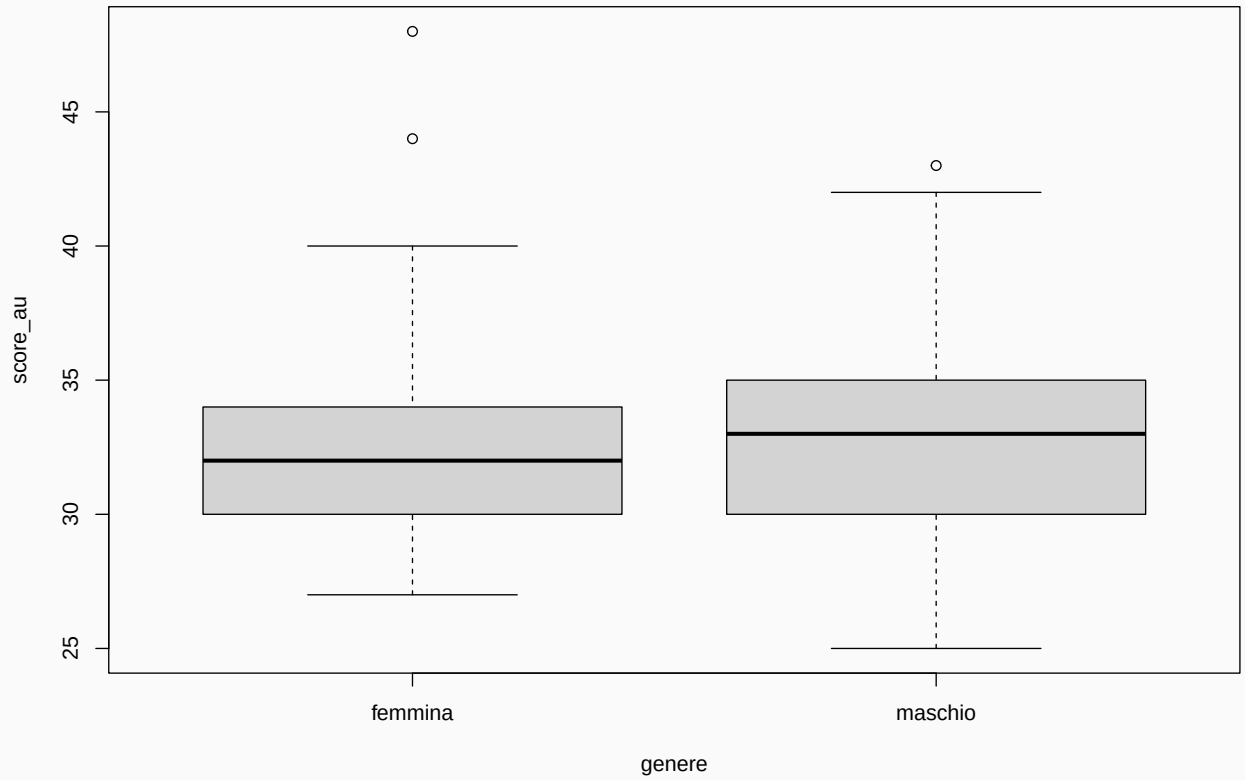
```
benessere$genere <- factor(ifelse(benessere$genere == 1,
                                  "maschio", "femmina"))
plot(score_au ~ genere, data = benessere)
```



Attenzione!

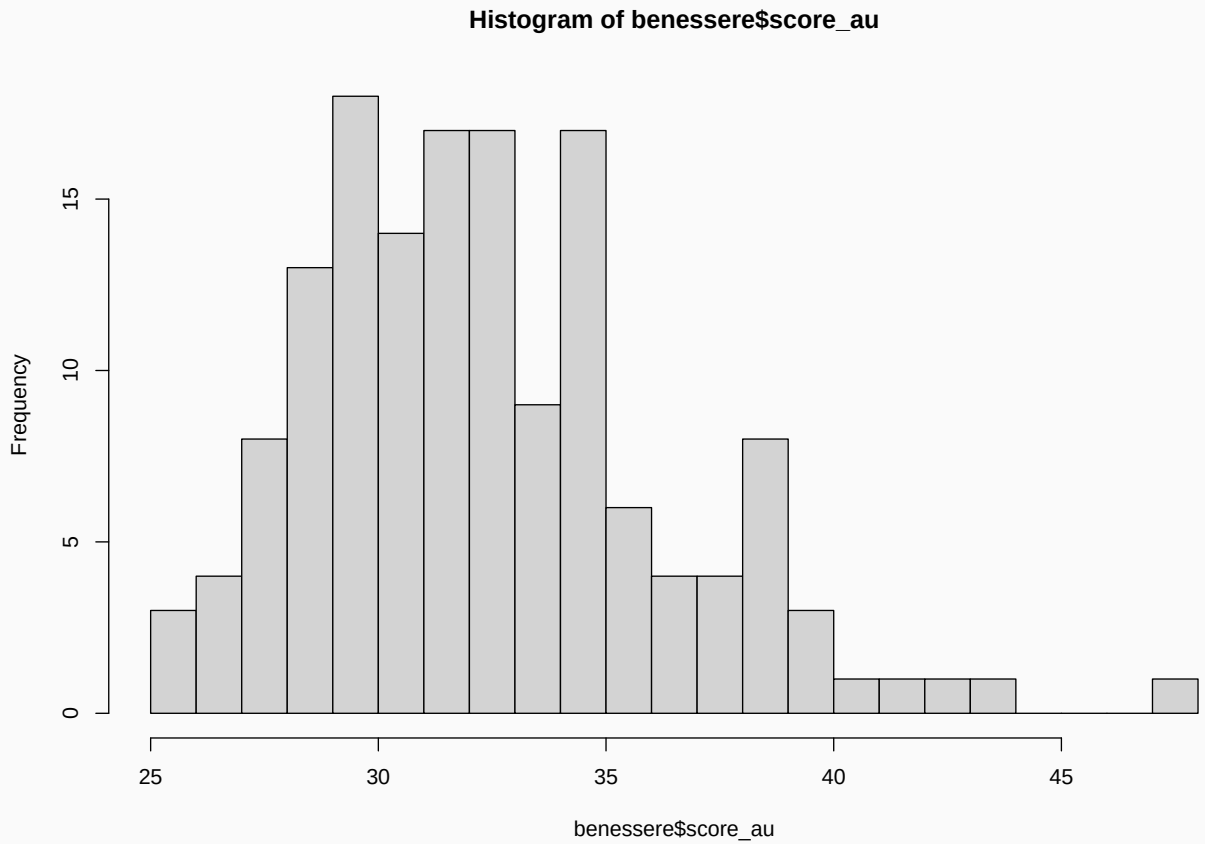
`plot(y ~ x)` con x categoriale è uguale a `boxplot(y ~ x)`

```
boxplot(score_au ~ genere, data = benessere)
```



hist(): Frequenze

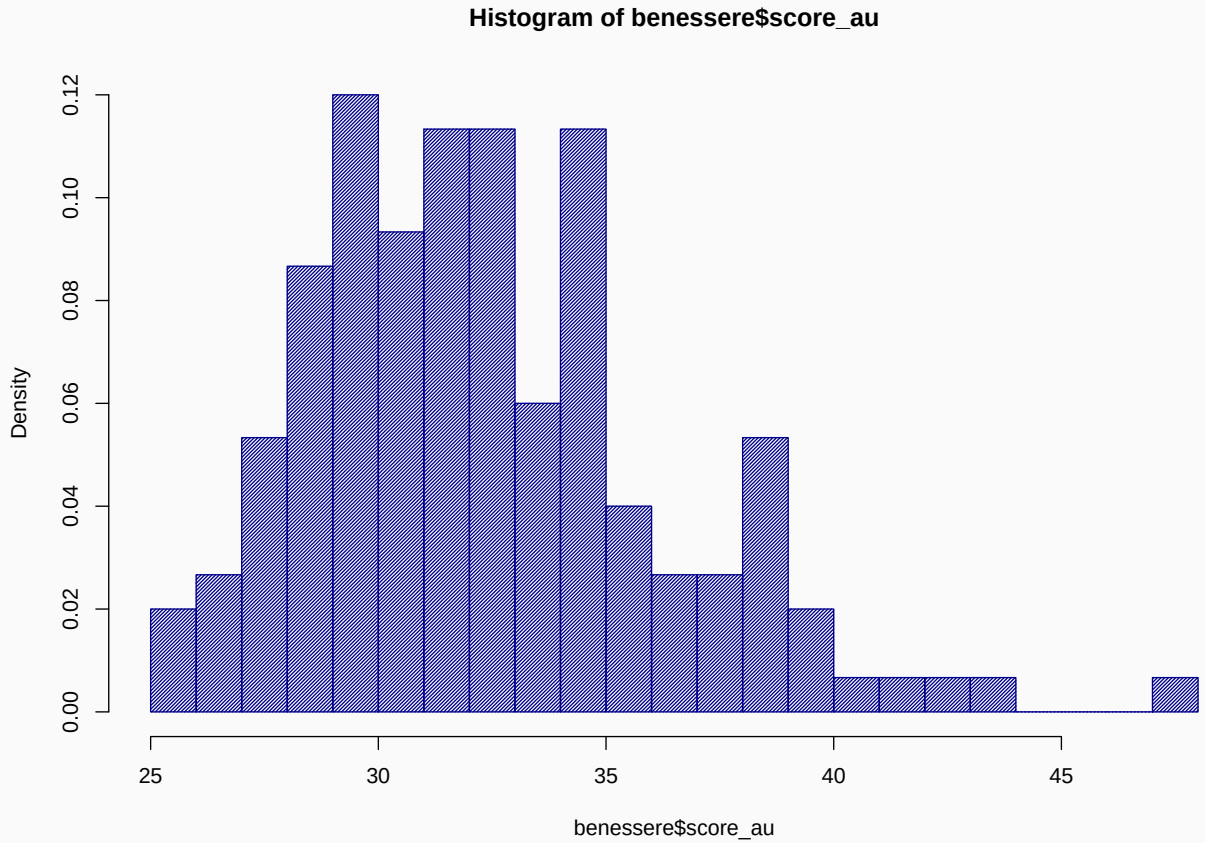
```
hist(benessere$score_au, breaks = 20)
```



