**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РФ**

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ **«БЕЛГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ ИМ. В. Г. ШУХОВА»**

Кафедра программного обеспечения вычислительной техники и автоматизированных систем

**ЛАБОРАТОРНАЯ РАБОТА №5**

**Дисциплина: Системный анализ**

**Тема: Уравнение Регрессии**

Выполнил: ст. группы ВТ-31

Подкопаев Антон Валерьевич

Проверил: проф. ПО и ВТАС

Полунин Александр Иванович

**Белгород 2020**

**Цель лабораторной работы**: получить математическую модель процесса, заданного его числовыми значениями при разных значениях двух аргументов. Числовые значения процесса измеряются со случайной ошибкой, распределенной по нормальному закону, мат. ожидание ошибки равно нулю, дисперсия неизвестна. Процесс описывается алгебраической зависимостью.

**Вариант 9**

**Изображение выглядит как текст, доска

Автоматически созданное описание**

**Ход выполнения лабораторной работы**

Данная задача решается при помощи регрессионного анализа. На первом шаге модели представляют в виде

где — оценки неизвестных параметров, — регрессоры.

На основании экспериментальных данных составляют матрицу:

где — значение j-го регрессора в i-ом эксперименте.

Изображение выглядит как компьютер, монитор, внутренний, снимок экрана

Автоматически созданное описание

Методом наименьших квадратов находим неизвестные коэффициенты:

где Y – вектор значений исследуемого процесса, полученный в экспериментах. Для этого вычислим среднее арифметическое результатов эксперимента:

и величины ,

где — значение модели, полученное при наборе регрессоров, вычисленных для i-го эксперимента.

Изображение выглядит как компьютер, монитор, внутренний, снимок экрана

Автоматически созданное описание

Изображение выглядит как снимок экрана, монитор, внутренний, компьютер

Автоматически созданное описание

По эти значениям вычисляем оценки дисперсий:

Чтобы выяснить, есть ли корреляция между реальный процессом и его моделью, вычисляем функцию Фишера:

Задаем уровень значимости 0.05. Если , коэффициент множественной корреляции R значим.

Изображение выглядит как снимок экрана, монитор, внутренний, компьютер

Автоматически созданное описание

Распределение Фишера F (0,05 ; 2,9) = 4,26, т.е. коэффициент множественной корреляции R значим

На третьем шаге проверяют значимость коэффициентов уравнения регрессии. Для этого оцениваем по результатам экспериментов дисперсию неизвестной погрешности ε, влияющей на эксперимент.

Вычислим дисперсии оценки коэффициентов математической модели. Для этого используем зависимости

=(FTF)-1, = .

Изображение выглядит как снимок экрана, монитор, внутренний, компьютер

Автоматически созданное описание

Для проверки значимости вычисленного коэффициента вычисляем величину

.

Изображение выглядит как снимок экрана, монитор, внутренний, компьютер

Автоматически созданное описание

Коэффициент Стьюдента: F (0,05 ; 2,7) = tα = 2.2621

145,70545664 > 2,2621 ; 110779,76932031 > 2.2621 ; 7570,79836378 > 2.2621

Тогда trp=tα , все , то его нельзя объяснить действием случайных факторов, он значим и его можно использовать в модели.

На четвертом шаге вычисляют доверительные интервалы для коэффициентов уравнения регрессии.

