Міністерство освіти і науки України
Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
Факультет інформатики та обчислювальної техніки
Кафедра обчислювальної техніки

Лабораторна робота №4

«Проведення трьохфакторного експерименту з використанням лінійного рівняння регресії»

Виконав:
студент II курсу ФІОТ
групи ІВ-93
Ровишин Андрій

Перевірив:
Регіда П.Г.

Київ – 2021

Мета роботи: Провести повний трьохфакторний експеримент. Знайти рівняння регресії адекватне об'єкту.

Завдання на лабораторну роботу

1. Скласти матрицю планування для повного трьохфакторного експерименту.

2. Провести експеримент, повторивши N раз досліди у всіх точках факторного простору і знайти значення відгуку Y. Знайти значення Y шляхом моделювання випадкових чисел у певному діапазоні відповідно варіанту. Варіанти вибираються за номером в списку в журналі викладача.

$$y_{max} = 200 + x_{cp\ max};$$

$$y_{min} = 200 + x_{cp\ min}$$

де $x_{cp\,max} = \frac{x_{1max} + x_{2max} + x_{3max}}{3}$, $x_{cp\,min} = \frac{x_{1min} + x_{2min} + x_{3min}}{3}$

2. Знайти коефіцієнти лінійного рівняння регресії. Записати лінійне рівняння регресії.

3. Провести 3 статистичні перевірки.

4. Зробити висновки по адекватності регресії та значимості окремих коефіцієнтів і записати скореговане рівняння регресії.

5. Написати комп'ютерну програму, яка усе це виконує.

