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

Лабораторна робота №4
З дисципліни «*Проведення трьохфакторного експерименту при використанні рівняння регресії з урахуванням ефекту взаємодії.*»

Виконав:
Студент 2-го курсу ФІОТ
Залікова книжка ІВ-8217
Мотора В. С.

Перевірив:
Ас. Регіда П.Г.

Київ 2020 р.

Варіант 232

Лістинг програми:

Код

```
import
math
          import random
          from functools import partial


          import numpy as np
          from scipy.stats import f, t
          from prettytable import PrettyTable


          def get_fisher_critical(prob,f3, f4):
              for i in [j*0.001 for j in range(int(10/0.001))]:
                  if abs(f.cdf(i,f4,f3)-prob) < 0.0001:
                      return i




          def get_student_critical(prob, f3):
              for i in [j*0.0001 for j in range(int(5/0.0001))]:
                  if abs(t.cdf(i,f3)-(0.5 + prob/0.1*0.05)) < 0.000005:
                      return i


          def get_cohren_critical(prob, f1, f2):
              f_crit = f.isf((1-prob)/f2, f1, (f2-1)*f1)
              return f_crit/(f_crit+f2-1)


          m = 3
          N = 8


          p = .95
          q = 1 - p


          x1max = 60
          x1min = 10
          x2max = 15
          x2min = -35
```

```
x3max = 15
x3min = 10


X_max = [x1max, x2max, x3max]
X_min = [x1min, x2min, x3min]


x_av_min = (x1min + x2min + x3min) / 3
x_av_max = (x1max + x2max + x3max) / 3


Y_max = int(round(200 + x_av_max, 0))
Y_min = int(round(200 + x_av_min, 0))
X0=1




X_matr = [
    [-1, -1, -1],
    [-1, -1, 1],
    [-1, 1, -1],
    [-1, 1, 1],
    [1, -1, -1],
    [1, -1, 1],
    [1, 1, -1],
    [1, 1, 1]]



x_for_beta = [
    [1, -1, -1, -1],
    [1, -1, -1, 1],
    [1, -1, 1, -1],
    [1, -1, 1, 1],
    [1, 1, -1, -1],
    [1, 1, -1, 1],
    [1, 1, 1, -1],
    [1, 1, 1, 1]
]
x_12_13_23 = [
    [1, 1, 1],
    [1, -1, -1],
    [-1, 1, -1],
    [-1, -1, 1],
    [-1, -1, 1],
```

```
    [-1, 1, -1],
    [1, -1, -1],
    [1, 1, 1],
]
x_123 = [
    -1,
    1,
    1,
    -1,
    1,
    -1,
    -1,
    1
]
X_matr_natur = [
    [x1min, x2min, x3min],
    [x1min, x2min, x3max],
    [x1min, x2max, x3min],
    [x1min, x2max, x3max],
    [x1max, x2min, x3min],
    [x1max, x2min, x3max],
    [x1max, x2max, x3min],
    [x1max, x2max, x3max],
]
x_12_13_23_natur = [[X_matr_natur[j][0] * X_matr_natur[j][1], X_matr_natur[j][0]
* X_matr_natur[j][2],
                    X_matr_natur[j][1] * X_matr_natur[j][2]] for j in range(N)]
x_123_natur = [X_matr_natur[j][0] * X_matr_natur[j][1] * X_matr_natur[j][2] for j in
range(N)]




flag = True
ct = 3 #кількість ітерацій
while (flag):
    flag = False
    Y_matr = [[random.randint((Y_min), (Y_max)) for i in range(m)] for j in range(N)]


    Y_average = [sum(j) / m for j in Y_matr]


    results_nat = [
        sum(Y_average),
        sum([Y_average[j] * X_matr_natur[j][0] for j in range(N)]),
```

```
    sum([Y_average[j] * X_matr_natur[j][1] for j in range(N)]),
