Data Analyst: modulo Intelligenza Artificiale

Unità IA.2

Distribuzioni di probabilità discrete, distribuzioni notevoli

Docente: Davide Passaro

Ripasso probabilità

Le tre definizioni di probabilità: classica, frequentista e soggettiva

La probabilità può essere interpretata in modi diversi a seconda del contesto, dei dati disponibili e dell'obiettivo dell'analisi.

Le tre principali definizioni sono:

- **Probabilità classica**
- **Probabilità frequentista**
- **Probabilità soggettiva (o bayesiana)**

---

1. Probabilità Classica

La definizione classica, introdotta da Laplace, si applica quando tutti gli esiti sono **ugualmente probabili**.

Definizione:

$$P(A) = \frac{\text{numero di casi favorevoli}}{\text{numero di casi possibili}}$$

Esempio: Probabilità di ottenere un 3 lanciando un dado:

$$P(3) = \frac{1}{6}$$

Quando è applicabile:

- giochi di fortuna (dadi, monete, carte)
- modelli con simmetria perfetta negli esiti

Limiti:

- richiede esiti equiprobabili (condizione spesso irrealistica)

---

2. Probabilità Frequentista

È la definizione utilizzata nella statistica classica. Si basa sulla frequenza relativa osservata dopo molte ripetizioni di un esperimento.

Definizione:

$$P(A) = \lim_{n \to \infty} \frac{\text{numero di volte in cui A si verifica}}{n}$$

Esempio: Se un farmaco produce risposta in 72 pazienti su 100 osservati:

$$P(\text{risposta}) \approx 0.72$$

Quando è applicabile:

- esperimenti ripetibili (studi clinici, produzione industriale)
- grandi quantità di dati

Limiti:

- richiede molte osservazioni
- non gestisce eventi singoli (es. probabilità che un *singolo* paziente risponda)
- non utilizza informazioni precedenti o conoscenza esperta

---

3. Probabilità Soggettiva (Bayesiana)

Interpreta la probabilità come un **grado di credenza razionale**, basato sulle informazioni disponibili. È il fondamento dell'approccio **bayesiano**.

Parentesi "storica": un importante esponente di questo approccio è stato il matematico Bruno de Finetti.

Concetto cardine:

La probabilità è vista come il grado di fiducia soggettivo di un individuo nell'avverarsi di un evento, basato sulle informazioni disponibili.

La probabilità rappresenta la nostra incertezza e può essere **aggiornata** quando arrivano nuovi dati tramite il teorema di Bayes.

Esempio: Un medico stima una probabilità del 40% che un paziente sviluppi una complicanza. Arrivano nuovi esami → la probabilità viene aggiornata.

Limiti:

- dipende dalle infiormazioni iniziali (gli *a priori*)
- può essere percepita come meno oggettiva

In [ ]:
```
# Esercizio di ripasso

# caso lancio di un dado probabilità di ottenere 6
lista_valori= [1,2,3,4,5,6]
eventi_favorevoli= [6]
p= len(eventi_favorevoli)/len(lista_valori)
print(p)


# oppure
lista_valori= [1,2,3,4,5,6]
esito_favorevole= 6
eventi_favorevoli= lista_valori.count(esito_favorevole) # conta quante volte è presente il 6
eventi_totali= len(lista_valori)
p= eventi_favorevoli/eventi_totali
print(f"il valore della probabilità è:  ",p)
print(f"il valore della probabilità è {p}")  # uso formato f-string
print(f"il valore della probabilità è {p:.5f}") # modo per approssimare stampa
```

```
0.16666666666666666
il valore della probabilità è:   0.16666666666666666
il valore della probabilità è 0.16666666666666666
il valore della probabilità è 0.16667
```

Esempio lancio di due monete

Il numero di teste X è una v.a. discreta | x | 0 | 1 | 2 | Totale | |:-----:|:---:|:---:|:---:|:---:|:------:| | p(x) | 1/4 | 1/2 | 1/4 | 1 |

Notazione: $p(x) = P(X = x)$ = P(numero di teste = x)

```
Primo lancio
├─ T
│  ├─ T  (TT)
│  └─ C  (TC)
└─ C
   ├─ T  (CT)
   └─ C  (CC)
```

N.B.

In alcuni casi è utile rappresentare il diagramma ad albero delle possibilità e contare gli eventi finali ottenuti.

Per esercizi prova a rappresentare il diagramma ad albero del lancio di tre monete (eventi Testa o Croce)

Verifica che gli esiti ottenuti sono:

TTT

TTC

TCT

TCC

CTT

CTC

CCT

CCC

In [ ]:
```
import random

# Possibili risultati
moneta = ['Testa', 'Croce']

# Simula un lancio
lancio = random.choice(moneta)
print(lancio)
```

```
Croce
```

In [ ]:
```
import random # modulo per generare numeri casuali

lancio = random.randint(0, 1)  # 0 = Croce, 1 = Testa
if lancio == 1:
    print("Testa")
else:
    print("Croce")
```

```
Testa
```

E per simulare più lanci?

In [ ]:
```
import random

moneta = ['Testa', 'Croce']

lanci = [random.choice(moneta) for _ in range(10)]  # 10 lanci
print(lanci)
print(lanci.count('Testa'))  # conta il numero di teste
```

```
['Testa', 'Croce', 'Testa', 'Croce', 'Croce', 'Testa', 'Testa', 'Testa', 'Testa', 'Croce']
6
```

Introduzione/ripasso fattoriale

Il simbolo del fattoriale

Il **fattoriale** di un numero intero non negativo (n) è definito come il prodotto di tutti i numeri interi positivi da (1) a (n):

$$n! = n \cdot (n-1) \cdot (n-2) \cdots 2 \cdot 1$$

Esempi:

$$4! = 4 \cdot 3 \cdot 2 \cdot 1 = 24$$
$$1! = 1$$

Per convenzione:

$$0! = 1$$

Il fattoriale cresce molto velocemente ed è fondamentale nelle formule combinatorie, nelle permutazioni, nelle disposizioni e nelle combinazioni.

Cenni al calcolo combinatorio

Il calcolo combinatorio è una branca della matematica discreta che si occupa di contare in quanti modi diversi è possibile raggruppare, ordinare o scegliere gli elementi di un insieme finito, sotto determinate condizioni.

Lo scopo principale è quello di determinare il numero di configurazioni possibili senza doverle elencare tutte esplicitamente.

Il calcolo combinatorio studia i modi in cui è possibile selezionare e disporre elementi. Le idee principali sono: permutazioni, disposizioni e combinazioni.

Principio del Prodotto (Regola della Moltiplicazione)

Definizione

Se un processo decisionale è composto da una sequenza di scelte consecutive e indipendenti, il numero totale di modi in cui l'intero processo può essere completato è dato dal prodotto del numero di opzioni disponibili per ciascuna singola scelta.

Esempio:

Se devo scegliere fra andare in vacanza in treno o in auto, fra le possibili mete mare, montagna o città d'arte e fra i mesi luglio, agosto o settembre avremo:
$2 \cdot 3 \cdot 3 = 18$ possibili scelte

Permutazioni

Le **permutazioni** sono i modi possibili di ordinare tutti gli elementi di un insieme.

