

Aggiungere una colonna ad un dataframe

Dopo aver inizializzato un dataframe possiamo aggiungere uno o più vettori colonna mediante una semplice assegnazione; nell'esempio seguente inizialmente il dataframe *df* è costituito da una sola colonna (vettore) di numeri da 1 a 6; successivamente viene aggiunta una seconda colonna con i quadrati dei numeri da 1 a 6. Per riferirci alla seconda colonna, o a una colonna qualsiasi, utilizzeremo l'operatore `[[...]]` che può servirci anche per modificare una colonna esistente.

Esempio 1

```
Console ~/R/ ↵
> v=1:6
> df=data.frame(v)
> df
  v
1 1
2 2
3 3
4 4
5 5
6 6
> df[[2]]=v^2
> df
  v  v2
1 1   1
2 2   4
3 3   9
4 4  16
5 5  25
6 6  36
```

Il comando `expand.grid`

Se lanciamo due dadi i possibili risultati sono $6 \cdot 6 = 36$ coppie di valori, ad esempio (2, 6) cioè 2 sul primo dado e 6 sul secondo oppure (6, 2) cioè 6 sul primo dado e 2 sul secondo. Come possiamo ottenere con R tutte le possibili coppie? Utilizzeremo il comando `expand.grid` che ha, nel nostro caso, sia in entrata sia in uscita un dataframe.

Esempio 1

```
Console ~/R/ ↵
> df=data.frame(dado1=1:6,dado2=1:6)
> df
  dado1 dado2
1     1     1
2     2     2
3     3     3
4     4     4
5     5     5
6     6     6
> expand.grid(df)
  dado1 dado2
1     1     1
2     2     1
3     3     1
4     4     1
5     5     1
6     6     1
7     1     2
8     2     2
9     3     2
10    4     2
11    5     2
12    6     2
```

```
13    1     3
14    2     3
15    3     3
16    4     3
17    5     3
18    6     3
19    1     4
20    2     4
21    3     4
22    4     4
23    5     4
24    6     4
25    1     5
26    2     5
27    3     5
28    4     5
29    5     5
30    6     5
31    1     6
32    2     6
33    3     6
34    4     6
35    5     6
36    6     6
```

Se lanciamo 3 dadi i possibili risultati sono $6^3=216$ terne di valori. Per ottenere tutte le terne utilizzeremo i comandi

```
df=data.frame(dado1=1:6, dado2=1:6, dado3=1:6)
expand.grid(df)
```

Provate!

Per ottenere tutte le possibili somme ad esempio di due dadi utilizzeremo il comando `rowSums(dataframe)` che ci fornisce in uscita il vettore delle somme dei valori di ciascuna riga del dataframe in entrata.

Esempio 2

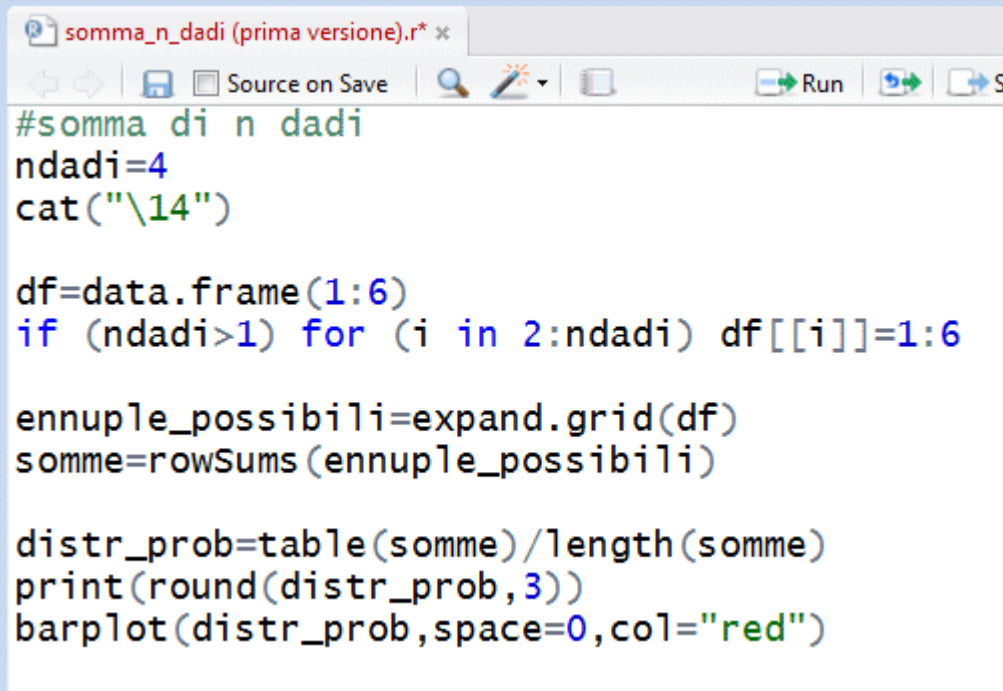
```
Console ~/R/ ↵
> df=data.frame(dado1=1:6, dado2=1:6)
> coppie_possibili=expand.grid(df)
> rowSums(coppie_possibili)
 [1]  2  3  4  5  6  7  3  4  5  6  7  8  4  5  6  7  8  9  5  6  7  8  9 10
[25]  6  7  8  9 10 11  7  8  9 10 11 12
```

Per capire come sono state ottenute le somme considerate le 36 coppie dell'esempio 1; quindi $2=1+1$, $3=2+1$, $4=3+1$, ecc.

