Tarea 11

fl.gomez10 at uniandes.edu.co

23 de abril de 2019

Horario de atención: Principalmente de 2:00pm a 5:00pm en la oficina i-109. También se pueden enviar dudas al correo electrónico. Entregar antes de finalizar la clase.

Trabaje iniciando sesión en la máquina virtual en línea mybinder.org/ [1].

1. Ejercicio 1 (20 puntos) Trabajo en Casa - Generar y graficar una distribución de datos Gaussiana (o Normal)

Vamos a estudiar cómo el tamaño de una muestra afecta lo que vemos de una población.

En un notebook llamado `hw11.ipynb` genere un conjunto de datos llamado "data" de n=10 datos distribuidos aleatoriamente según la distribución normal (`np.random.norm(loc=3.0, scale=1.5, size=n)`), centrados en 3.0, con un ancho de 1.5.

- Generar un histograma de los datos con 10, otro con 20 y otro con 30 columnas usando la opción (bins=10) de la función `plt.hist()`.
- calcular la media de "data" usando `np.mean(data)`
- calcular la mediana de "data" usando `np.median(data)`
- En una celda responda lo siguiente ¿Son cercanas la media y la mediana?
- ¿Esperaría que fueran iguales la media y la mediana? ¿Por qué?

2. Ejercicio 2 (20 puntos) Trabajo en Casa - Más gaussianas.

Repita para n = 100, 1000, 10000, 100000

[1] https://mybinder.org/v2/gh/ComputoCienciasUniandes/FISI2026-201910/master?urlpath=lab

3. Ejercicio 3 (20 puntos) Ajuste de curvas

Vamos a trabajar con este conjunto de datos `some_statistical_stuff.dat`[2]
Escriba esta función de distribución de probabilidad normalizada:

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

Utilizando `scipy.optimize.curve_fit` busque los parámetros óptimos y grafique la función con los parámetros de mejor ajuste.

4. Ejercicio 4 (40 puntos) Ajuste de curvas

Escriba esta función de distribución de probabilidad normalizada:

$$g(k, \lambda) = \frac{\lambda^k e^{-\lambda}}{k!} \tag{2}$$

Esta función acepta sólamente un argumento de entrada entero k. Así que se puede utilizar `bins[:-1]` como k. Puede ser útil `scipy.special.factorial`.

Utilizando `scipy.optimize.curve_fit` busque los parámetros óptimos y grafique la función con los parámetros de mejor ajuste.

[2] https://raw.githubusercontent.com/ComputoCienciasUniandes/FISI2026-201910/master/Talleres/Grupo_1/some_statistical_stuff.dat