hist(): Densità

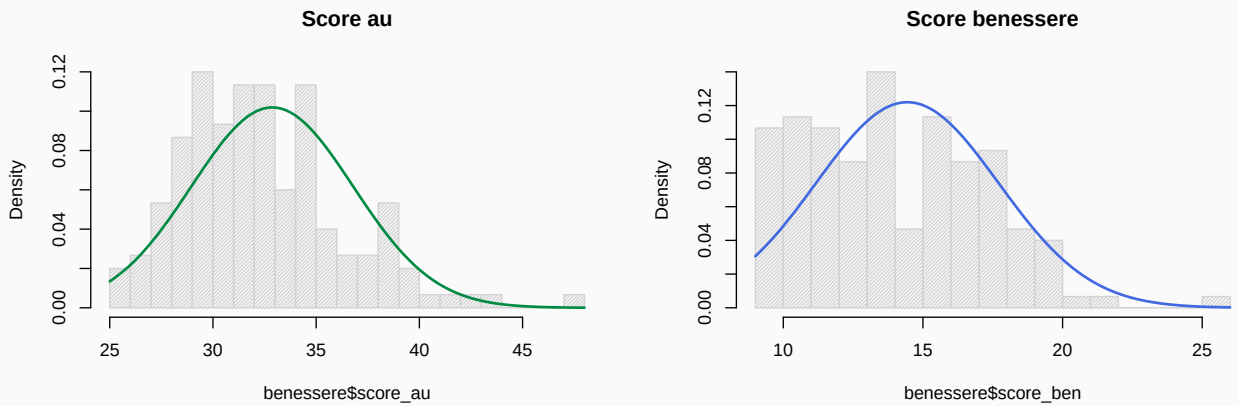
Densità

```
hist(benessere$score_au,
     density=50, breaks=20,
     prob=TRUE, col = "darkblue")
```

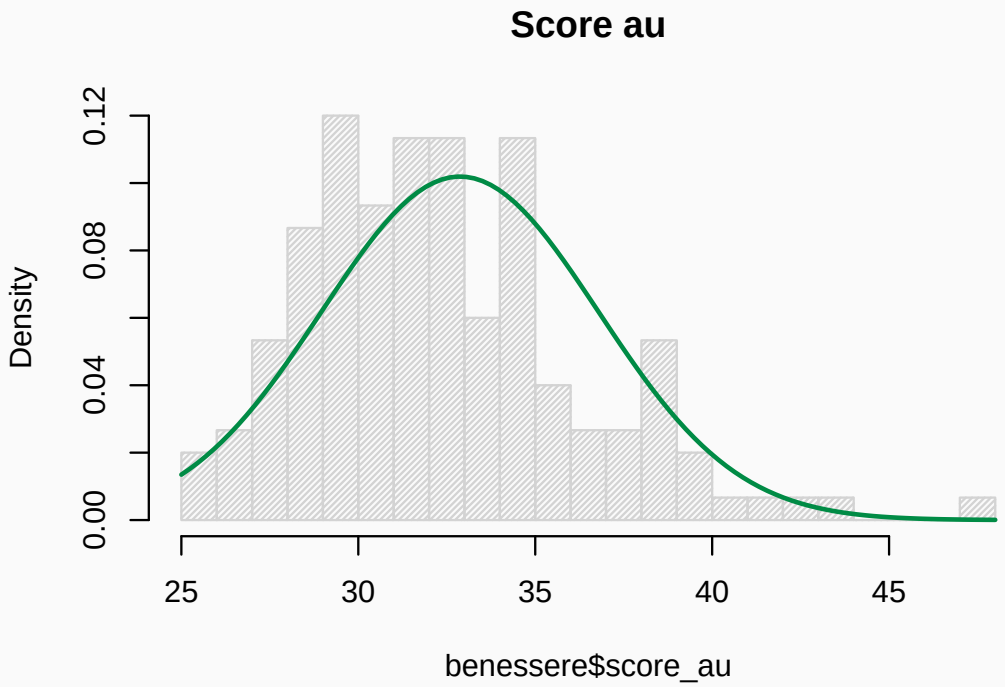


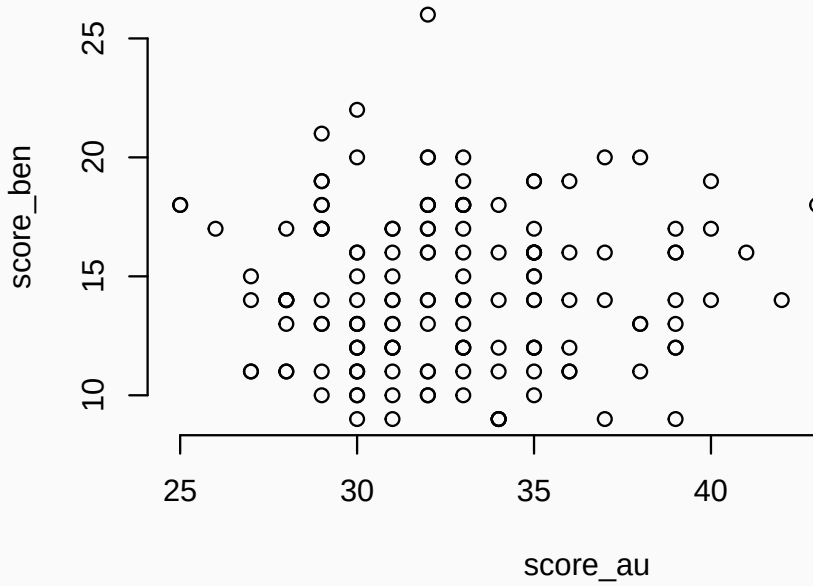
Multi plot (in riga)

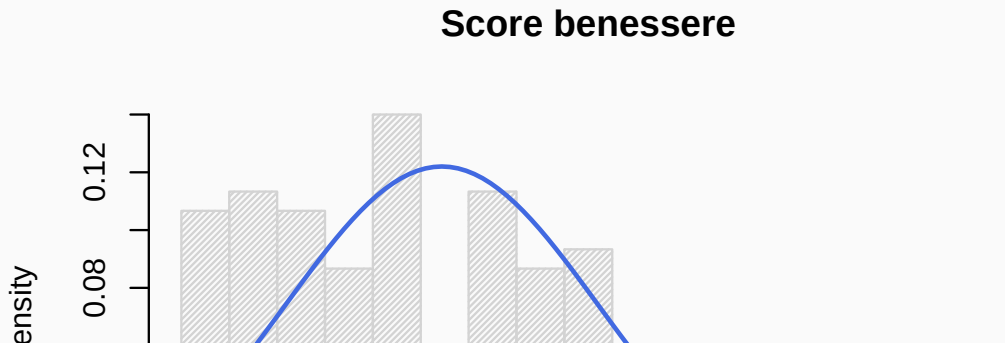
```
par(mfrow=c(1, 2))

hist(benessere$score_au,density=50, breaks=20, prob=TRUE,
     main = "Score au")
curve(dnorm(x, mean=mean(benessere$score_au),
            sd=sd(benessere$score_au)),
      col="springgreen4", lwd=2, add=TRUE, yaxt="n")

[...]
```

