Algunos gráficos para pensar el ejercicio 3 de la guía 2

Abril 2026

La pregunta: ¿Cuántas veces debería tirar dos dados para que lo más probable sea ganar jugándole a que sale 10 veces?

Como discutimos en clase, la variable aleatoria X : "N° de veces que la suma de los dos dados es 7" tiene distribución binomial con probabilidad de éxito $p = \frac{1}{6}$. En la Figura 1 graficamos la función de probabilidad puntual (o de masa) para $B(k; n, 1/6) = \binom{n}{k}(\frac{1}{6})^k(\frac{5}{6})^{n-k}$ para n lanzamientos de los dados, con $59 \leq n \leq 65$.

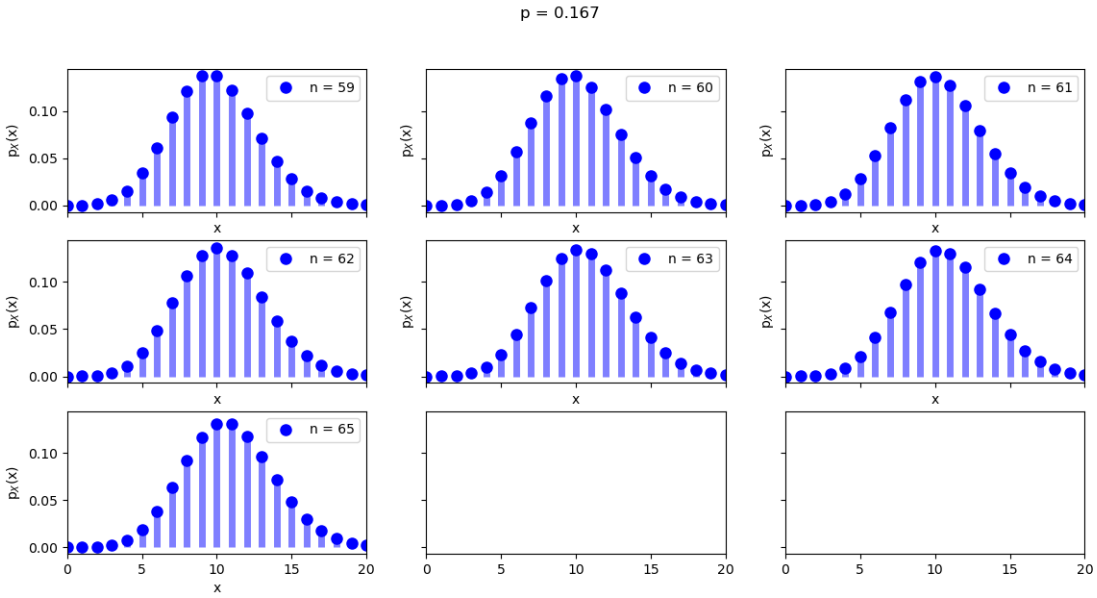

Figure 1. La distribución es bimodal para n = 59 y n = 65. Para estos se cumple que $(n + 1)p \in N$. Para $n \in \{60, 61, 62, 63, 64\}$ la distribución es unimodal. Entonces, para maximizar las chances de ganar apostando al diez, deberían tirar los dados entre 60 y 64 veces. Se ve muy simétrica, ¿no?

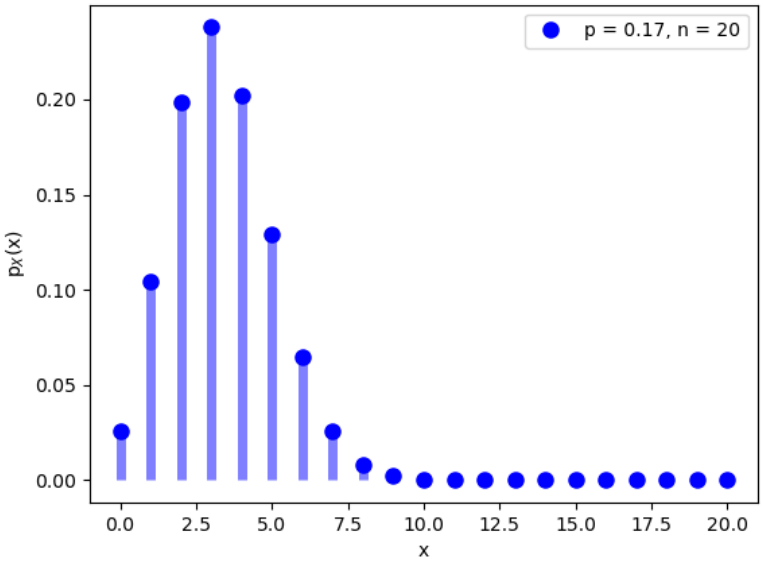

Figure 2. Para $p \neq 0.5$ y n pequeño, la distribución binomial no es simétrica.

El combinatorio es simétrico pues $\binom{n}{k} = \frac{n!}{n!(n-k)!} = \frac{n!}{(n-k)!n!} = \binom{n}{n-k}$. Se puede ver esto gráficamente en la Figura 3, donde representamos la función de masa para el caso en el que la probabilidad de éxito es igual a la de fracaso.