Изображение выглядит как снимок экрана, монитор, внутренний, компьютер

Автоматически созданное описание

**Результат работы программы**

**Изображение выглядит как компьютер, монитор, внутренний, снимок экрана

Автоматически созданное описаниеИзображение выглядит как компьютер, монитор, внутренний, снимок экрана

Автоматически созданное описание**

**Изображение выглядит как компьютер, монитор, внутренний, снимок экрана

Автоматически созданное описание**

Список возможных полиномов:

5. x₁x₂² + x₁

12. x₁x₂² + x₂

18. x₁x₂² + x₁x₂

23. x₁x₂² + x₁²

27. x₁x₂² + x₂²

31. x₁²x₂ + x₁x₂²

32. x₁^3 + x₁x₂²

33. x₂^3 + x₁x₂²

Лучшее уравнение регрессии:

0.99892 x₁x₂² + 6.15533 x₂ + 7.99605

Rr: 0.9998007281128493

Rост: 0.9998007281128498

Дисперсия гауссовского шума по Y:

21.036426572654157

Изображение выглядит как снимок экрана, компьютер, внутренний, ноутбук

Автоматически созданное описание

*Приложение 1*

from itertools import \*

import numpy as np

import matplotlib.pyplot as plt

import math

import scipy

import scipy.optimize as opt

import scipy.integrate as integrate

def r0(x1,x2):

return 0

def r1(x1,x2):

return 1

def rx1(x1,x2):

return x1

def rx2(x1,x2):

return x2

def rx1\_2(x1,x2):

return x1\*x1

def rx2\_2(x1,x2):

return x2\*x2

def rx2x1(x1,x2):

return x2\*x1

arm=[rx1,rx2,rx2\_2,rx2x1]

def Get\_F(i:int,RRR):#возвр.матр. F

count\_r=len(RRR[i])

F=np.zeros((n1,count\_r ))#M: 12 x кол-во регрессоров

for it in range(n1):

for j in range(count\_r):

F[it][j]=RRR[i][j](X1[it][0],X2[it][0])

return F

def Get\_res\_s(it:int,A):

xx0=[]

xx1=[]

xx2=[]

yy0=[]

yynew=[]

for i in range(n1):

xx0.append(i)

xx1.append(X1[i][0])

xx2.append(X2[i][0])

yy0.append(Y0[i][0])

tmp=0.0

for j in range(len(A)):

tmp+=A[j][0]\*RRR[it][j](X1[i][0],X2[i][0]) #Y^=Fa^

yynew.append(tmp) #y^=sum<i=1, k>(a^\_i\*f\_i(x))

return xx0,xx1,xx2,yy0,yynew

def Get\_Grafic(xx0,yy0,yynew):

fig, ax = plt.subplots()

ax.scatter(xx0, yy0)

ax.plot(xx0, yy0, 'r', lw=2, label="Theoretical")

ax.plot(xx0, yynew, 'b', lw=2, label="Fit")

ax.legend()

ax.set\_xlim(0, 13)

ax.set\_xlabel(r"$x$", fontsize=18)

ax.set\_ylabel(r"$y$", fontsize=18)

plt.show()

return

n1=12

n2=10

##вектор x1,x2,y

X1=np.array([

[7.0],

[5.0],

[7.0],

[2.0],

[11.0],

[3.0],

[3.0],

[9.0],

[3.0],

[7.0],

[4.0],

[5.0],

[4.0]

])

X2=np.array([

[7.0],

[12.0],

[15.0],

[1.0],

[7.0],

[2.0],

[5.0],

[15.0],

[15.0],

[8.0],

[14.0],

[10.0],

[6.0]

])

Y0=np.array([

[383.9852410],

[787.7938352],

[1671.2846123],

[13.3291853],

[583.8121339],

[32.8034594],

[115.0882527],

[2125.4422674],

[772.1392667],

[527.2359452],

[883.1937554],

[572.7850655],

])

Y=np.array([

182.2517787,

188.0421263,

185.3350833,

189.5512556,

192.6351647,

193.5865340,

182.4558612,

182.4558612,

194.8749390,

190.5872336

])

def Get\_Qo(yy0,yynew):

Qo=0.0 #Q остаток

for i in range(n1):

Qo+=(yy0[i]-yynew[i])\*\*2

return Qo

def Get\_RRR():

tf=[]

for i in range(2, len(arm)):

j = combinations(arm, i)

tf+=list(j)

return tf

global r\_min

r\_min=999999.9

global r\_num

r\_num=0

RRR=Get\_RRR()

for iterat in range(len(RRR)):

F=Get\_F(iterat,RRR)

A=(np.linalg.inv(((F.transpose()).dot(F))).dot(F.transpose())).dot(Y0)

xx0,xx1,xx2,yy0,yynew=Get\_res\_s(iterat,A)

Qoo=Get\_Qo(yy0,yynew)

print('i =', iterat,'Q\_ост =', Qoo)

if(r\_min>Qoo):

r\_min=Qoo

r\_num=iterat

F=Get\_F(r\_num,RRR)

print('F =\n',F)

A=(np.linalg.inv(((F.transpose()).dot(F))).dot(F.transpose())).dot(Y0) #a^

print('A =\n',A)

xx0,xx1,xx2,yy0,yynew=Get\_res\_s(r\_num,A)

print('i =', r\_num, 'Q\_ост =', r\_min, 'func =', RRR[r\_num]) #STR\_RRR

Get\_Grafic(xx0,yy0,yynew)

############################################################################

my\_t=np.mean(yy0)#MY теор

print('мат. ожидание теор. =',my\_t)

Q=0.0 #Q теор

Qr=0.0 #Q реал

Qo=0.0 #Q остаток

for i in range(n1):

Q += (yy0[i]-my\_t)\*\*2

Qr += (yynew[i]-my\_t)\*\*2

Qo += (yy0[i]-yynew[i])\*\*2

R\_2=Qr/Q

R\_22=1.0-Qo/Q

print('Q\_ост =', Qo,'Q\_реал =', Qr,'Q\_теор=', Q)

print('R^2 =',R\_2,'=',R\_22)

K = len(A) #кол-во регрессоров

Sr\_2 = Qr/(K-1)

So\_2 = Qo/(n1-K)

FF = Sr\_2/So\_2

print('F =', FF)

print('K =', K, '; N - K =', n1 - K, '; F\_t({}, {}, {}) = 4,26'.format(0.05, K-1, n1-K))

y\_13\_22=np.average(Y)

q\_e\_2=0.0 #(σ^)^2\_e

l=len(Y)

for i in range(l):

q\_e\_2+=((Y[i]-y\_13\_22)\*\*2)/(l-1)

q\_e\_2=np.sqrt(q\_e\_2)

#print(F)

#print(q\_e\_2)

C = np.abs(np.linalg.inv(np.dot(F.transpose(),F)))

q\_a = np.zeros(len(C))

for i in range(len(q\_a)):

q\_a[i] = np.sqrt(C[i][i])\*q\_e\_2

print('σ^\_a:\n', q\_a)

q\_a = q\_a\*\*2

T = np.zeros(len(q\_a))

sk = np.abs(np.sqrt(A[0][0]\*A[0][0]+A[1][0]\*A[1][0]+A[2][0]\*A[2][0]))

for i in range(len(q\_a)):

T[i] = sk/q\_a[i]

print('T =', T)

q\_a = np.sqrt(q\_a)

t\_r = 2.2621

for i in range(len(q\_a)):

print("Довер.интервалы t[{}];".format(i+1), " ", A[i][0]-t\_r\*q\_a[i], " <= ", A[i][0], " <= ", A[i][0]+t\_r\*q\_a[i])

print(q\_e\_2)

*Приложение 2*

from itertools import combinations

import matplotlib.pyplot as plt

import numpy as np

subscript\_dict = str.maketrans("0123456789", "₀₁₂₃₄₅₆₇₈₉")

