Національний технічний університет України

«Київський політехнічний інститут імені Ігоря Сікорського»

Факультет інформатики та обчислювальної техніки

Кафедра обчислювальної техніки

Методи оптимізації та планування експерименту

Лабораторна робота №3

“ ПРОВЕДЕННЯ ТРЬОХФАКТОРНОГО ЕКСПЕРИМЕНТУ З ВИКОРИСТАННЯМ ЛІНІЙНОГО РІВНЯННЯ РЕГРЕСІЇ ”

Виконав:

студент групи ІВ-83

Верьовочкін І. С.

Перевірив:

ас. Регіда П.Г.

Київ 2020 р.

**Мета:** провести двофакторний експеримент, перевірити однорідність дисперсії за критерієм Романовського, отримати коефіцієнти рівняння регресії, провести

натуралізацію рівняння регресії.

Номер у списку: 4

Віаріант: 304



Лістинг програми:

1. import numpy as np  
     
   x1\_min = 15  
   x1\_max = 45  
   x2\_min = 30  
   x2\_max = 80  
   x3\_min = 15  
   x3\_max = 45  
     
   x\_average\_max = (x1\_max + x2\_max + x3\_max) / 3  
   x\_average\_min = (x1\_min + x2\_min + x3\_min) / 3  
   y\_max = 200 + x\_average\_max  
   y\_min = 200 + x\_average\_min  
     
   m, n = 3, 4  
     
     
   def main(m, n):  
    print("\nМатриця кодованих значень:")  
    norm\_x = np.array([  
    [+1, -1, -1, -1],  
    [+1, -1, +1, +1],  
    [+1, +1, -1, +1],  
    [+1, +1, +1, -1]  
    ])  
    for i in range(len(norm\_x)):  
    print("{}.".format(i + 1), end="")  
    for j in range(len(norm\_x[i])):  
    print("{:4}".format(norm\_x[i][j]), end="")  
    print()  
     
    print("\nМатриця X:")  
    x = np.array([  
    [x1\_min, x2\_min, x3\_min],  
    [x1\_min, x2\_max, x3\_max],  
    [x1\_max, x2\_min, x3\_max],  
    [x1\_max, x2\_max, x3\_min]  
    ])  
     
    for i in range(len(x)):  
    print("{}.".format(i + 1), end="")  
    for j in range(len(x[i])):  
    print("{:4}".format(x[i][j]), end="")  
    print()  
     
    print("\nМатриця Y:")  
    y = np.random.randint(y\_min, y\_max, size=(n, m))  
    for i in range(len(y)):  
    print("{}.".format(i + 1), end="")  
    for j in range(len(y[i])):  
    print("{:4}".format(y[i][j]), end="")  
    print()  
     
    print("\nСередні значення Y:")  
    y\_av = np.sum(y, axis=1) / len(y[0])  
    y\_1, y\_2, y\_3, y\_4 = y\_av  
    print(f"y\_1 = {y\_1:.2f}\ny\_2 = {y\_2:.2f}\ny\_3 = {y\_3:.2f}\ny\_4 = {y\_4:.2f}")  
    mx\_1, mx\_2, mx\_3 = [i / len(x) for i in np.sum(x, axis=0)]  
    my = sum(y\_av) / len(y\_av)  
     
    a\_1 = sum([x[i][0] \* y\_av[i] for i in range(len(x))]) / len(x)  
    a\_2 = sum([x[i][1] \* y\_av[i] for i in range(len(x))]) / len(x)  
    a\_3 = sum([x[i][2] \* y\_av[i] for i in range(len(x))]) / len(x)  
     
