Міністерство освіти і науки України

Національний технічний університет України

«Київський політехнічний інститут ім. Ігоря Сікорського»

Факультет інформатики та обчислювальної техніки

ЛАБОРАТОРНА РОБОТА № 2

з дисципліни «Методи наукових досліджень»

на тему «ПРОВЕДЕННЯ ДВОФАКТОРНОГО ЕКСПЕРИМЕНТУ З

ВИКОРИСТАННЯМ ЛІНІЙНОГО РІВНЯННЯ РЕГРЕСІЇ»

ВИКОНАВ:

студент 2 курсу

групи ІВ-91

Петрук С.В.

Залікова – 9124

ПЕРЕВІРИВ:

ас. Регіда П. Г.

Київ – 2021

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

1. Записати лінійне рівняння регресії.

2. Обрати тип двофакторного експерименту і скласти матрицю планування для нього з використанням додаткового нульового фактору **(хо=1).**

3. Провести експеримент в усіх точках повного факторного простору (знайти значення функції відгуку y). Значення функції відгуку задати випадковим чином у відповідності до варіанту у діапазоні ymin ÷ ymax

**ymax = (30 - Nваріанту)\*10,**

**ymin = (20 - Nваріанту)\*10.**

4. Перевірити однорідності дисперсії за критерієм Романовського

5. Знайти коефіцієнти нормованих рівнянь регресії і виконати перевірку

(підставити значення нормованих факторів і коефіцієнтів у рівняння).

6. Провести натуралізацію рівняння регресії й виконати перевірку

натуралізованого рівняння.

7. Написати комп'ютерну програму, яка все це виконує.

**Програмний код**

from random import randint  
import numpy as np  
  
  
def normalize(array):  
 x0 = (max(array) + min(array)) / 2  
 dx = x0 - min(array)  
 normalized = [(array[i] - x0) / dx for i in range(len(array))]  
 return normalized  
  
  
def find\_closest(m, table):  
 key\_list = list(table.keys())  
 if m in key\_list:  
 return m  
 else:  
 diffs = ([abs(m-i) for i in key\_list])  
 index = diffs.index(min(diffs))  
 return key\_list[index]  
  
  
def main(m):  
 var = 122  
 y\_max = (30 - var) \* 10  
 y\_min = (20 - var) \* 10  
 print(f'y\_max = {y\_max}, y\_min = {y\_min}')  
  
 romanovski\_values\_table = {2: 1.73, 6: 2.16, 8: 2.43, 10: 2.62, 12: 2.75, 15: 2.9, 20: 3.08}  
  
 x1 = [-5, randint(-5, 15), 15]  
 x2 = [10, randint(10, 60), 60]  
 print(f'X1: {x1}')  
 print(f'X2: {x2}')  
  
 x1\_n = normalize(x1)  
 x2\_n = normalize(x2)  
 print(f'X1 normalized: {x1\_n}')  
 print(f'X2 normalized: {x2\_n}')  
  
 matrix\_of\_experiments = [[randint(y\_min, y\_max) for \_ in range(3)] for \_ in range(m)]  
 print(f'Y from 1 to m: {matrix\_of\_experiments}')  
  
 y\_average = []  
 dispersions = []  
  
 for i in range(len(matrix\_of\_experiments[0])):  
 sum1 = 0  
 for j in range(m):  
 sum1 += matrix\_of\_experiments[j][i]  
  
 y\_current = sum1 / len(matrix\_of\_experiments)  
 y\_average.append(y\_current)  
  
 sum2 = 0  
 for k in range(m):  
 sum2 += (matrix\_of\_experiments[k][i] - y\_current) \*\* 2  
 dispersions.append(sum2 / m)  
  
 print(f'Y average: {y\_average}')  
 print(f'Dispersions: {dispersions}')  
  
 main\_dispersion = ((2 \* (2 \* m - 2)) / (m \* (m - 4))) \*\* (1 / 2)  
 print(f'Main dispersion = {main\_dispersion}')  
  
 f\_uv = []  
 theta\_uv = []  
 r\_uv = []  
 res = 0  
 for i in range(3):  
 for j in range(i + 1, 3):  
 if dispersions[i] >= dispersions[j]:  
 res = dispersions[i] / dispersions[j]  
 else:  
 res = dispersions[j] / dispersions[i]  
 f\_uv.append(res)  
  