# Возвращает все возможные полиномы из набора слагаемых polynom\_parts размером polynom\_parts\_count

def get\_polynom\_list(polynom\_parts, polynom\_parts\_count=3):

polynoms = []

combination\_list = combinations(range(len(polynom\_parts)), polynom\_parts\_count)

for combination in combination\_list:

combination = sorted(combination, reverse=True)

result\_element = []

for part\_num in combination:

result\_element.append(polynom\_parts[part\_num])

polynoms.append(result\_element)

return polynoms

# Возвращает строковые имена функций из списка functions

def get\_functions\_labels(functions):

labels = []

for func in functions:

labels.append(func.\_\_name\_\_)

return labels

# Получение матриц F для каждого полинома в polynoms\_list

def get\_all\_F\_matrices(polynoms\_list, x\_vectors):

F\_matrices = []

for polynom in polynoms\_list:

F\_matrix = []

for x\_vector in x\_vectors:

F\_matrix\_str = []

for polynom\_part in polynom:

F\_matrix\_str.append(polynom\_part(x\_vector[0], x\_vector[1]))

F\_matrix.append(F\_matrix\_str)

F\_matrices.append(np.array(F\_matrix))

return F\_matrices

# Получение векторов а для каждого полинома в списке polynoms\_list

def get\_all\_a\_vectors(F\_matrices, Y\_vector):

a\_vectors = []

for F\_matrix in F\_matrices:

a\_vector = np.linalg.inv((F\_matrix.T.dot(F\_matrix))).dot(

F\_matrix.T.dot(Y\_vector)

)

a\_vectors.append(a\_vector)

return a\_vectors

def get\_best\_polynom(F\_matrices, a\_vectors, Y\_vector,polynoms):

# Вычисляем y c крышечкой

y\_cap\_vectors = []

for F, a in zip(F\_matrices, a\_vectors):

y\_cap\_vectors.append(F.dot(a))

R2\_1\_list = []

R2\_2\_list = []

Q = sum(map(lambda y: (y - np.mean(Y\_vector)) \*\* 2, Y\_vector))

for index,y\_cap\_vector in enumerate(y\_cap\_vectors):

QR = sum(map(lambda y\_cap\_i: (y\_cap\_i - np.mean(Y\_vector)) \*\* 2, y\_cap\_vector))

Qstop = sum(

map(lambda y\_cap\_i, Y\_i: (Y\_i - y\_cap\_i) \*\* 2, y\_cap\_vector, Y\_vector)

)

# Вычислим коэффициент детерминации 2 способами

R2\_1 = QR / Q

R2\_2 = 1 - Qstop / Q

R2\_1\_list.append(R2\_1)

R2\_2\_list.append(R2\_2)

print('{}. '.format(index+1),get\_regression\_equation(get\_functions\_labels(polynoms[index]),a\_vectors[index]), ' Rr:' ,R2\_1,'Rост: ',R2\_2)

# Возвращаем индекс максимального R2 (соответственно индекс полинома) для двух способов

return (R2\_1\_list.index(max(R2\_1\_list)), R2\_2\_list.index(max(R2\_2\_list)))

def get\_regression\_equation(polynom\_labels, a\_vector):

result\_str = ""

for index, label in enumerate(polynom\_labels, start=1):

result\_str += str(round(a\_vector[index - 1],5))

result\_str += " " + label + " + "

return result\_str[:-3]

import matplotlib.pyplot as plt

from mpl\_toolkits.mplot3d import Axes3D

print(lab5\_regression.get\_regression\_equation

(

lab5\_regression.get\_functions\_labels(polynoms[best2\_index])

,a\_vectors[best2\_index])

)

x1\_list,x2\_list = [],[]

for i in x\_vectors:

x1\_list.append(i[0])

x2\_list.append(i[1])

y\_cap\_vector = F\_matrices[best2\_index].dot(a\_vectors[best2\_index])

fig = plt.figure()

axes = Axes3D(fig)

axes.scatter(np.array(x1\_list), np.array(x2\_list), np.array(Y\_vector),label = 'Исходные данные')

axes.scatter(np.array(x1\_list), np.array(x2\_list), np.array(y\_cap\_vector),label = 'Полученные значения')

plt.show()