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

Методи оптимізації та планування

Лабораторна робота №3

«ПРОВЕДЕННЯ ТРЬОХФАКТОРНОГО ЕКСПЕРИМЕНТУ З ВИКОРИСТАННЯМ ЛІНІЙНОГО РІВНЯННЯ РЕГРЕСІЇ»

Виконав:
студент групи ІО-82
Вербовський І. М.
Залікова книжка № 8205
Номер у списку групи: 4
Перевірив ас. Регіда П. Г.

Київ 2020 р.

Лістинг програми

```
import random
import numpy
from scipy.stats import t,f

x1_min = 15
x1_max = 45

x2_min = 15
x2_max = 50

x3_min = 15
x3_max = 30

xm_min = (x1_min + x2_min + x3_min) / 3
xm_max = (x1_max + x2_max + x3_max) / 3
y_min = 200 + xm_min
y_max = 200 + xm_max

xn = [[-1, -1, -1],
      [-1, 1, 1],
      [1, -1, 1],
      [1, 1, -1]]

x = [[x1_min, x2_min, x3_min],
     [x1_min, x2_max, x3_max],
     [x1_max, x2_min, x3_max],
     [x1_max, x2_max, x3_min]]

m = 2
y = [[random.randint(int(y_min), int(y_max)) for i in range(m)] for j in range(4)]


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


def table_fisher(prob, d, f3):
    x_vec = [i*0.001 for i in range(int(10/0.001))]
    for i in x_vec:
        if abs(f.cdf(i, 4-d, f3)-prob) < 0.0001:
            return i


def kohren(dispersion, m):
    fisher = table_fisher(0.95, 1, (m - 1) * 4)
    gt = fisher/(fisher+(m-1)-2)
    return max(dispersion) / sum(dispersion) < gt


def student(dispersion_reproduction, m, y_mean, xn):
    dispersion_statistic_mark = (dispersion_reproduction / (4 * m)) ** 0.5

    beta = [1 / 4 * sum(y_mean[j] for j in range(4))]
    for i in range(3):
        b = 0
        for j in range(4):
            b += y_mean[j] * xn[j][i]
        beta.append(1 / 4 * b)

    t = []
    for i in beta:
        t.append(abs(i) / dispersion_statistic_mark)

    f3 = (m - 1) * 4

    return t[0] > table_student(0.95, f3), t[1] > table_student(0.95, f3), t[2] >
table_student(0.95, f3), t[3] > table_student(0.95, f3)
```

```
def normalized_multiplier(x, y_mean):
    mx = [0, 0, 0]
    axx = [0, 0, 0]
    ax = [0, 0, 0]
    for i in range(3):
        for j in range(4):
            mx[i] += x[j][i]
            axx[i] += x[j][i] ** 2
            ax[i] += x[j][i] * y_mean[j]
        mx[i] /= 4
        axx[i] /= 4
        ax[i] /= 4

    my = sum(y_mean) / 4

    a12      = (x[0][0] * x[0][1] + x[1][0] * x[1][1] + x[2][0] * x[2][1] + x[3][0] * x[3][1]) / 4
    a13      = (x[0][0] * x[0][2] + x[1][0] * x[1][2] + x[2][0] * x[2][2] + x[3][0] * x[3][2]) / 4
    a23      = (x[0][1] * x[0][2] + x[1][1] * x[1][2] + x[2][1] * x[2][2] + x[3][1] * x[3][2]) / 4

    a = numpy.array([[1, *mx],
                     [mx[0], axx[0], a12, a13],
                     [mx[1], a12, axx[1], a23],
                     [mx[2], a13, a23, axx[2]]])
    c = numpy.array([my, *ax])
    b = numpy.linalg.solve(a, c)
    return b


def fisher(m, d, y_mean, yo, dispersion_reproduction):

    dispersion_ad = 0
    for i in range(4):
        dispersion_ad += (yo[i] - y_mean[i]) ** 2

    dispersion_ad = dispersion_ad * m / (4 - d)

    fp = dispersion_ad / dispersion_reproduction

    f3 = (m - 1) * 4

    return fp < table_fisher(0.95, d, f3)


while True:
    while True:
        if m > 8:
            print("Current m is more than max number in database of Student criterion. Please
restart")
            exit(0)

        y_mean = []
        for i in range(4):
            y_mean.append(sum(y[i]) / m)

        dispersion = []
        for i in range(len(y)):
            dispersion.append(0)
            for j in range(m):
                dispersion[i] += (y_mean[i] - y[i][j]) ** 2
            dispersion[i] /= m

        dispersion_reproduction = sum(dispersion) / 4

        if kohren(dispersion, m):
            break
        else:
            m += 1
            for i in range(4):
                y[i].append(random.randint(int(y_min), int(y_max)))

    k = student(dispersion_reproduction, m, y_mean, xn)
    d = sum(k)

    b = normalized_multiplier(x, y_mean)
    b = [b[i] * k[i] for i in range(4)]
```

```
    yo = []
    for i in range(4):
        yo.append(b[0] + b[1] * x[i][0] + b[2] * x[i][1] + b[3] * x[i][2])

    if d == 4:
        m += 1
        for i in range(4):
            y[i].append(random.randint(int(y_min), int(y_max)))

    elif fisher(m, d, y_mean, yo, dispersion_reproduction):
        break
    else:
        m += 1
        for i in range(4):
            y[i].append(random.randint(int(y_min), int(y_max)))

#console output
print("\n| № | X1 | X2 | X3 |", end="")

for i in range(m):
    print(" Yi{:d} |".format(i+1), end="")

print()
for i in range(4):
    print("| {:1d} | {:2d} | {:2d} | {:2d} |".format(i+1, *x[i]), end="")
    for j in y[i]:
        print(" {:3d} |".format(j), end="")
    print()

print("\nUsing G(Kohren) - criterion, current dispersion is uniform.")
print("Usind T(Student) - criterion, relevance of \n\tb0 is {}, b1 - {} b2 - {}, b3 -
{}".format(*k))
print("Using F(Fisher) - criterion, current regerecy equation is adequate.")

print("\n\tLinear regrecy equation:\tY = {:.2f}".format(b[0]), end="")
for i in range(1,4):
    if b[i] != 0:
        print(" + {0:.2f}".format(b[i]) + "*X" + str(i), end="")

print("\n\nControl result:")
for i in range(4):
    print("\t\t\t\t\tYs{:d}\t= {:.2f}\n\t b0 + b1*X1 + b2*X2 + b3*X3\t= {:.2f}"
          .format(i+1, y_mean[i], b[0] + b[1] * x[i][0] + b[2] * x[i][1]))
    print()
```

