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

Методи оптимізації та планування

Лабораторна робота №5

**«Проведення трьохфакторного експерименту при використанні**

**рівняння регресії з урахуванням квадратичних членів (центральний ортогональний композиційний план).»**

Виконала:

студенка групи ІО-82

Халіл Хана

Залікова книжка № 8225

Номер у списку групи 23

Перевірив ас. Регіда П. Г.

Київ 2020 р

**Лістинг програми**

import random, numpy

from scipy.stats import t,f

def kohren(mat\_y, m, n):

s = []

for i in range(n):

ks = 0

for j in range(m):

ks += (mat\_y[i][-1] - mat\_y[i][j]) \*\* 2

s.append(ks / m)

gp = max(s) / sum(s)

fisher = table\_fisher(0.95, n, m, 1)

gt = fisher/(fisher+(m-1)-2)

return gp < gt

def geny(n, m, y\_max, y\_min):

mat\_y = [[random.randint(y\_min, y\_max) for j in range(m)]for i in range(n)]

for elem in mat\_y:

elem.append(sum(elem) / len(elem))

return mat\_y

#give combinations of xnat elements or others

def cmb(arr):

return [1, \*arr,

arr[0]\*arr[1],

arr[0]\*arr[2],

arr[1]\*arr[2],

arr[0]\*arr[1]\*arr[2],

arr[0]\*arr[0],

arr[1]\*arr[1],

arr[2]\*arr[2]]

# calculate b koefficients

def get\_b(lmaty):

a00 = [[],

[xnatmod[0]], [xnatmod[1]], [xnatmod[2]],

[xnatmod[0], xnatmod[1]],

[xnatmod[0], xnatmod[2]],

[xnatmod[1], xnatmod[2]],

[xnatmod[0], xnatmod[1], xnatmod[2]],

[xnatmod[0], xnatmod[0]],

[xnatmod[1], xnatmod[1]],

[xnatmod[2], xnatmod[2]]]

def calcxi(n, listx):

sumxi = 0

for i in range(n):

lsumxi = 1

for j in range(len(listx)):

lsumxi \*= listx[j][i]

sumxi += lsumxi

return sumxi

a0 = [15]

for i in range(10):

a0.append(calcxi(n, a00[i + 1]))

a1 = [calcxi(n, a00[i] + a00[1]) for i in range(len(a00))]

a2 = [calcxi(n, a00[i] + a00[2]) for i in range(len(a00))]

a3 = [calcxi(n, a00[i] + a00[3]) for i in range(len(a00))]

a4 = [calcxi(n, a00[i] + a00[4]) for i in range(len(a00))]

a5 = [calcxi(n, a00[i] + a00[5]) for i in range(len(a00))]

a6 = [calcxi(n, a00[i] + a00[6]) for i in range(len(a00))]

a7 = [calcxi(n, a00[i] + a00[7]) for i in range(len(a00))]

a8 = [calcxi(n, a00[i] + a00[8]) for i in range(len(a00))]

a9 = [calcxi(n, a00[i] + a00[9]) for i in range(len(a00))]

a10 = [calcxi(n, a00[i] + a00[10]) for i in range(len(a00))]

a = numpy.array([[a0[0], a0[1], a0[2], a0[3], a0[4], a0[5],

a0[6], a0[7], a0[8], a0[9], a0[10]],

[a1[0], a1[1], a1[2], a1[3], a1[4], a1[5],

a1[6], a1[7], a1[8], a1[9], a1[10]],

[a2[0], a2[1], a2[2], a2[3], a2[4], a2[5],

a2[6], a2[7], a2[8], a2[9], a2[10]],

[a3[0], a3[1], a3[2], a3[3], a3[4], a3[5],

a3[6], a3[7], a3[8], a3[9], a3[10]],

[a4[0], a4[1], a4[2], a4[3], a4[4], a4[5],

a4[6], a4[7], a4[8], a4[9], a4[10]],

[a5[0], a5[1], a5[2], a5[3], a5[4], a5[5],

a5[6], a5[7], a5[8], a5[9], a5[10]],

[a6[0], a6[1], a6[2], a6[3], a6[4], a6[5],

a6[6], a6[7], a6[8], a6[9], a6[10]],

[a7[0], a7[1], a7[2], a7[3], a7[4], a7[5],

a7[6], a7[7], a7[8], a7[9], a7[10]],

[a8[0], a8[1], a8[2], a8[3], a8[4], a8[5],

a8[6], a8[7], a8[8], a8[9], a8[10]],

[a9[0], a9[1], a9[2], a9[3], a9[4], a9[5],

a9[6], a9[7], a9[8], a9[9], a9[10]],

[a10[0], a10[1], a10[2], a10[3], a10[4], a10[5],

a10[6], a10[7], a10[8], a10[9], a10[10]]])

c0 = [calcxi(n, [lmaty])]

for i in range(len(a00) - 1):

c0.append(calcxi(n, a00[i + 1] + [lmaty]))

c = numpy.array([c0[0], c0[1], c0[2], c0[3], c0[4], c0[5],

c0[6], c0[7], c0[8], c0[9], c0[10]])

b = numpy.linalg.solve(a, c)

return b

def table\_student(prob, n, m):

x\_vec = [i\*0.0001 for i in range(int(5/0.0001))]

par = 0.5 + prob/0.1\*0.05

f3 = (m - 1) \* n

for i in x\_vec:

if abs(t.cdf(i, f3) - par) < 0.000005:

return i

def table\_fisher(prob, n, m, d):

x\_vec = [i\*0.001 for i in range(int(10/0.001))]

f3 = (m - 1) \* n

for i in x\_vec:

if abs(f.cdf(i, n-d, f3)-prob) < 0.0001:

return i

def student(n, m, mat\_y):

disp = []

for i in mat\_y:

s = 0

for k in range(m):

s += (i[-1] - i[k]) \*\* 2

disp.append(s / m)

sbt = (sum(disp) / n / n / m) \*\* (0.5)

bs = []

for i in range(11):

ar = []

for j in range(len(mat\_y)):

ar.append(mat\_y[j][-1] \* cmb(xnorm[j])[i] / n)

bs.append(sum(ar))

t = [(bs[i] / sbt) for i in range(11)]

tt = table\_student(0.95, n, m)

st = [i > tt for i in t]

return st

def fisher(b\_0, x\_mod, n, m, d, mat\_y):

if d == n:

return True

disp = []

for i in mat\_y:

s = 0

for k in range(m):

s += (i[-1] - i[k]) \*\* 2

disp.append(s / m)

sad = sum([(sum([cmb(xnat[i])[j] \* b\_0[j] for j in range(11)]) - mat\_y[i][-1]) \*\* 2 for i in range(n)])

sad = sad \* m / (n - d)

fp = sad / sum(disp) / n

ft = table\_fisher(0.95, n, m, d)

return fp < ft

def all\_print():

titles\_x = ["№", "X1", "X2", "X3", "X1\*X2", "X1\*X3", "X2\*X3", "X1\*X2\*X3", "X1^2", "X2^2", "X3^2"]

# cycle for pretty printing title of table with normal parameters

for j in range(11):

s = ""

if j == 0:

s = " {:2s} " # for №

if j >= 1 and j < 4:

s = " {:6s} " # for X0

if j >= 4 and j < 7:

s = " {:7s} " # for X + num

if j == 7:

s = " {:8s} " # for X\*X, with different combinations

if j > 7 and j < 11:

s = " {:6s} " # for X\*X\*X

print(s.format(titles\_x[j]), end="") # taking all titles from list

# this cycle is used for printing Yi in title of table

for i in range(m):

print(" Yi{:d} ".format(i+1), end="")

# printing Y middle, Y experimental and dispersion

print(" Ys Ye ", end="")

print()

# fill table with data

for i in range(n):

print(" {:2d} ".format(i), end="")

for j in range(1, 11):

s = ""

if j >= 1 and j < 4:

s = " {:6.2f} "

if j >= 4 and j < 7:

s = " {:7.2f} "

if j == 7:

s = " {:8.2f} "

if j > 7 and j < 11:

s = " {:6.2f} "

print(s.format(cmb(xnat[i])[j]), end="")

for j in maty[i][:-1]:

print(" {:3d} ".format(j), end="")

print(" {:6.2f} {:6.2f} "

.format(maty[i][-1],

sum([cmb(xnat[i])[j] \* b0[j] \* dmas[j] for j in range(11)])), end="")

print()

print("\nФункція відгуку зі значущими коефіцієнтами:\n\tY = ", end="")

if dmas[0] != 0:

print("{:.5f}".format(b0[0]), end="")

for i in range(1, 11):

if dmas[i] != 0:

print(" + {:.5f}\*{}".format(b0[i], titles\_x[i]), end="")

print()

l = 1.215

x1min = -10

x1max = 10

x01 = (x1min + x1max) / 2

xl1 = l\*(x1max-x01)+x01

x2min = -10

x2max = 2

x02 = (x2min + x2max) / 2

xl2 = l\*(x2max-x02)+x02

x3min = -10

x3max = 4

x03 = (x3min + x3max) / 2

xl3 = l\*(x3max-x03)+x03

ymax = round(200 + (x1max + x2max + x3max) / 3)