In una permutazione:

- si usano tutti gli elementi
- l'ordine conta

La formula è:

$$P_n = n!$$

Esempio:

i modi per ordinare 5 libri su uno scaffale sono:

$$5! = 120$$

Disposizioni

Le **disposizioni** considerano un sottoinsieme di k elementi scelti tra n, in cui l'ordine è importante.

La formula delle disposizioni semplici è:

$$D_{n,k} = \frac{n!}{(n-k)!}$$

Esempio:

i modi per scegliere primo, secondo e terzo posto fra 10 atleti sono:

$$D_{10,3} = \frac{10!}{7!} = 720$$

Ragionamento intuitivo: Il primo atleta lo posso scegliere fra 10, il secondo fra i 9 rimanenti, il terzo fra gli otto rimanenti.

Quindi si hanno: $10 \cdot 9 \cdot 8 = 720$ possibilità.

N.B Conta l'ordine!

Combinazioni

Le **combinazioni** scelgono un sottoinsieme di k elementi da n, senza considerare l'ordine.

La formula è:

$$C_{n,k} = \binom{n}{k} = \frac{n!}{k!(n-k)!}$$

Esempio: i gruppi di 3 amici che si possono formare da 10 persone sono:

$$\binom{10}{3} = 120$$

N.B In questo caso NON conta l'ordine!

Esercizio 1

Calcola i seguenti fattoriali:

1. 5!
2. 7! - 6!
3. Spiega perché per convenzione 0! = 1

Soluzione:

1. 5! = 120
2. $7! = 5040,\ 6! = 720 \Rightarrow 5040 - 720 = 4320$. Ma volendo anche $7! - 6! = 7 \cdot 6! - 6! = 6!(7-1) = 6 \cdot 6! = 4320$.

3. Osservazione importante: la convenzione 0! = 1 garantisce la validità della formula delle combinazioni:

$$\binom{n}{0} = \frac{n!}{0! \cdot n!} = 1$$

C'è, infatti, una sola combinazione che prende 0 elementi da un gruppo di n: l'insieme vuoto.

Esercizio 2

Quanti sono gli anagrammi anche senza significato della parola **LUNA**?

Soluzione:

$4! = 24$

Esercizio 3

Da 12 atleti, quanti modi ci sono per assegnare primo, secondo e terzo posto?

Soluzione:

$$D_{12,3} = \frac{12!}{9!} = 12 \cdot 11 \cdot 10 = 1320$$

Esercizio 4

Quanti gruppi di 4 studenti si possono formare da una classe di 20?

Soluzione:

$$\binom{20}{4} = \frac{20!}{4!, 16!} = 4845$$

Esercizio 5 (approfondimento non svolto a lezione)

Per calcolare il numero di anagrammi della parola **MAMMA**, che contiene lettere ripetute, devi usare la formula per le **permutazioni con ripetizione**.

La parola MAMMA è composta da $n = 5$ lettere. Le lettere che si ripetono sono:

- **M** si ripete $k_1 = 3$ volte
- **A** si ripete $k_2 = 2$ volte

Formula per le Permutazioni con Ripetizione

La formula generale è:

$$P'(n; k_1, k_2, \ldots, k_m) = \frac{n!}{k_1! k_2! \cdots k_m!}$$

Dove:

- $n!$ è il fattoriale del numero totale di lettere.
- $k_i!$ è il fattoriale del numero di volte che ciascuna lettera si ripete.

Calcolo per "MAMMA"

Sostituendo i valori:

1. **Totale lettere** (n): 5
2. **Ripetizioni della M** (k_1): 3
3. **Ripetizioni della A** (k_2): 2

$$P' = \frac{5!}{3! \cdot 2!}$$

Ci sono **10** anagrammi distinti della parola MAMMA.

Librerie utili in Python

Esistono molti moduli (librerie) in Python che sono utili per il calcolo della probabilità

Uno degli aspetti più utili (approfondiremo in seguito) è quello di poter simulare fenomeni casuali.

Ad esempio è possibile verificare secondo l'approccio frequentista il valore di una probabilità calcolata secondo l'approccio classico

Il modulo `random` di Python

Il modulo `random` di Python fornisce funzioni per generare numeri casuali e simulare fenomeni aleatori. Non crea vera casualità fisica: utilizza un **generatore pseudocasuale**, cioè produce numeri che *sembrano* casuali ma derivano da algoritmi deterministici.

Con `random` è possibile:

- generare numeri casuali in un intervallo, ad esempio:

```
random.random()     # numero tra 0 e 1
random.randint(1,6) # intero tra 1 e 6
```

- estrarre elementi da una lista:

```
random.choice(lista)
```

- mescolare una sequenza:

```
random.shuffle(lista)
```

- estrarre campioni casuali:

```
random.sample(lista, k=3)
```

Altre possibili librerie (che vedremo in seguito)

SciPy

SciPy è la libreria di riferimento per la computazione scientifica. Il sottomodulo scipy.stats è particolarmente potente per la probabilità e la statistica. Offre funzionalità alcuni degli argomenti che vedremo in questa e nelle prossime lezioni:

Distribuzioni di Probabilità: Implementazioni complete di tutte le principali distribuzioni (Normale, Binomiale, Poisson, Esponenziale, ecc.), che permettono di:

Calcolare la PMF (Probability Mass Function) per le distribuzioni discrete o la PDF (Probability Density Function) per le distribuzioni continue (vedi più avanti in questo testo per la definizione di PDF)

```
# Esempio uso nel caso della binomiale (vedi dopo)
from scipy.stats import binom

n = 10   # numero di prove
p = 0.5  # probabilità di successo

# P(X = 3)
print("P(X=3):", binom.pmf(3, n, p))
```

Statsmodels

Statsmodels si concentra sull'inferenza statistica e sui modelli, offrendo classi e funzioni per la stima di molti modelli statistici, test statistici e l'esplorazione dei dati statistici.

In [ ]:
```
# Esercizio: simula il lancio di due dadi D6
# restituisci in output una lista con n=100 risultati della somma somma dei
# due dadi


# parte2:
# quale risultato ti aspetti sia più frequente?
# verifica l'ipotesi (fai contare a python :-) )

import random

numero_lanci= 100000
risultati = []

for i in range(numero_lanci):
  dado1= random.randint(1,6)
  dado2= random.randint(1,6)
  risultato= dado1+dado2
  risultati.append(risultato)

#print(risultati)

frequenza_7 = risultati.count(7)
print("Simulazione", frequenza_7/numero_lanci)
print("Risultato probabilità classica", 6/36)
```

```
Simulazione 0.16575
Risultato probabilità classica 0.16666666666666666
```

In [ ]:
```
# Generazione numeri "pseudo-casuali"
#(approfondiremo successivamente idea generare numeri pseudocasuali)

import random

a= random.random()     # numero tra 0 e 1
b=  random.randint(1,6) # intero tra 1 e 6
print(a)
print(b)
```

```
0.42335369271462453
5
```

In [ ]:
```
# creo una lista di valori vuola e ci appendo n valori di random.int

lista_dadi= []
for i in range(10):
  lista_dadi.append(random.randint(1,6))
print(lista_dadi)
print(lista_dadi.count(2))
```

```
[3, 1, 3, 4, 1, 1, 1, 6, 3, 3]
0
```

Variabili casuali

Una variabile aleatoria (o casuale) è definita da un insieme di modalità cui sono associate delle probabilità

Le variabili aleatorie possono essere

- discrete
- continue

Esempio di variabile continua:

- altezza di una persona
- velocità di un corpo

Esempio di variabile discreta:

- risultato del lancio di un dado o di una moneta
- conteggio di un evento

Distribuzioni Discrete e Continue

Ad ogni variabile casuale X è sempre associata una distribuzione di probabilità.