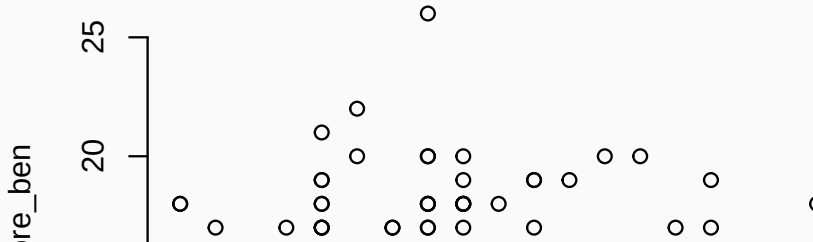


Multiplot (in colonna)









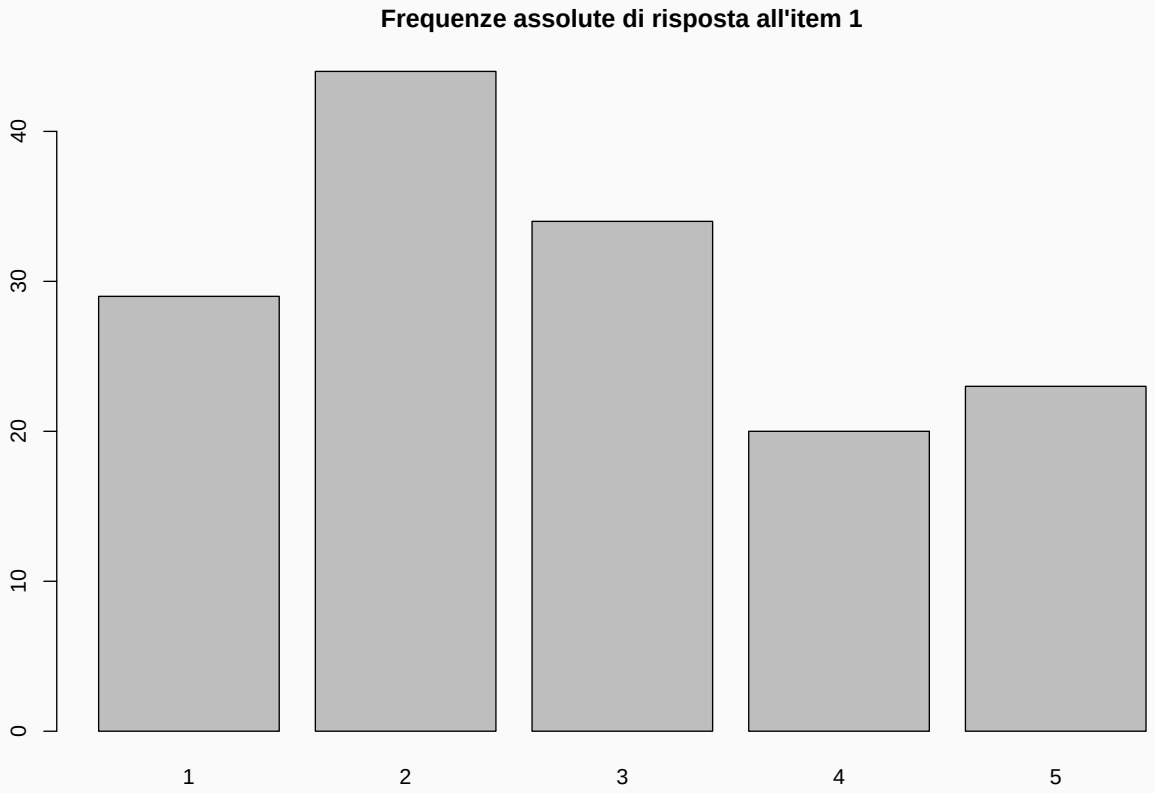
Multiplot (in colonna), codice

```
par(mfcol= c(2,2))
hist(benessere$score_au,density=50, breaks=20, prob=TRUE,
     main = "Score au")
curve(dnorm(x, mean=mean(benessere$score_au),
            sd=sd(benessere$score_au)),
      col="springgreen4", lwd=2, add=TRUE, yaxt="n")

hist(benessere$score_ben,density=50, breaks=20, prob=TRUE,
     main = "Score benessere")
curve(dnorm(x, mean=mean(benessere$score_ben),
            sd=sd(benessere$score_ben)),
      col="royalblue", lwd=2, add=TRUE, yaxt="n")

with(benessere,
     plot(score_au, score_ben, frame = FALSE))
```

barplot(): Frequenze assolute

Per creare i grafici a barre quando si hanno variabili discrete o categoriali

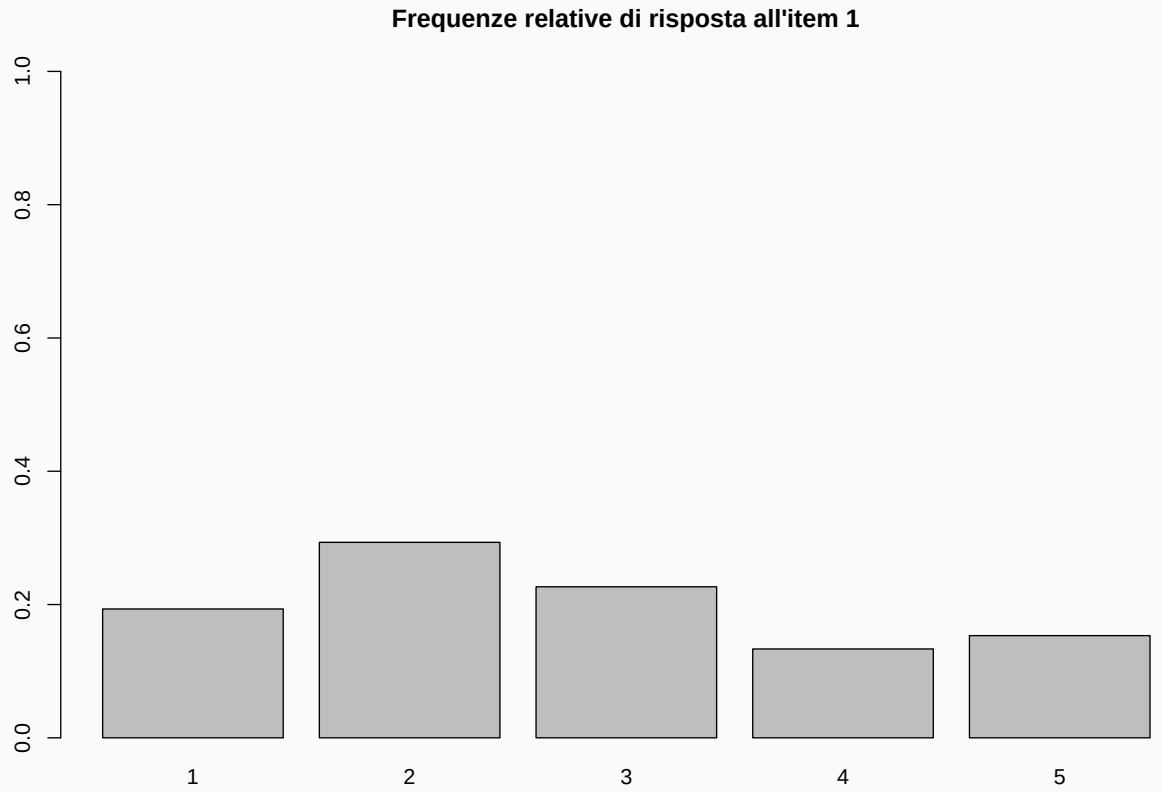
```
freq_item1 = table(benessere$item1)
barplot(freq_item1,
        main = "Frequenze assolute di risposta all'item 1")
```



