Tema 11: Distribuciones

Alonso Pizarro Lagunas

8/12/2021

Distribuciones en *R* y *Python*

Algunas funciones de *R* para cualquier variable aleatoria $v.a$

- `dva(x, ...)` se refiere a la función de densidad o probabilidad $f(x)$ de la $v.a$
- `pva(x, ...)` se refiere a la funcion de distribución $F(x)$ de la $v.a$

- `qva(p, ...)` Cuantil p-ésimo de la $v.a$ (el valor de x más pequeño tal que $F(x) \geq p$)
- `rva(n, ...)` Generador de n observaciones siguiendo la distribución de la $v.a$

En `va` debemos indicar la distribución. Por ejemplo si es una Bernoulli indicamos `dbern`, `pbern`, `qbern` y `rbern`. Para una normal `dnorm`, `pnorm`, `qnorm` y `rnorm`. Así, con todas las distribuciones.

Algunas funciones de *Python* para cualquier variable aleatoria $v.a$

- `pmf(k, ...)` o `pdf(x, ...)` se refiere a la función de densidad $(f(x))$ o probabilidad $f(k)$ de la $v.a$
- `cdf(x, ...)` se refiere a la funcion de distribución $F(x)$ de la $v.a$

- `ppf(p, ...)` Cuantil p-ésimo de la $v.a$ (el valor de x más pequeño tal que $F(x) \geq p$)
- `rvs(size, ...)` Generador de *size* observaciones siguiendo la distribución de la $v.a$

Distribución de Bernoulli

Si X es una $v.a$ que mide el número de éxitos (fracasos) y se realiza un único experimento diremos que X se distribuye como una Bernoulli con parámetro p

$$X \sim Be(p)$$

donde $q = 1 - p$ es la probabilidad de fracaso.

El dominio de X es $X(\Omega) = \{0, 1\}$

La función de probabilidad es

$$f(k) = p^k \cdot (1-p)^{1-k}$$

La función de distribución es

$$F(k) = \begin{cases} 0 & \text{si } k < 0 \\ q & \text{si } 0 \leq k < 1 \\ 1 & \text{si } k \geq 1 \end{cases}$$

Con *Esperanza* $E(X) = p$ y *Varianza* $Var(X) = pq$

Un ejemplo con la distribución Bernoulli en R

Sea $X \sim Be(p = 0.7)$, la distribución que modela la probabilidad de obtener una cara usando una moneda trucada.

```
library(Rlab) # Recordamos cargar el paquete Rlab que contiene toda la info sobre distribuciones
```

```
## Rlab 2.15.1 attached.

##
## Attaching package: 'Rlab'

## The following objects are masked from 'package:stats':
##
##     dexp, dgamma, dweibull, pexp, pgamma, pweibull, qexp, qgamma,
##     qweibull, rexp, rgamma, rweibull

## The following object is masked from 'package:datasets':
##
##     precip
```

```
dbern(0, prob = 0.7) # Prob de sacar 0, f`n densidad
```

```
## [1] 0.3
```

```
dbern(1, prob = 0.7) # Prob de sacar 1, f`n densidad
```

```
## [1] 0.7
```

```
pbern(0, prob = 0.7) # F`n acumulada
```

```
## [1] 0.3
```

```
pbern(1, prob = 0.7) # F`n acumulada
```

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

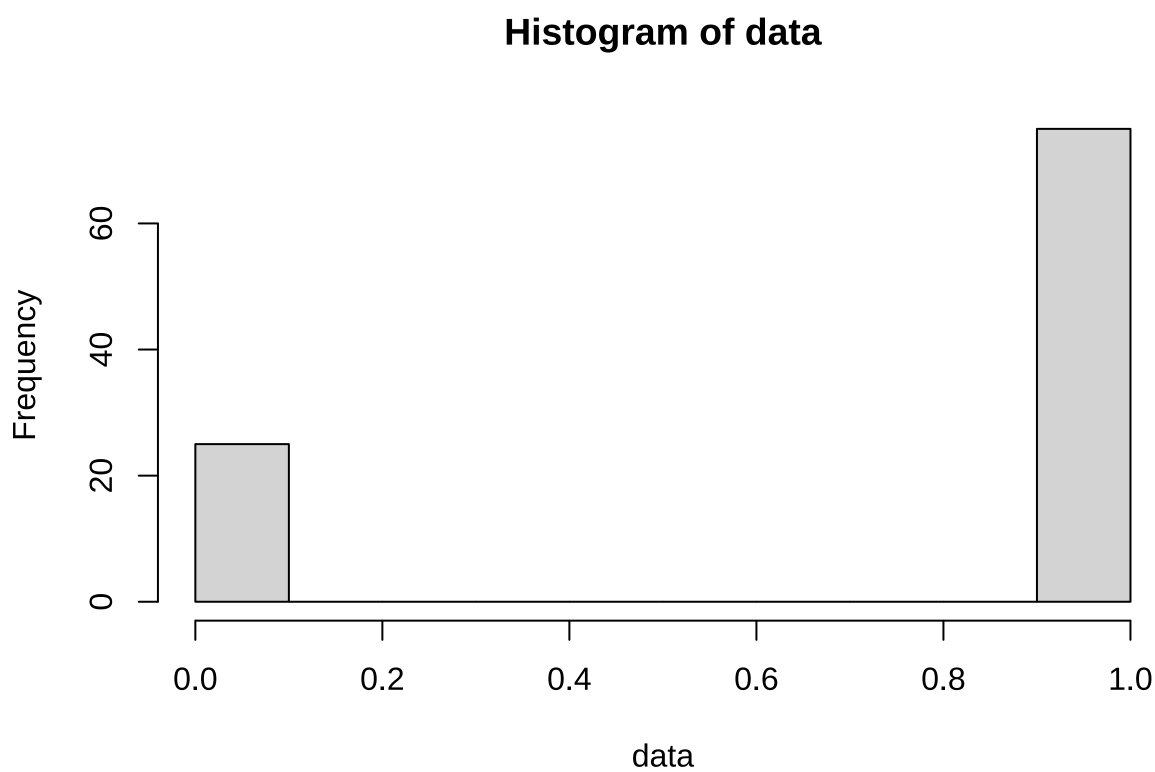
```
qbern(0.5, prob = 0.7) # segundo cuartil
```