№варіанта	x_1		x_2		x_3	
	min	max	min	Max	min	max
319	20	70	-15	45	20	35

Код програми

```
import random
from prettytable import PrettyTable

m = 3
n = 8
x1_min = 20
x1_max = 70
x2_min = -15
x2_max = 45
x3_min = 20
x3_max = 35
y_min = 200 + (x1_min + x2_min + x3_min) / 3
y_max = 200 + (x1_max + x2_max + x3_max) / 3
```

```
y_matrix = [[random.randint(int(y_min), int(y_max)) for _ in range(m)] for _ in
range(n)]
y_avrg = [round(sum(i) / len(i), 3) for i in y_matrix]
xn = [[-1, -1, -1],
      [-1, -1, 1],
      [-1, 1, -1],
      [-1, 1, 1],
      [1, -1, -1],
      [1, -1, 1],
      [1, 1, -1],
      [1, 1, 1]]

b0 = sum(y_avrg) / n
b1 = sum([y_avrg[i] * xn[i][0] for i in range(n)]) / n
b2 = sum([y_avrg[i] * xn[i][1] for i in range(n)]) / n
b3 = sum([y_avrg[i] * xn[i][2] for i in range(n)]) / n
b12 = sum([y_avrg[i] * xn[i][0] * xn[i][1] for i in range(n)]) / n
b13 = sum([y_avrg[i] * xn[i][0] * xn[i][2] for i in range(n)]) / n
b23 = sum([y_avrg[i] * xn[i][1] * xn[i][2] for i in range(n)]) / n
b123 = sum([y_avrg[i] * xn[i][0] * xn[i][1] * xn[i][2] for i in range(n)]) / n

plan_matrix = [[x1_min, x2_min, x3_min, x1_min * x2_min, x1_min * x3_min, x2_min *
x3_min, x1_min * x2_min * x3_min],
               [x1_min, x2_min, x3_max, x1_min * x2_min, x1_min * x3_max, x2_min *
x3_max, x1_min * x2_min * x3_max],
               [x1_min, x2_max, x3_min, x1_min * x2_max, x1_min * x3_min, x2_max *
x3_min, x1_min * x2_max * x3_min],
               [x1_min, x2_max, x3_max, x1_min * x2_max, x1_min * x3_max, x2_max *
x3_max, x1_min * x2_max * x3_max],
               [x1_max, x2_min, x3_min, x1_max * x2_min, x1_max * x3_min, x2_min *
x3_min, x1_max * x2_min * x3_min],
               [x1_max, x2_min, x3_max, x1_max * x2_min, x1_max * x3_max, x2_min *
x3_max, x1_max * x2_min * x3_max],
               [x1_max, x2_max, x3_min, x1_max * x2_max, x1_max * x3_min, x2_max *
x3_min, x1_max * x2_max * x3_min],
               [x1_max, x2_max, x3_max, x1_max * x2_max, x1_max * x3_max, x2_max *
x3_max, x1_max * x2_max * x3_max]]

y_result = []
for i in range(n):
    y_result.append(b0 + b1 * plan_matrix[i][0] + b2 * plan_matrix[i][1] + b3 *
plan_matrix[i][2] +
                    b12 * plan_matrix[i][3] + b13 * plan_matrix[i][4] + b23 *
plan_matrix[i][5] +
                    b123 * plan_matrix[i][6])

dispersion = [round(sum([(y_matrix[j][i] - y_avrg[i]) ** 2 for i in range(m)]) / m, 3)
for j in range(n)]

table1 = PrettyTable()
table1.field_names = ["X0", "X1", "X2", "X3", "X12", "X13", "X23", "X123", "Y1", "Y2",
"Y3", "Y average", "S^2"]
x0 = [[1] for _ in range(n)]
for i in range(n):
    table1.add_row([*x0[i], *plan_matrix[i], *y_matrix[i], y_avrg[i], dispersion[i]])
print('Матриця планування:')
print(table1)
print()

# Критерій Кохрена
print("Перевірка за критерієм Кохрена:")
gp = max(dispersion) / sum(dispersion)
gt = 0.5157
if gp < gt:
```

```
    print("За критерієм Кохрена дисперсія однорідна")
    print("{} < {}".format(round(gp, 3), round(gt, 3)))
else:
    print("За критерієм Кохрена дисперсія однорідна")
    print("{} > {}".format(round(gp, 3), round(gt, 3)))
print()

# Критерій Стьюденса
print("Перевірка значущості коефіцієнтів за критерієм Стьюдента")
d = 8
sb = sum(dispersion) / n
s_beta_2 = sb / (n * m)
s_beta = s_beta_2 ** (1 / 2)
bb = [b0, b1, b2, b3, b12, b13, b23, b123]
t_list = [abs(bb[i]) / s_beta for i in range(n)]
tt = 2.120
b_list = [b0, b1, b2, b3, b12, b13, b23, b123]
for i in range(n):
    if t_list[i] < tt:
        b_list[i] = 0
        d -= 1
for i in range(len(t_list)):
    print('t{} = {}'.format(i, round(t_list[i], 3)))
print()

# Критерій Фішера
print("Перевірка адекватності за критерієм Фішера")
y_reg = [b0 + b1 * plan_matrix[i][0] + b2 * plan_matrix[i][1] + b3 * plan_matrix[i][2]
+
         b12 * plan_matrix[i][3] + b13 * plan_matrix[i][4] + b23 * plan_matrix[i][5] +
         b123 * plan_matrix[i][6] for i in range(n)]
sad = (m / (n - d)) * int(sum([(y_reg[i] - y_avrg[i]) ** 2 for i in range(n)]))
fp = sad / sb
if fp < 4.5:
    print('Рівняння регресії адекватне оригіналу на рівні 0.05')
else:
    print('Рівняння регресії неадекватне оригіналу на рівні 0.05')
print()

print('Рівняння:')
print('y = {} + {} * x1 + {} * x2 + {} * x3 + {} * x1x2 + {} * x1x3 + {} * x2x3 + {} *
x1x2x3'
      .format(round(b0, 3), round(b1, 3), round(b2, 3), round(b3, 3), round(b12, 3),
round(b13, 3), round(b23, 3),
              round(b123, 3)))
print()
for i in range(len(y_result)):
    print('y{} = {}'.format(i+1, round(y_result[i], 3)))
```

Висновок:

У ході лабораторної роботи я змоделював трьохфакторний експеримент. Також я реалізував 3 статистичні перевірки за критерієм Кохрена, Стьюдента та Фішера. Знайшов рівняння регресії адекватне об'єкту.