Osservate che una stessa somma si può ottenere più volte, ad esempio la somma 7 compare sei volte.

Somma di n dadi

Ora abbiamo tutti gli strumenti per scrivere un programma che ci fornisca la distribuzione di probabilità della variabile casuale S ="somma di n dadi equi". Ecco il programma:



```
somma_n_dadi (prima versione).r* x
Source on Save
Run

#somma di n dadi
ndadi=4
cat("\14")

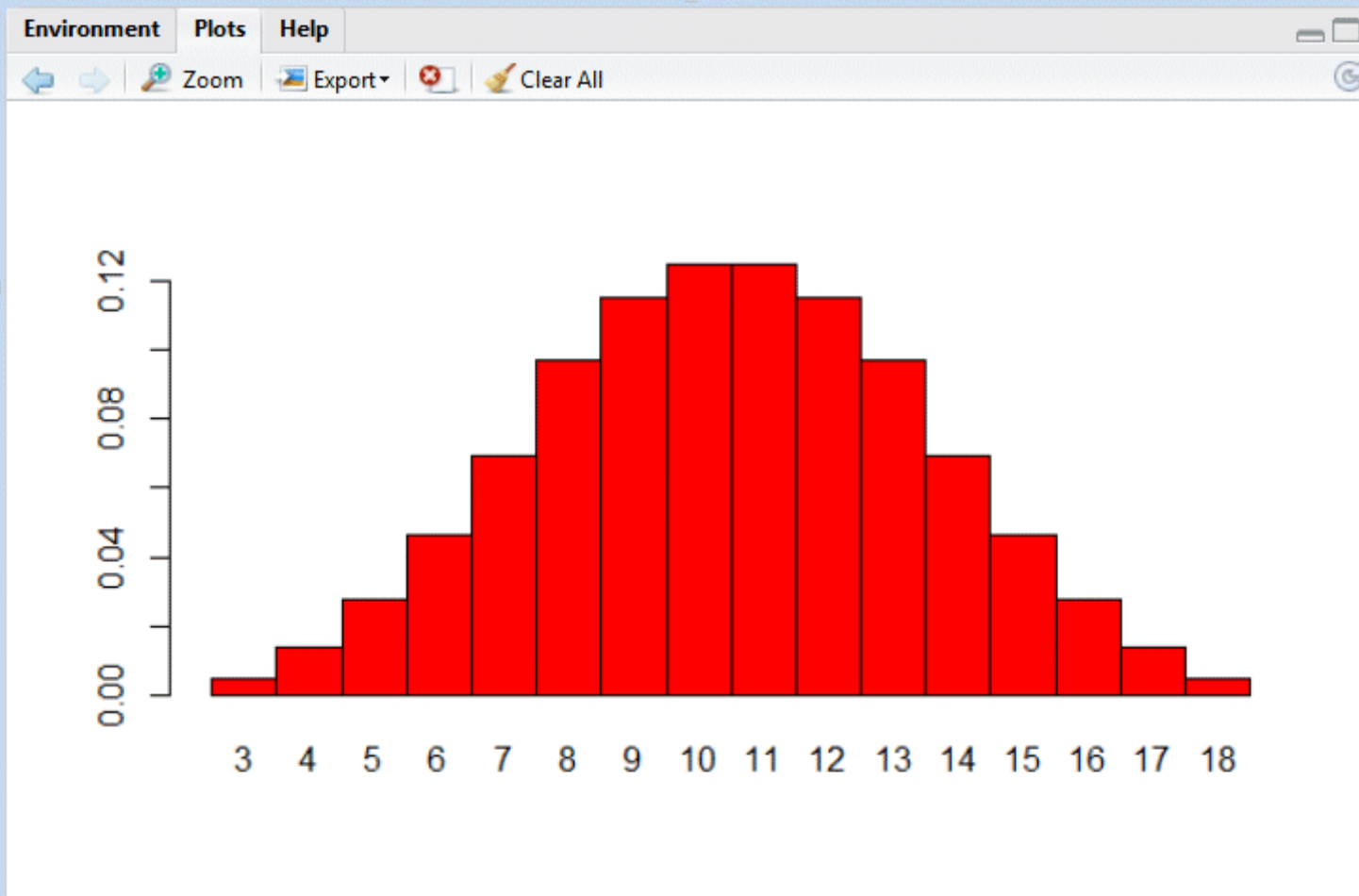
df=data.frame(1:6)
if (ndadi>1) for (i in 2:ndadi) df[[i]]=1:6

ennuple_possibili=expand.grid(df)
somme=rowSums(ennuple_possibili)

distr_prob=table(somme)/length(somme)
print(round(distr_prob,3))
barplot(distr_prob,space=0,col="red")
```

Notare la presenza del comando *if* (se): il comando *for* che aggiunge colonne al data-frame *df* viene eseguito solo se $ndadi > 1$. E questo è l'output del nostro programma:

```
Console ~/R/ ↵
somme
  3    4    5    6    7    8    9   10   11   12
0.005 0.014 0.028 0.046 0.069 0.097 0.116 0.125 0.125 0.116
 13   14   15   16   17   18
0.097 0.069 0.046 0.028 0.014 0.005
>
```



Qui la distribuzione di S nel caso di 4 dadi:

