Tecnológico de Monterrey

Instituto Tecnológico y de Estudios Superiores de Monterrey

Algunas distribuciones de probabilidad

TC3004B.104 Inteligencia Artificial Avanzada para la Ciencia de Datos I

Profesores:

Ivan Mauricio Amaya Contreras

Blanca Rosa Ruiz Hernandez

Antonio Carlos Bento

Frumencio Olivas Alvarez

Hugo Terashima Marín

Julian Lawrence Gil Soares - A00832272

16 de Agosto de 2023

In [ ]:
```
import numpy as np
import matplotlib.pyplot as plt
from math import gamma
import math

def normal():
    miu = 10
    sigma = 2
    x = np.arange(miu - 4 * sigma, miu + 4 * sigma, 0.01)
    y = (1 / (sigma * np.sqrt(2 * np.pi))) * np.exp(-0.5 * ((x - miu) / sigma)**2)

    plt.plot(x, y, color='blue')
    plt.title('Distribución Normal (10, 2)')
    plt.xlabel('Valores')
    plt.ylabel('Densidad de Probabilidad')
    plt.grid(True)
    plt.show()

def t():
    df = 12

    x = np.linspace(-5, 5, 500)

    pdf_values = (gamma((df + 1) / 2) / (np.sqrt(df * np.pi) * gamma(df / 2))) * \
                 (1 + (x**2) / df)**(-((df + 1) / 2))

    plt.plot(x, pdf_values, color='blue')
    plt.title(f'T (df={df})')
    plt.xlabel('Valores')
    plt.ylabel('Densidad de Probabilidad')
    plt.grid(True)
    plt.show()

def chi():
    df = 8
    x = np.linspace(0, 30, 500)

    pdf_values = (1 / (2 ** (df / 2))) * (x ** ((df / 2) - 1)) * (np.exp(-x / 2)) /

    plt.plot(x, pdf_values, color='blue')
    plt.title(f'Distribución Chi-cuadrada (df={df})')
    plt.xlabel('Valores')
    plt.ylabel('Densidad de Probabilidad')
    plt.grid(True)
    plt.show()

def f():
    v1 = 9
    v2 = 13

    x = np.linspace(0, 5, 500)

    pdf_values = ((gamma((v1 + v2) / 2) * (v1 / v2)**(v1 / 2) * x**(v1 / 2 - 1)) /
                  (gamma(v1 / 2) * gamma(v2 / 2) * (1 + (v1 / v2) * x)**((v1 + v2) /
```

```
    plt.plot(x, pdf_values, color='blue')
    plt.title(f'Distribución F (v1={v1}, v2={v2})')
    plt.xlabel('Valores')
    plt.ylabel('Densidad de Probabilidad')
    plt.grid(True)
    plt.show()

def z():
    print("\nQ5\n")

    prob_a = 1 - 0.5 * (1 + math.erf(0.7 / math.sqrt(2)))
    print("P(Z > 0.7) = ",prob_a)

    prob_b = 0.5 * (1 + math.erf(0.7 / math.sqrt(2)))
    print("P(Z < 0.7) = " , prob_b)

    prob_c = math.exp(-0.7**2 / 2) / math.sqrt(2 * math.pi)
    print(f"P(Z = 0.7) = " , prob_c)

def left():
    print("\nQ6\n")
    df = 0.45

    print("df =", df , "area = " , math.sqrt(2) * math.erf(-1 + 2 * df))



def norm(x):
    mean = 100
    stDev = 7

    z = (x - mean) / stDev
    return 0.5 * (1 + math.erf(z / math.sqrt(2)))

prob_a = norm(87)
prob_b = 1 - norm(87)
prob_c = norm(110) - norm(87)

print("\nQ7\n")
print("P(X < 87) = " , prob_a)
print("P(X > 87) = " , prob_b)
print("P(87 < X < 110) = " , prob_c)

def t_student_cdf(t):
        return 0.5 * (1 + math.erf(t / math.sqrt(2)))

print("\nQ8\n")

df = 10
t_score_a = (0.5 - 0) / math.sqrt((df / (df - 2)))
prob_a = t_student_cdf(t_score_a)
print("P(X < 0.5) =" , prob_a)

t_score_b = (1.5 - 0) / math.sqrt((df / (df - 2)))
prob_b = 1 - t_student_cdf(t_score_b)
```

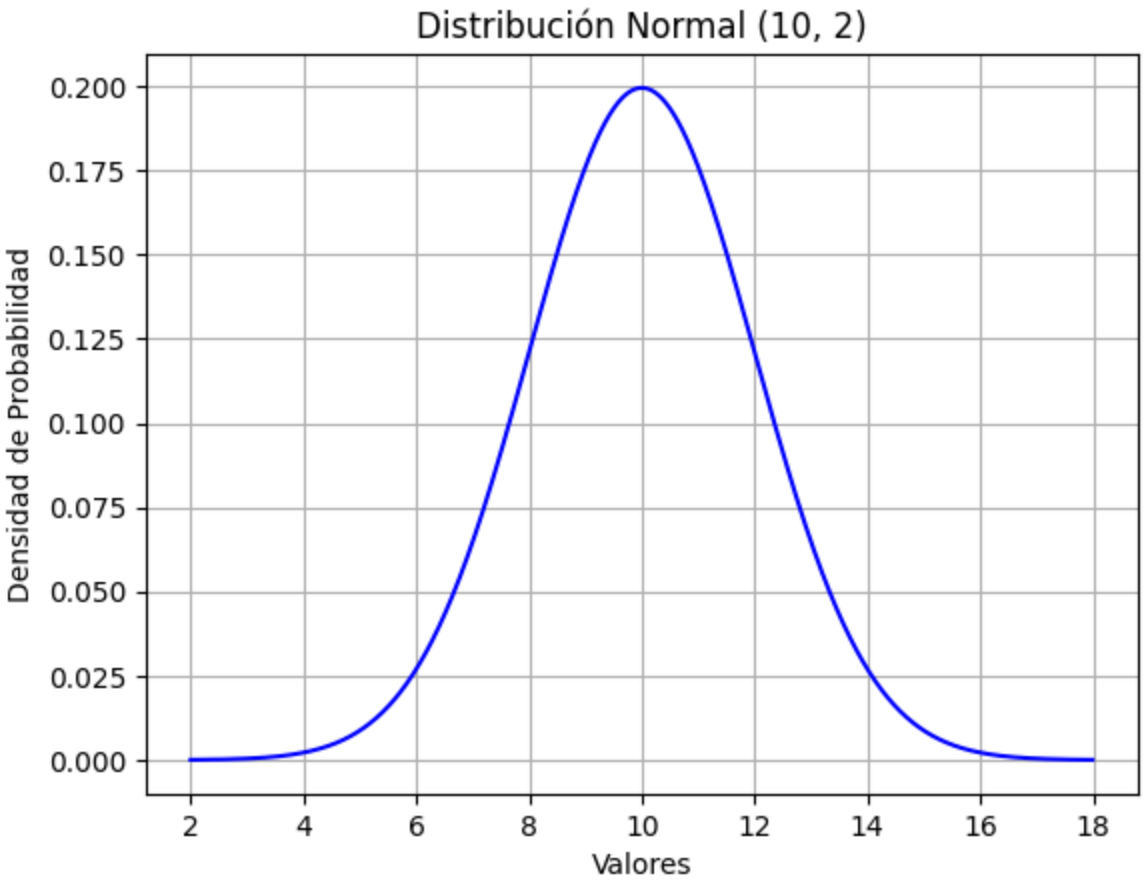
```
print("P(X > 1.5) =", prob_b)

def calc(mean, std_dev, threshold):
    z_score = (threshold - mean) / std_dev
    proportion = 0.5 * (1 + math.erf(z_score / math.sqrt(2)))
    return proportion

mean = 65
std_dev = 20
threshold = 60

percent = round((calc(mean, std_dev, threshold) * 100), 2)

print("Porcentage: " ,percent)
```

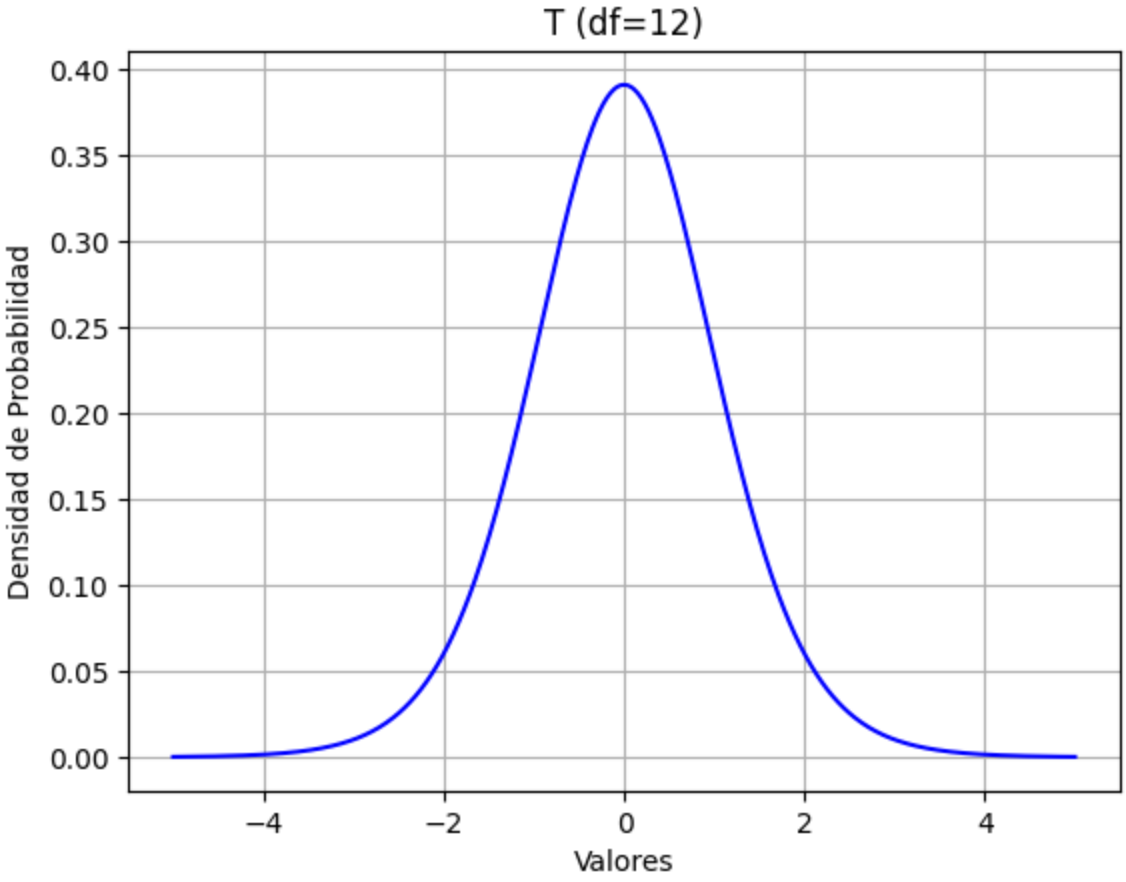
Q7

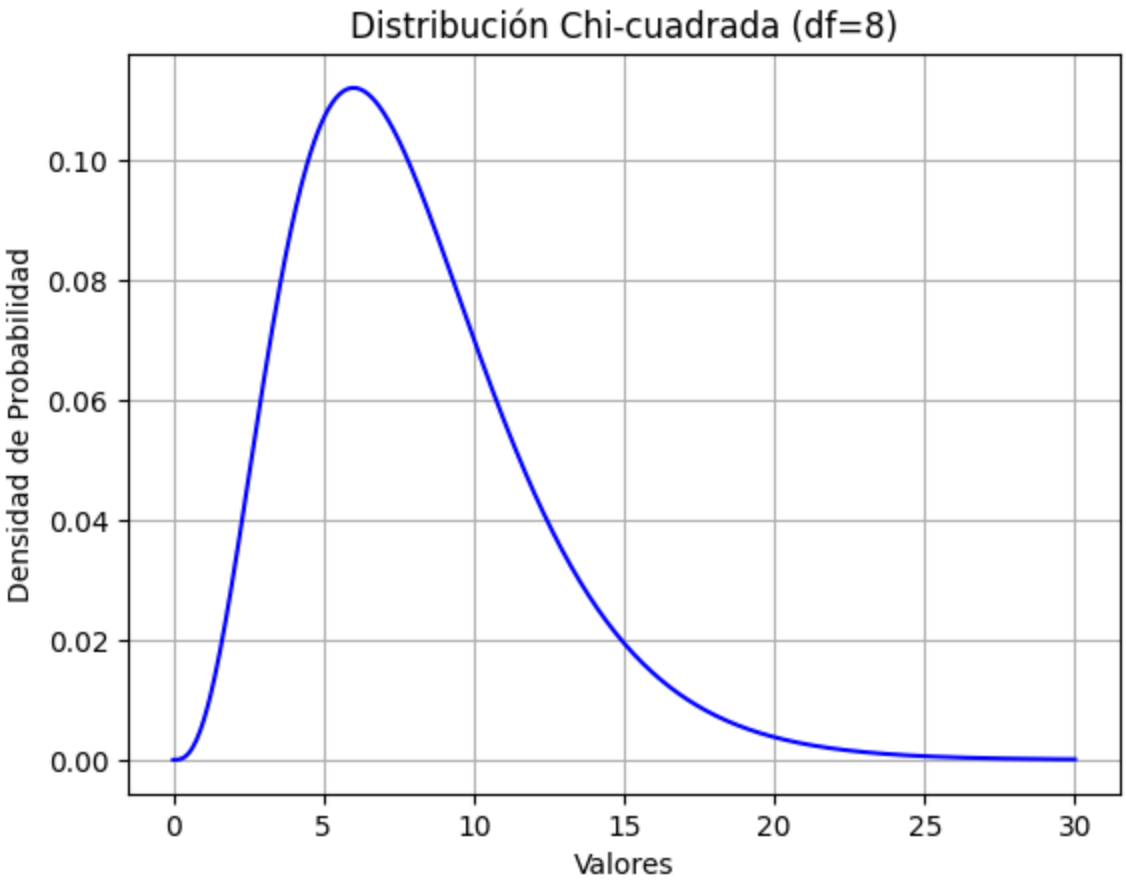
```
P(X < 87) =  0.03164541611667265
P(X > 87) =  0.9683545838833274
P(87 < X < 110) =  0.8917908583734926
```