    sum([Y_average[j] * X_matr_natur[j][2] for j in range(N)]),
    sum([Y_average[j] * x_12_13_23_natur[j][0] for j in range(N)]),
    sum([Y_average[j] * x_12_13_23_natur[j][1] for j in range(N)]),
    sum([Y_average[j] * x_12_13_23_natur[j][2] for j in range(N)]),
    sum([Y_average[j] * x_123_natur[j] for j in range(N)]),
]



mj0 = [N,
       sum([X_matr_natur[j][0] for j in range(N)]),
       sum([X_matr_natur[j][1] for j in range(N)]),
       sum([X_matr_natur[j][2] for j in range(N)]),
       sum([x_12_13_23_natur[j][0] for j in range(N)]),
       sum([x_12_13_23_natur[j][1] for j in range(N)]),
       sum([x_12_13_23_natur[j][2] for j in range(N)]),
       sum([x_123_natur[j] for j in range(N)]),
       ]
mj1 = [sum([X_matr_natur[j][0] for j in range(N)]),
       sum([X_matr_natur[j][0] ** 2 for j in range(N)]),
       sum([x_12_13_23_natur[j][0] for j in range(N)]),
       sum([x_12_13_23_natur[j][1] for j in range(N)]),
       sum([(X_matr_natur[j][0] ** 2) * X_matr_natur[j][1] for j in range(N)]),
       sum([(X_matr_natur[j][0] ** 2) * X_matr_natur[j][2] for j in range(N)]),
       sum([x_123_natur[j] for j in range(N)]),
       sum([(X_matr_natur[j][0] ** 2) * x_12_13_23_natur[j][2] for j in range(N)]),
       ]
mj2 = [sum([X_matr_natur[j][1] for j in range(N)]),
       sum([x_12_13_23_natur[j][0] for j in range(N)]),
       sum([X_matr_natur[j][1] ** 2 for j in range(N)]),
       sum([x_12_13_23_natur[j][2] for j in range(N)]),
       sum([(X_matr_natur[j][1] ** 2) * X_matr_natur[j][0] for j in range(N)]),
       sum([x_123_natur[j] for j in range(N)]),
       sum([(X_matr_natur[j][1] ** 2) * X_matr_natur[j][2] for j in range(N)]),
       sum([(X_matr_natur[j][1] ** 2) * x_12_13_23_natur[j][1] for j in range(N)]),
       ]
mj3 = [sum([X_matr_natur[j][2] for j in range(N)]),
       sum([x_12_13_23_natur[j][1] for j in range(N)]),
       sum([x_12_13_23_natur[j][2] for j in range(N)]),
       sum([X_matr_natur[j][2] ** 2 for j in range(N)]),
       sum([x_123_natur[j] for j in range(N)]),
       sum([(X_matr_natur[j][2] ** 2) * X_matr_natur[j][0] for j in range(N)]),
       sum([(X_matr_natur[j][2] ** 2) * X_matr_natur[j][1] for j in range(N)]),
       sum([(X_matr_natur[j][2] ** 2) * x_12_13_23_natur[j][0] for j in range(N)]),
       ]
mj4 = [sum([x_12_13_23_natur[j][0] for j in range(N)]),
       sum([(X_matr_natur[j][0] ** 2) * X_matr_natur[j][1] for j in range(N)]),
       sum([(X_matr_natur[j][1] ** 2) * X_matr_natur[j][0] for j in range(N)]),
```

```
        sum([x_123_natur[j] for j in range(N)]),
        sum([x_12_13_23_natur[j][0] ** 2 for j in range(N)]),
        sum([(X_matr_natur[j][0] ** 2) * x_12_13_23_natur[j][2] for j in range(N)]),
        sum([(X_matr_natur[j][1] ** 2) * x_12_13_23_natur[j][1] for j in range(N)]),
        sum([(x_12_13_23_natur[j][0] ** 2) * X_matr_natur[j][2] for j in range(N)]),
        ]
    mj5 = [sum([x_12_13_23_natur[j][1] for j in range(N)]),
