Variables Aleatorias

Pablo L. De Nápoli

Departamento de Matemática
Facultad de Ciencias Exactas y Naturales
Universidad de Buenos Aires

Probabilidades y Estadística para Computación
Primer cuatrimestre de 2023

Herramientas computacionales

Existen varias herramientas computacionales que pueden utilizarse para programar algunas de las cosas que estuvimos viendo.

Algunas de las más populares son:

- Python con los módulos scipy.stats o statsmodels.

  Recomiendo usarlo con el entorno Jupyter.

- R

  Página: `https://www.r-project.org/`

  Recomiendo usarlo con el entorno R-studio.

En la página de la materia pueden encontrar un enlace para descargar el libro

Essential Statistics with Python and R.

Hoy vamos a ver como hacer algunas cosas en Python usando scipy.stats.

Variables discretas

Recordamos que la distribución de una variable discreta X con valores (x_k) se especifica mediante las probabilidades puntuales

$$p_k = P\{X = x_k\}$$

Definiendo una disstribución discreta

```
import scipy.stats
xk = (1, 2, 3)
pk = (0.5, 0.25, 0.25)
distribucion = scipy.stats.rv_discrete(values=(xk, pk))
```

Esperanzas en la computadora

Podemos calcular distintas caracerísticas de la distribución:

Esperanza, variancia, desviación estándar de un dado

```
import numpy as np
import scipy.stats
xk = (1, 2, 3, 4, 5, 6)
pk = (1 / 6, 1 / 6, 1 / 6, 1 / 6, 1 / 6, 1 / 6)
distribucion = scipy.stats.rv_discrete(values=(xk, pk))
print("esperanza=", distribucion.mean())
print("varianza=", distribucion.var())
print("desviación estándar=", distribucion.std())
```

Salida del programa

```
esperanza= 3.5
varianza= 2.9166666666666666
desviación estándar= 1.707825127659933
```

Simulando una variable aleatoria

Tiramos un dado 100 veces

```
# continuamos el programita anterior
cuantas_veces = 100
frecuencia = np.zeros(7, dtype=int)

# np.random.seed(12345)

for j in range(0, cuantas_veces):
    dado = distribucion.rvs()
    frecuencia[dado] += 1

s = 0
for k in range(1, 7):
    s = s + k * frecuencia[k]

# continua
```

Resultados de la simulación

Salida del programa

```
cuantas_veces= 100

valor  frecuencia  frecuencia relativa
1    10   0.1
2    17   0.17
3    15   0.15
4    17   0.17
5    22   0.22
6    19   0.19

Media muestral= 3.81

esperanza= 3.5
```

Ahora probamos 10.000 veces

Salida del programa

```
cuantas_veces= 10000

valor  frecuencia  frecuencia relativa
1    1652    0.1652
2    1677    0.1677
3    1649    0.1649
4    1611    0.1611
5    1692    0.1692
6    1719    0.1719

Media muestral= 3.5171

esperanza= 3.5
```

La distribución binomial

Las distribuciones conocidas que estuvimos viendo en el curso, ya están implementadas en scipy.stats.

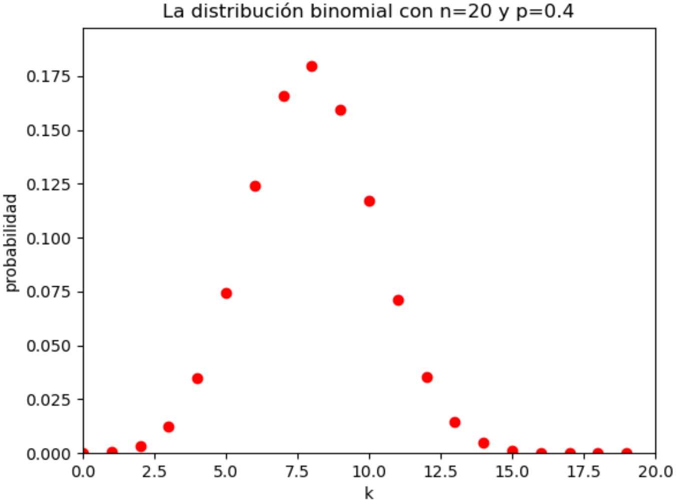
Graficamos la distribución binomial

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

n = 20
p = 0.4
distribucion = scipy.stats.binom(n, p)
x = np.arange(n)
y = distribucion.pmf(x)