Результат роботи програми

```
| № | X1 | X2 | X3 | Yi1 | Yi2 | Yi3 |
| 1 | 15 | 15 | 15 | 231 | 232 | 215 |
| 2 | 15 | 50 | 30 | 217 | 216 | 223 |
| 3 | 45 | 15 | 30 | 233 | 231 | 231 |
| 4 | 45 | 50 | 15 | 239 | 237 | 240 |

Using G(Kohren) - criterion, current dispersion is uniform.
Usind T(Student) - criterion, relevance of
        b0 is True, b1 - True b2 - False, b3 - True
Using F(Fisher) - criterion, current regerecy equation is adequate.

        Linear regrecy equation:        Y = 226.82 + 0.43*X1 + -0.48*X3

Control result:
                                Ys1     = 226.00
         b0 + b1*X1 + b2*X2 + b3*X3     = 233.24

                                Ys2     = 218.67
         b0 + b1*X1 + b2*X2 + b3*X3     = 233.24

                                Ys3     = 231.67
         b0 + b1*X1 + b2*X2 + b3*X3     = 246.07

                                Ys4     = 238.67
         b0 + b1*X1 + b2*X2 + b3*X3     = 246.07
```

Контрольні запитання

1) У деяких випадках немає необхідності проводити повний факторний експеримент (ПФЕ). Якщо буде використовуватися лінійна регресія, то можливо зменшити кількість рядків матриці ПФЕ до кількості коефіцієнтів регресійної моделі. Кількість дослідів слід скоротити, використовуючи для планування так звані регулярні дробові репліки від повного факторного експерименту, що містять відповідну кількість дослідів і зберігають основні властивості матриці планування – це означає дробовий факторний експеримент (ДФЕ). Репліка, що включає тільки половину експериментів ПФЕ, називається напів-реплікою, що включає четверту частину дослідів – чверть-реплікою і т. д. Дробовий факторний експеримент відповідає всім властивостям повного факторного експерименту. При ПФЕ і ДФЕ використовується кількість рівнів 2, так як нормовані значення факторів в матриці планування приймають два значення -1 або 1. Тобто дробовий факторний експеримент містить у собі 2^(к-1) (де k-кількість факторів) дослідів, які під час знаходження коефіцієнтів для лінійної моделі можуть повністю не використовуватися. Загалом ДФЕ дозволяє знайти 2^к коефіцієнтів регресії при 2^к базисних функціях (для планів більш високого порядку).

2) У випадку ідеальної однорідності дисперсій у рядках коефіцієнт Gp прямував би до значення 1/N, де N - кількість експериментів (кількість рядків у матриці планування). Якщо виконується умова Gp <Gт, то з обраним рівнем статистичної значимості q (з ймовірністю p = 1 - q) усі дисперсії у рядках визнаються однорідними. Якщо Gp> Gт, то гіпотезу відкидають, тобто m недостатньо (m - кількість дослідів). Тоді необхідно збільшити кількість дослідів: m = m +1.

3) Якщо виконується нерівність ts< tтабл, то приймається нуль-гіпотеза, тобто вважається, що знайдений коефіцієнт β s є статистично незначущим і його слід виключити з рівняння регресії. Якщо ts > tтабл то гіпотеза не підтверджується, тобто β s – значимий коефіцієнт і він залишається в рівнянні регресії.

4) Якщо Fp< Fт то отримана математична модель з прийнятим рівнем статистичної значимості q адекватна експериментальним даним.