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

```
qbern(0.25, prob = 0.7) # primer cuartil
```

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

```
data <- rbern(100, prob = 0.7) # Generemos 100 números aleatorios con esta distribución
hist(data) # Visualizamos los datos en un histograma
```



Un ejemplo con la distribución Bernoulli en Python

```
from scipy.stats import bernoulli
import matplotlib.pyplot as plt
p = 0.7
mean, var, skew, kurt = bernoulli.stats(p, moments = 'mvsk')
print("Media %f"%mean)
```

```
## Media 0.700000
```

```
print("Varianza %f"%var)
```

```
## Varianza 0.210000
```

```
print("Sesgo %f"%skew)
```

```
## Sesgo -0.872872
```

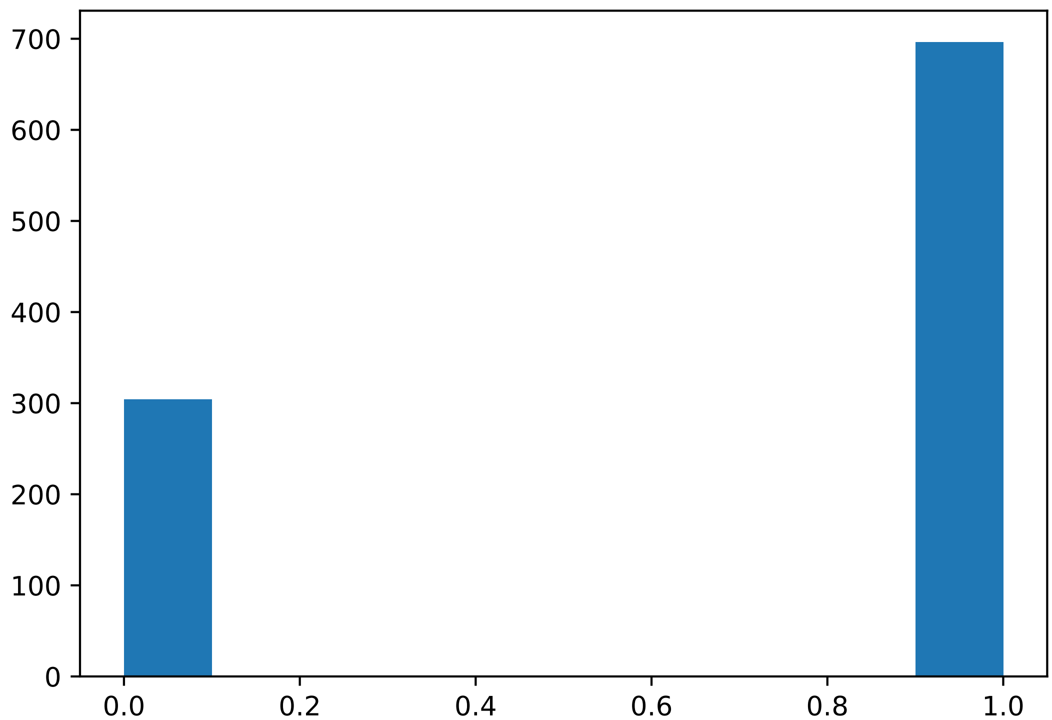
```
print("Curtosis %f"%kurt)
```

```
## Curtosis -1.238095
```

```
fix, ax = plt.subplots(1,1)
x = bernoulli.rvs(p,size=1000)
ax.hist(x)
```

```
## (array([304.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0., 696.]), array([0. , 0.1, 0.2, 0.3, 0.4
```

```
plt.show()
```



Distribución Binomial

Si X es una $v.a$ que mide el número de éxitos y se realizan n ensayos Bernoulli independientes entre sí diremos que X se distribuye como una binomial con parámetros n y p

$$X \sim B(n,p)$$

donde p es la probabilidad de éxito y $q = 1 - p$ es la probabilidad de fracaso.

El dominio de X será $D_X = \{0, 1, 2, ..., n\}$

La función de densidad viene dada por

$$f(k) = \binom{n}{k} p^k (1-p)^{n-k}$$

La función de distribución viene dado por

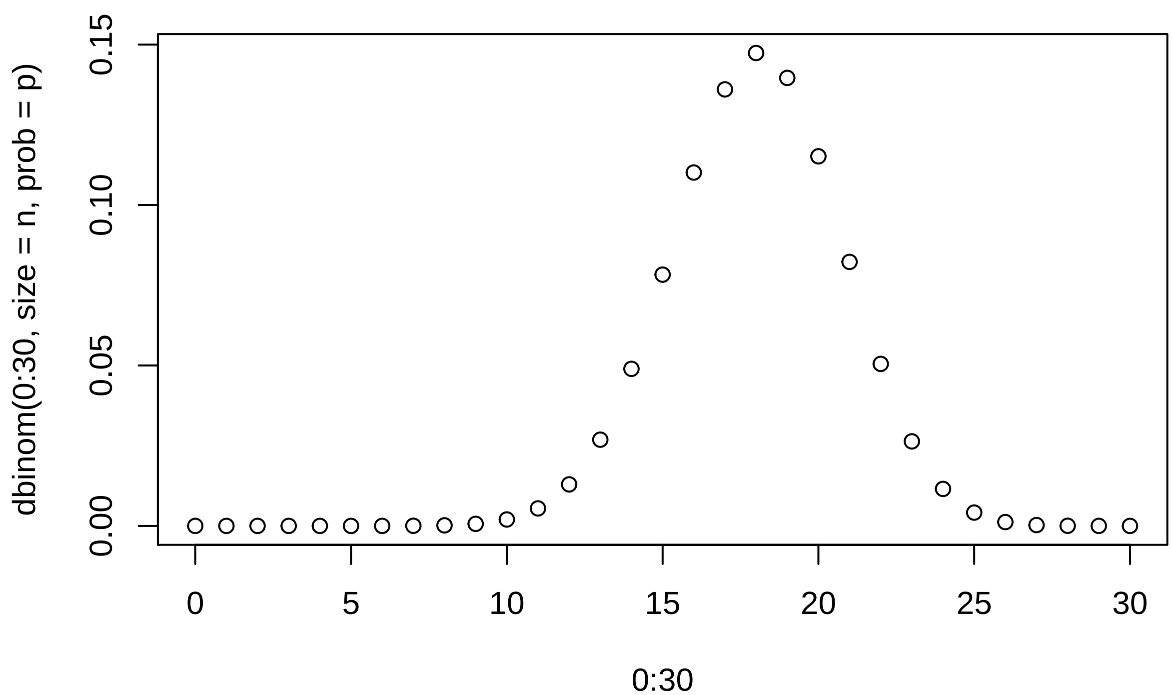
$$F(x) = \begin{cases} 0 & \text{si } x < 0 \\ \sum_x^{k=0} f(k) & \text{si } 0 \leq x < 1 \\ 1 & \text{si } x \geq n \end{cases}$$

Con una *Esperanza* igual a $E(X) = np$ y una *Varianza* igual a $Var(X) = npq$.

