Distribución Binomial

Imanol

3/3/2021

Función de densidad

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

TODO: escribir la FDens y la FDistr

EN R

```r
library(Rlab)
```

```
## Rlab 2.15.1 attached.
```

```
##
## Attaching package: 'Rlab'
```

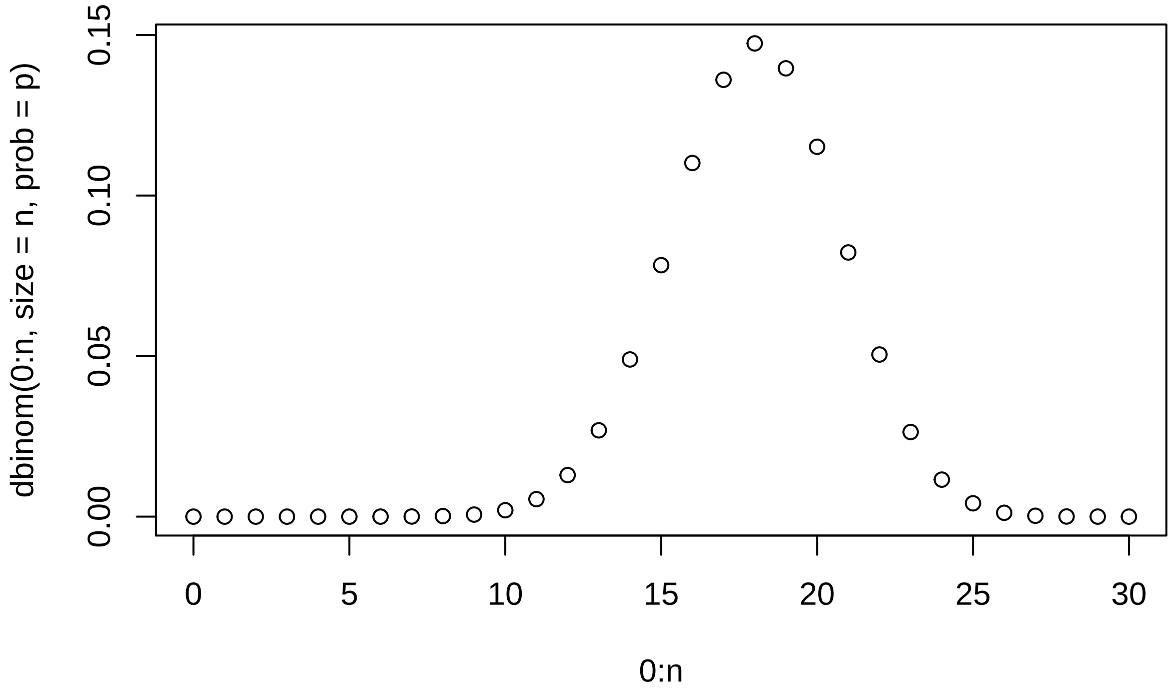
```
## The following objects are masked from 'package:stats':
##
##     dexp, dgamma, dweibull, pexp, pgamma, pweibull, qexp, qgamma,
##     qweibull, rexp, rgamma, rweibull
```