 theta = ((m - 2) / m) \* res  
 theta\_uv.append(theta)  
  
 r\_uv.append(abs(theta - 1) / main\_dispersion)  
  
 print(f'Fuv : {f\_uv}')  
 print(f'0uv: {theta\_uv}')  
 print(f'Ruv: {r\_uv}')  
  
 r\_kr = romanovski\_values\_table[find\_closest(m, romanovski\_values\_table)]  
 print(f'При m = {m} для p = 0.99 Rkr = {r\_kr}')  
 for i in r\_uv:  
 if i >= r\_kr:  
 print('Дисперсії неоднорідні')  
 print(f'Проводимо експеримент при m = {m+1}')  
 main(m+1)  
 exit()  
 print('Дисперсії однорідні')  
  
 mx1 = sum(x1\_n) / 3  
 mx2 = sum(x2\_n) / 3  
 my = sum(y\_average) / 3  
  
 print(f'mx1: {mx1}')  
 print(f'mx2: {mx2}')  
 print(f'my: {my}')  
  
 a1 = (x1\_n[0] \*\* 2 + x1\_n[1] \*\* 2 + x1\_n[2] \*\* 2) / 3  
 a2 = (x1\_n[0] \* x2\_n[0] + x1\_n[1] \* x2\_n[1] + x1\_n[2] \* x2\_n[2]) / 3  
 a3 = (x2\_n[0] \*\* 2 + x2\_n[1] \*\* 2 + x2\_n[2] \*\* 2) / 3  
 print(f'a1 = {a1}')  
 print(f'a2 = {a2}')  
 print(f'a3 = {a3}')  
  
 a11 = (x1\_n[0] \* y\_average[0] + x1\_n[1] \* y\_average[1] + x1\_n[2] \* y\_average[2]) / 3  
 a22 = (x2\_n[0] \* y\_average[0] + x2\_n[1] \* y\_average[1] + x2\_n[2] \* y\_average[2]) / 3  
 print(f'a11 = {a11}')  
 print(f'a22 = {a22}')  
  
 b0 = np.linalg.det([[my, mx1, mx2],  
 [a11, a1, a2],  
 [a22, a2, a3]]) / \  
 np.linalg.det([[1, mx1, mx2],  
 [mx1, a1, a2],  
 [mx2, a2, a3]])  
  
 b1 = np.linalg.det([[1, my, mx2],  
 [mx1, a11, a2],  
 [mx2, a22, a3]]) / \  
 np.linalg.det([[1, mx1, mx2],  
 [mx1, a1, a2],  
 [mx2, a2, a3]])  
  
 b2 = np.linalg.det([[1, mx1, my],  
 [mx1, a1, a11],  
 [mx2, a2, a22]]) / \  
 np.linalg.det([[1, mx1, mx2],  
 [mx1, a1, a2],  
 [mx2, a2, a3]])  
  
 print(f'Нормоване рівняння регресії: \ny = {b0} + {b1}\*x1 + {b2}\*x2')  
 for i in range(3):  
 print(f'Для X1{i + 1} та X2{i + 1}')  
 print(f'y = {b0 + b1 \* x1\_n[i] + b2 \* x2\_n[i]}')  
  
 dx1 = abs(x1[2] - x1[0]) / 2  
 dx2 = abs(x2[2] - x2[0]) / 2  
 x10 = (x1[2] + x1[0]) / 2  
 x20 = (x2[2] + x2[0]) / 2  
 print(f'dx1 = {dx1}')  
 print(f'dx2 = {dx2}')  
 print(f'x10 = {x10}')  
 print(f'x20 = {x20}')  
  
 a0\_res = b0 - b1 \* (x10 / dx1) - b2 \* (x20 / dx2)  
 a1\_res = b1 / dx1  
 a2\_res = b2 / dx2  
 print(f'a0 = {a0\_res}')  
 print(f'a1 = {a1\_res}')  
 print(f'a2 = {a2\_res}')  
  
 print(f'Натуралізоване рівняння регресії: \ny = {a0\_res} + {a1\_res}\*x1 + {a2\_res}\*x2')  
 for i in range(3):  
 print(f'Для X1{i + 1} та X2{i + 1}')  
 print(f'y = {a0\_res + a1\_res \* x1[i] + a2\_res \* x2[i]}')  
  
  
if \_\_name\_\_ == '\_\_main\_\_':  
 main(m=5)

**Результат роботи програми**