La distribuzione di probabilità è l'elemento che definisce la variabile casuale, descrivendo il modo in cui la probabilità è ripartita tra i possibili valori che la variabile può assumere.

In altri termini, in statistica e probabilità, una **distribuzione** descrive come sono assegnate le probabilità ai valori di una variabile casuale.

Le distribuzioni (come la variabili) si dividono principalmente in due categorie:

- **discrete**
- **continue**

---

1. Distribuzioni Discrete

Una **variabile casuale discreta** assume un numero **finito** o **numerabile** di valori. Esempi:

- numero di lanci di moneta che risultano testa
- numero di pazienti con una certa patologia
- numero di chiamate al pronto soccorso in un'ora

Caratteristiche principali

- I valori possibili sono separati e distinti.
- La probabilità si assegna tramite una **funzione di massa di probabilità (PMF, Probability Mass Function)**.
- La probabilità di ogni singolo punto è definita e può essere diversa da zero.

Esempi di distribuzioni discrete

- **Distribuzione di Bernoulli** (successo/fallimento)
- **Distribuzione Binomiale** (numero di successi su n prove)
- **Distribuzione Poisson** (numero di eventi in un intervallo)

Proprietà fondamentali

- Somma delle probabilità = **1**
- $P(X = x)$ è definito per ogni valore x

---

2. Distribuzioni Continue

Una **variabile casuale continua** assume **infiniti** valori in un intervallo. Esempi:

- pressione sanguigna
- tempo di reazione
- peso corporeo
- altezza

Caratteristiche principali

- I valori possibili formano un **continuo** (infinite possibilità).
- La probabilità si descrive tramite una **funzione di densità di probabilità (PDF, Probability Density Function)**.
- La probabilità che la variabile assuma un **valore esatto** è **zero**:

$$P(X = x) = 0$$

Invece si calcola la probabilità su un **intervallo**:

$$P(a < X < b) = \int_a^b f(x)dx$$

N.B. vedremo successivamente le distribuzioni continue. Se non hai familiarità con gli integrali ($\int$) non ti preoccupare. A questo livello li puoi considerare in modo intuitutivo come una sommatoria su intervalli "piccolissimi".

Esempi di distribuzioni continue

- **Distribuzione Normale (Gauss)**
- **Distribuzione Uniforme continua**
- **Distribuzione Esponenziale**

Proprietà

- La **Proprietà fondamentale**

$$\int_{-\infty}^{\infty} f(x)dx = 1$$

---

3. Differenza tra PMF e PDF

Aspetto	Discreta (PMF)	Continua (PDF)
Simbolo	P(X = x)	f(x)
Probabilità singolo valore	può essere > 0	sempre = 0
Somma o integrale	$\sum P(X = x) = 1$	$\int f(x)dx = 1$
Calcolo probabilità	punti	intervalli

---

4. Quando una variabile è discreta o continua?

Discreta:

quando conta gli oggetti (numero di ricoveri, numero di successi).

Continua:

quando misura quantità fisiche (tempo, pressione, biomarcatori).

Attenzione alla terminologia

La **massa di probabilità** (in inglese *Probability Mass Function*, PMF) è un concetto usato nelle **distribuzioni di probabilità discrete**.

La **massa di probabilità** associa **ad ogni valore possibile** di una variabile casuale discreta la **probabilità che quella variabile assuma quel valore**.

Se X è una variabile casuale discreta, la massa di probabilità $P(X = x_i)$ soddisfa:

1. $0 \leq P(X = x_i) \leq 1$ per ogni x_i possibile
2. $\sum_i P(X = x_i) = 1$

Esempio semplice: lancio di una moneta

Se definiamo:

$$X = \text{numero di teste in un lancio di moneta}$$

Allora la PMF è:

x	0	1
P(X=x)	0.5	0.5

Qui:

- $P(X = 0) = 0.5$ → probabilità di ottenere croce
- $P(X = 1) = 0.5$ → probabilità di ottenere testa

La somma delle probabilità: $0.5 + 0.5 = 1$

Proprietà della massa di probabilità

1. **Discreta:** si applica solo a variabili casuali che assumono valori separati (ad esempio 0,1,2… successi).
2. **Non negativa:** tutte le probabilità sono ≥0.

3. **Somma = 1:** la probabilità totale di tutti i possibili valori è 1.
4. **Funzione:** possiamo pensare alla PMF come a una **funzione** che a ciascun valore x associa P(X=x).

Nota

La PMF è diversa dalla **densità di probabilità** (PDF) che si usa per variabili casuali continue.

- Con variabili continue, la probabilità di un singolo valore è **0** e si considera l'area sotto la curva.
- Con variabili discrete, la probabilità di ciascun valore è **non nulla** ed è direttamente la PMF.

Caratteristiche fondamentali di una distribuzione di probabilità

Una **distribuzione di probabilità** descrive come i valori di una variabile casuale sono distribuiti. Per caratterizzarla si usano diversi indicatori numerici detti **momenti** o **misure di posizione e dispersione**.

Media (o valore atteso)

Indica il “centro” della distribuzione, ovvero il valore verso cui tendono le osservazioni.

Distribuzione discreta

$$\mathbb{E}[X] = \sum_i x_i \cdot p(x_i)$$

Distribuzione continua (vedremo nelle prossime lezioni)

$$\mathbb{E}[X] = \int_{-\infty}^{+\infty} x f(x) dx$$

Valore atteso del lancio di una moneta

Consideriamo una variabile aleatoria X che assume i valori:

- 1 se esce **Testa**
- 0 se esce **Croce**

Per una moneta equa:

$P(X = 1) = \frac{1}{2}, \qquad P(X = 0) = \frac{1}{2}$

Calcolo del valore atteso

Per definizione:

$\mathbb{E}[X] = \sum_{k=0}^{1} k \cdot P(X = k)$

Sostituendo i valori possibili:

$\mathbb{E}[X] = 0 \cdot \frac{1}{2} + 1 \cdot \frac{1}{2}$

$\mathbb{E}[X] = \frac{1}{2}$

Esempio: valore atteso del lancio di un dado

Consideriamo una variabile aleatoria X che rappresenta il risultato del lancio di un dado equilibrato a 6 facce. I possibili valori sono: $1, 2, 3, 4, 5, 6$ e ciascuno ha probabilità:

$P(X = k) = \frac{1}{6}$

Calcolo del valore atteso

Per definizione:

$\mathbb{E}[X] = \sum_{k=1}^{6} k \cdot P(X = k)$

Sostituendo:

$\mathbb{E}[X] = 1 \cdot \frac{1}{6} + 2 \cdot \frac{1}{6} + 3 \cdot \frac{1}{6} + 4 \cdot \frac{1}{6} + 5 \cdot \frac{1}{6} + 6 \cdot \frac{1}{6}$

$\mathbb{E}[X] = \frac{1+2+3+4+5+6}{6}$

$\mathbb{E}[X] = \frac{21}{6} = 3.5$

---

2. Varianza

Misura quanto la distribuzione è “sparsa”, cioè quanto i valori si discostano dalla media.

$$\mathrm{Var}(X) = \mathbb{E}[(X - \mathbb{E}[X])^2]$$

In forma esplicita

Distribuzione discreta:

Se chiamiamo μ la stima del valore atteso $\mathbb{E}[X]$

$$\mathrm{Var}(X) = \sum_i (x_i - \mu)^2 \cdot p(x_i)$$

Distribuzione continua:

$$\mathrm{Var}(X) = \int_{-\infty}^{+\infty} (x - \mu)^2 \cdot f(x) dx$$

---

3. Deviazione standard

È la radice quadrata della varianza:

$$\sigma = \sqrt{\mathrm{Var}(X)}$$

N.B σ è usata perché ha la **stessa unità di misura della variabile**, a differenza della varianza.

---

4. Moda

È il valore più probabile della distribuzione.

- **Distribuzione discreta:** il valore con probabilità massima.
- **Distribuzione continua:** il punto in cui la densità (f(x)) è massima.

---

5. Mediana

È il punto che divide la distribuzione in due parti uguali:

$P(X \leq \text{mediana}) = 0.5$

È **robusta** rispetto a valori estremi o outlier.

Breve approfondimento su Outlier

Un outlier è un dato molto distante dalla maggior parte degli altri, tanto da apparire “strano” o “sospetto”. Può essere:

- un errore di misura o di inserimento,
- un valore corretto ma raro,
- un dato che rivela un fenomeno importante (es. un paziente con valori clinici molto alterati).

Perché è importante riconoscere gli outlier

Gli outlier alterano la media. La media, infatti, che è molto sensibile ai valori estremi. Gli outlier rendono più grande anche la varianza e la deviazione standard

Problema outlier: influenzeno negativamente modelli statistici e machine learning, fanno apparire erroneamente correlazioni che non esistono

---

Ulteriore approfondimento (facoltativo)

Asimmetria (Skewness)

Misura quanto la distribuzione è asimmetrica.

- Skewness = 0 → distribuzione simmetrica
- Skewness > 0 → coda destra più lunga
- Skewness < 0 → coda sinistra più lunga

Distribuzione di Bernoulli e Binomiale

Distribuzione di Bernoulli

La **distribuzione di Bernoulli** descrive un esperimento con **due soli esiti possibili**: successo 1 o insuccesso 0.

- Probabilità di successo: p
- Probabilità di insuccesso: q = 1 - p

La funzione di massa di probabilità (PMF) è:

$$P(X = x) = \begin{cases} p & \text{se } x = 1 \\ 1 - p & \text{se } x = 0 \end{cases}$$

Valore atteso o Media:

$$\mu = E[X] = p$$

Varianza:

$$\sigma^2 = \mathrm{Var}(X) = p(1 - p)$$

Esempio in Python:

```
p = 0.3  # probabilità di successo
mu = p
var = p * (1 - p)
print("Media:", mu)
print("Varianza:", var)
```

Distribuzione Binomiale

La **distribuzione binomiale** generalizza la Bernoulli a **n prove indipendenti** identiche, ciascuna con probabilità di successo p.

- Numero di prove: n
- Probabilità di successo: p

- Numero di successi: X

La funzione di massa di probabilità (PMF) della distribuzione binomiale calcola la probabilità esatta di ottenere esattamente k successi in n prove.

La funzione di massa di probabilità (PMF) è:

$$P(X = k) = \binom{n}{k} p^k (1-p)^{n-k}, \quad k = 0, 1, \ldots, n$$

Dove:

- k è il numero di successi desiderati ($k = 0, 1, \ldots, n$)
- p^k è la probabilità di ottenere k successi
- $(1-p)^{n-k}$ è la probabilità di ottenere $n-k$ insuccessi (A volte è usata anche la lettera $q = 1-p$).
- $\binom{n}{k}$ è il coefficiente binomiale che indica il numero di modi diversi in cui si possono ottenere k successi in n prove:

$$\binom{n}{k} = \frac{n!}{k!(n-k)!}$$

Media (valore atteso):

$$\mu = E[X] = np$$

Varianza:

$$\sigma^2 = \mathrm{Var}(X) = np(1-p)$$

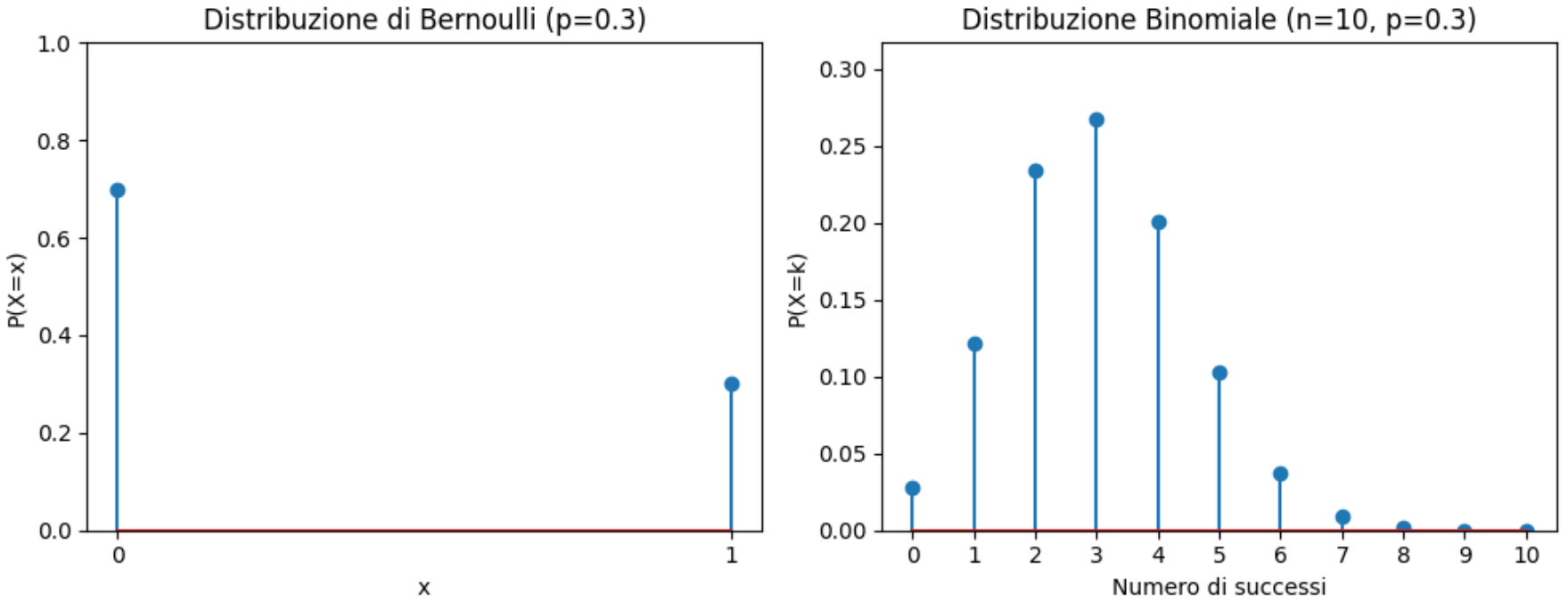
Esempio in Python:

```
n = 10   # numero di prove
p = 0.3  # probabilità di successo

mu = n * p
var = n * p * (1 - p)
print("Media:", mu)
print("Varianza:", var)
```