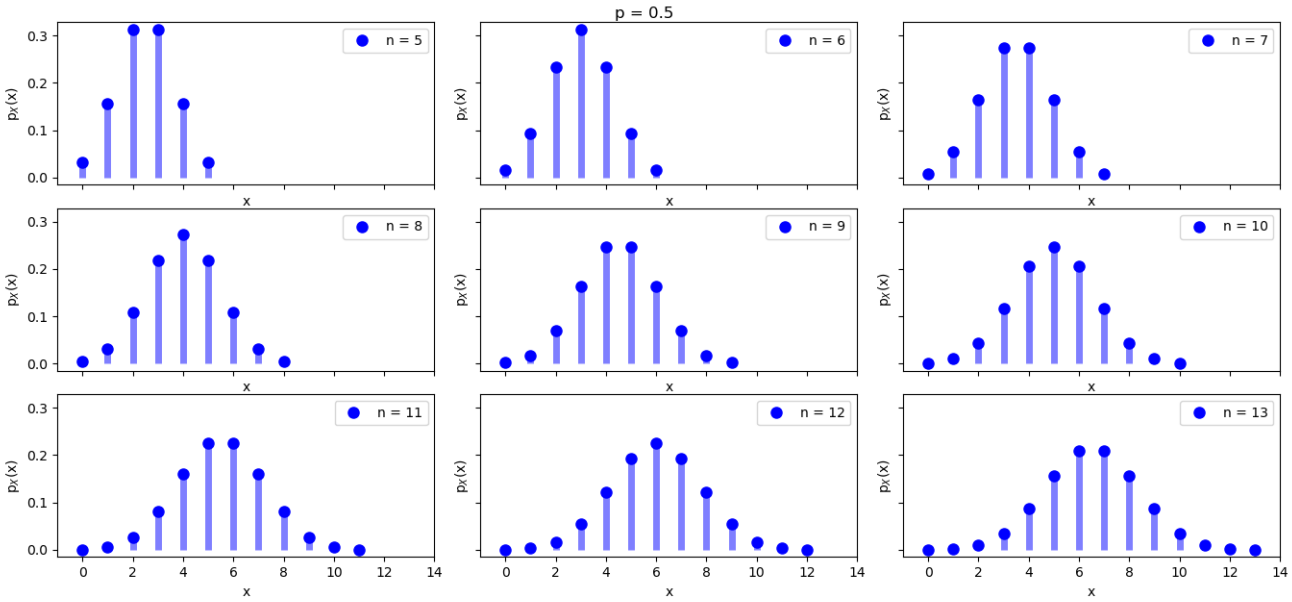

Figure 3. Para p = 0.5 se recupera la simetría del combinatorio. pues $B(k; n, 0.5) = \binom{n}{k}0.5^n = B(n-k; n, 0.5)$

Les dejamos el modulito PYthon que generó estos gráficos para que juegen con los parámatros a gusto

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

# Funciones ------------------------------------------------------------------
def graficar_ncreciente_p_fijo(p, nn, columnas, filas, fig1, ax1):

 print("len(nn)", len(nn))
 print("filas", filas)
 axs = ax1.flatten()
 print(axs)

 for i, ax in enumerate(axs):
   if i ==len(nn):
    break

   rva = binom(nn[i], p)
   Rxa = rva.support()
   xa = np.arange(Rxa[0], Rxa[1]+1)
   ax.plot(xa, rva.pmf(xa), 'bo', ms=8, label = f'n = {nn[i]}')#
   ax.vlines(xa, 0, rva.pmf(xa), colors='b', lw=5, alpha=0.5)
   #ax1[i][j].set_title(f'p = {p}, n = {nn[l]}')
   ax.legend(loc=1)

 fig1.suptitle(f'p = {round(p,3)}')

 #plt.tight_layout()

#-----------------------------------------------
#v.a. X = "cantidad de éxitos en n intentos "
#Defino los parámetros de mi binomial
n = 20
p = 1/6.0

#objeto congelado
rv = binom(n, p)
#Rango de X
Rx = rv.support()
print("Rango de la variable aleatoria = ", Rx)
#Esperanza y varianza
mean, var = rv.stats(moments='mv')
print('Esperanza = ', round(mean, 2), 'Varianza = ', round(var,2))

#Array de numpy de números enteros entre los percentiles 0.01 y 0.99 de la binomial
x = np.arange(Rx[0], Rx[1]+1)#Array del rango de X
#x = np.arange(rv.ppf(0.01), rv.ppf(0.999))#Usando pasos por defecto (1)
print("RX = ", x, "Tipo de datos: ", type(x),)
print("p_X(x) = ", rv.pmf(x))
print("Suma de p_X(x) sobre RX", round(sum(rv.pmf(x)),2))

#Grafico la función de probabilidad puntual o de masa de X
fig, ax = plt.subplots(1, 1)
```

```
#ax.plot(x, binom.pmf(x, n, p), 'bo', ms=8, label='binom pmf')
ax.plot(x, rv.pmf(x), 'bo', ms=8, label = f'p = {round(p, 2)}, n = {n}' )#, alpha=0.5
ax.vlines(x, 0, rv.pmf(x), colors='b', lw=5, alpha=0.5)
ax.set_ylabel('p$_{X}$(x)')
ax.set_xlabel('x')
plt.legend()

#Varias distribuciones binomiales con n creciente y p fijo
nn = np.arange(59, 66)
columnas = 3
filas = math.ceil(len(nn)/columnas)
fig1, ax1 = plt.subplots(filas, columnas, figsize=(15,10))
graficar_ncreciente_p_fijo(1/6, nn, filas, columnas, fig1, ax1)
fig2, ax2 = plt.subplots(filas, columnas, figsize=(15,10))
graficar_ncreciente_p_fijo(0.5, np.arange(5, 17), filas, columnas, fig2, ax2)
plt.tight_layout()
plt.show()
#Fin ----------------------------------------------------------------
```