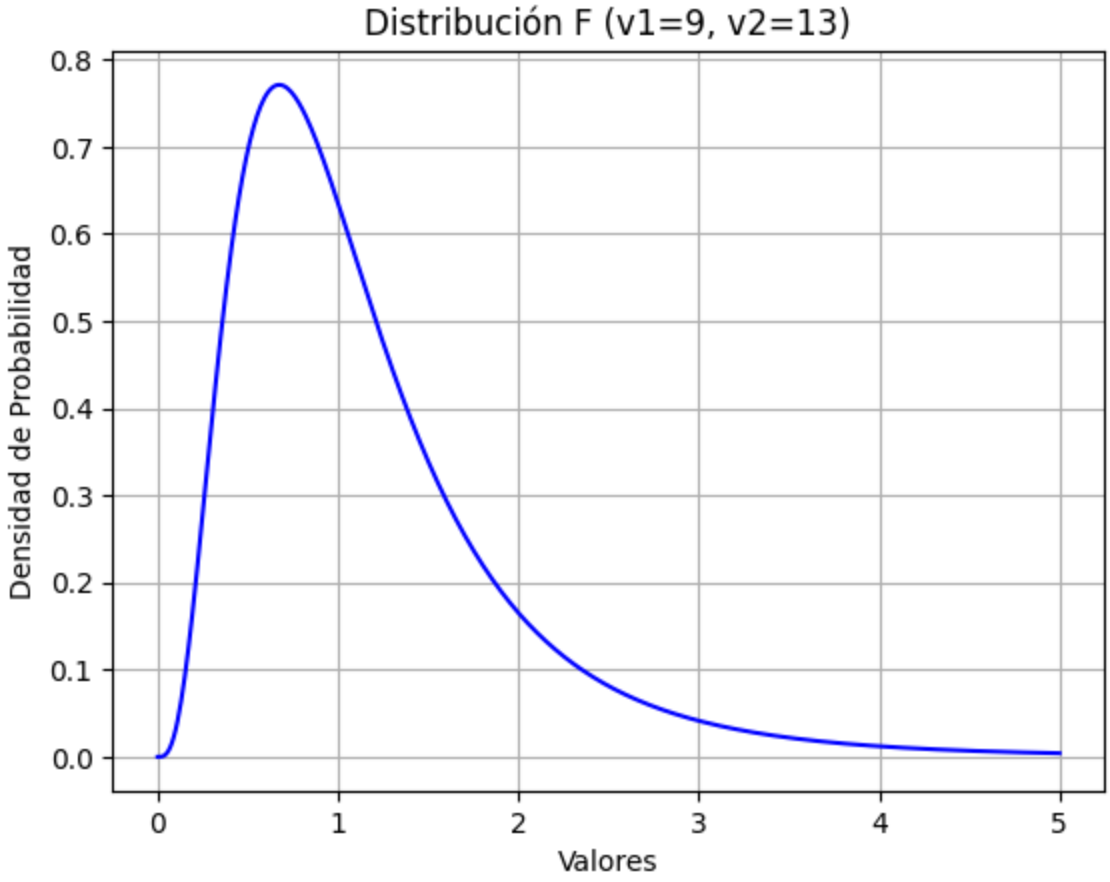
Q8

```
P(X < 0.5) = 0.6726395769907114
P(X > 1.5) = 0.08985624743949994
```




T (df=12)
Densidad de Probabilidad
0.40
0.35
0.30
0.25
0.20
0.15
0.10
0.05
0.00
−4
−2
0
2
4
Valores


Distribución Chi-cuadrada (df=8)
Densidad de Probabilidad
0.10
0.08
0.06
0.04
0.02
0.00
0
5
10
15
20
25
30
Valores



Q5

P(Z > 0.7) = 0.24196365222307303
P(Z < 0.7) = 0.758036347776927
P(Z = 0.7) = 0.31225393336676127

Q6

df = 0.45 area = -0.15904658109708486
Porcentage: 40.13