In [ ]:
```
# Esercizio in aula: scrivi la funzione binomiale che prende in input
# k, n,p
# confronta il risultato con la liberia scipy.stats.binom.pmf()

def binomiale():
  """ Completa la funzione """



# stampa risultato ottenuto e confrontalo con quello ottenuto
# dalla libreria scipy.stats.binom
```

In [1]:
```
import matplotlib.pyplot as plt
from scipy.stats import bernoulli, binom
import numpy as np

# -------------------------
# Distribuzione di Bernoulli
# -------------------------
p = 0.3 # probabilità di successo

x_bern = [0, 1]
y_bern = bernoulli.pmf(x_bern, p)

plt.figure(figsize=(10,4))

plt.subplot(1, 2, 1)
plt.stem(x_bern, y_bern)
plt.xticks([0,1])
plt.xlabel('x')
plt.ylabel('P(X=x)')
plt.title('Distribuzione di Bernoulli (p=0.3)')
plt.ylim(0,1)

# -------------------------
# Distribuzione Binomiale
# -------------------------
n = 10  # numero di prove

x_binom = np.arange(0, n+1)
y_binom = binom.pmf(x_binom, n, p)

plt.subplot(1, 2, 2) # qui faccio subplot ovvero due sotto grafici nello stesso grafico
# plt.stem() viene utilizzata per creare grafici a stelo (o stem plot)
plt.stem(x_binom, y_binom)
plt.xticks(range(0,n+1))
plt.xlabel('Numero di successi')
plt.ylabel('P(X=k)')
plt.title(f'Distribuzione Binomiale (n={n}, p={p})')
plt.ylim(0, max(y_binom)+0.05)
```

```
plt.tight_layout()
plt.show()
```



Variazione al codice di prima

Qui di seguito inseriamo lo stesso codice ma mettiamo la probabilità p=0.5

Ragionamenti da fare: Come mi aspetto che cambino i grafici?

E' sempre utile riflettere sui risultati che si ottengono.

Ecco il testo riscritto senza icone, pronto per essere incollato in un markdown su Colab:

Che cos'è `plt.stem()`?

`plt.stem` è una funzione di Matplotlib usata per creare grafici a *stems* (grafici a stelo), molto comuni per visualizzare distribuzioni discrete o serie di impulsi.

Mostra:

- un punto (marker) per ogni valore discreto
- una linea verticale (lo "stelo") che parte dall'asse e arriva al punto

È perfetto per rappresentare:

- PMF (probability mass function)
- valori discreti in generale
- sequenze numeriche

Esempio base

```
import matplotlib.pyplot as plt

x = [1, 2, 3, 4, 5, 6]
y = [1/6] * 6

plt.stem(x, y)
plt.xlabel("Valore del dado")
plt.ylabel("Probabilità")
plt.title("PMF del lancio di un dado")
plt.show()
```

N.B. Rimuovere la linea di base (finezza estetica)

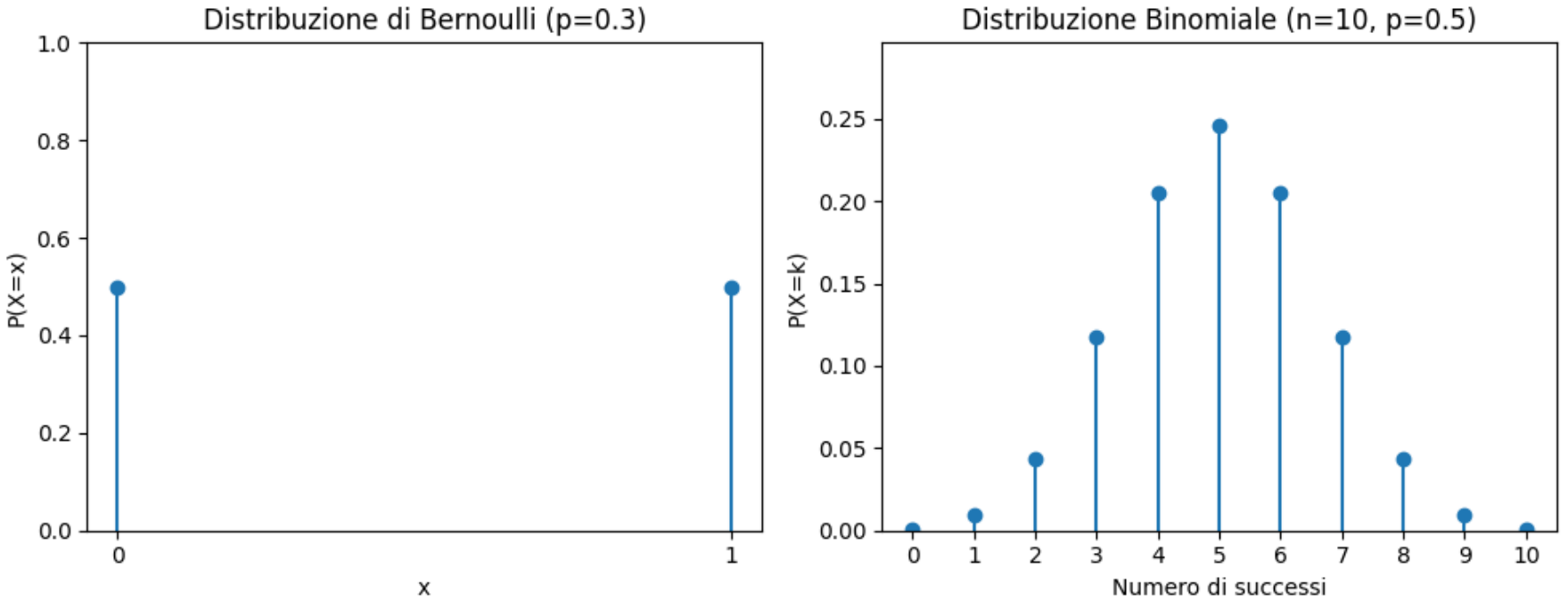
```
plt.stem(x, y, basefmt=" ")
```

Esempio con distribuzione binomiale:

```
import numpy as np
from scipy.stats import binom
import matplotlib.pyplot as plt

n, p = 10, 0.4
x = np.arange(0, n+1)
y = binom.pmf(x, n, p)

plt.stem(x, y)
plt.title("Distribuzione Binomiale (n=10, p=0.4)")
plt.show()
```