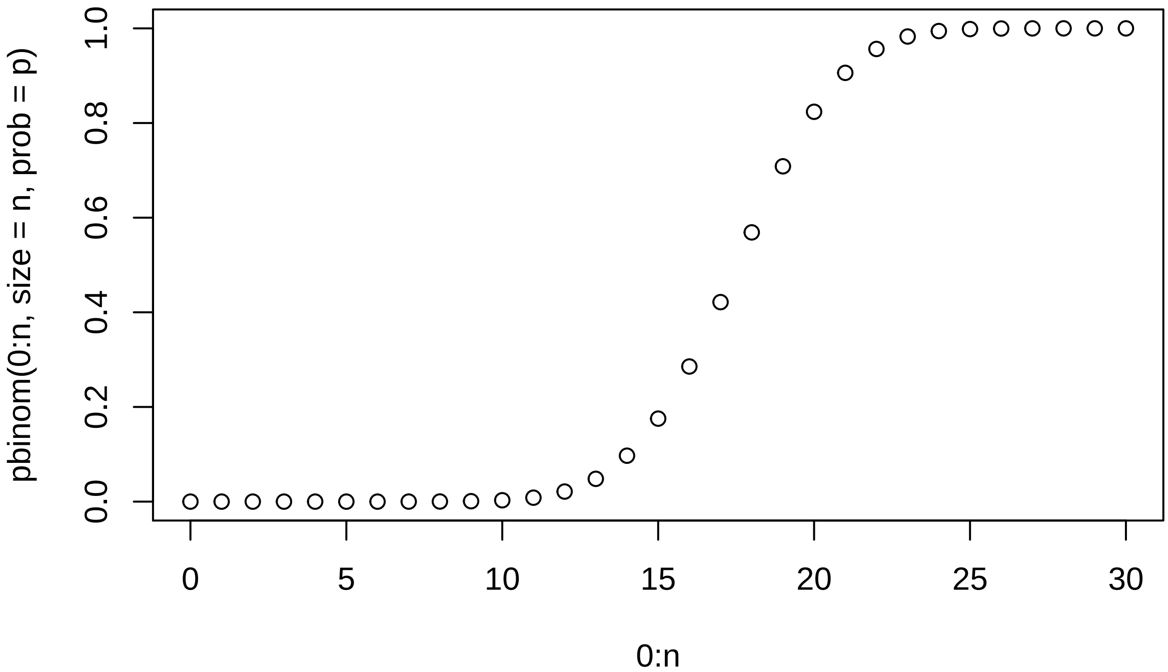
Ejemplo de la distribución binomial en R.

Sea $X \sim B(n = 30, p = 0.6)$

```
n =30
p =0.6
plot(0:30,dbinom(0:30, size = n, prob =p))
```




```
plot(0:n, pbinom(0:n, size = n, prob = p))
```



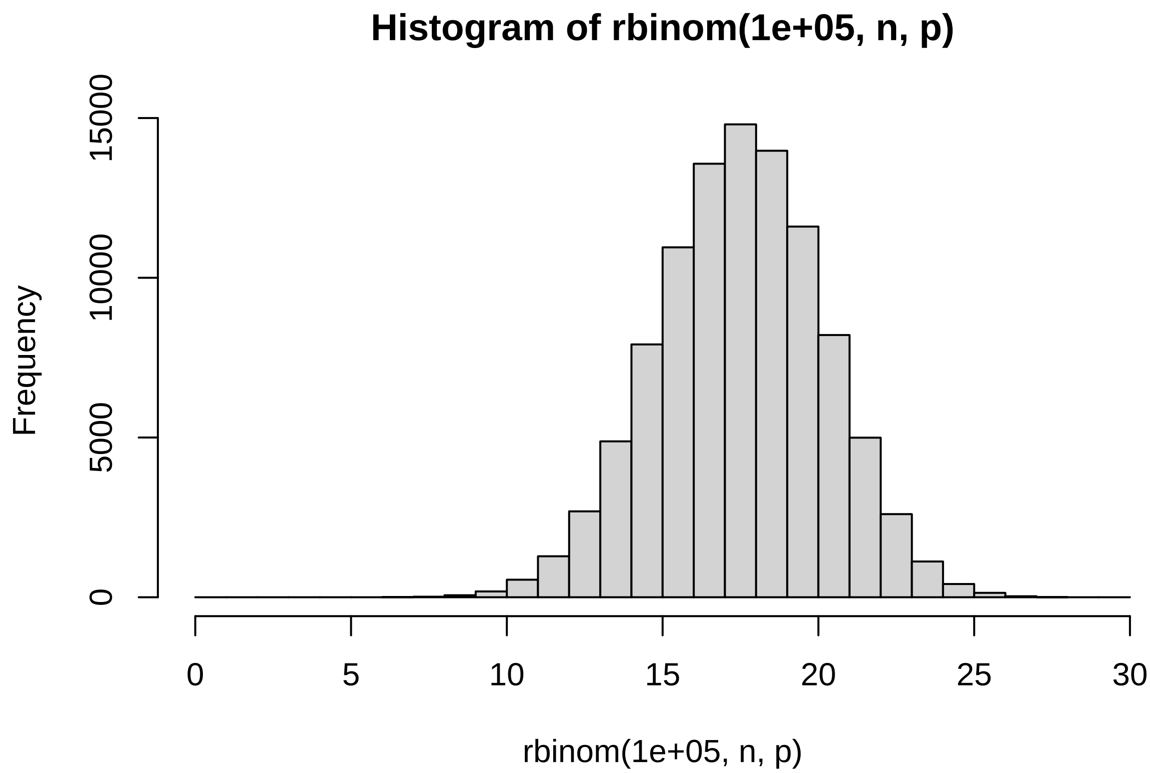

```
qbinom(0.5, n, p)
```

```
## [1] 18
```

```
qbinom(0.25, n, p)
```

```
## [1] 16
```

```
hist(rbinom(100000, n, p), breaks =0:30)
```



Ejemplo de la distribución binomial en Pyhton.