        sum([(X_matr_natur[j][0] ** 2) * X_matr_natur[j][2] for j in range(N)]),
        sum([x_123_natur[j] for j in range(N)]),
        sum([(X_matr_natur[j][2] ** 2) * X_matr_natur[j][0] for j in range(N)]),
        sum([(X_matr_natur[j][0] ** 2) * x_12_13_23_natur[j][2] for j in range(N)]),
        sum([x_12_13_23_natur[j][1] ** 2 for j in range(N)]),
        sum([(X_matr_natur[j][2] ** 2) * x_12_13_23_natur[j][0] for j in range(N)]),
        sum([(x_12_13_23_natur[j][1] ** 2) * X_matr_natur[j][1] for j in range(N)]),
        ]
    mj6 = [sum([x_12_13_23_natur[j][2] for j in range(N)]),
        sum([x_123_natur[j] for j in range(N)]),
        sum([(X_matr_natur[j][1] ** 2) * X_matr_natur[j][2] for j in range(N)]),
        sum([(X_matr_natur[j][2] ** 2) * X_matr_natur[j][1] for j in range(N)]),
        sum([(X_matr_natur[j][1] ** 2) * x_12_13_23_natur[j][1] for j in range(N)]),
        sum([(X_matr_natur[j][2] ** 2) * x_12_13_23_natur[j][0] for j in range(N)]),
        sum([(X_matr_natur[j][2] ** 2) * X_matr_natur[j][1] for j in range(N)]),
        sum([(x_12_13_23_natur[j][2] ** 2) * X_matr_natur[j][0] for j in range(N)]),
        ]
    mj7 = [sum([x_123_natur[j] for j in range(N)]),
        sum([(X_matr_natur[j][0] ** 2) * x_12_13_23_natur[j][2] for j in range(N)]),
        sum([(X_matr_natur[j][1] ** 2) * x_12_13_23_natur[j][1] for j in range(N)]),
        sum([(X_matr_natur[j][2] ** 2) * x_12_13_23_natur[j][0] for j in range(N)]),
        sum([(x_12_13_23_natur[j][0] ** 2) * X_matr_natur[j][2] for j in range(N)]),
        sum([(x_12_13_23_natur[j][1] ** 2) * X_matr_natur[j][1] for j in range(N)]),
        sum([(x_12_13_23_natur[j][2] ** 2) * X_matr_natur[j][0] for j in range(N)]),
        sum([x_123_natur[j] ** 2 for j in range(N)]) ]



    B_nat1 = np.linalg.solve([mj0, mj1, mj2, mj3, mj4, mj5, mj6, mj7], results_nat) # list
of B's
    B_nat = list(B_nat1)



    B_norm = [
        sum(Y_average) / N,
        sum([Y_average[j] * X_matr[j][0] for j in range(N)]) / N,
        sum([Y_average[j] * X_matr[j][1] for j in range(N)]) / N,
        sum([Y_average[j] * X_matr[j][2] for j in range(N)]) / N,
        sum([Y_average[j] * x_12_13_23[j][0] for j in range(N)]) / N,
        sum([Y_average[j] * x_12_13_23[j][1] for j in range(N)]) / N,
        sum([Y_average[j] * x_12_13_23[j][2] for j in range(N)]) / N,
```

```
        sum([Y_average[j] * x_123[j] for j in range(N)]) / N,
    ]


    print('##' * 40, '\n')
    print("\nМатриця планування експерименту:")


    tb = PrettyTable()
    tb.field_names = ["N", "x1", " x2", "x3", "Y1", "Y2", "Y3"]


    for i in range(N):
        tb.add_row([i + 1, X_matr[i][0], X_matr[i][1], X_matr[i][2],
Y_matr[i][0], Y_matr[i][1], Y_matr[i][2]])


    print(tb)


    def criterion_of_Student(value, criterion, check):
        if check < criterion:
            return 0
        else:
            return value




    y1_nat = B_nat[0] + B_nat[1] * X_matr_natur[0][0] + B_nat[2] * X_matr_natur[0][1] +
B_nat[3] * X_matr_natur[0][2] + \