ymin = round(200 + (x1min + x2min + x3min) / 3)

xnorm = [[-1, -1, -1],

[-1, 1, 1],

[1, -1, 1],

[1, 1, -1],

[-1, -1, 1],

[-1, 1, -1],

[1, -1, -1],

[1, 1, 1],

[-l, 0, 0],

[l, 0, 0],

[0, -l, 0],

[0, l, 0],

[0, 0, -l],

[0, 0, l],

[0, 0, 0]]

xnat = [[x1min, x2min, x3min],

[x1min, x2min, x3max],

[x1min, x2max, x3min],

[x1min, x2max, x3max],

[x1max, x2min, x3min],

[x1max, x2min, x3max],

[x1max, x2max, x3min],

[x1max, x2max, x3max],

[-xl1, x02, x03],

[xl1, x02, x03],

[x01, -xl2, x03],

[x01, xl2, x03],

[x01, x02, -xl3],

[x01, x02, xl3],

[x01, x02, x03]]

n = 15

m = 3

while True:

while True:

print("\nПоточний m = {}\n".format(m))

xnatmod = [[xnat[i][j] for i in range(15)] for j in range(3)]

maty = geny(n, m, ymax, ymin)

matymod = [maty[i][-1] for i in range(len(maty))]

kohren\_flag = kohren(maty, 3, 15)

print("Дисперсія {}однорідна, з ймовірністю = {:.2}"

.format("" if kohren\_flag else "не ", 0.95))

if kohren\_flag:

break

else:

m += 1

b0 = get\_b(matymod)

dmas = student(n, m, maty)

d = sum(dmas)

fishercheck = fisher(b0, xnatmod, n, m, d, maty)

print("Рівняння {}адекватне, з ймовірністю = {:.2f}\n"

.format("" if fishercheck else "не ", 0.95))

all\_print()

if fishercheck:

break

**Результати роботи програми**

Поточний m = 3

Дисперсія однорідна, з ймовірністю = 0.95

Рівняння адекватне, з ймовірністю = 0.95

№ X1 X2 X3 X1\*X2 X1\*X3 X2\*X3 X1\*X2\*X3 X1^2 X2^2 X3^2 Yi1 Yi2 Yi3 Ys Ye

0 -10.00 -10.00 -10.00 100.00 100.00 100.00 -1000.00 100.00 100.00 100.00 204 202 200 202.00 196.16

1 -10.00 -10.00 4.00 100.00 -40.00 -40.00 400.00 100.00 100.00 16.00 204 196 195 198.33 195.25

2 -10.00 2.00 -10.00 -20.00 100.00 -20.00 200.00 100.00 4.00 100.00 194 191 205 196.67 198.89

3 -10.00 2.00 4.00 -20.00 -40.00 8.00 -80.00 100.00 4.00 16.00 202 201 195 199.33 197.98

4 10.00 -10.00 -10.00 -100.00 -100.00 100.00 1000.00 100.00 100.00 100.00 200 199 195 198.00 196.16

5 10.00 -10.00 4.00 -100.00 40.00 -40.00 -400.00 100.00 100.00 16.00 201 191 203 198.33 195.25

6 10.00 2.00 -10.00 20.00 -100.00 -20.00 -200.00 100.00 4.00 100.00 196 193 198 195.67 198.89

7 10.00 2.00 4.00 20.00 40.00 8.00 80.00 100.00 4.00 16.00 191 203 194 196.00 197.98

8 -12.15 -4.00 -3.00 48.60 36.45 12.00 -145.80 147.62 16.00 9.00 204 194 204 200.67 197.30

9 12.15 -4.00 -3.00 -48.60 -36.45 12.00 145.80 147.62 16.00 9.00 199 197 195 197.00 197.30

10 0.00 -3.29 -3.00 -0.00 -0.00 9.87 0.00 0.00 10.82 9.00 198 196 200 198.00 198.27

11 0.00 3.29 -3.00 0.00 -0.00 -9.87 -0.00 0.00 10.82 9.00 201 197 194 197.33 198.27

12 0.00 -4.00 -5.51 -0.00 -0.00 22.02 0.00 0.00 16.00 30.31 198 195 204 199.00 198.35

13 0.00 -4.00 5.51 -0.00 0.00 -22.02 -0.00 0.00 16.00 30.31 205 199 199 201.00 198.35

14 0.00 -4.00 -3.00 -0.00 -0.00 12.00 0.00 0.00 16.00 9.00 200 204 194 199.33 198.12

Функція відгуку зі значущими коефіцієнтами:

Y = 198.47760 + -0.00557\*X1^2 + -0.02847\*X2^2 + 0.01088\*X3^2