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

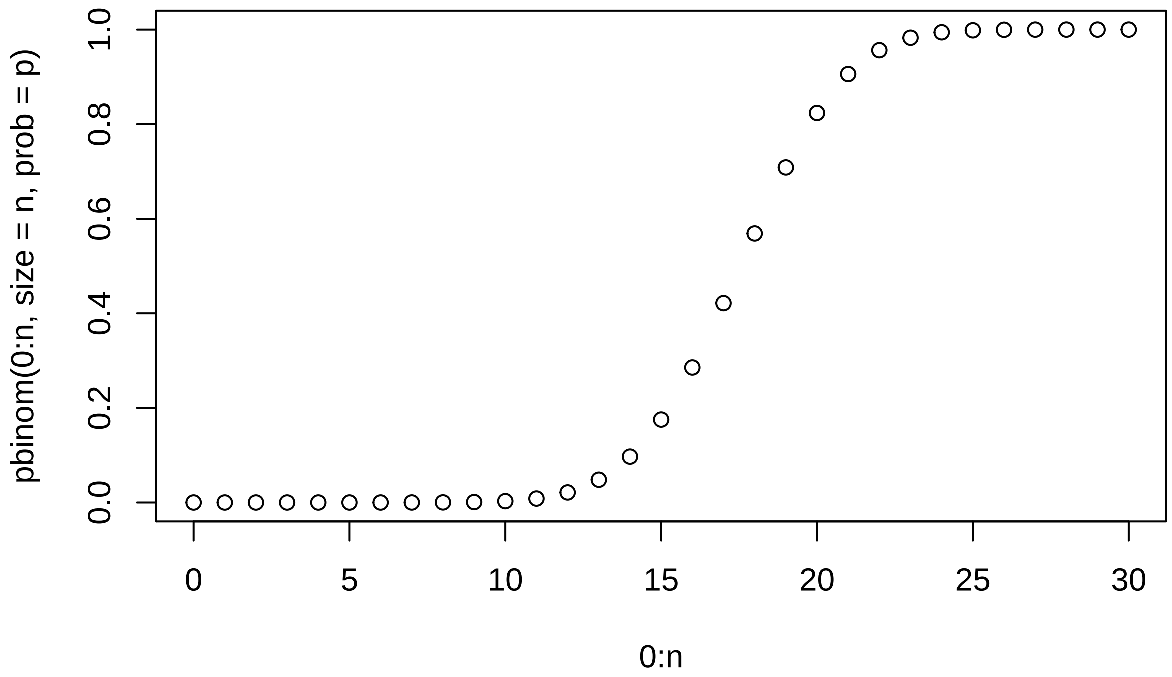
```r
n = 30
p = 0.6

dbinom(0:30, size = n, prob = p) # Cual seria la distribución de sacar
```

```
##  [1] 1.152922e-12 5.188147e-11 1.128422e-09 1.579791e-08 1.599538e-07
##  [6] 1.247640e-06 7.797748e-06 4.010270e-05 1.729429e-04 6.341240e-04
## [11] 1.997491e-03 5.447702e-03 1.293829e-02 2.687184e-02 4.894513e-02
## [16] 7.831221e-02 1.101265e-01 1.360387e-01 1.473752e-01 1.396186e-01
## [21] 1.151854e-01 8.227527e-02 5.048710e-02 2.634109e-02 1.152423e-02
## [26] 4.148722e-03 1.196747e-03 2.659437e-04 4.274096e-05 4.421478e-06
## [31] 2.210739e-07
```

```
                                 #entre 0 y 30 exitos, sabiendo que
                                 #utilizamos una muestra generada con tamaño
                                 #30 veces el del experimento y que
                                 #la probabilidad es 0.6

# Para verlo mejor hacer un plot
plot(0:n, dbinom(0:n, size = n, prob = p)) # Función de densidad de la binomial
```



```
plot(0:n, pbinom(0:n, size = n, prob = p)) # Función de probabilidad acumulada
```



```
# Calcular cuantiles
qbinom(0.5, n, p) # La mediana de la binomial de size n y probabilidad p
```