barplot(): Frequenze relative

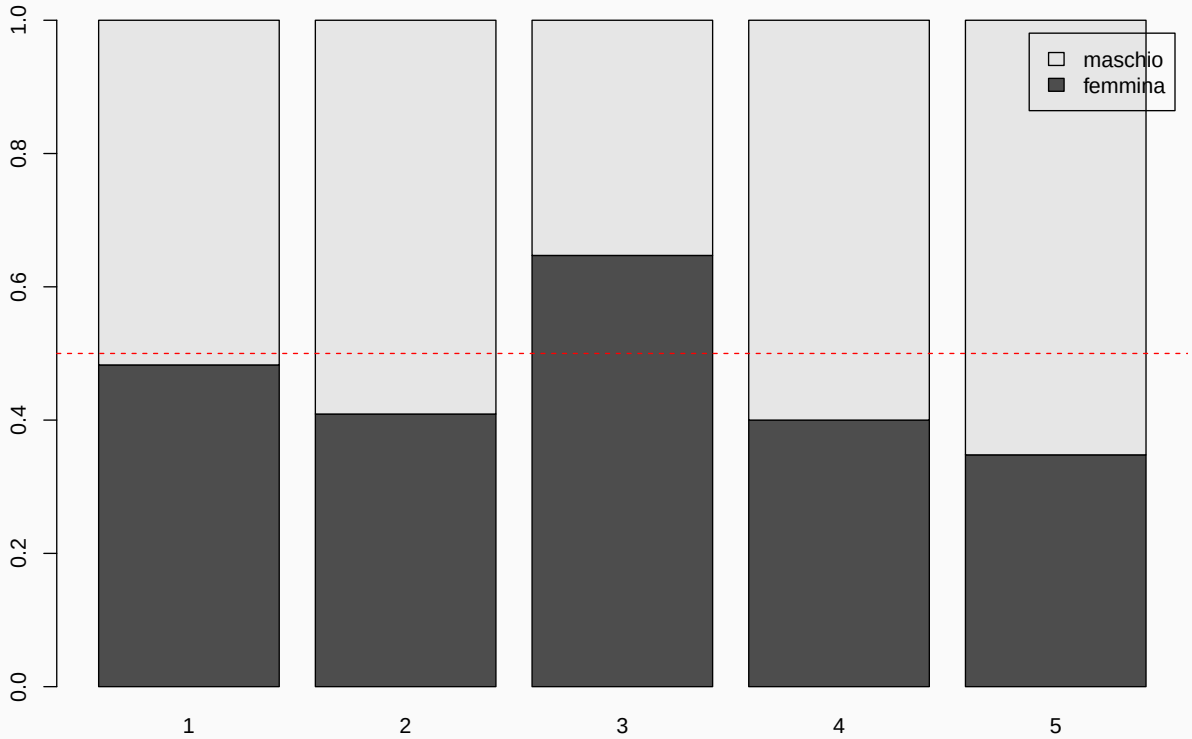
Richiede uno step in più → la creazione della tabella delle frequenze

```
perc_item1 = freq_item1/sum(freq_item1)
barplot(perc_item1, ylim = c(0, 1),
        main = "Frequenze relative di risposta all'item 1")
```

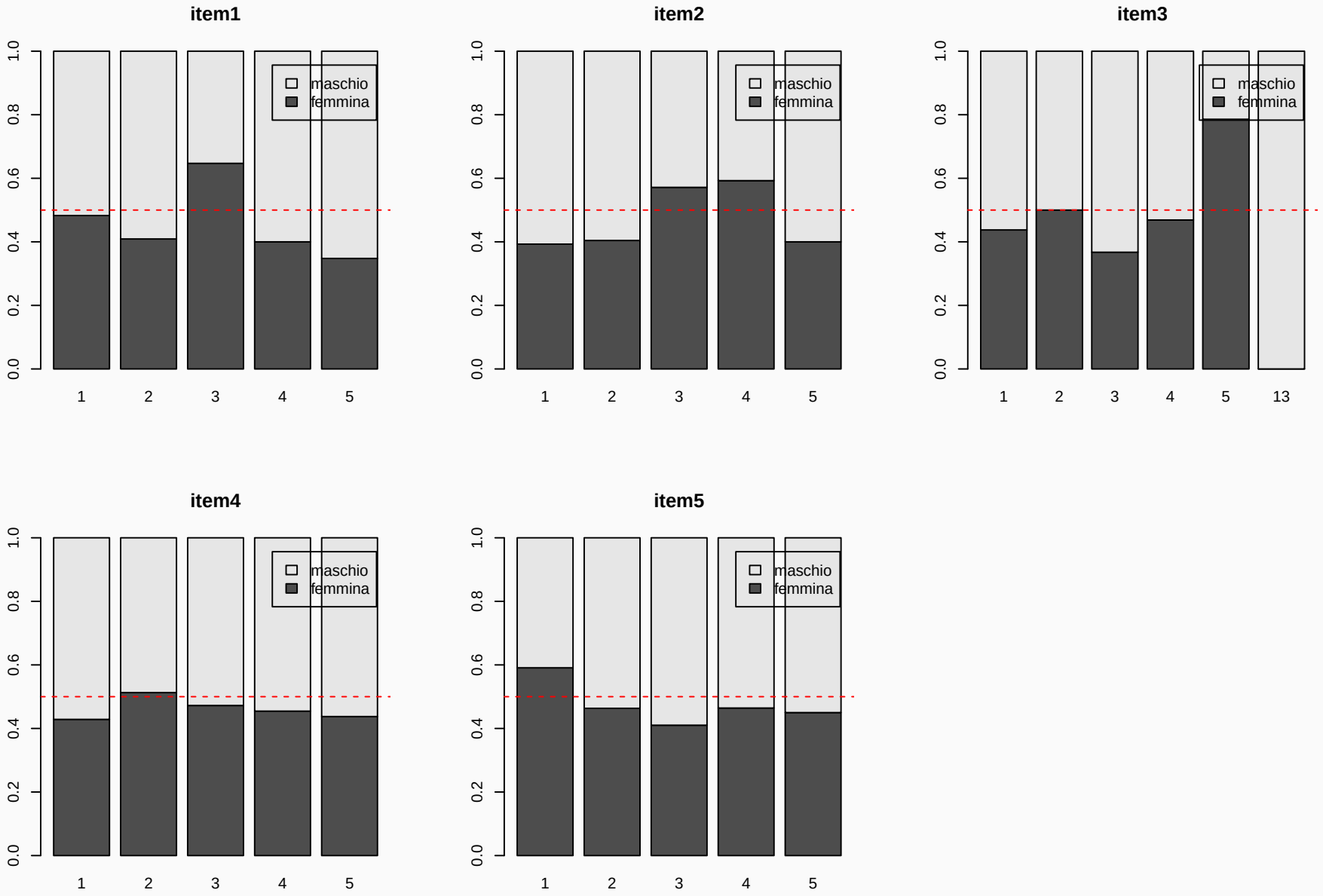


barplot() con più variabili

```
perc_item1_gender = table(benessere$genere, benessere$item1)
perc_item1_gender[1,] = perc_item1_gender[1,]/table(benessere$item1)
perc_item1_gender[2,] = perc_item1_gender[2,]/table(benessere$item1)

barplot(perc_item1_gender, ylim=c(0,1),
        legend = rownames(perc_item1_gender))
abline(h = .5, lty = 2, col = "red")
```



Un altro esempio di multiplot



Un esempio di multiplot (codice)

```
item_ben = benessere[, grep("item", colnames(benessere))]

par(mfrow = c(2, round(ncol(item_ben)/2  + 0.2)))
temp = NULL
for (i in 1:ncol(item_ben)) {
  temp = table(benessere$genere, item_ben[,i])
  for (j in 1:nrow(temp)) {
    temp[j,] = temp[j,]/table(item_ben[,i])
  }
barplot(temp, ylim=c(0,1), legend = rownames(temp),
        main = colnames(item_ben)[i])
abline(h = .5, lty = 2, col = "red")
}
```

interaction.plot()

Permette di vedere l'interazione tra due variabili a seconda di una terza variabile:

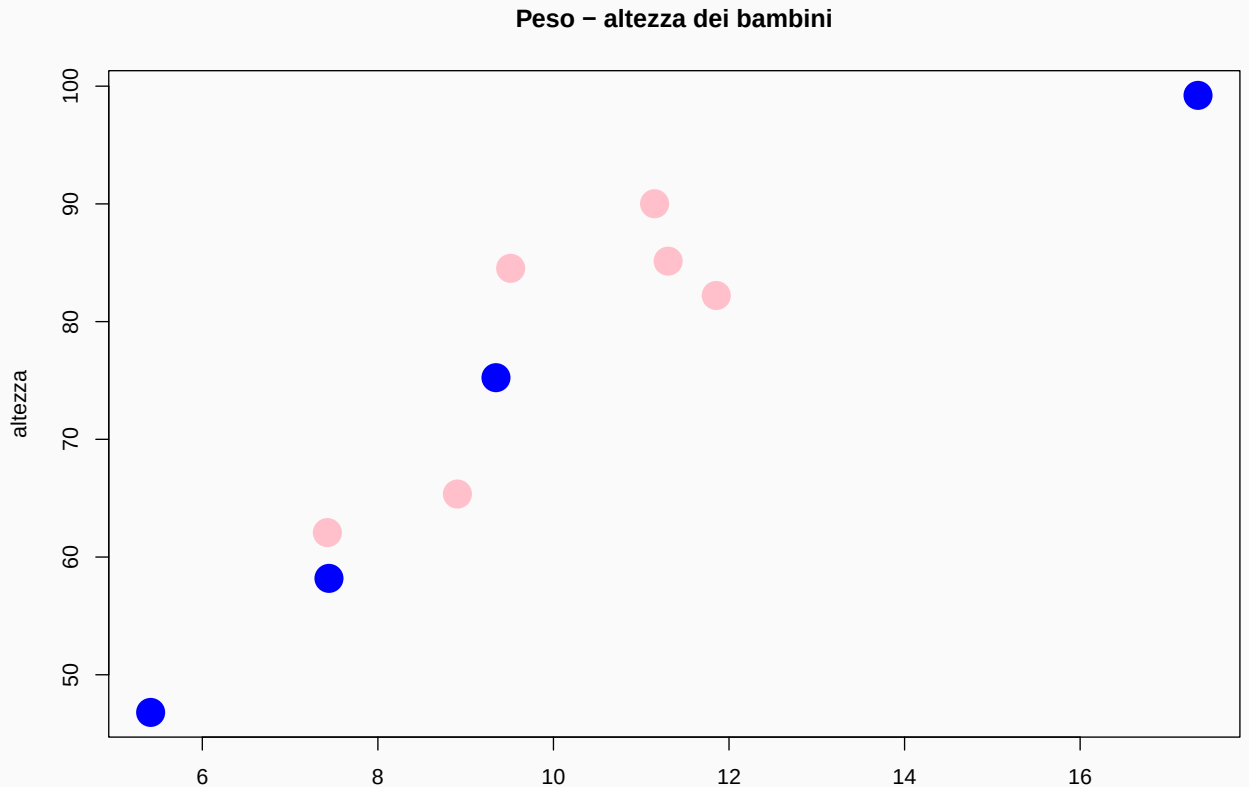
```
interaction.plot(x, v. categoriale, y)
```