plt.axis([0, n, 0, np.max(y) * 1.1])
plt.plot(x, y, 'ro')
plt.xlabel("k")
plt.ylabel("probabilidad")
plt.show()
```

Graficamos la distribución binomial



Ahora vamos a trabajar con variables continuas

Supongamos que tenemos una distribución de probabilidades continua con densidad $f(x)$ y función de distribución acumulada

$$F(x) = \int_{-\infty}^{x} f(t)\, dt$$

Si X es una variable aleatoria con esta distribución

$$F(x) = P\{X \leq x\}$$

y si $I = (a, b]$ es un intervalo semiabierto

$$P\{X \in I\} = F(b) - F(a) = \int_a^b f(x)\, dx$$

Supondremos hoy (para evitar complicaciones técnicas) que $f(x)$ es continua y positiva. Con lo que F será estrictamente creciente y $F' = f$.

Parámetros de Posición

Podemos considerar tres parámetros de posición diferentes:

- La esperanza, media o valor medio:

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

- La mediana es aquel valor de x_0 para el cual

$$P\{X \leq x_0\} = P\{X > x_0\} = \frac{1}{2}$$

  es decir mediana$(X) = F^{-1}(1/2)$.

- La moda es aquél valor de x para el cuál $f(x)$ es máximo. (podría haber más de uno, se distingue entonces entre distribuciones unimodales o multimodales).

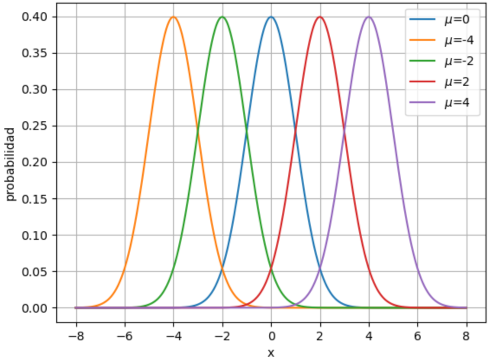
Un ejemplo: la distribución normal

Considermaos como un primer ejemplo, la distribución normal $N(\mu, \sigma^2)$:

$$f_X(x) = \frac{1}{\sigma\sqrt{2\pi}} \ e^{-(x-\mu)^2/(2\sigma^2)}$$

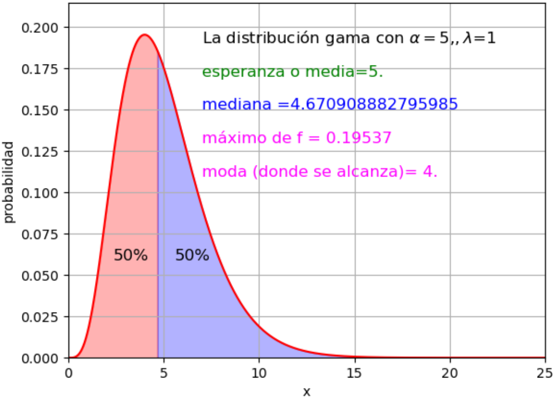
donde $\mu, \sigma \in \mathbb{R}$ y $\sigma > 0$. En este caso

$$E[X] = \text{mediana}(X) = \text{moda}(X) = \mu$$



Otro ejemplo: las densidades gama $\Gamma(\alpha, \lambda)$ (que introdujimos la clase pasada)

$$f_{\alpha,\lambda}(x) = \frac{\lambda^\alpha}{\Gamma(\alpha)}\, x^{\alpha-1}\, e^{-\lambda x} I_{(0,+\infty)}(x), \quad E(X) = \frac{\alpha}{\lambda}$$



Cálculo de la moda de las distribuciones gamma $\Gamma(\alpha, \lambda)$

Si $x > 0$

$$\begin{aligned} f'_{\alpha,\lambda}(x) &= \frac{\lambda^\alpha}{\Gamma(\alpha)} \left[(\alpha-1)x^{\alpha-2}\, e^{-\lambda x} + x^{\alpha-1}\,(-\lambda)e^{-\lambda x}\right] \\ &= \frac{\lambda^\alpha}{\Gamma(\alpha)} \left[(\alpha-1) - \lambda x\right] x^{\alpha-2} e^{-\lambda x} \end{aligned}$$

que sólo se anula en

$$x = \frac{\alpha - 1}{\lambda}$$

Entonces vemos que la distribución $\Gamma(\alpha, \lambda)$ es unimodal, y su moda es

$$moda(X) = \frac{\alpha - 1}{\lambda}$$

Vemos que en general, para las distribuciones gama, la moda y la esperanza no coinciden.

¡En la computadora!

Cómo calculé los parámetros de posición en Python 3