In [4]:
```
## Esempio 2: stesso codice di prima ma probabilità p=0.5
## Ragionamenti da fare: Come mi aspetto che cambino i grafici?
## E' sempre utile riflettere sui risultati che si ottengono

import matplotlib.pyplot as plt
from scipy.stats import bernoulli, binom
import numpy as np

# -------------------------
# Distribuzione di Bernoulli
# -------------------------
p = 0.5 # probabilità di successo

x_bern = [0, 1]
y_bern = bernoulli.pmf(x_bern, p)

plt.figure(figsize=(10,4))

plt.subplot(1, 2, 1)
plt.stem(x_bern, y_bern,basefmt=" ")
plt.xticks([0,1])
plt.xlabel('x')
plt.ylabel('P(X=x)')
plt.title('Distribuzione di Bernoulli (p=0.3)')
plt.ylim(0,1)

# -------------------------
# Distribuzione Binomiale
# -------------------------
n = 10  # numero di prove

x_binom = np.arange(0, n+1)
y_binom = binom.pmf(x_binom, n, p)

plt.subplot(1, 2, 2) # qui faccio subplot ovvero due sotto grafici nello stesso grafico
# plt.stem() viene utilizzata per creare grafici a stelo (o stem plot)
plt.stem(x_binom, y_binom,basefmt=" ")
plt.xticks(range(0,n+1))
plt.xlabel('Numero di successi')
```

```
plt.ylabel('P(X=k)')
plt.title(f'Distribuzione Binomiale (n={n}, p={p})')
plt.ylim(0, max(y_binom)+0.05)

plt.tight_layout()
plt.show()
```



Ragionamento sul grafico fatto in classe

I nuovi grafici ottenuti erano quello che mi aspettavo? Perché ?

Distribuzione di Poisson

- **Definizione:** Modella il numero di eventi che si verificano in un intervallo di tempo o spazio fissato, quando gli eventi sono rari e indipendenti.
- **Parametro:** λ = tasso medio di occorrenza

In altri termini: x = numero di eventi in un determinato intervallo

λ = il numero medio di eventi per intervallo

- **PMF:**

$$P(X = k) = \frac{\lambda^k e^{-\lambda}}{k!}, \quad k = 0, 1, 2, \ldots$$

dove:

- $P(X = k)$ è la probabilità che il numero di eventi sia esattamente x.
- k è il numero di eventi (un intero non negativo: 0,1,2,3,…).
- λ (lambda) è il parametro della distribuzione e rappresenta il numero medio di eventi nell'intervallo di tempo/spazio specificato.
- e è il numero di Nepero (e≈2.71828).
- $k!$ è il fattoriale di k.

Si dimostra, inoltre, che la distribuzione di Poisson ha:

- **Media:** $E[X] = \lambda$
- **Varianza:** $\mathrm{Var}(X) = \lambda$

Esercizio

Un call center riceve in media 5 chiamate ogni ora. Sia X il numero di chiamate ricevute in un'ora.

1. Qual è la probabilità di ricevere **esattamente 3 chiamate** in una ora?

2. Qual è la probabilità di ricevere **almeno 4 chiamate** in una ora?
3. Qual è il **numero medio di chiamate** e la **varianza**?

Soluzione in Python

```
from scipy.stats import poisson

# Parametro lambda (numero medio di chiamate)
lambda_call = 5

# 1. Probabilità di esattamente 3 chiamate
prob_3 = poisson.pmf(3, lambda_call)
print("Probabilità di ricevere esattamente 3 chiamate:", prob_3)

# 2. Probabilità di almeno 4 chiamate
prob_ge_4 = 1 - poisson.cdf(3, lambda_call)  # P(X >= 4) = 1 - P(X <= 3)
print("Probabilità di ricevere almeno 4 chiamate:", prob_ge_4)

# 3. Media e varianza
mean = lambda_call
variance = lambda_call
print("Media:", mean)
print("Varianza:", variance)
```

Spiegazione

1. **`poisson.pmf(k, lambda)`** calcola ($P(X = k)$).
2. **`poisson.cdf(k, lambda)`** calcola ($P(X \leq k)$).
3. Per la distribuzione di Poisson, **media e varianza coincidono** e sono entrambe (\lambda).

N.B.

- *pmf* sta per probability mass function
- *cdf* sta per Cumulative Distrivution Function (vedi dopo)

Vedi di seguito il codice di esempio scritto in più modi

Cenni: Cumulative Distribution Function (CDF)

La **cumulative distribution function (CDF)**, in italiano **funzione di distribuzione cumulativa**, è uno dei concetti fondamentali in probabilità.

La **CDF** di una variabile aleatoria X è una funzione che associa a ogni numero reale x la probabilità che X assuma un valore **minore o uguale** a x.

La definiamo come: $F_X(x) = P(X \leq x)$

Alcune caratteristiche dalla CDF:

1. È sempre **compresa tra 0 e 1**: $0 \leq F_X(x) \leq 1$
2. È **non decrescente**: $x_1 < x_2 \Rightarrow F_X(x_1) \leq F_X(x_2)$

Esempio per capire nel caso di variabili discrete:

la CDF si ottiene sommando le probabilità di tutti gli esiti fino al valore considerato, inclusi quelli uguali ad esso. Facciamo l'esempio di un dado equo a sei facce, la CDF in $x = 4$ non è altro che la somma delle probabilità di ottenere 1, 2, 3 o 4, cioè $4/6$.

In breve

La CDF è:

- una funzione che rappresenta **la probabilità che X sia ≤ x**
- cresce da 0 a 1

In [6]:
```
# esempio 1
from scipy.stats import poisson

# Parametro lambda (numero medio di chiamate)
lambda_call = 5

#  Probabilità di esattamente 3 chiamate
prob_3 = poisson.pmf(3, lambda_call)
print("Probabilità di ricevere esattamente 3 chiamate:", prob_3)

# Probabilità di almeno 4 chiamate
prob_ge_4 = 1 - poisson.cdf(3, lambda_call)  # P(X >= 4) = 1 - P(X <= 3)
print("Probabilità di ricevere almeno 4 chiamate:", prob_ge_4)

#  Media e varianza
mean = lambda_call
variance = lambda_call
print("Media:", mean)
print("Varianza:", variance)
```

```
Probabilità di ricevere esattamente 3 chiamate: 0.1403738958142805
Probabilità di ricevere almeno 4 chiamate: 0.7349740847026385
Media: 5
Varianza: 5
```

In [5]:
```
# esempio 2
import math

# Parametri
lambda_call = 5
k = 3

# Calcolo della probabilità usando la formula di Poisson
prob = (lambda_call**k * math.exp(-lambda_call)) / math.factorial(k)

print(f"Probabilità di ricevere esattamente 3 chiamate: {prob:.9f}")


prob2 = 1-  ((lambda_call**3 * math.exp(-lambda_call)) / math.factorial(3)+ \
 (lambda_call**2 * math.exp(-lambda_call)) / math.factorial(2)+\
  (lambda_call**1 * math.exp(-lambda_call)) / math.factorial(1)+\
   (lambda_call**0 * math.exp(-lambda_call)) / math.factorial(0))

print(f"Probabilità di ricevere almeno 4 chiamate: {prob2:.9f}")
```

```
Probabilità di ricevere esattamente 3 chiamate: 0.140373896
Probabilità di ricevere almeno 4 chiamate: 0.734974085
```

Cenni: distribuzione Geometrica

- **Definizione:** Modella il numero di prove Bernoulli necessarie per ottenere il **primo successo**.