interaction.plot()

Permette di vedere l'interazione tra due variabili a seconda di una terza variabile:

```
interaction.plot(x, v. categoriale, y)
```

La relazione tra peso e altezza cambia a seconda del genere?



```
babies$cat_weight = with(babies,
                         ifelse(
                           peso <= quantile(babies$peso)[2],
                           "light",
                           ifelse(peso > quantile(babies$peso)[2] &
                                    peso > quantile(babies$peso)[4],
       "medium",
       "heavy")))
babies$cat_weight = factor(babies$cat_weight,
                           levels = c("light",
                                      "medium",
                                      "heavy"))


babies
```

```
      id genere      peso  altezza cat_weight
1   baby1      f  7.424646 62.07722      light
2   baby2      m  7.442727 58.18877      light
3   baby3      f  9.512598 84.52737      heavy
4   baby4      f 11.306349 85.13573     medium
5   baby5      m  9.345165 75.23783      heavy
6   baby6      m  5.411290 46.80163      light
```

```
with(babies,
     interaction.plot(cat_weight, genere, altezza))
```

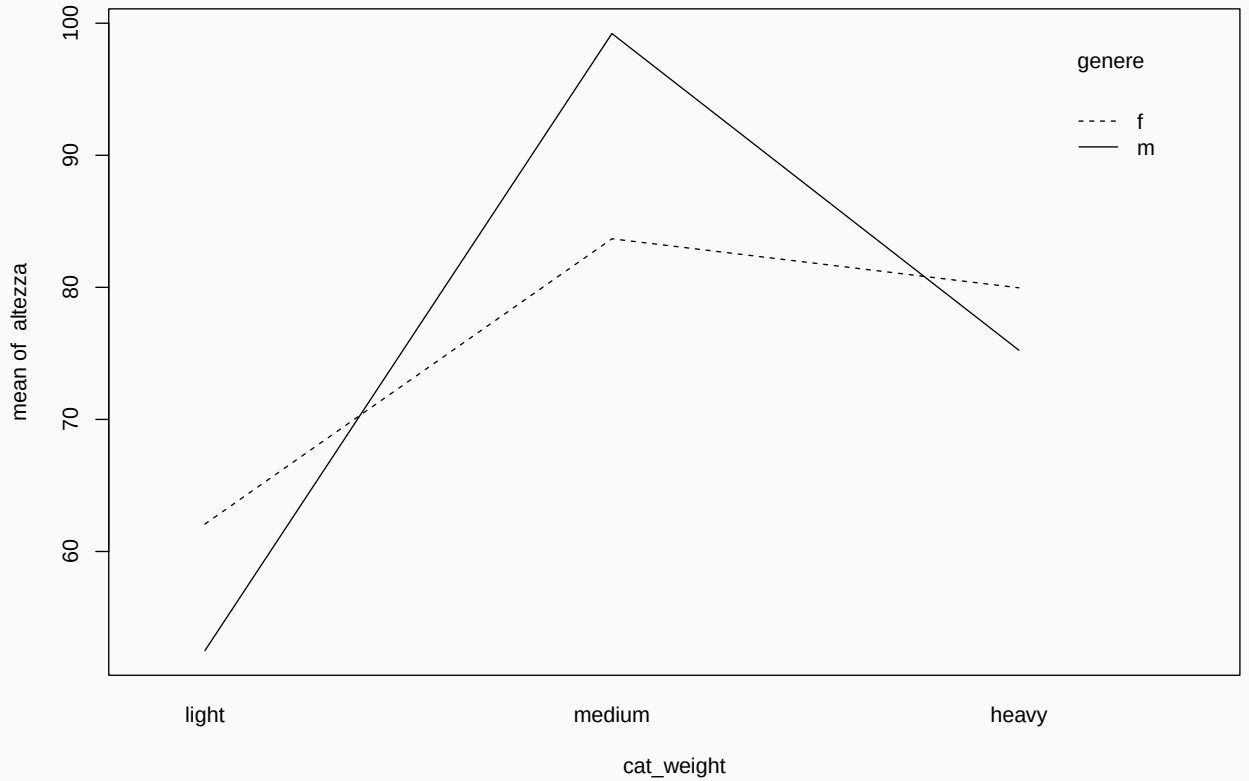


Table of Contents

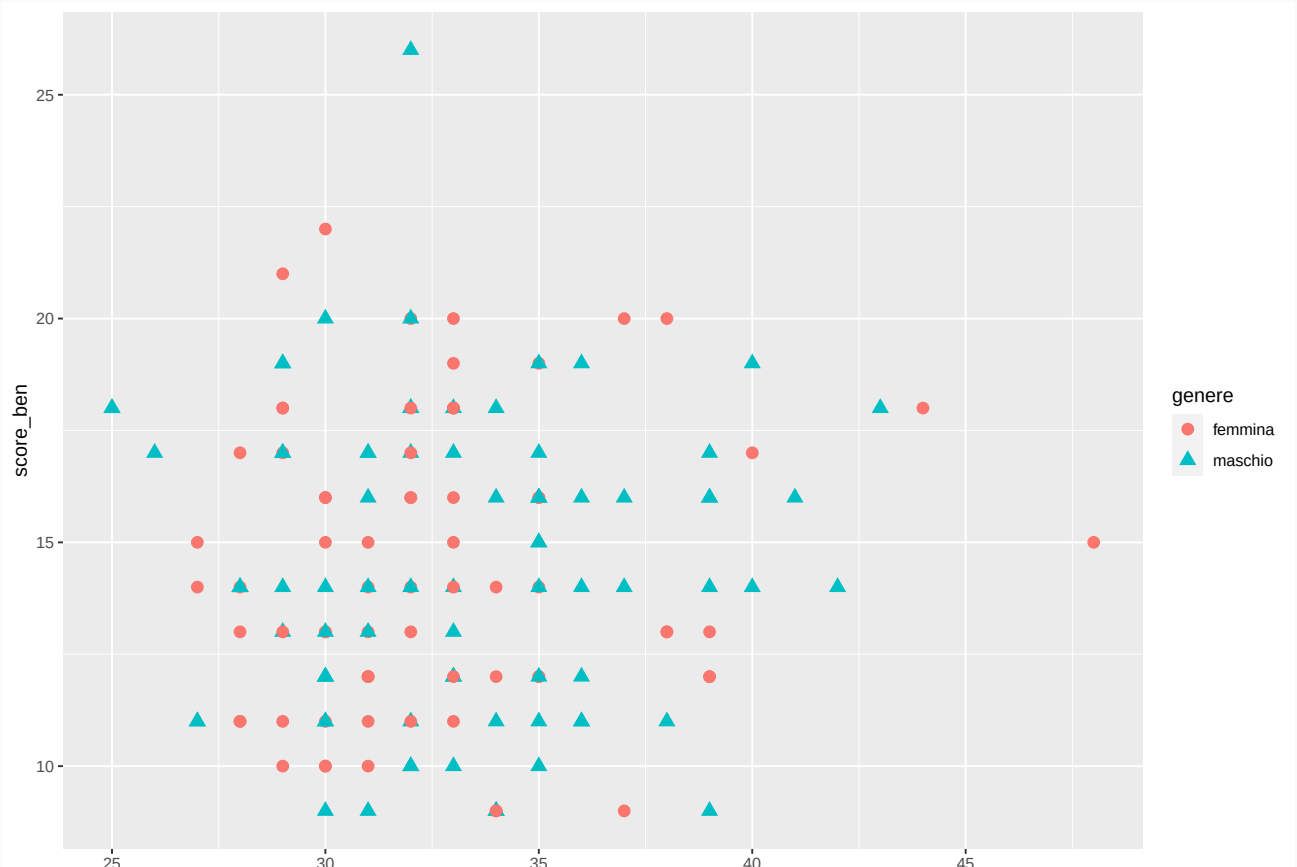
è più difficile ma è anche più facile:

```
ggplot(dati,
       ars(x = variabile.x,
           y = variabile.y,
           col = variabile.colore.contorno,
           fill =  variabile.colore.filling,
           shape =  variabile.shape,
           size = variabile.size,
           ...)) + geom_tipo.grafico() + ...
```

Solitamente vuole i dati in formato long

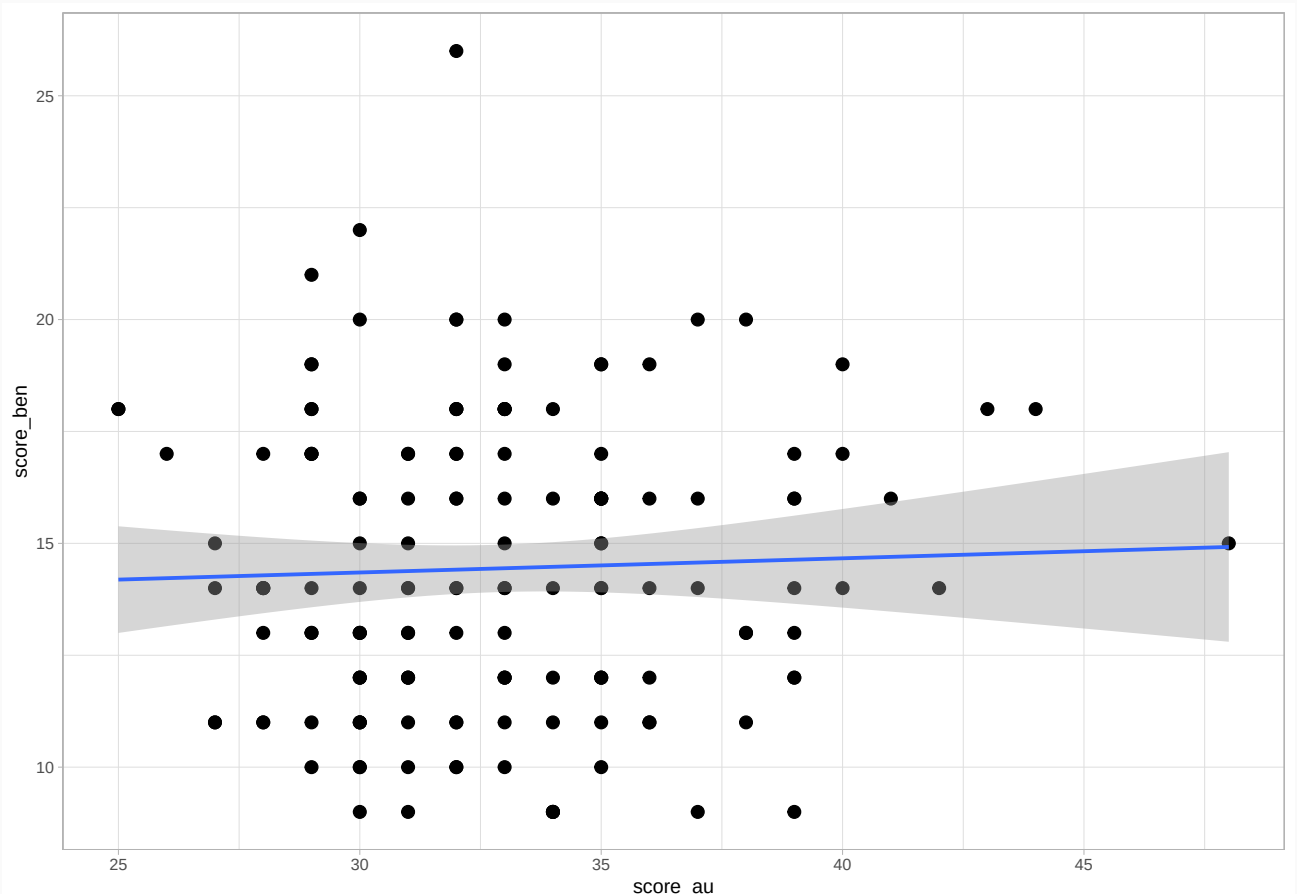
Scatter plot

```
ggplot(benessere,
       aes(x = score_au, y = score_ben,
           col = genere,
           shape = genere)) +
  geom_point(size = 3)
```



Scatter plot con retta di regressione

```
ggplot(benessere,
       aes(x = score_au, y = score_ben)) +
  geom_point(size = 3) +
  geom_smooth(method = "lm", formula = y ~ x) + theme_light()
```



boxplot() e violinplot()

Sono i grafici prediletti per far vedere le distribuzioni dei dati (specie il violin)

Richiedono che i dati siano in formato long:

```
    id condition mean_time
1 sbj1         A  4.136174
2 sbj1         B  2.639523
3 sbj2         A  2.547628
4 sbj2         B  4.319068
5 sbj3         A  4.265100
6 sbj3         B  4.113846
```

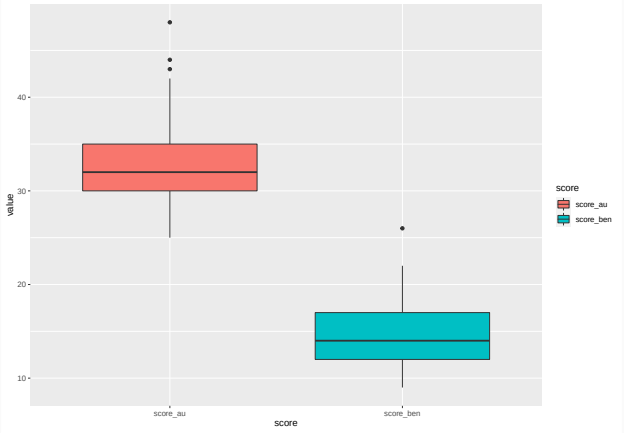
```r
small = benessere[, c("ID", "score_au", "score_ben")]
score_long  = reshape(small,
        idvar = "ID",
        times =names(small)[-1],
        timevar = "score", v.names = "value",
        varying = list(names(small)[-1]),
        direction = "long")
head(score_long)
```

```
           ID    score value
1.score_au  1 score_au    33
2.score_au  2 score_au    40
3.score_au  3 score_au    38
4.score_au  4 score_au    38
5.score_au  5 score_au    26
6.score_au  6 score_au    33
```

boxplot vs violinplot

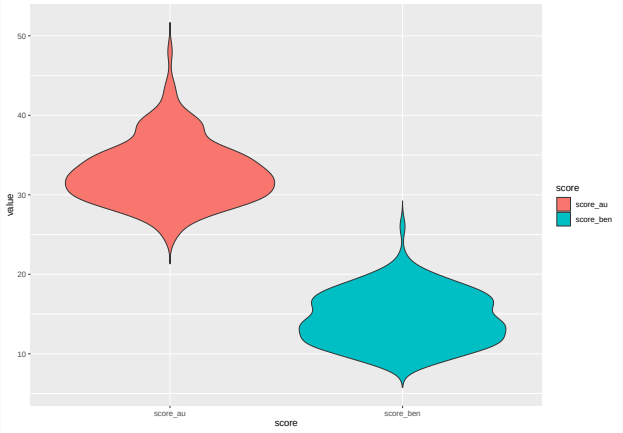
Boxplot

```
ggplot(score_long,
       aes(x = score, y = value,
           fill = score)) +
  geom_boxplot()
```

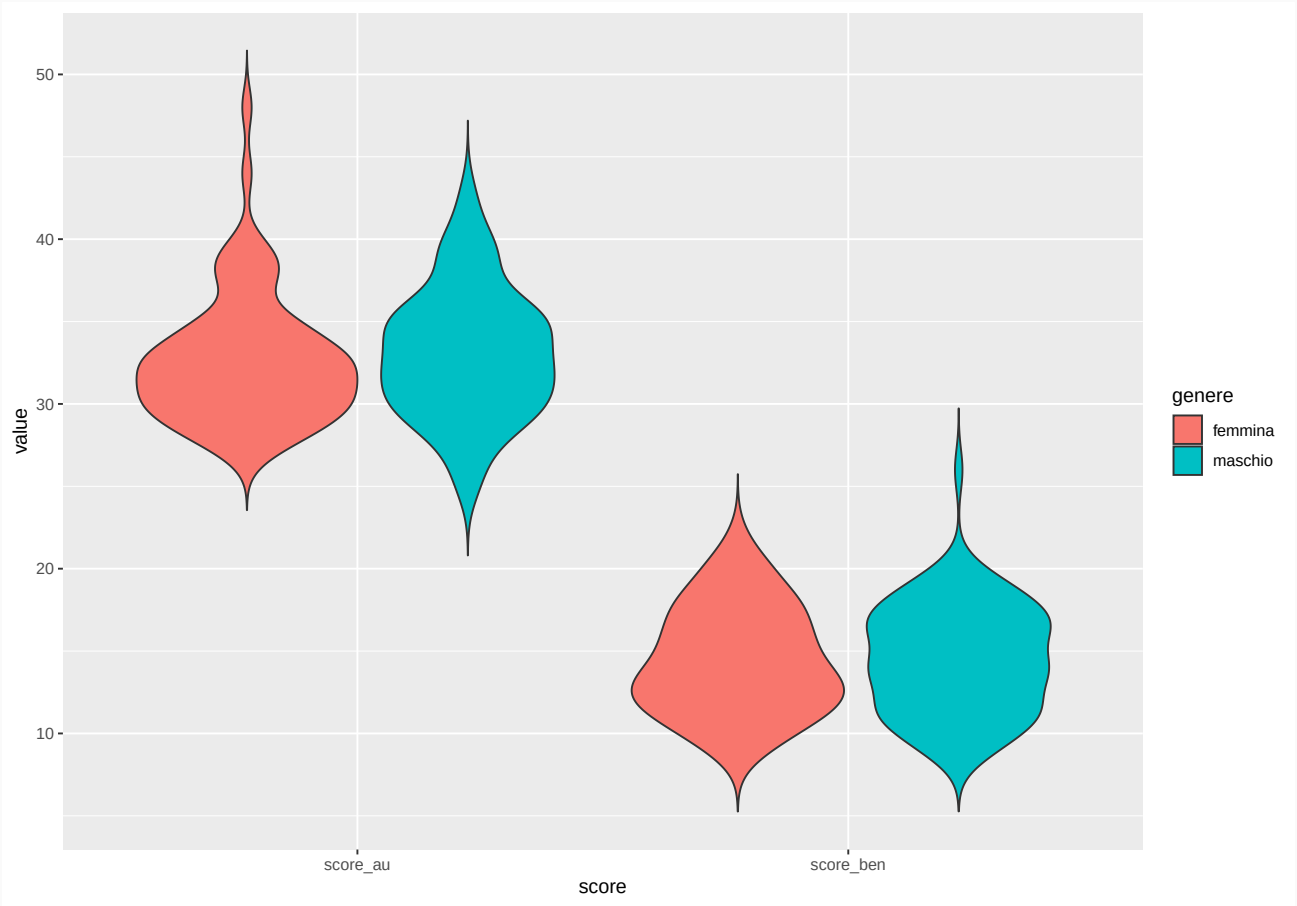


Violinplot

```
ggplot(score_long,
       aes(x = score, y = value,
           fill = score)) +
  geom_violin(trim = FALSE)
```



```r
score_long = merge(score_long, benessere[, c("ID", "genere")])

ggplot(score_long,
       aes(x = score, y = value,
           fill = genere)) + geom_violin(trim = FALSE)
```



barplot e istogrammi

`geom_hist()`: istogramma (per variabili continue)

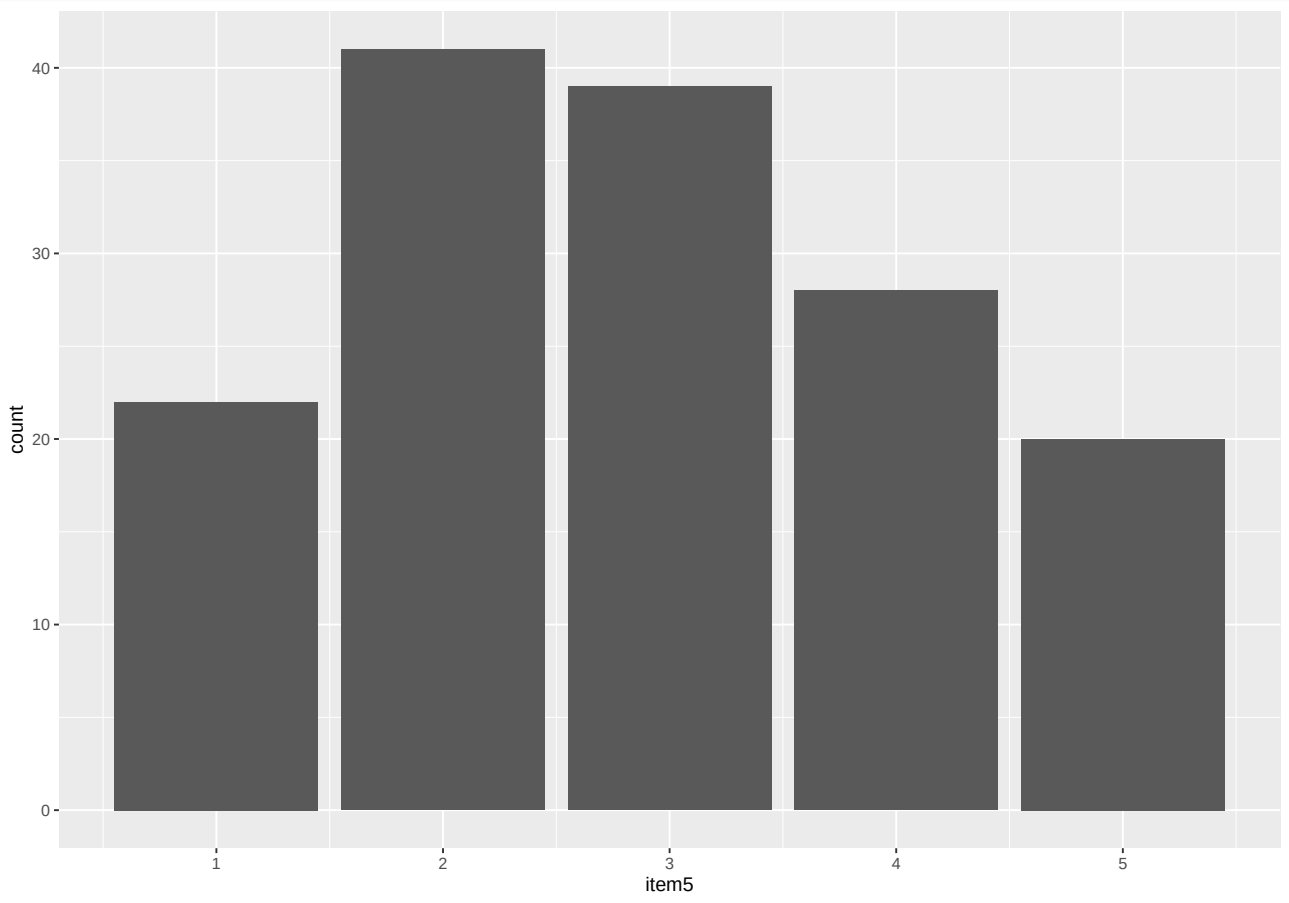
`geom_bar()`: grafico a barre

Prevede degli argomenti:

- `geom_bar(stat = "count")`: conta in automatico le frequenze per ogni modalità del carattere, **non vuole una variabile y**
- `geom_bar(stat = "identity")`: plotta un valore associato ad ogni modalità del carattere, **vuole una variabile y**

geom_bar(stat = "count"):

```
ggplot(benessere,
       aes(x = item5)) + geom_bar(stat = "count")
```



geom_bar(stat = "identity")