```
import numpy as np
import scipy.stats
import matplotlib.pyplot as plt
alpha = 5
lambda_ = 1
distribucion = scipy.stats.gamma(a=alpha, scale=1 / lambda_)
media = distribucion.mean()
mediana = distribucion.median()
x = np.arange(start=0, stop=25, step=0.01)
y = distribucion.pdf(x)
plt.plot(x,y)
y_maximo = max(y)
maximo_donde = np.where(y == y_maximo)[0][0]
moda = x[maximo_donde]
```

https://bitbucket.org/pdenapo/programitas-proba/src/main/gama_pdf.py

La varianza en la distribución normal

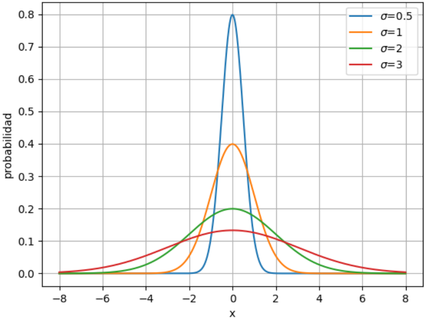
En cambio, la varianza

$$\mathrm{Var}(X) = E[(X-\mu_X)^2] = \int_{-\infty}^{\infty} (x-\mu_X)^2 \; f(x)\, dx \text{ donde } \mu_X = E[X]$$

y la deviacion estándar

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

son parámetros de dispersión. Veamos por ejemplo la distribución normal



Sobre la distribución normal

- Podemos accder a la distribución normal con normal=scipy.stats.norm(loc=mu,scale=sigma)

La normal y sus parámetros en Python 3

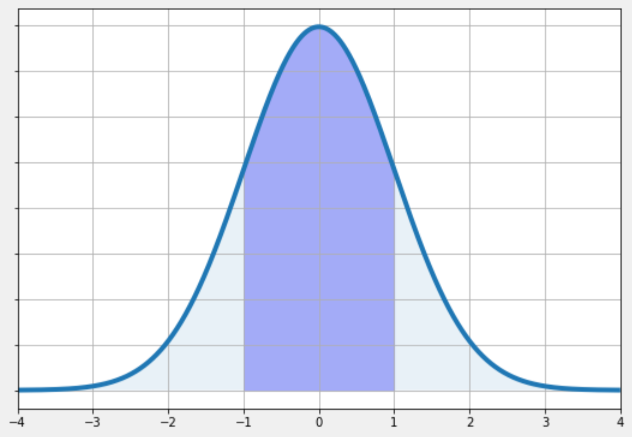
```
>>> normal=scipy.stats.norm(loc=1,scale=2)
>>> normal.mean()
1.0
>>> normal.var()
4.0
>>> normal.std()
2.0
```

- Entonces tenemos varios métodos que podemos usar:
  - La distribución acumulada f (Probability density function)) : normal.pdf
  - La distribución acumulada F (Cumulative distribution function): normal.cdf
  - La inversa de F (Percent point function): normal.ppf
- Notemos que si $X \sim N(\mu, \sigma^2)$, entonces $X^* = \frac{X-\mu}{\sigma} \sim N(0,1)$.

Interpretación de la variancia en la distribución normal

$$
\begin{aligned}
P\{\mu - \sigma \leq X \leq \mu + \sigma\} &= P\{-1 \leq X^* \leq 1\} \\
&= F_{X^*}(1) - F_{X^*}(-1) \approx 0{,}6826894921370859
\end{aligned}
$$

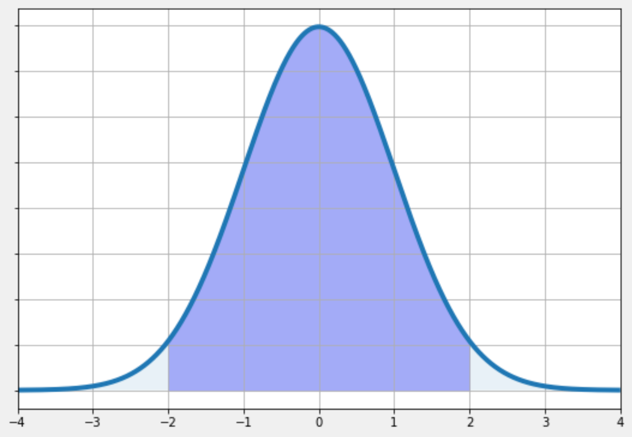
Es decir que (aproximadamente) el $68,26\,\%$ de las obsevaciones cae en esta región.



