**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО**

**ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ  
ВЫСШЕГО ОБРАЗОВАНИЯ  
**«БЕЛГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ  
ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ им. В.Г.ШУХОВА»  
(БГТУ им. В.Г.Шухова)**

Кафедра программного обеспечения вычислительной техники и автоматизированных систем

Лабораторная работа №5

Дисциплина: Системный анализ

по теме Линейная регрессия

Выполнил: ст. группы ВТ-31  
Новожен Н.В

Проверил: Полунин А.И

**Белгород 2020**

**Цель 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.

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

где — оценки неизвестных параметров, — регрессоры.

**ВАРИАНТ 7**

|  |  |  |
| --- | --- | --- |
| Yi | x1 | x2 |
| 276.6234933 | 10.0 | 11.0 |
| 63.8808072 | 6.0 | 4.0 |
| 324.6093583 | 5.0 | 15.0 |
| 69.0876417 | 6.0 | 4.0 |
| 70.7131909 | 8.0 | 3.0 |
| 217.9847897 | 6.0 | 11.0 |
| 273.6863587 | 4.0 | 14.0 |
| 488.8796705 | 11.0 | 16.0 |
| 80.0760474 | 4.0 | 6.0 |
| 164.2341817 | 5.0 | 9.0 |
| 428.4742487 | 8.0 | 16.0 |
| 72.1589693 | 3.0 | 6.0 |
| 142.4072056 |  |  |
| 145.5690592 |  |  |
| 144.0908624 |  |  |
| 146.3931282 |  |  |
| 148.0771151 |  |  |
| 148.5966160 |  |  |
| 142.5186460 |  |  |
| 142.5556745 |  |  |
| 149.3001572 |  |  |
| 146.9588302 |  |  |

На основании экспериментальных данных составляют матрицу:

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

**F= [ 10. 121. 110.]**

**[ 6. 16. 24.]**

**[ 5. 225. 75.]**

**[ 6. 16. 24.]**

**[ 8. 9. 24.]**

**[ 6. 121. 66.]**

**[ 4. 196. 56.]**

**[ 11. 256. 176.]**

**[ 4. 36. 24.]**

**[ 5. 81. 45.]**

**[ 8. 256. 128.]**

**[ 3. 36. 18.]**

Методом наим. 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.

**a= [4.75412987 ,1.02300112 ,0.99189688]**

мат.ожидание теор= 210.86739645

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 коррелиции R значим.

**F= 4740.912024426484**

**Распред. Фишера F(0.05,2,9)=19.38 ,т.у коэффициент множественной коррелиции R значим ,математическая модель процесса содержит аргументы действительного процесса и необходимо дальше уточнять модель**

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, = .

Для проверки значимости вычисленного коэффициента вычисляем величину

.

ti=[ **60.98217924 ; 18925.37071603 ; 2839.99046995**]

Коэффициент Стьюдента: F(0.05,2,7)=tα=2.2621

**60.98217924 > 2.2621**

**18925.37071603 > 2.2621**

**2839.99046995> 2.2621**

**Тогда** tгр=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.

**Довер.интервалы t\_i= 1 ;**

**4.1087939042741395 <= 4.754129874261401 <= 5.399465844248662**

**Довер.интервалы t\_i= 2 ;**

**0.9863687373116362 <= 1.0230011181057503 <= 1.0596334988998646**

**Довер.интервалы t\_i= 3 ;**

**0.8973322104733225 <= 0.9918968782988733 <= 1.086461546124424**

**Листинг программы:**

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])

    yynew.append(tmp)

  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([

   [10.0],

   [6.0],

   [5.0],

   [6.0],

   [8.0],

   [6.0],

   [4.0],

   [11.0],

   [4.0],

   [5.0],

   [8.0],

   [3.0],

   [3.0]

])

X2=np.array([

   [11.0],

   [4.0],

   [15.0],

   [4.0],

   [3.0],

   [11.0],

   [14.0],

   [16.0],

   [6.0],

   [9.0],

   [16.0],

   [6.0],

   [10.0]

])

Y0=np.array([

   [276.6234933],

   [63.8808072],

   [324.6093583],

   [69.0876417],

   [70.7131909],

   [217.9847897],

   [273.6863587],

   [488.8796705],

   [80.0760474],

   [164.2341817],

   [428.4742487],

   [72.1589693],

])

Y=np.array([

   142.4072056,

   145.5690592,

   144.0908624,

   146.3931282,

   148.0771151,

   148.5966160,

   142.5186460,

   142.5556745,

   149.3001572,

   146.9588302

])

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

**# MAIN**

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)

  #print('A=',A)

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

  #print(yy0,yynew)

  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)

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

print('A=',A)

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

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

print('Q остаток=',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)

#Sr\_2=0#?????????????????????????????????????????????????????????

So\_2=Qo/(n1-K)

FF=Sr\_2/So\_2

print('F=',FF)

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

q\_e\_2=0.0

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

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

print('q\_a=',q\_a)

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  i=",i+1,"   ;",A[i][0]-t\_r\*q\_a[i]," <= ",A[i][0]," <= ",A[i][0]+t\_r\*q\_a[i])