    a\_11 = sum([x[i][0] \*\* 2 for i in range(len(x))]) / len(x)  
    a\_22 = sum([x[i][1] \*\* 2 for i in range(len(x))]) / len(x)  
    a\_33 = sum([x[i][2] \*\* 2 for i in range(len(x))]) / len(x)  
    a\_12 = sum([x[i][0] \* x[i][1] for i in range(len(x))]) / len(x)  
    a\_13 = sum([x[i][0] \* x[i][2] for i in range(len(x))]) / len(x)  
    a\_23 = a\_32 = sum([x[i][1] \* x[i][2] for i in range(len(x))]) / len(x)  
     
    det = np.linalg.det(  
    [[1, mx\_1, mx\_2, mx\_3], [mx\_1, a\_11, a\_12, a\_13], [mx\_2, a\_12, a\_22, a\_32], [mx\_3, a\_13, a\_23, a\_33]])  
    det\_0 = np.linalg.det(  
    [[my, mx\_1, mx\_2, mx\_3], [a\_1, a\_11, a\_12, a\_13], [a\_2, a\_12, a\_22, a\_32], [a\_3, a\_13, a\_23, a\_33]])  
    det\_1 = np.linalg.det(  
    [[1, my, mx\_2, mx\_3], [mx\_1, a\_1, a\_12, a\_13], [mx\_2, a\_2, a\_22, a\_32], [mx\_3, a\_3, a\_23, a\_33]])  
    det\_2 = np.linalg.det(  
    [[1, mx\_1, my, mx\_3], [mx\_1, a\_11, a\_1, a\_13], [mx\_2, a\_12, a\_2, a\_32], [mx\_3, a\_13, a\_3, a\_33]])  
    det\_3 = np.linalg.det(  
    [[1, mx\_1, mx\_2, my], [mx\_1, a\_11, a\_12, a\_1], [mx\_2, a\_12, a\_22, a\_2], [mx\_3, a\_13, a\_23, a\_3]])  
     
    b\_0 = det\_0 / det  
    b\_1 = det\_1 / det  
    b\_2 = det\_2 / det  
    b\_3 = det\_3 / det  
    b = [b\_0, b\_1, b\_2, b\_3]  
     
    print(f"\nНормалізоване рівняння регресії: y = {b\_0:.5f} + {b\_1:.5f} \* x1 + {b\_2:.5f} \* x2 + {b\_3:.5f} \* x3\n")  
    print("Перевірка:")  
    y\_1\_exp = b\_0 + b\_1 \* x[0][0] + b\_2 \* x[0][1] + b\_3 \* x[0][2]  
    y\_2\_exp = b\_0 + b\_1 \* x[1][0] + b\_2 \* x[1][1] + b\_3 \* x[1][2]  
    y\_3\_exp = b\_0 + b\_1 \* x[2][0] + b\_2 \* x[2][1] + b\_3 \* x[2][2]  
    y\_4\_exp = b\_0 + b\_1 \* x[3][0] + b\_2 \* x[3][1] + b\_3 \* x[3][2]  
    print(f"y\_1 = {b\_0:.3f} + {b\_1:.3f} \* {x[0][0]} + {b\_2:.3f} \* {x[0][1]} + {b\_3:.3f} \* {x[0][2]} = {y\_1\_exp:.3f}"  
    f"\ny\_2 = {b\_0:.3f} + {b\_1:.3f} \* {x[1][0]} + {b\_2:.3f} \* {x[1][1]} + {b\_3:.3f} \* {x[1][2]} = {y\_2\_exp:.3f}"  
    f"\ny\_3 = {b\_0:.3f} + {b\_1:.3f} \* {x[2][0]} + {b\_2:.3f} \* {x[2][1]} + {b\_3:.3f} \* {x[2][2]} = {y\_3\_exp:.3f}"  
    f"\ny\_4 = {b\_0:.3f} + {b\_1:.3f} \* {x[3][0]} + {b\_2:.3f} \* {x[3][1]} + {b\_3:.3f} \* {x[3][2]} = {y\_4\_exp:.3f}")  
     
    print("\nКритерій Кохрена:")  
    f\_1 = m - 1  
    f\_2 = n  
    s\_1 = sum([(i - y\_1) \*\* 2 for i in y[0]]) / m  
    s\_2 = sum([(i - y\_2) \*\* 2 for i in y[1]]) / m  
    s\_3 = sum([(i - y\_3) \*\* 2 for i in y[2]]) / m  
    s\_4 = sum([(i - y\_4) \*\* 2 for i in y[3]]) / m  
    s\_array = np.array([s\_1, s\_2, s\_3, s\_4])  
    gP = max(s\_array) / sum(s\_array)  
     
    table = {3: 0.6841, 4: 0.6287, 5: 0.5892, 6: 0.5598, 7: 0.5365, 8: 0.5175, 9: 0.5017, 10: 0.4884,  
    range(11, 17): 0.4366, range(17, 37): 0.3720, range(37, 145): 0.3093}  
    gT = table.get(m)  
     
    if (gP < gT):  
    print(f"Дисперія однорідна: Gp = {gP:.5} < Gt = {gT}")  
    else:  
    print(f"Дисперсія не однорідна Gp = {gP:.5} < Gt = {gT}")  
    m = m + 1  
    main(m + 1, n, q)  
    return  
     
    print("\nКритерій Ст'юдента")  
    s2\_B = s\_array.sum() / n  
    s2\_beta\_S = s2\_B / (n \* m)  
    s\_beta\_S = pow(s2\_beta\_S, 1 / 2)  
     
    beta\_0 = sum([norm\_x[i][0] \* y\_av[i] for i in range(len(norm\_x))]) / n  
    beta\_1 = sum([norm\_x[i][1] \* y\_av[i] for i in range(len(norm\_x))]) / n  
    beta\_2 = sum([norm\_x[i][2] \* y\_av[i] for i in range(len(norm\_x))]) / n  
    beta\_3 = sum([norm\_x[i][3] \* y\_av[i] for i in range(len(norm\_x))]) / n  
     
    t = [abs(beta\_0) / s\_beta\_S, abs(beta\_1) / s\_beta\_S, abs(beta\_2) / s\_beta\_S, abs(beta\_3) / s\_beta\_S]  
     
    f3 = f\_1 \* f\_2  
    t\_table = {8: 2.306, 9: 2.262, 10: 2.228, 11: 2.201, 12: 2.179, 13: 2.160, 14: 2.145, 15: 2.131, 16: 2.120,  
    17: 2.110, 18: 2.101, 19: 2.093, 20: 2.086, 21: 2.08, 22: 2.074, 23: 2.069, 24: 2.064, 25: 2.06}  
    d = 4  
     
    for i in range(len(t)):  
    if (t\_table.get(f3) > t[i]):  
    b[i] = 0  
    d -= 1  
     
    print(f"Рівняння регресії: y = {b[0]:.3f} + {b[1]:.3f} \* x1 + {b[2]:.3f} \* x2 + {b[3]:.3f} \* x3")  
    check\_0 = b[0] + b[1] \* x[0][0] + b[2] \* x[0][1] + b[3] \* x[0][2]  
    check\_1 = b[0] + b[1] \* x[1][0] + b[2] \* x[1][1] + b[3] \* x[1][2]  
    check\_2 = b[0] + b[1] \* x[2][0] + b[2] \* x[2][1] + b[3] \* x[2][2]  
    check\_3 = b[0] + b[1] \* x[3][0] + b[2] \* x[3][1] + b[3] \* x[3][2]  
    ckeck\_list = [check\_0, check\_1, check\_2, check\_3]  
    print("Нормалізовані значення: ", ckeck\_list)  
     