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

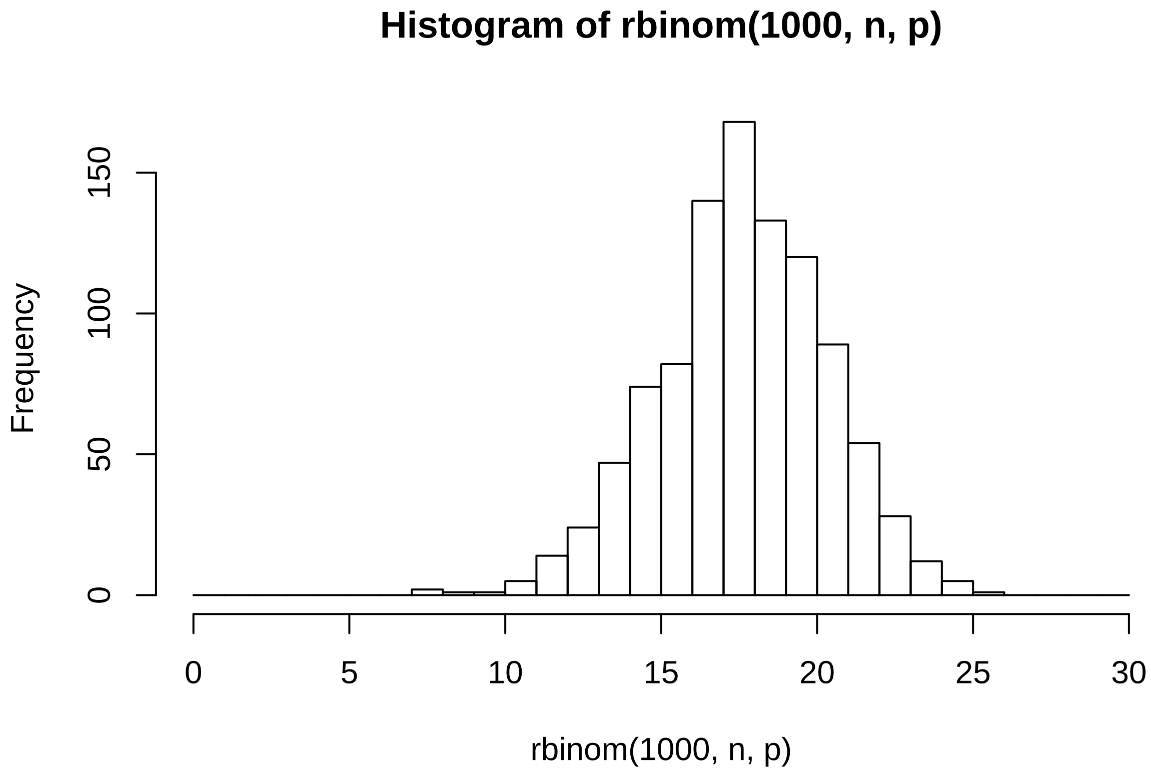
```
qbinom(0.25, n, p) # El Q1 de la binomial de size n y probabilidad p
```