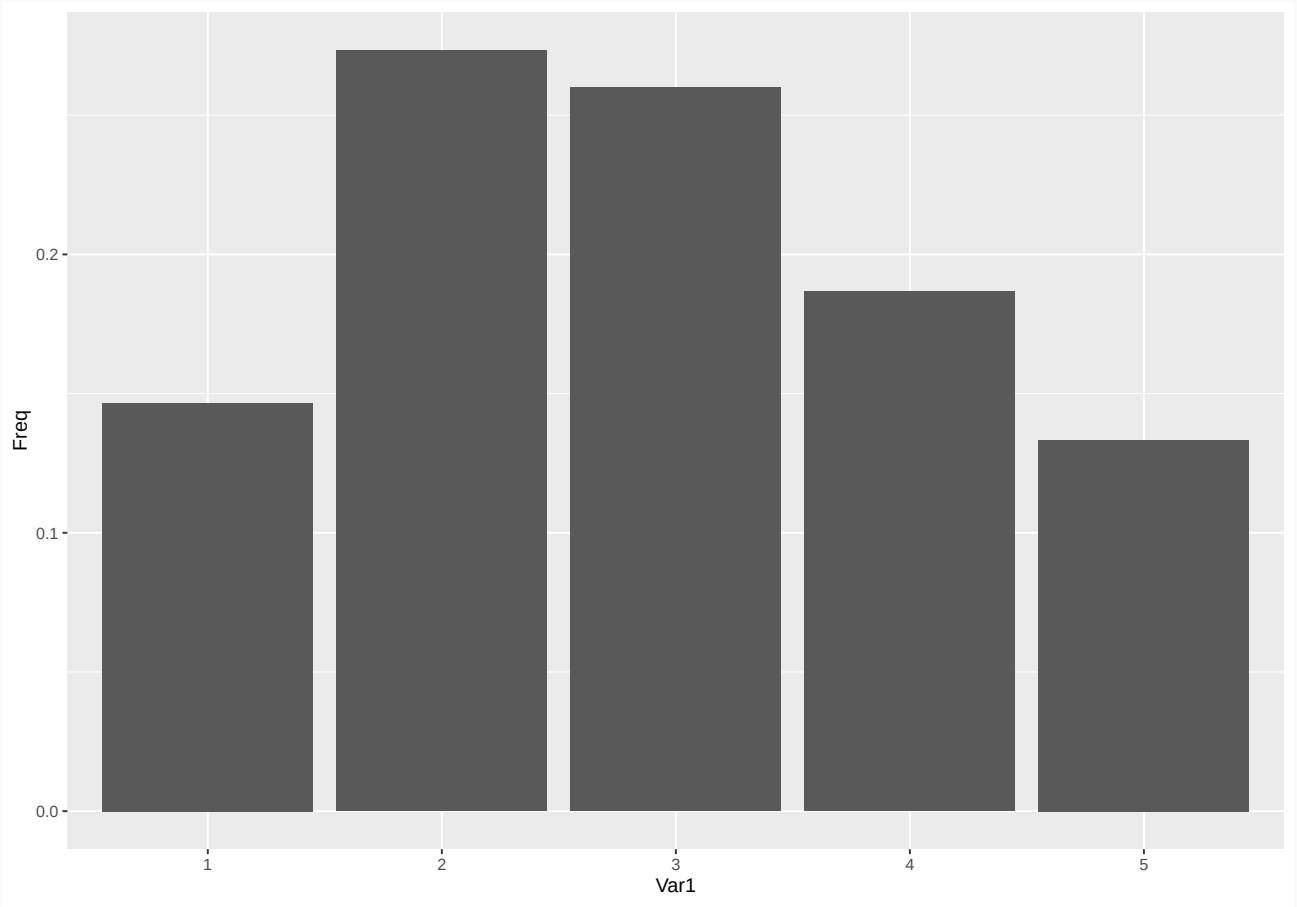
Vanno prima calcolati i valori per ogni modalità del carattere (e.g., proporzioni o percentuali):

```
item_5 = data.frame(table(benessere$item5)/nrow(benessere))
item_5
```

```
  Var1      Freq
1    1 0.1466667
2    2 0.2733333
3    3 0.2600000
4    4 0.1866667
5    5 0.1333333
```

Poi arriva il grafico

```
ggplot(item_5,
       aes(x = Var1, y = Freq)) + geom_bar(stat = "identity")
```



Anche tutti insieme!

Creo il dataset in formato long dei miei dati

```
new_item = benessere[, c(grep("ID", colnames(benessere)),
                         grep("item", colnames(benessere)))]
new_item = reshape(new_item,
        idvar = "ID",
        times =names(new_item)[-1],
        timevar = "item", v.names = "value",
        varying = list(names(new_item)[-1]),
        direction = "long")
new_item
```

```
           ID  item value
1.item1     1 item1     1
2.item1     2 item1     2
3.item1     3 item1     2
4.item1     4 item1     3
5.item1     5 item1     4
....
```

Calcolo le proporzioni

Di ogni opzione di risposta `value` per ogni item `item`

```
proporzione = new_item %>%
  group_by(item, value) %>%
  summarise(prop = n()/nrow(benessere))

proporzione
```

```
# A tibble: 26 x 3
# Groups:   item [5]
   item  value  prop
   <chr> <int> <dbl>
 1 item1     1 0.193
 2 item1     2 0.293
....
```

Finalmente plotto

```
ggplot(proporzione,
       aes(x = value, y = prop)) +
  geom_bar(stat = "identity") +
  facet_wrap(~item) # mi crea i pannelli
```

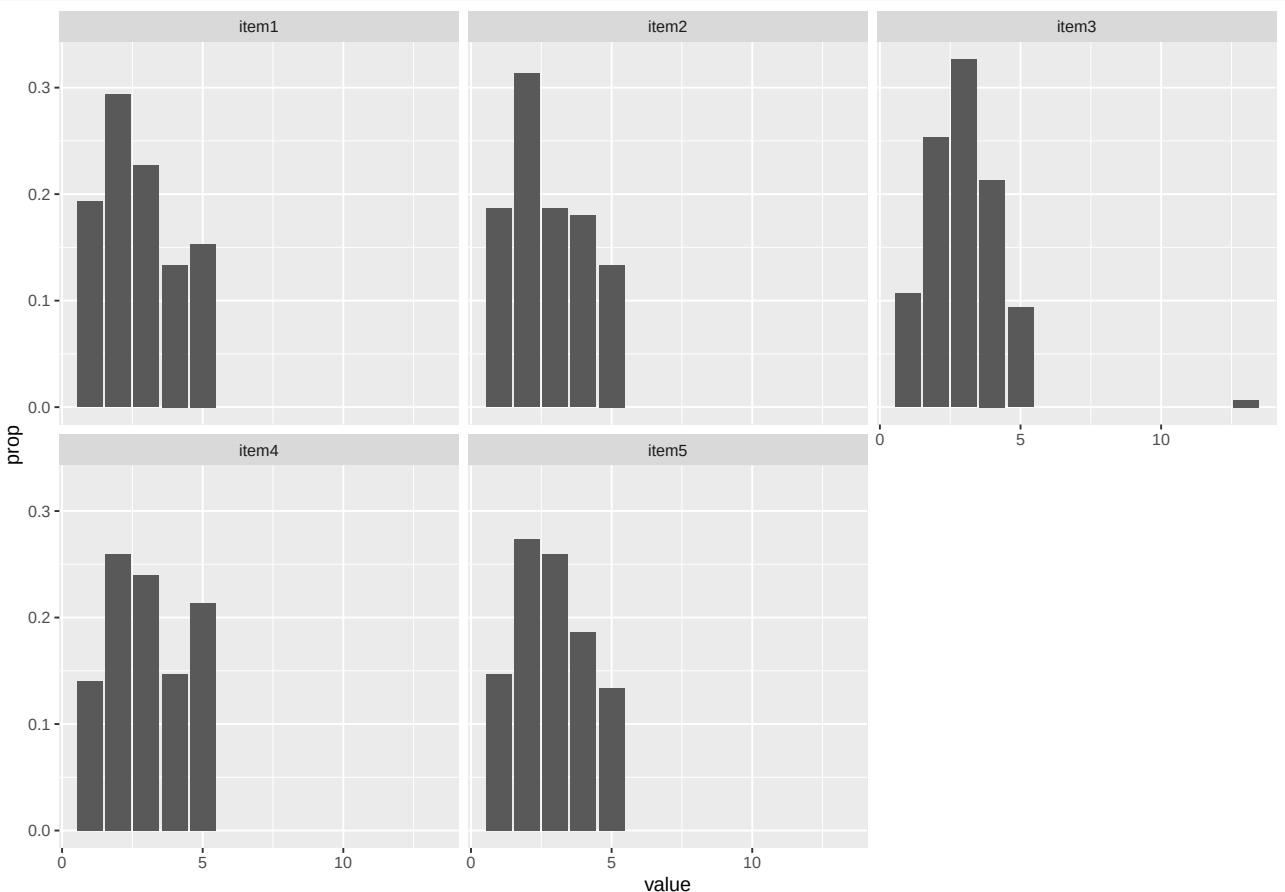


Table of Contents

In automnatico

```
pdf("nome_grafico.pdf")
png("nome_grafico.png")
tiff("nome_grafico.tiff")
jpeg("nome_grafico.jpeg")
bmp("nome_grafico.bmp")
```

Dopo che si è esportato il grafico:

```
dev.off()
```

Esempio

```
pdf("il_mio_violin.pdf")
ggplot(score_long,
       aes(x = score, y = value,
           fill = score)) + geom_violin(trim = FALSE)
dev.off()
```

A mano

```
knitr::include_graphics("data/esporta.png")
```