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

fig, ax = plt.subplots()
n = 7
p = 0.4

mean, var, skew, kurt = binom.stats(n, p, moments = 'mvsk')
print("Media %f"%mean)
```

```
## Media 2.800000
```

```
print("Varianza %f"%var)
```

```
## Varianza 1.680000
```

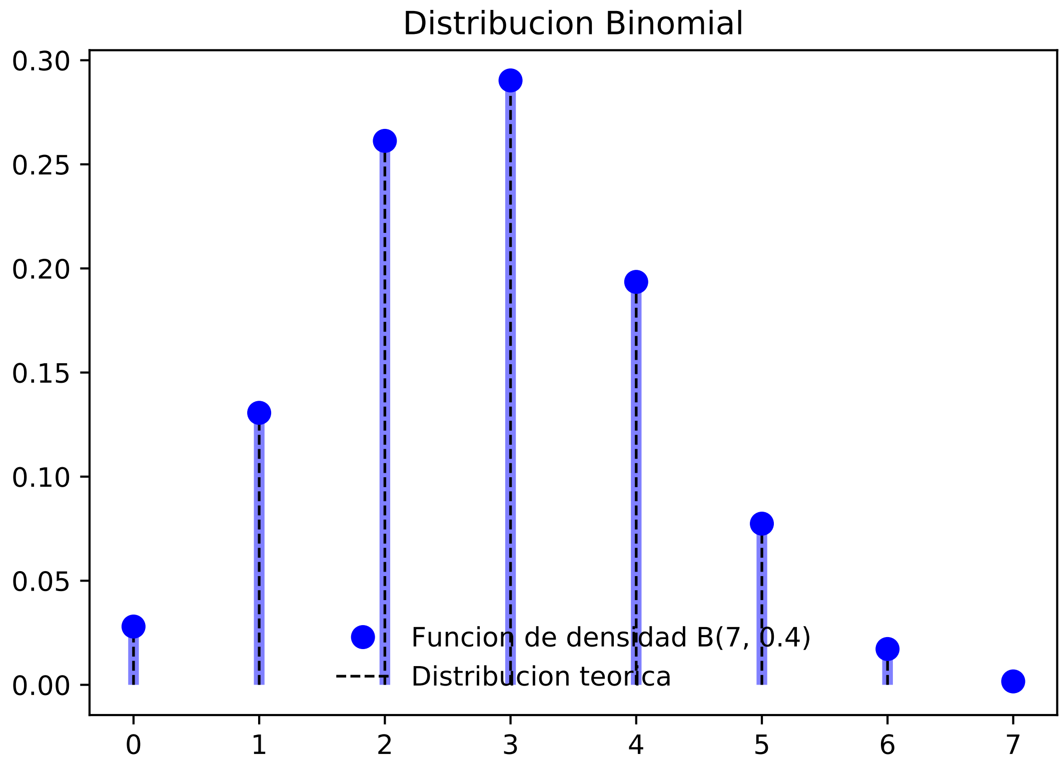
```
print("Sesgo %f"%skew)
```

```
## Sesgo 0.154303
```

```
print("Curtosis %f"%kurt)
```

```
## Curtosis -0.261905
```

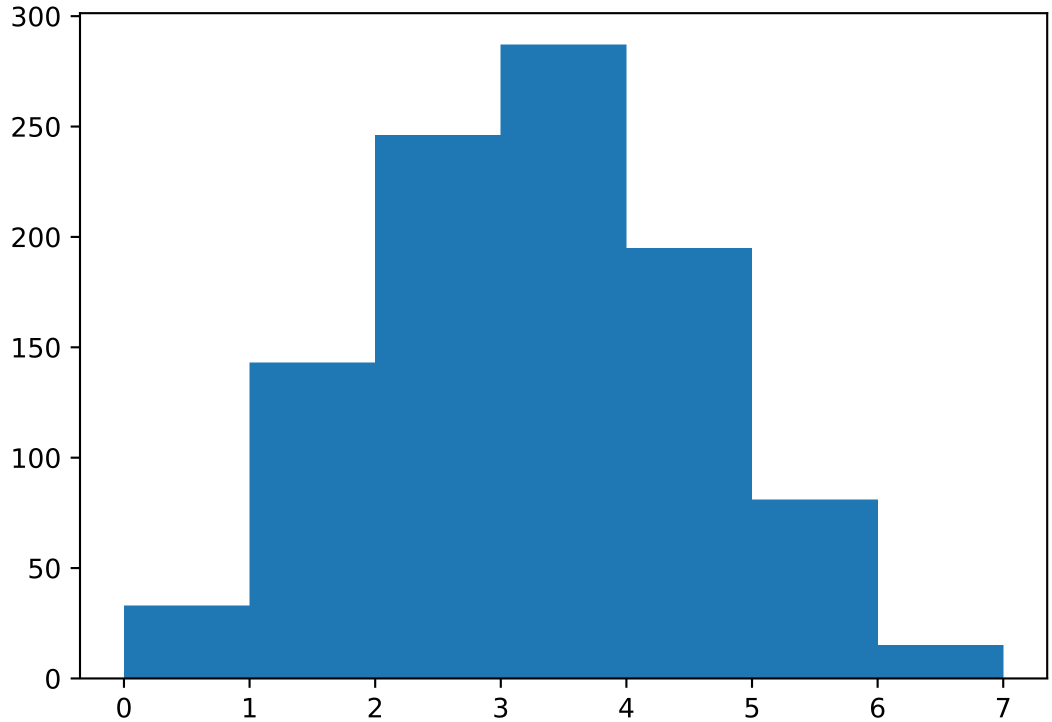
```
x = np.arange(0,n+1)
ax.plot(x, binom.pmf(x,n,p),'bo',ms = 8, label = 'Funcion de densidad B(7, 0.4)')
ax.vlines(x, 0, binom.pmf(x,n,p), colors = 'b', lw = 4, alpha = 0.5)

rv = binom(n,p)
ax.vlines(x,0, rv.pmf(x), colors ='k', linestyles = '--', lw = 1, label ='Distribucion teorica')
ax.legend(loc = 'best', frameon = False)
ax.set_title(r'Distribucion Binomial')
plt.show()
```




```
fig, ax = plt.subplots(1,1)
r = binom.rvs(n,p, size = 1000)
ax.hist(r, bins = n)
```

```
## (array([ 33., 143., 246., 287., 195.,  81.,  15.]), array([0., 1., 2., 3., 4., 5., 6., 7.]), <BarCon
```

```
plt.show()
```



Podemos encontrar más información sobre la visualización en la página de Matplotlib y más.

Distribución Geométrica

Sea X una $v.a$ que mide el número de repeticiones independientes del experimento hasta haber conseguido un éxito diremos que X se distribuye como una Geométrica con parámetro p

$$X \sim Ge(p)$$

donde p es la probabilidad de éxito y $q = 1 - p$ es la probabilidad de fracaso.

El dominio de X es $D_X = \{0, 1, 2, ...\}$ o bien $D_X = \{1, 2, ...\}$ si comienza con cero o comienza en uno.

La *función de densidad* viene dada por

$$f(k) = (1 - p)^k$$

si comienza en cero o

$$f(k) = (1 - p)^{k-1} \cdot p$$

si comienza con 1.

La función de distribución viene dado por

$$F(x) = \begin{cases} 0 & \text{si } x < 0 \\ 1 - (1-p)^{k+1} & \text{si } k \leq x < k+1, k \in \mathbb{N} \end{cases}$$

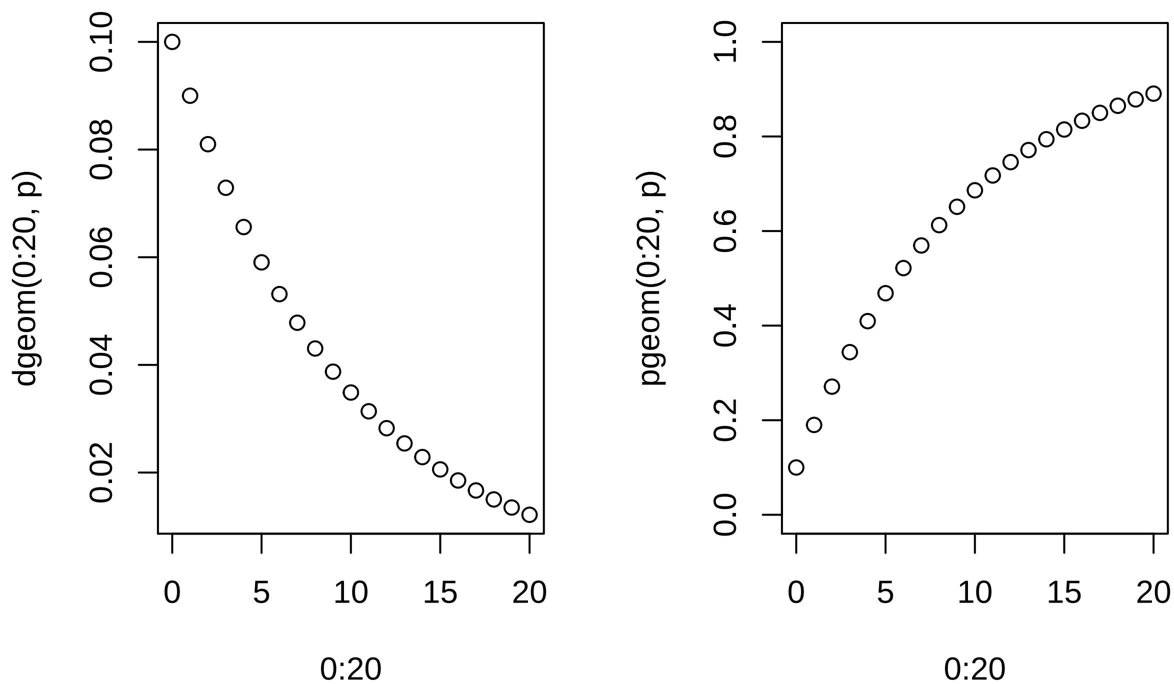
La esperanza $E(X) = \frac{1-p}{p}$ si comienza en 0 y $E(X) = \frac{1}{p}$ si es uno.

La varianza es $Var(x) = \frac{1-p}{p^2}$.

Ejemplo en R

Sea $X = Ge(p = 0.1)$ las distribución que modela la probabilidad de abrir una puerta hasta conseguirlo.

```
p = 0.1
par(mfrow=c(1,2))
plot(0:20,dgeom(0:20, p)) # Función de densidad
plot(0:20,pgeom(0:20, p), ylim = c(0,1)) # Función distribución
```




```
qgeom(0.5,p) # Cuantiles
```

```
## [1] 6
```

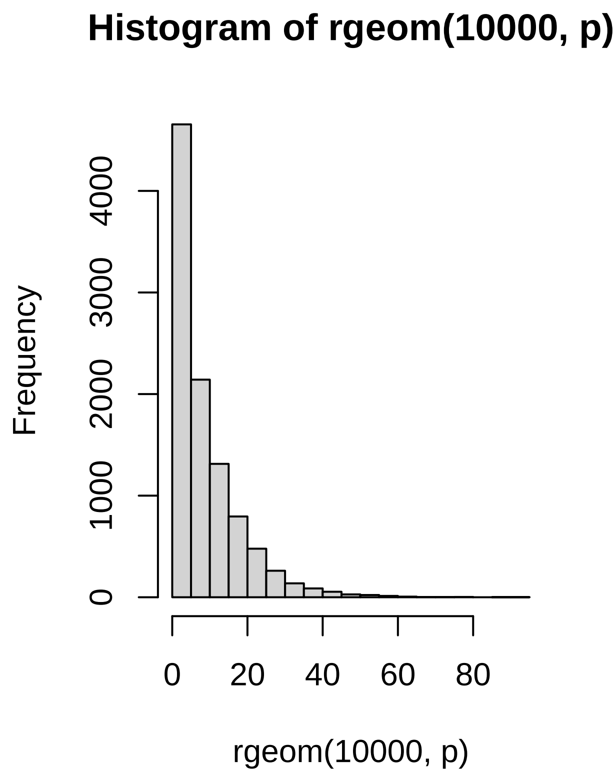
```
qgeom(0.75,p)
```

```
## [1] 13
```

```
rgeom(10, p) # generamos datos aleatorios de esta dist (hasta el éxito)
```

```
##  [1]  0 27  1  0  0 21  8 29  1  6
```

```
hist(rgeom(10000, p)) # un histograma con 10000 experimentos
```



Ejemplo en python

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

fig, ax = plt.subplots(1,1)
p = 0.3
mean, var, skew, kurt = geom.stats(p, moments='mvsk')

print("Media %f"%mean)
```

```
## Media 3.333333
```

```
print("Varianza %f"%var)
```

```
## Varianza 7.777778
```

```
print("Sesgo %f"%skew)
```

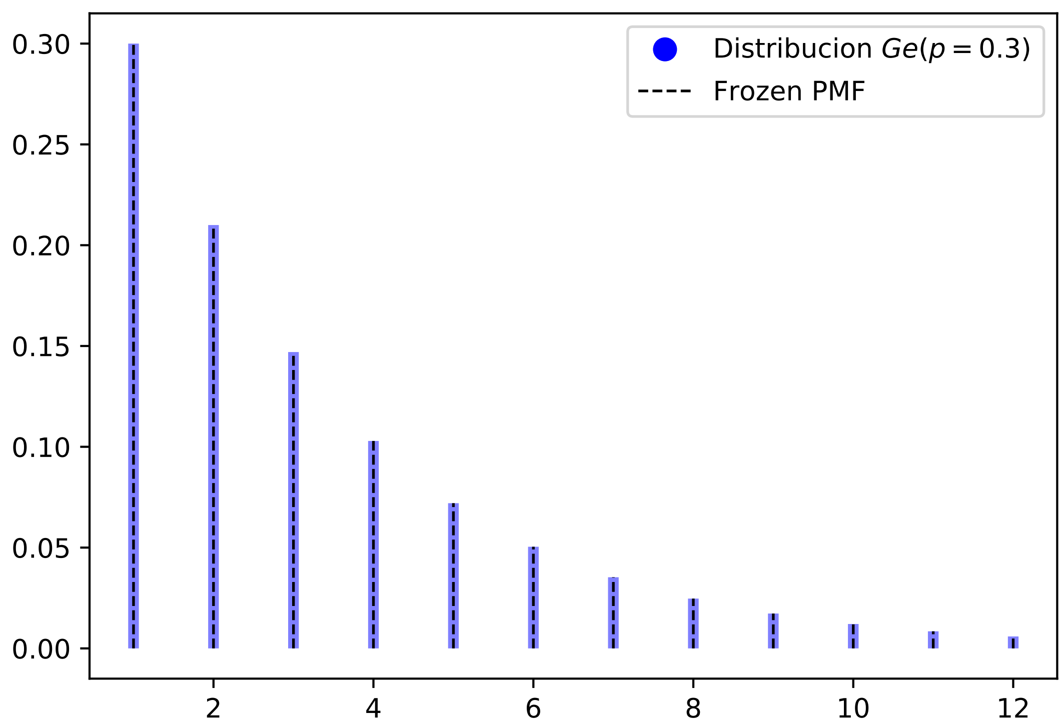
```
## Sesgo 2.031889
```

```
print("Curtosis %f"%kurt)
```

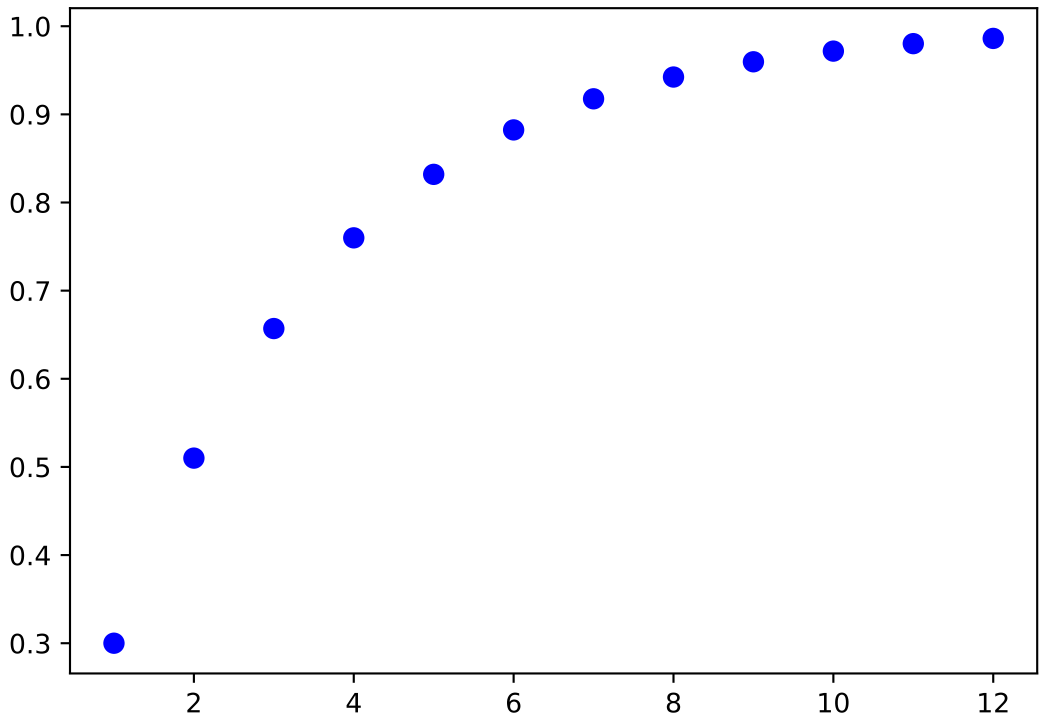
```
## Curtosis 6.128571
```

```
x = np.arange(geom.ppf(0.01,p), geom.ppf(0.99,p))
ax.plot(x,geom.ppf(x,p), 'bo', ms = 8, label = "Distribucion $Ge(p = 0.3)$")
```

```
ax.vlines(x,0, geom.pmf(x,p), colors = 'b', lw = 4, alpha = 0.5)



rv = geom(p)
ax.vlines(x,0, geom.pmf(x,p), colors = 'k', linestyles = '--', lw = 1, label = 'Frozen PMF')
ax.legend(loc = 'best')
plt.show()
```