             B_nat[4] * x_12_13_23_natur[0][0] + B_nat[5] * x_12_13_23_natur[0][1] +
B_nat[6] * x_12_13_23_natur[0][2] + \
             B_nat[7] * x_123_natur[0]
    y1_norm = B_norm[0] + B_norm[1] * X_matr[0][0] + B_norm[2] * X_matr[0][1] + B_norm[3]
* X_matr[0][2] + B_norm[4] * \
              x_12_13_23[0][0] + B_norm[5] * x_12_13_23[0][1] + B_norm[6] * x_12_13_23[0][2]
+ B_norm[7] * x_123[0]


    dx = [((X_max[i] - X_min[i]) / 2) for i in range(3)]
    A = [sum(Y_average) / len(Y_average), B_nat[0] * dx[0], B_nat[1] * dx[1], B_nat[2]
* dx[2]]


    S_kv = [(sum([((Y_matr[i][j] - Y_average[i]) ** 2) for j in range(m)]) / m) for i
in range(N)]
```

```
    Gp = max(S_kv) / sum(S_kv)


    f1 = m - 1
    f2 = N




    Gt = get_cohren_critical(p, f1, f2)


    if Gp < Gt:
        print('Дисперсії однорідні')


        flag = False
    else:
        print('Дисперсії неоднорідні')
        m += 1


    S_average = sum(S_kv) / N


    S2_beta_s = S_average / (N * m)


    S_beta_s = S2_beta_s ** .5


    beta = [(sum([x_for_beta[j][i] * Y_average[j] for j in range(N)]) / N) for i in
    range(4)] ts = [(math.fabs(beta[i]) / S_beta_s) for i in range(4)]


    f3 = f1 * f2


    criterion_of_St = get_student_critical(p, f3)


    result_2 = [criterion_of_Student(B_nat[0], criterion_of_St, ts[0]) +
                criterion_of_Student(B_nat[1], criterion_of_St, ts[1]) * X_matr_natur[i][0] +
                criterion_of_Student(B_nat[2], criterion_of_St, ts[2]) * X_matr_natur[i][1] +
                criterion_of_Student(B_nat[3], criterion_of_St, ts[3]) * X_matr_natur[i][2]
for i in range(N)]
```

```
    znach_koef = []
    for i in ts:
        if i > criterion_of_St:
            znach_koef.append(i)
        else:
            pass


    d = len(znach_koef)
    f4 = N - d
    f3 = (m - 1) * N


    deviation_of_adequacy = (m / (N - d)) * sum([(result_2[i] - Y_average[i]) ** 2 for i
in range(N)])


    Fp = deviation_of_adequacy / S2_beta_s


    Ft = get_fisher_critical(p, f3, f4)
    """+"""


    print("Значення після критерія Стюдента:")
    print("Y1 = {0:.3f};   Y2 = {1:.3f};   Y3 = {2:.3f};   Y4 = {3:.3f}.".format(result_2[0],
                                                                                 result_2[1],
                                                                                 result_2[2],

result_2[3]))
    print("Y1a = {0:.3f}; Y2a = {1:.3f}; Y3a = {2:.3f}; Y4a =
{3:.3f}.".format(Y_average[0],

Y_average[1],

Y_average[2],

Y_average[3]))


    print(Ft)
    if Fp > Ft:
        print('Fp = {} > Ft = {}'.format(round(Fp, 3), Ft))
        print('Рівняння регресії неадекватно оригіналу при рівні значимості
{}'.format(round(q, 2)))
        m += 1
        flag = ct != 0
```

```
            ct -= 1


        else:
            print('Fp = {} < Ft = {}'.format(round(Fp, 3), Ft))
            print('Рівняння регресії адекватно оригіналу при рівні значимості {}'.format(round(q,
2)))
            flag = False
```