    print("\nКритерій Фішера")  
    f\_4 = n - d  
    s2\_ad = m / f\_4 \* sum([(ckeck\_list[i] - y\_av[i]) \*\* 2 for i in range(len(y\_av))])  
    fP = s2\_ad / s2\_B  
    fT = [  
    [164.4, 199.5, 215.7, 224.6, 230.2, 234],  
    [18.5, 19.2, 19.2, 19.3, 19.3, 19.3],  
    [10.1, 9.6, 9.3, 9.1, 9, 8.9],  
    [7.7, 6.9, 6.6, 6.4, 6.3, 6.2],  
    [6.6, 5.8, 5.4, 5.2, 5.1, 5],  
    [6, 5.1, 4.8, 4.5, 4.4, 4.3],  
    [5.5, 4.7, 4.4, 4.1, 4, 3.9],  
    [5.3, 4.5, 4.1, 3.8, 3.7, 3.6],  
    [5.1, 4.3, 3.9, 3.6, 3.5, 3.4],  
    [5, 4.1, 3.7, 3.5, 3.3, 3.2],  
    [4.8, 4, 3.6, 3.4, 3.2, 3.1],  
    [4.8, 3.9, 3.5, 3.3, 3.1, 3],  
    [4.7, 3.8, 3.4, 3.2, 3, 2.9],  
    [4.6, 3.7, 3.3, 3.1, 3, 2.9],  
    [4.5, 3.7, 3.3, 3.1, 2.9, 2.8],  
    [4.5, 3.6, 3.2, 3, 2.9, 2.7],  
    [4.5, 3.6, 3.2, 3, 2.8, 2.7],  
    [4.4, 3.6, 3.2, 2.9, 2.8, 2.7],  
    [4.4, 3.5, 3.1, 2.9, 2.7, 2.6],  
    [4.4, 3.5, 3.1, 2.9, 2.7, 2.6]  
    ]  
    if (fP > fT[f3][f\_4]):  
    print(f"fp = {fP} > ft = {fT[f3][f\_4]}.\nМатематична модель не адекватна\n")  
    else:  
    print(f"fP = {fP} < fT = {fT[f3][f\_4]}.\nМатематична модель адекватна\n")  
     
     
   print("\nРівняння регресії: y = b0 + b1 \* x1 + b2 \* x2 +b3 \* x3")  
   main(m, n)

Результат:

Рівняння регресії: y = b0 + b1 \* x1 + b2 \* x2 +b3 \* x3

Матриця кодованих значень:

1. 1 -1 -1 -1

2. 1 -1 1 1

3. 1 1 -1 1

4. 1 1 1 -1

Матриця X:

1. 15 30 15

2. 15 80 45

3. 45 30 45

4. 45 80 15

Матриця Y:

1. 253 245 242

2. 250 253 245

3. 221 230 239

4. 245 250 238

Середні значення Y:

y\_1 = 246.67

y\_2 = 249.33

y\_3 = 230.00

y\_4 = 244.33

Нормалізоване рівняння регресії: y = 249.90000 + -0.36111 \* x1 + 0.17000 \* x2 + -0.19444 \* x3

Перевірка:

y\_1 = 249.900 + -0.361 \* 15 + 0.170 \* 30 + -0.194 \* 15 = 246.667

y\_2 = 249.900 + -0.361 \* 15 + 0.170 \* 80 + -0.194 \* 45 = 249.333

y\_3 = 249.900 + -0.361 \* 45 + 0.170 \* 30 + -0.194 \* 45 = 230.000

y\_4 = 249.900 + -0.361 \* 45 + 0.170 \* 80 + -0.194 \* 15 = 244.333

Критерій Кохрена:

Дисперія однорідна: Gp = 0.48795 < Gt = 0.6841

Критерій Ст'юдента

Рівняння регресії: y = 249.900 + -0.361 \* x1 + 0.170 \* x2 + 0.000 \* x3

Нормалізовані значення: [249.58333333333246, 258.08333333333263, 238.7499999999992, 247.24999999999935]

Критерій Фішера

fp = 18.448795180719067 > ft = 4.3.

Математична модель не адекватна