Interpretación de la varianza en la distribución normal (2)

$$P\{\mu - 2\sigma \leq X \leq \mu + 2\sigma\} = P\{-2 \leq X^* \leq 2\}$$
$$= F_{X^*}(2) - F_{X^*}(-2) \approx 0{,}9544997361036416$$

Es decir que (aproximadamente) el 95,44 % de las obsevaciones cae en esta región.



Percentiles y Cuartiles

Dado k con $0 \leq 100$ los percentiles de la distribución

$$P\{X \leq P_k\} = \frac{k}{100} \text{ o sea } P_k = F^{-1}\left(\frac{k}{100}\right)$$

Tenemos también los cuartiles

$$Q_1 = P_{25} = F^{-1}(0{,}25)$$

$$Q_2 = P_{50} = \text{mediana} = F^{-1}(0{,}5)$$

$$Q_3 = P_{75} = F^{-1}(0{,}75)$$

$$\text{iqr} = Q_3 - Q_1$$

Se denomina rango intercuartílico o rango intercuartil. Es otro parámetro de dispersión.

Ejemplo: Calculemos el rango intercuartílico para una distribución normal (1)

Supongamos que $X \sim N(\mu, \sigma^2)$. Recordamos que $X^* = \frac{X-\mu}{\sigma} \sim N(0,1)$
Luego para encontrar los cuartiles, buscamos los de X^*.

$$P\{X^* \leq Q_1^*\} = 1/4, \text{ o sea } Q_1^* = F_{X^*}^{-1}(0{,}25)$$

$$P\{X^* \leq Q_3^*\} = 1/4, \text{ o sea } Q_3^* = F_{X^*}^{-1}(0{,}75)$$

Luego serán

$$Q_1 = \mu + \sigma Q_1^*, Q_3 = \mu + \sigma Q_3^*$$

entonces

$$\text{iqr}(X) = \sigma(Q_3^* - Q_1^*) = \text{iqr}(X^*) \cdot \sigma = k \cdot \sigma$$

donde

$$k \approx 1{,}3489795003921634$$

Ejemplo: Calculemos el rango intercuartílico para una distribución normal (2)

Cómo calculé el rango intercuartílico de una normal estándar

```
from scipy.stats import norm
q1 = norm.ppf(0.25)
q3 = norm.ppf(0.75)
iqr = q3 - q1
print(q1)
print(q3)
print(iqr)
```

Salida

```
-0.6744897501960817
0.6744897501960817
1.3489795003921634
```

Aquí ppf = Percent point function (inversa de la función de distribución), es un método que permite calcular los percentiles de una distribución.

Interpretación gráfica

Notamos que $Q_1^* = -Q_3^*$ por la simetría de la curva normal.