In altri termini: è la probabilità che il primo successo (o evento in generale) richieda l'esecuzione di k prove indipendenti, ognuna con probabilità di successo p.

- **Parametro:** p = probabilità di successo
- **PMF:**

$$P(X = k) = (1 - p)^{k-1} p, \quad k = 1, 2, 3, \ldots$$

- **Media:** $E[X] = \frac{1}{p}$
- **Varianza:** $\mathrm{Var}(X) = \frac{1-p}{p^2}$

Esercizi con soluzioni per approfondire

Esercizio — Media, varianza, deviazione standard (discreta)

Testo: La variabile casuale X assume i valori 0,1,2 con probabilità rispettive p(0)=0.2; p(1)=0.5; p(2)=0.3. Calcola la media $\mathbb{E}[X]$, la varianza $\mathrm{Var}(X)$ e la deviazione standard (σ).

Soluzione:

Nel caso del valore atteso:

$$\mathbb{E}[X] = 0 \cdot 0.2 + 1 \cdot 0.5 + 2 \cdot 0.3 = 1.1$$

Nel caso della varianza si possono usare diverse strade. Qui ne seguiamo una parte dall'osservare che è possibile dimostrare la seguente relazione:

$Var(X) = \mathbb{E}[X^2] - (\mathbb{E}[X])^2$

Procederemo quindi calcolando $\mathbb{E}[X^2]$, ovvero la somma dei quadrati dei valori moltiplicati per le loro rispettive probabilità:

$$\mathbb{E}[X2] = \sum_i x_i^2 \cdot p(x_i)$$

$$\mathbb{E}[X^2] = 0^2 \cdot 0.2 + 1^2 \cdot 0.5 + 2^2 \cdot 0.3 = 1.7$$

Calcoliamo infine la Var(X)

$$\mathrm{Var}(X) = \mathbb{E}[X^2] - (\mathbb{E}[X])^2 = 1.7 - 1.1^2 = 1.7 - 1.21 = 0.49$$
$$\sigma = \sqrt{0.49} = 0.7$$

Codice (verifica):

```
import numpy as np # uso libreria numpy che gestisce array ovvero vettori

vals = np.array([0,1,2])
probs = np.array([0.2,0.5,0.3])

print(vals * probs) # stampo prodotto per capire che tipo di prodotto fa
mean = (vals * probs).sum()
ex2 = (vals**2 * probs).sum()
var = ex2 - mean**2
std = np.sqrt(var)

print("mean =", mean)
print("var =", var)
print("std =", std)
```

Esercizio — Media e mediana di un campione, robustezza della mediana

Testo: Dato il campione osservato ([2,3,7,10,12]):

1. Calcola la media e la mediana.
2. Aggiungi l'outlier 100 al campione e ricalcola media e mediana: cosa osservi?

Soluzione (passo-passo):

Campione originale:

$$\text{media} = \frac{2+3+7+10+12}{5} = \frac{34}{5} = 6.8$$

Ordinato il campione, la mediana è il valore al centro: (7).

Aggiungendo 100: nuovo campione ([2,3,7,10,12,100]).

- Media = $\frac{34+100}{6} = \frac{134}{6} \approx 22.333$ (fortemente influenzata dall'outlier).
- Mediana (valori ordinati) = media dei due valori centrali ((7+10)/2 = 8.5) — la mediana cambia poco rispetto alla media, quindi è **più robusta** agli outlier.

Codice (verifica):

```
import numpy as np
data = np.array([2,3,7,10,12], dtype=float)
print("mean original:", data.mean())
print("median original:", np.median(data))

data_out = np.append(data, 100)
print("mean with outlier:", data_out.mean())
print("median with outlier:", np.median(data_out))
```

Esercizio 4 — Moda (frequenza)

Testo: Dati i valori campionari ([1,2,2,3,3,3,4]), quale è la moda?

Soluzione: La moda è il valore più frequente: qui il valore (3) compare 3 volte, più di ogni altro, quindi **moda = 3**.

Codice (verifica):

```
from collections import Counter
data = [1,2,2,3,3,3,4]
cnt = Counter(data)
mode = cnt.most_common(1)[0][0]
print("mode =", mode)
```

In [8]:
```
import numpy as np # uso libreria numpy che gestisce array ovvero vettori

vals = np.array([0,1,2])
probs = np.array([0.2,0.5,0.3])
print("vals: ", vals)
print("probs: ", probs)
print("vals*probs: ",vals * probs) # stampo prodotto per capire che tipo di prodotto fa
mean = (vals * probs).sum()
ex2 = (vals**2 * probs).sum()
var = ex2 - mean**2
std = np.sqrt(var)

print("mean =", mean)
print("var =", var)
print("std =", std)
```

```
vals:  [0 1 2]
probs:  [0.2 0.5 0.3]
vals*probs:  [0.  0.5 0.6]
mean = 1.1
var = 0.48999999999999977
std = 0.6999999999999998
```

In [10]:
```
import numpy as np
data = np.array([2,3,7,10,12], dtype=float)
print("mean original:", data.mean())
print("median original:", np.median(data))

data_out = np.append(data, 100)
print("mean with outlier:", data_out.mean())
print("median with outlier:", np.median(data_out))
```

```
mean original: 6.8
median original: 7.0
mean with outlier: 22.333333333333332
median with outlier: 8.5
```

In [12]:
```
# qui usiamo modulo collections
# non necessario per forza usare collections
# come potresti scrivere codice senza usare questo modulo?

from collections import Counter
data = [1,2,2,3,3,3,3,4]
print(data)
cnt = Counter(data)
print("Counter: ",cnt)
mode = cnt.most_common(1)[0][0]
print("mode =", mode)
```

```
[1, 2, 2, 3, 3, 3, 3, 4]
Counter:  Counter({3: 4, 2: 2, 1: 1, 4: 1})
mode = 3
```