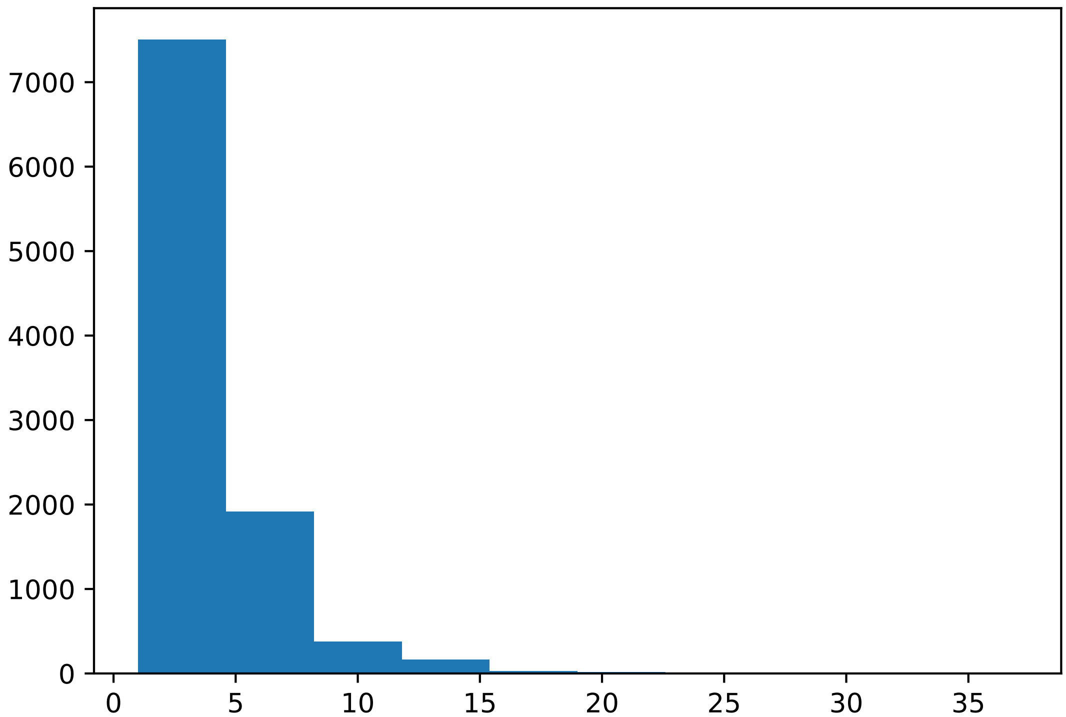

```
fig, ax = plt.subplots(1,1)
prob = (geom.cdf(x,p))
ax.plot(x, prob, 'bo', ms = 7, label = 'Funcion de distribucion acumulada')
plt.show()
```




```
fig, ax = plt.subplots(1,1)
r = geom.rvs(p, size = 10000)
plt.hist(r)
```

```
## (array([7.501e+03, 1.912e+03, 3.740e+02, 1.650e+02, 2.800e+01, 1.300e+01,
##        3.000e+00, 3.000e+00, 0.000e+00, 1.000e+00]), array([ 1. ,  4.6,  8.2, 11.8, 15.4, 19. , 22.6
```

```
plt.show()
```



Distribución Hipergeométrica

Consideremos un experimento de extraer, una detrás de otra sin reemplazo, n objetos donde hay N de tipo A y M de tipo B. Si X es una $v.a$ que mide el "el número de objetos del tipo A" diremos que X se distribuye como una hipergeométrica con parámeros N, My n.

El dominio de X será $D_X = \{0, 1, 2, ..., N\}$

La *función de densidad* vendrá dado por

$$f(k) = \frac{\binom{N}{k}\binom{M}{n-k}}{\binom{N+M}{n}}$$

La *función de distribución* viene dada por

$$F(x) = \begin{cases} 0 & \text{si } x < 0 \\ \sum_{k=0}^{x} f(x) & \text{si } 0 \leq x < n \\ 1 & \text{si } x \geq n \end{cases}$$

Con una *Esperanza* $E(X) = \frac{nN}{N+M}$ y una *Varianza* $Var(X) = \frac{nNM}{(N+M)^2} \cdot \frac{N+M-n}{N+M-1}$

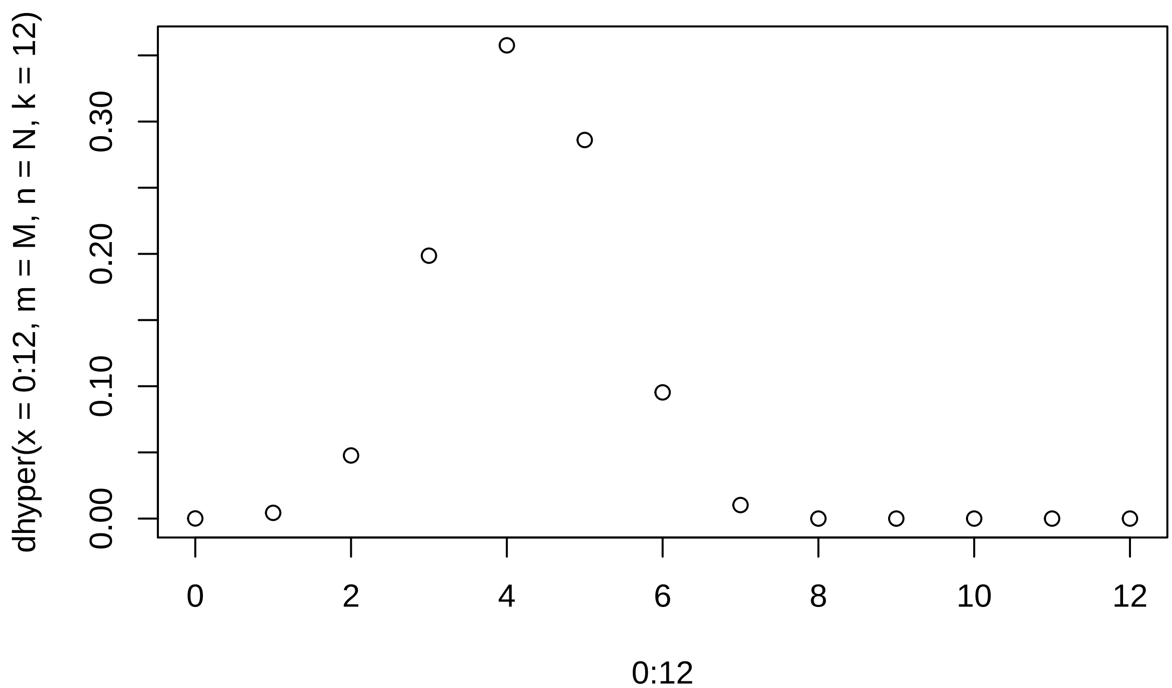
Veamos un ejemplo de distribución Hipergeométrica en `R`.