Результат виконання програми:

```
andrew at Enigma in Python_Code
 /usr/bin/python3 /home/andrew/My_Code/Python_Code/Mixed/main.py
##########################################################################################


Матриця планування експерименту:
+---+----+-----+----+-----+-----+-----+
| N | x1 |  x2 | x3 |  Y1 |  Y2 |  Y3 |
+---+----+-----+----+-----+-----+-----+
| 1 | -1 |  -1 | -1 | 221 | 216 | 218 |
| 2 | -1 |  -1 | 1  | 197 | 226 | 216 |
| 3 | -1 |  1  | -1 | 226 | 229 | 217 |
| 4 | -1 |  1  | 1  | 205 | 226 | 217 |
| 5 | 1  |  -1 | -1 | 217 | 214 | 225 |
| 6 | 1  |  -1 | 1  | 205 | 214 | 197 |
| 7 | 1  |  1  | -1 | 220 | 211 | 208 |
| 8 | 1  |  1  | 1  | 228 | 227 | 226 |
+---+----+-----+----+-----+-----+-----+
Дисперсії однорідні
Значення після критерія Стюдента:
Y1 = 236.688;   Y2 = 236.688;   Y3 = 240.046;   Y4 = 240.046.
Y1a = 218.333;   Y2a = 213.000;   Y3a = 224.000;   Y4a = 216.000.
2.74
Fp = 1096.166 > Ft = 2.74
Рівняння регресії неадекватно оригіналу при рівні значимості 0.05
##########################################################################################


 Матриця планування експерименту:
 +---+----+-----+----+-----+-----+-----+
 | N | x1 |  x2 | x3 |  Y1 |  Y2 |  Y3 |
 +---+----+-----+----+-----+-----+-----+
 | 1 | -1 |  -1 | -1 | 223 | 206 | 208 |
 | 2 | -1 |  -1 | 1  | 210 | 201 | 198 |
 | 3 | -1 |  1  | -1 | 200 | 196 | 213 |
 | 4 | -1 |  1  | 1  | 230 | 201 | 195 |
 | 5 | 1  |  -1 | -1 | 229 | 230 | 229 |
 | 6 | 1  |  -1 | 1  | 225 | 208 | 198 |
 | 7 | 1  |  1  | -1 | 218 | 203 | 228 |
 | 8 | 1  |  1  | 1  | 224 | 222 | 199 |
 +---+----+-----+----+-----+-----+-----+
 Дисперсії однорідні
 Значення після критерія Стюдента:
 Y1 = 209.065;   Y2 = 209.065;   Y3 = 209.065;   Y4 = 209.065.
 Y1a = 208.750;   Y2a = 205.750;   Y3a = 208.750;   Y4a = 206.250.
 2.422
 Fp = 43.453 > Ft = 2.422
 Рівняння регресії неадекватно оригіналу при рівні значимості 0.05
 ##########################################################################################
```

```
Матриця планування експерименту:
+---+----+-----+----+-----+-----+-----+
| N | x1 |  x2 | x3 |  Y1 |  Y2 |  Y3 |
+---+----+-----+----+-----+-----+-----+
| 1 | -1 |  -1 | -1 | 208 | 218 | 201 |
| 2 | -1 |  -1 | 1  | 230 | 226 | 204 |
| 3 | -1 |  1  | -1 | 213 | 219 | 220 |
| 4 | -1 |  1  | 1  | 229 | 196 | 214 |
| 5 | 1  |  -1 | -1 | 209 | 216 | 200 |
| 6 | 1  |  -1 | 1  | 206 | 204 | 206 |
| 7 | 1  |  1  | -1 | 216 | 206 | 226 |
| 8 | 1  |  1  | 1  | 211 | 207 | 217 |
+---+----+-----+----+-----+-----+-----+
Дисперсії однорідні
Значення після критерія Стюдента:
Y1 = 204.869;   Y2 = 204.869;   Y3 = 204.869;   Y4 = 204.869.
Y1a = 208.800;   Y2a = 215.800;   Y3a = 217.000;   Y4a = 214.000.
2.312
Fp = 279.407 > Ft = 2.312
Рівняння регресії неадекватно оригіналу при рівні значимості 0.05
################################################################################
```

```
Матриця планування експерименту:
+---+----+-----+----+-----+-----+-----+
| N | x1 |  x2 | x3 |  Y1 |  Y2 |  Y3 |
+---+----+-----+----+-----+-----+-----+
| 1 | -1 |  -1 | -1 | 230 | 198 | 212 |
| 2 | -1 |  -1 | 1  | 228 | 204 | 219 |
| 3 | -1 |  1  | -1 | 202 | 213 | 207 |
| 4 | -1 |  1  | 1  | 225 | 216 | 218 |
| 5 | 1  |  -1 | -1 | 226 | 209 | 227 |
| 6 | 1  |  -1 | 1  | 208 | 215 | 230 |
| 7 | 1  |  1  | -1 | 207 | 207 | 210 |
| 8 | 1  |  1  | 1  | 214 | 209 | 227 |
+---+----+-----+----+-----+-----+-----+
Дисперсії однорідні
Значення після критерія Стюдента:
Y1 = 207.403;   Y2 = 207.403;   Y3 = 207.403;   Y4 = 207.403.
Y1a = 210.333;   Y2a = 211.667;   Y3a = 209.500;   Y4a = 209.000.
2.248
Fp = 107.765 > Ft = 2.248
Рівняння регресії неадекватно оригіналу при рівні значимості 0.05
```

Висновок:
На цій лабораторні роботі ми провели повний трьохфакторний експеримент. Знайшли рівняння регресії адекватне об'єкту.