**ИДЗ №6  
Обработка двумерной выборки**

**Оценивание параметров двумерной случайной величины**

**Вариант №7**

*Работу выполнили:*Батманов Даниил, P3207

Шорников Сергей, P3211

**Цель работы:**

На основании анализа двумерной выборки:

1. Обработать двумерную выборку
2. Оценить параметры двумерной случайной величины

**Исходные данные:**

Изображение выглядит как текст, Шрифт, число, снимок экрана

Автоматически созданное описание

**Ход работы:**

|  |  |  |  |
| --- | --- | --- | --- |
|  | **22** | **32** | **42** |
|  |  |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **27** | **37** | **42** | **47** |
|  |  |  |  |  |

Оценка мат. ожидания:

Оценка дисперсии:

Оценка мат. ожидания:

Оценка дисперсии:

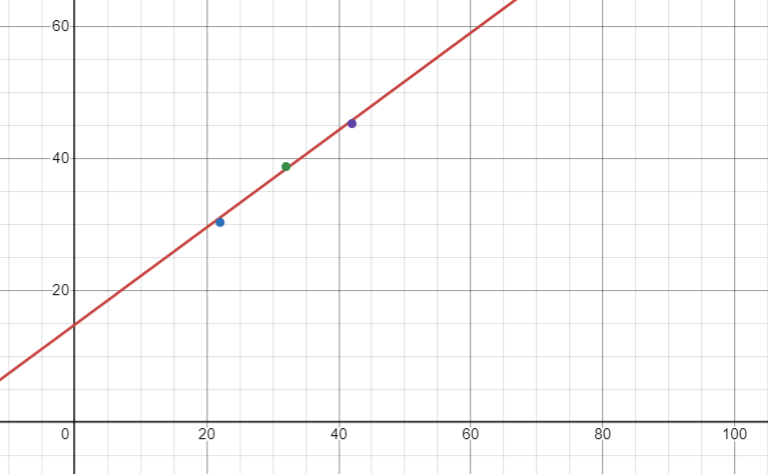
Корреляционная матрица:

Зависимость условного математического ожидания компоненты Y от значений компоненты X

|  |  |  |  |
| --- | --- | --- | --- |
|  | 22 | 32 | 42 |
|  | 30.33 | 38,8 | 45.33 |

Уравнение выборочной линии регрессии:

= 14.839255499153978 + 0.7377326565143824 \* x

****

*Исходный код:*

*import* math;  
  
matrix = [[20, 10, 0, 0],  
 [0, 80, 45, 0],  
 [0, 0, 15, 30]]  
yLabels = [27, 37, 42, 47]  
xLabels = [22, 32, 42]  
  
xDispersion = 0  
yDispersion = 0  
xExpectedValue = 0  
yExpectedValue = 0  
  
*# ищем мат ожидания  
for* i *in* range(len(xLabels)):  
 buff = 0  
 *for* j *in* range(len(yLabels)):  
 buff += matrix[i][j]  
 xDispersion += (buff \* xLabels[i] \* xLabels[i])  
 xExpectedValue += buff \* xLabels[i]  
  
xExpectedValue /= 200  
print("Мат ожидание X: ", xExpectedValue)  
  
  
*for* i *in* range(len(yLabels)):  
 buff = 0  
 *for* j *in* range(len(xLabels)):  
 buff += matrix[j][i]  
 yDispersion += (buff \* yLabels[i] \* yLabels[i])  
 yExpectedValue += buff \* yLabels[i]  
  
yExpectedValue /= 200  
print("Мат ожидание Y: ", yExpectedValue)  
  
xDispersion /= 200  
yDispersion /= 200  
  
yDispersion -= yExpectedValue \* yExpectedValue  
xDispersion -= xExpectedValue \* xExpectedValue  
print("disp X: ", xDispersion)  
print("disp Y: ", yDispersion)  
  
  
  
xSko = math.sqrt(xDispersion)  
ySko = math.sqrt(yDispersion)  
print("sko X: ", xSko)  
print("sko Y: ", ySko)  
covariance = 0  
  
*for* i *in* range(len(xLabels)):  
 *for* j *in* range(len(yLabels)):  
 covariance += xLabels[i] \* yLabels[j] \* matrix[i][j] / 200  
covariance -= yExpectedValue \* xExpectedValue  
  
print("correlation", covariance)  
coefficientOfCovariance = covariance / xSko / ySko  
  
correlationMatrix = [[xExpectedValue, covariance],  
 [covariance, yExpectedValue]]  
print("matrix:\n", correlationMatrix)  
  
*# матрица точек линии регрессии*matrixOfP = []  
matrixOfPx = []  
*for* i *in* range(len(xLabels)):  
 buffMatrix = []  
 buff = 0  
 *for* j *in* range(len(yLabels)):  
 buff += matrix[i][j] / 200  
 buffMatrix.append(matrix[i][j] / 200)  
 matrixOfP.append(buffMatrix)  
 matrixOfPx.append(buff)  
  
finalMatrix = [xLabels, [0, 0, 0]]  
  
*for* i *in* range(len(xLabels)):  
 buff = 0  
 *for* j *in* range(len(yLabels)):  
 buff += (matrixOfP[i][j] / matrixOfPx[i]) \* yLabels[j]  
 finalMatrix[1][i] = buff  
print("Коэффициент корреляции:", coefficientOfCovariance)  
print("Матрица функции регрессии:", \*finalMatrix)  
print("Выборочная линия регрессии:", yExpectedValue - coefficientOfCovariance \* (ySko / xSko) \* xExpectedValue, "+",  
 coefficientOfCovariance \* (ySko / xSko), "\* x")

*Вывод программы:*

Мат ожидание X: 32.75

Мат ожидание Y: 39.0

disp X: 36.9375

disp Y: 28.5

sko X: 6.077622890571609

sko Y: 5.338539126015656

correlation 27.25

matrix:

[[32.75, 27.25], [27.25, 39.0]]

Коэффициент корреляции: 0.8398666328966911

Матрица функции регрессии: [22, 32, 42] [30.333333333333332, 38.8, 45.333333333333336]

Выборочная линия регрессии: 14.839255499153978 + 0.7377326565143824 \* x

**Вывод:**

В результате выполнения задания нам удалось оценить параметры двумерной случайной величины, найти выборочную линию регрессии, матрицу функции регрессии и коэффициент корреляции.