Suponga que tenemos 20 animales de los cuales 7 son perros. Queremos medir la probabilidad de encontrar un número determinado de perros si elegimos $k = 12$ animales al azar.

```r
# dhyper gives the density, phyper gives the distribution function,
# qhyper gives the quantile function, and rhyper generates random deviates.

x = 12
M = 7
N = 13

dhyper(x = 0:12, m = M, n = N, k = 12)
```

```
##  [1] 0.0001031992 0.0043343653 0.0476780186 0.1986584107 0.3575851393
##  [6] 0.2860681115 0.0953560372 0.0102167183 0.0000000000 0.0000000000
## [11] 0.0000000000 0.0000000000 0.0000000000
```

```r
plot(x = 0:12, dhyper(x = 0:12, m = M, n = N, k = 12))
```



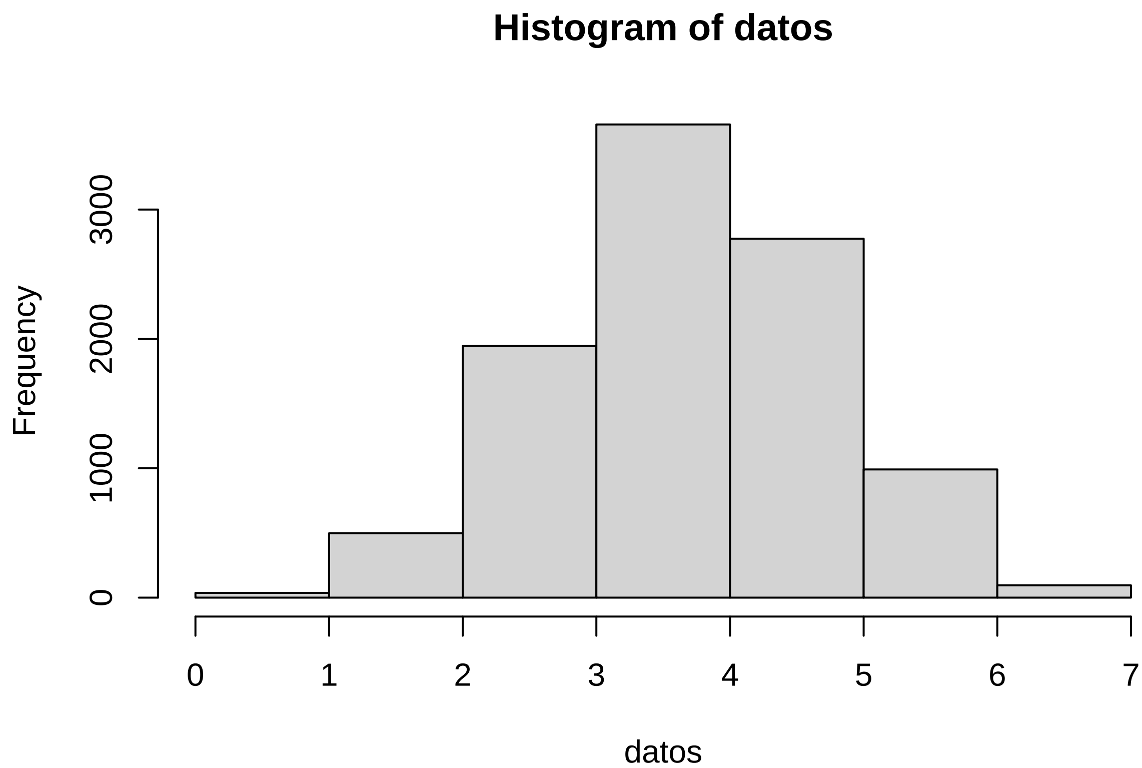

```r
phyper(q = 0:12, m =M, n = N, k = 12)
```

```
##  [1] 0.0001031992 0.0044375645 0.0521155831 0.2507739938 0.6083591331
##  [6] 0.8944272446 0.9897832817 1.0000000000 1.0000000000 1.0000000000
## [11] 1.0000000000 1.0000000000 1.0000000000
```

```r
qhyper(p = 0.5, m = M, n = N, k = 12)
```

```
## [1] 4
```

```r
datos <- rhyper(nn = 10000, m = M, n = N,  k=12)

hist(datos, breaks = 8)
```



Veamos un ejemplo de distribución Hipergeométrica en `Python`.

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

[M, n, N] = [20,7,12]

rv = hypergeom(M, n, N)
x = np.arange(0, n+1)
y = rv.pmf(x)

mean, var, skew, kurt = rv.stats(moments='mvsk')

print("Media %f"%mean)
```

```
## Media 4.200000
```

```
print("Varianza %f"%var)
```

```
## Varianza 1.149474
```

```
print("Sesgo %f"%skew)
```

```
## Sesgo -0.062181
```

```
print("Curtosis %f"%kurt)
```

```
## Curtosis -0.152661
```

```
fig = plt.figure()
ax = fig.add_subplot(111)
ax.plot(x,y,'bo')
ax.vlines(x, 0,y, lw =2, alpha = 0.4)
ax.set_ylabel('Distribucion de probabilidades de H(13,7,12)')
ax.set_xlabel('Numero de perros entre los doce elegidos al azar')
plt.show()
```