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

```
# Generar numeros aleatorios binomiales
rbinom(1000, n, p) # Genera 1000 numeros que siguen una funcion de distribucion
```

```
##    [1] 22 16 17 18 14 21 13 20 21 17 14 19 16 14 17 17 16 18 18 17 17 17 20 18
##   [25] 17 21 18 22 16 20 23 19 21 20 19 16 20 21 20 16 11 13 18 15 17 18 20 16
##   [49] 14 13 17 17 18 16 18 19 19 20 16 22 17 14 17 17 20 11 23 16 17 16 15 16
##   [73] 17 22 20 18 14 18 16 16 14 21 20 17 19 15 24 24 20 23 21 18 19 23 17 16
##   [97] 18 19 20 16 15 17 20 17 18 15 21 19 18 19 20 17 15 19 20 17 19 24 21 21
##  [121] 19 21 20 18 21 18 20 16 20 19 21 20 19 16 16 15 21 21 16 19 16 16 20 23
##  [145] 16 15 20 17 18 15 12 17 18 18 17 20 19 22 21 15 17 21 17 20 19 19 18 16
##  [169] 18 17 16 23 15 15 20 17 18 15 19 19 20 20 20 16 14 18 20 18 17 19 18 20
##  [193] 14 17 19 19 18 18 21 19 17 20 21 14 15 17 19 18 14 16 22 20 18 20 18 18
##  [217] 17 18 16 17 17 18 22 20 20 14 21 23 22 19 16 18 21 17 18 22 14 18 19 17
##  [241] 15 21 18 20 19 19 20 20 20 16 20 17 21 18 22 21 15 19 16 19 16  8 18 22
##  [265] 20 16 21 20 19 18 19 19 20 18 17 14 17 18 19 19 18 20 14 25 18 19 19 13
##  [289] 18 23 24 21 22 19 17 22 18 19 18 19 18 22 20 19 20 18 20 21 15 18 14 21
##  [313] 22 22 18 18 15 21 14 13 17 20 22 21 17 13 20 22 19 20 14 19 16 10 19 21
##  [337] 16 15 18 18 21 20 15 19 18 20 18 22 18 19 18 19 15 17 17 21 18 21 19 18
##  [361] 13 19 16 21 19 17 18 18 20 16 22 20 20 19 10 22 18 24 20 19 20 16 14 15
##  [385] 17 18 19 19 20 18 18 21 17 17 15 18 18 11 19 24 19 15 17 18 20 16 17 18
##  [409] 21 19 22 19 17 17 21 13 19 14 17 19 17 19 21 17 20 12 22 16 15 18 20 16
##  [433] 18 15 23 19 18 20 20 18 19 22 23 15 16 18 19 21 18 16 16 22 18 17 19 15
##  [457] 16 24 20  9 21 20 17 20 12 19 22 18 17 17 14 19 19 16 23 19 19 20 19 24
##  [481] 19 16 19 14 17 20 21 22 15 15 18 18 18 17 13 20 17 19 23 15 21 20 16 18
##  [505] 14 20 18 18 15 18 17 22 16 21 18 15 16 14 21 20 13 16 22 16 15 22 19 17
##  [529] 18 19 21 17 18 14 18 15 17 16 14 18 13 18 18 15 18 17 17 14 20 18 18 15
##  [553] 16 19 18 19 14 19 18 17 23 17 18 15 19 21 17 10 14 15 22 19 21 22 18 21
##  [577] 20 23 17 14 21 17 20 22 19 16 14 16 21 21 14 18 22 17 18 18 17 24 18 22
##  [601] 16 22 14 16 16 21 15 21 18 17 20 17 18 14 22 18 17 18 21 23 14 17 15 19
##  [625] 16 17 20 17 16 17 15 17 18 18 21 17 19 20 17 23 20 22 20 15 22 22 17 18
##  [649] 19 21 16 13 17 20  9 17 22 17 18 16 19 16 17 22 19 19 17 14 16 18 25 17
##  [673] 12 20 16 18 21 16 18 17 18 20 16 18 20 15 20 19 16 18 22 17 15 23 17 16
##  [697] 18 16 18 15 17 20 18 19 18 18 21 15 22 19 14 23 16 23 17 12 19 14 19 21
##  [721] 19 22 18 20 16 15 20 22 18 22 17 13 17 18 16 20 20 20 20 16 20 18 17 18
##  [745] 18 19 21 15 17 18 15 22 20 16 16 17 18 20 21 17 19 18 17 20 16 20 20 19
##  [769] 21 13 16 15 15 17 16 20 21 20 19 22 24 13 21 19 15 14 20 19 19 18 18 20
##  [793] 17 17 17 20 18 21 19 20 17 16 17 13 21 22 16 18 17 21 16 19 18 19 17 18
##  [817] 20 21 22 22 13 18 20 22 18 17 22 17 16 16 21 16 21 21 18 19 19 20 16 17
##  [841] 19 15 16 19 24 18 13 22 16 18 15 22 20 19 12 20 14 18 19 17 19 21 16 15
##  [865] 19 18 14 16 13 19 17 17 17 20 19 13 15 15 19 21 14 19 21 23 18 14 22 19
##  [889] 19 17 21 17 18 18 19 20 21 15 16 16 20 22 16 16 13 22 17 19 16 17 17 23
##  [913] 21 18 15 19 18 17 19 20 19 21 19 17 20 20 16 21 18 14 17 21 20 21 21 13
##  [937] 17 20 13 23 18 21 20 17 13 10 21 14 19 20 16 20 18 18 15 16 15 18 18 17
##  [961] 14 17 20 16 20 18 16 15 22 15 18 14 22 21 21 20 19 18 19 18 17 19 14 19
##  [985] 19 17 18 15 19 17 14 16 19 21 19 19 20 19 21 23
```

```
                   #binomial
# Lo normale es hacer un histograma de estos numeros y no verlos tal cual
hist(rbinom(1000, n, p), breaks = 0:30)
```



En PYTHON

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

fig, ax = plt.subplots(1,1)
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
```

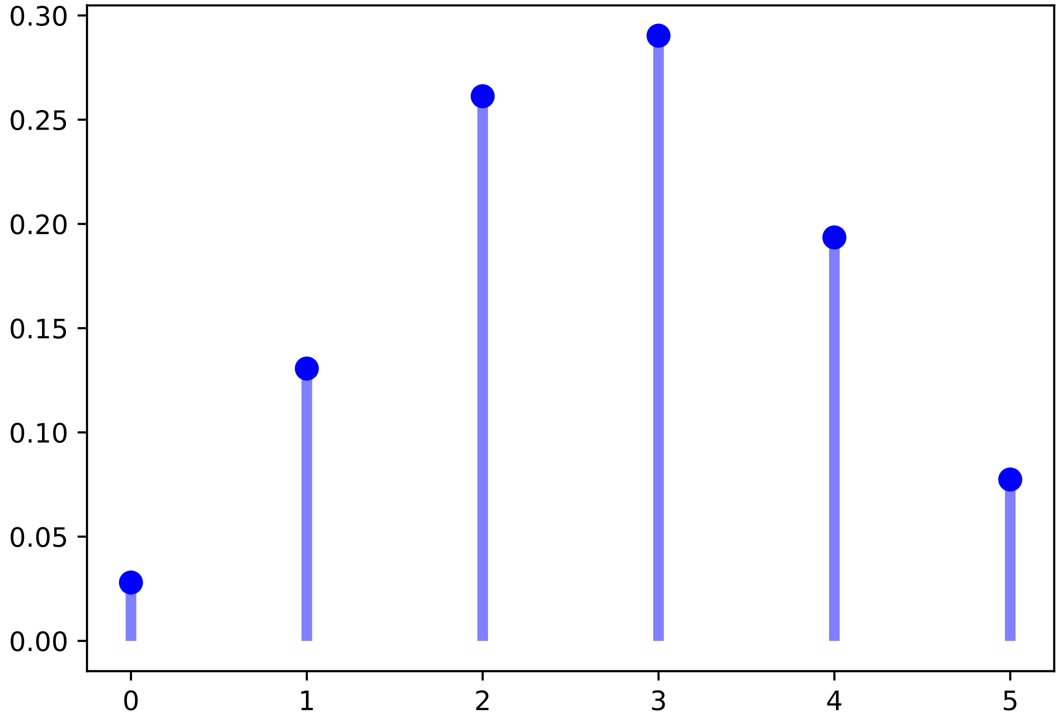
```
print("Sesgo %f"%skew)
```

```
## Sesgo 0.154303
```

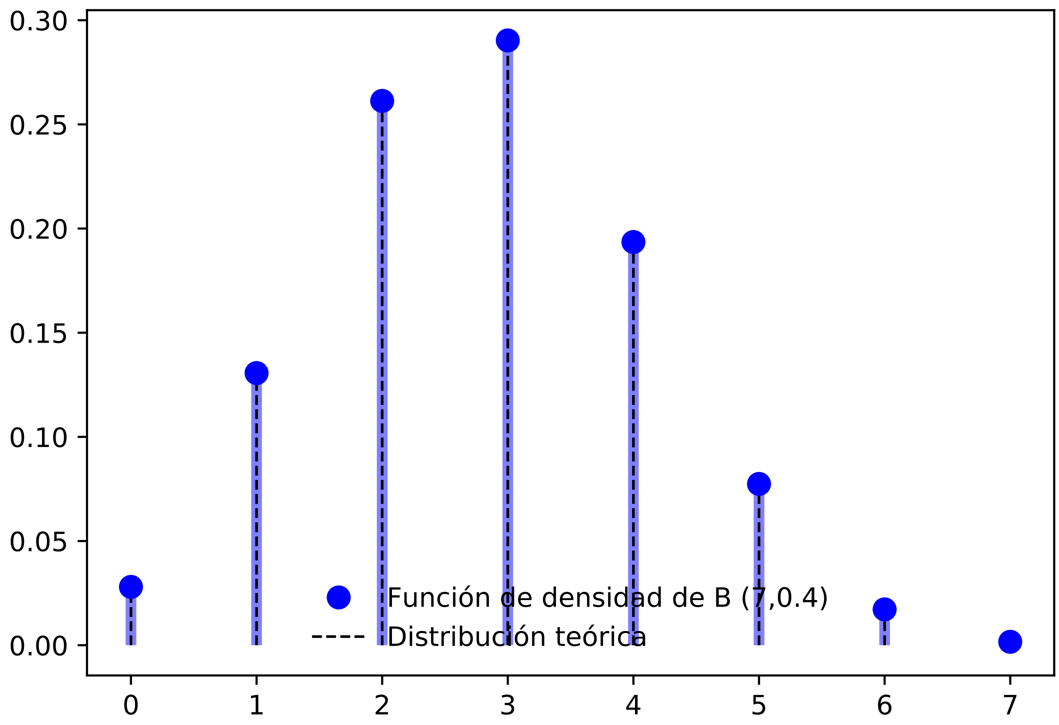
```
print("Curtosis %f"%kurt)
```

```
## Curtosis -0.261905
```

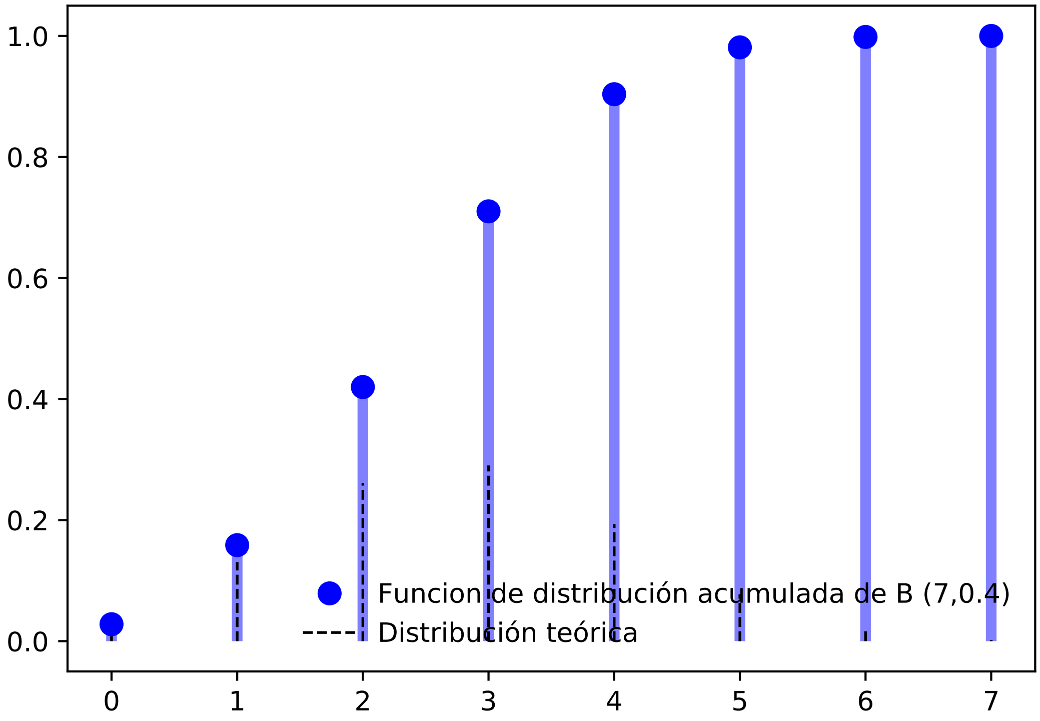
```
x = np.arange(binom.ppf(0.01, n, p), binom.ppf(0.99,n,p)) # Generar los valores de una binomial desde e
ax.plot(x, binom.pmf(x, n, p),’bo’, ms = 8, label = "Función de densidad de B (7,0.4)") # "bo" quiere d
ax.vlines(x, 0, binom.pmf(x,n,p), colors = ’b’, lw = 4, alpha = 0.5) # Le mete unas rayitas verticales
plt.show()


# Si tenemos claro el dominio mejor asi
```



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

# Le ponemos el extra de ver la binomial
rv = binom(n, p)
ax.vlines(x,0,rv.pmf(x), colors = ’k’, linestyles = ’--’, lw = 1, label = "Distribución teórica")
ax.legend(loc = ’best’, frameon = False)
plt.show()


# Par sacar la funcion de distribución acumulada
```



```
fig, ax = plt.subplots(1,1)
x = np.arange(0, n+1)
ax.plot(x, binom.cdf(x, n, p),'bo', ms = 8, label = "Funcion de distribución acumulada de B (7,0.4)")
ax.vlines(x, 0, binom.cdf(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 = "Distribución teórica")
ax.legend(loc = 'best', frameon = False)
plt.show()


# Generar numeros aleatorios binomiales
```



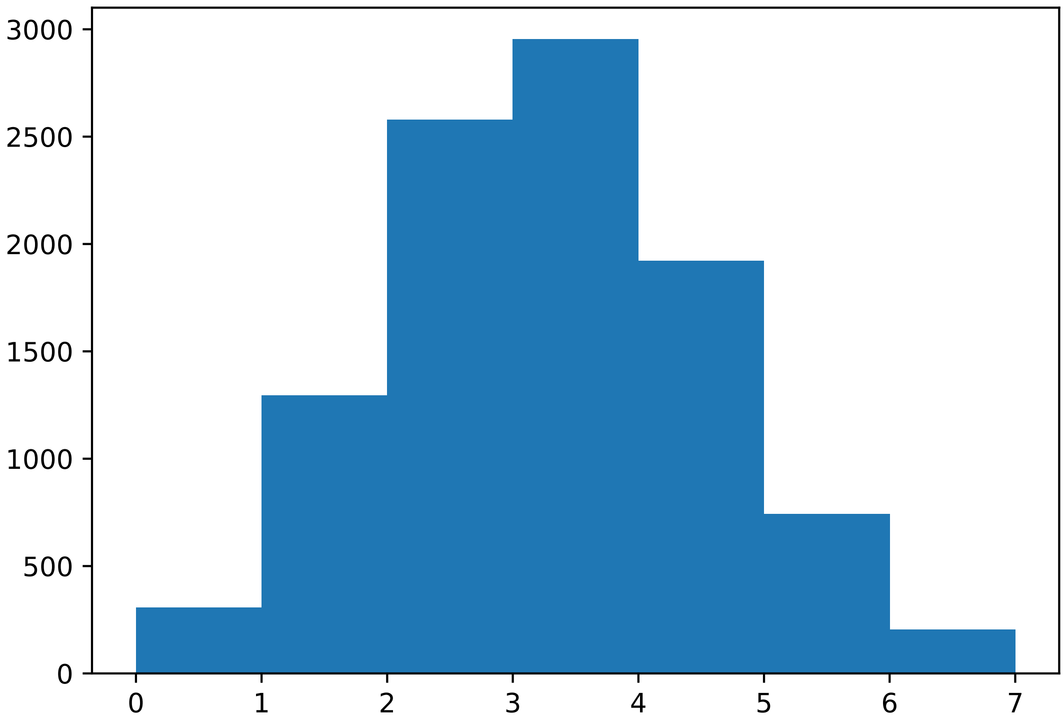
```
r = binom.rvs(n, p, size = 10) # Generar 10 valores aleatorios de repetir n
                               #veces la bernouilli con parametro p.
print(r)

# Histograma de estos valores aleatorios
```

```
## [3 3 1 4 1 4 2 2 3 4]
```

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

```
## (array([ 307., 1294., 2578., 2953., 1922.,  742.,  204.]), array([0., 1., 2., 3., 4., 5., 6., 7.]), <
```

```
plt.show()
```


3000
2500
2000
1500
1000
500
0
0
1
2
3
4
5
6
7