Ulteriori esercizi di approfondimento

In [21]:
```
# ============================================================
# ESERCIZIO:
# Un paziente ha una probabilità del 30% di rispondere a un trattamento.
# Simula 1000 pazienti e calcola:
# - quante risposte positive si ottengono
# - la frequenza empirica di successo
# ============================================================

import numpy as np

p = 0.3                 # probabilità di successo
n = 1000                # numero di pazienti simulati

# Simulazione Bernoulli: 1 = successo, 0 = fallimento
# N.B. Una variabile casuale Binomiale con n=1 è
# identica a una variabile casuale di Bernoulli.

#N.B. La funzione np.random.binomial(n, p, size=m) di NumPy genera campioni
# da una distribuzione Binomiale
risposte = np.random.binomial(1, p, size=n)

successi = risposte.sum()
frequenza = successi / n
print("Stampiamo lista risposte: \n",risposte)
print("Stampiamo i successi (sommiamo gli 1 nel vettore risposte):\n",successi)
print(f"Successi simulati: {successi}")
print(f"Numero pazienti simulati {n}")
print(f"Frequenza empirica: {frequenza:.3f}")


######################

print("\nSe aumentiamo il numero di pazienti che succede? ")
p = 0.3                 # probabilità di successo
n = 100000                # numero di pazienti simulati

# Simulazione Bernoulli: 1 = successo, 0 = fallimento
risposte = np.random.binomial(1, p, size=n)
```

```
successi = risposte.sum()
frequenza = successi / n

print(f"Successi simulati: {successi}")
print(f"Frequenza empirica: {frequenza:.3f}")
```

```
Stampiamo lista risposte:
 [0 1 1 0 1 1 0 0 0 0 1 1 0 1 0 1 0 0 0 0 1 0 0 0 0 0 0 0 1 1 0 0 0 0 1 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 0 1 1 0 0 0 1 0 1 0 1 0 0 1 0 0 0 0 1
 0 1 0 0 0 0 0 0 0 0 1 0 0 0 0 1 0 1 1 0 0 0 0 0 1 0 0 0 0 1 1 1 0 1 0 0 0
 1 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 1 0 1 1 0 0 0 0 0 1 0 0 0 0 0 1 1 0 0 1 0
 0 0 0 0 1 1 1 0 1 0 0 0 1 1 0 0 0 0 0 0 0 0 0 0 1 0 0 1 1 1 0 0 0 1 1 0 0 0
 0 0 0 0 0 0 0 1 0 0 1 0 1 0 0 0 0 0 0 0 1 0 0 0 0 1 0 0 1 1 0 0 0 1 0 1 0
 0 1 0 1 0 1 0 0 0 1 1 1 1 0 1 0 1 0 1 0 0 0 0 1 0 1 1 1 0 0 0 0 0 0 0 0 0
 0 0 1 1 0 0 1 0 0 0 0 0 0 0 0 0 0 1 1 1 1 0 0 1 1 0 1 0 0 1 0 0 0 0 0 0 1
 0 1 0 1 0 1 0 1 0 0 0 1 0 0 1 0 1 0 0 1 0 0 1 0 0 1 1 0 1 0 0 0 0 0 0 1 1
 1 0 0 0 0 0 0 0 0 1 1 0 1 1 0 1 0 1 0 0 0 1 1 0 1 0 1 1 0 1 0 0 1 0 0 0 1
 1 1 0 0 0 0 0 0 0 0 1 0 0 0 0 0 1 1 0 0 0 1 0 0 0 1 0 0 1 1 0 0 1 0 0 0 0
 0 1 0 1 0 1 0 0 0 1 1 0 0 1 0 0 0 0 0 1 1 1 0 0 0 1 1 1 0 0 0 0 0 1 1 1 1
 0 0 1 1 1 0 0 0 0 0 0 0 1 1 0 0 1 0 0 0 0 0 0 0 1 1 0 1 0 1 0 0 0 0 0 0 1
 0 0 0 0 0 0 1 1 1 0 0 0 1 0 1 0 0 1 0 1 0 0 0 0 0 0 0 0 0 0 1 0 0 1 1 1 0
 1 0 1 0 0 0 0 0 0 0 0 1 0 0 1 0 0 1 1 0 0 1 0 0 1 1 0 1 1 0 0 1 0 1 0 0 0
 0 1 0 0 1 0 0 0 0 0 0 0 0 0 0 1 1 0 0 0 0 0 0 1 1 0 0 0 0 0 1 0 0 0 0 0 1
 0 0 1 0 1 0 1 1 0 0 0 1 1 0 0 1 0 0 0 0 0 0 0 0 1 0 1 1 0 0 0 0 0 0 0 1 0
 0 0 0 1 1 0 1 0 0 1 0 1 1 0 1 1 0 0 0 0 0 0 0 1 0 0 1 0 1 1 1 0 0 0 1 1 1
 1 1 1 1 0 1 1 0 0 0 0 1 1 0 1 0 0 0 1 0 0 0 0 0 0 0 0 0 1 0 0 0 1 0 1 0 0
 1 1 1 1 0 0 0 0 0 0 1 0 0 1 0 0 0 0 1 1 1 0 0 0 1 0 0 0 1 1 1 0 0 0 1 0 0
 0 0 1 0 0 0 0 1 0 0 1 0 1 0 0 0 1 0 0 0 1 0 0 0 0 0 1 0 0 0 0 0 0 0 0 1 1
 0 0 0 0 1 0 1 0 1 0 0 0 0 0 1 0 0 0 1 1 1 0 0 0 0 0 0 0 1 0 1 0 0 1 0 0 0
 0 1 1 1 0 0 0 1 0 0 1 0 0 0 0 0 1 0 0 1 0 1 0 0 1 1 0 1 1 0 0 1 0 1 0 0 0
 0 0 1 0 1 1 0 1 0 0 1 0 0 0 1 1 0 1 1 1 1 0 0 0 1 0 0 0 0 0 0 0 1 0 0 0 0
 0 0 0 1 0 1 0 0 1 0 1 0 1 1 1 0 0 1 0 0 1 1 0 0 0 0 0 1 0 0 0 0 0 1 1 0 0
 0 0 1 1 1 1 0 0 0 0 0 0 1 1 1 0 1 0 0 0 1 0 0 0 1 1 1 1 0 0 1 0 1 0 1 0 1
 0 1 0 0 0 0 0 1 0 0 1 0 0 1 1 1 0 1 0 1 1 1 0 0 0 1 0 0 0 0 1 0 0 0 0 0 0
 0]
Stampiamo i successi (sommiamo gli 1 nel vettore risposte):
 329
Successi simulati: 329
Numero pazienti simulati 1000
Frequenza empirica: 0.329

Se aumentiamo il numero di pazienti che succede?
Successi simulati: 30098
Frequenza empirica: 0.301
```

In [13]:
```
# ============================================================
# ESERCIZIO:
# In un reparto un tampone rapido intercetta il virus nel 80% dei casi.
# Se si eseguono 20 tamponi su pazienti infetti, qual è la probabilità di
# ottenere esattamente 15 risultati positivi?
# Inoltre simula 10.000 prove e confronta la probabilità teorica con quella simulata.
# ============================================================

import numpy as np
from scipy.stats import binom

n = 20        # numero di prove (tamponi)
p = 0.80      # probabilità di successo

# Probabilità teorica di ottenere esattamente 15 successi
k = 15
prob_teorica = binom.pmf(k, n, p)

# Simulazione  (vedremo meglio cosa significa simulare)
print("Simulazione 1")
simulazioni = 1000
risultati = np.random.binomial(n, p, size=simulazioni)
prob_empirica = np.mean(risultati == k)

print(f"Probabilità teorica di ottenere 15 successi: {prob_teorica:.5f}")
print(f"Probabilità empirica da simulazione: {prob_empirica:.5f}")


# Simulazione 2
print("Simulazione 2")
simulazioni = 100000
risultati = np.random.binomial(n, p, size=simulazioni)
prob_empirica = np.mean(risultati == k)

print(f"Probabilità teorica di ottenere 15 successi: {prob_teorica:.5f}")
print(f"Probabilità empirica da simulazione: {prob_empirica:.5f}")
```

```
Simulazione 1
Probabilità teorica di ottenere 15 successi: 0.17456
Probabilità empirica da simulazione: 0.16000
Simulazione 2
Probabilità teorica di ottenere 15 successi: 0.17456
Probabilità empirica da simulazione: 0.17625
```