|  |  |  |  |
| --- | --- | --- | --- |
| МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ  КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ імені Тараса Шевченка  ФАКУЛЬТЕТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ  **Кафедра програмних систем і технологій**  Дисципліна  **«Ймовірнісні основи програмної інженерії»**  **Лабораторна робота № 3**  **«Двовимірна статистика»** | | | |
| **Виконав:** | Бойко Костянтин Богданович | **Перевірила**: |  |
| Група | ІПЗ-24(1) | Дата перевірки |  |
| Форма навчання | денна | Оцінка |  |
| Спеціальність | 121 |
| 2022 | | | |

**Мета**: Навчитись використовувати на практиці набуті знання про міри в двовимірній статистиці.

**Завдання**

1. Намалюйте діаграму розсіювання для даних. Укажіть, чи існує тренд у даних. Якщо так, то вкажіть, чи є це негативним трендом, чи позитивним.

2. Знайдіть центр ваги і коваріацію.

3. Знайти рівняння лініїї регресії y від x.

4. Розрахуйте коефіцієнт кореляції між даними.

5. Зробити висновок про залежності.

**Хід роботи**

*Математична модель*

Для побудови діаграми розсіювання відокремимо суму покупки та час проведений в магазині. Поставимо у відповідність на осі ОХ та ОY

Щоб знайти тренд, ми аналізуємо графік та перевіряємо чи перший елемент відрізняється від останнього. Якщо він буде більшим, то тренд є зростаючий(негативний), якщо він меншим буде, то спадаючий(негативний)

*Псевдокод алгоритму*

def trend(data):

if max(data) == data[len(data)-1]:

print("Trend of data is positive")

File.write("\nTrend of data is positive")

elif min(data) == data[len(data)-1]:

print("Trend of data is negative")

File.write("\nTrend of data is negative")

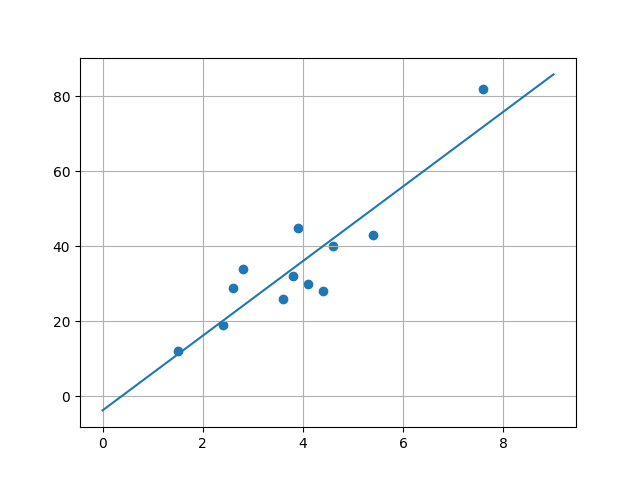
else:

print("The data does not have any trend")

File.write("\nThe data does not have any trend")

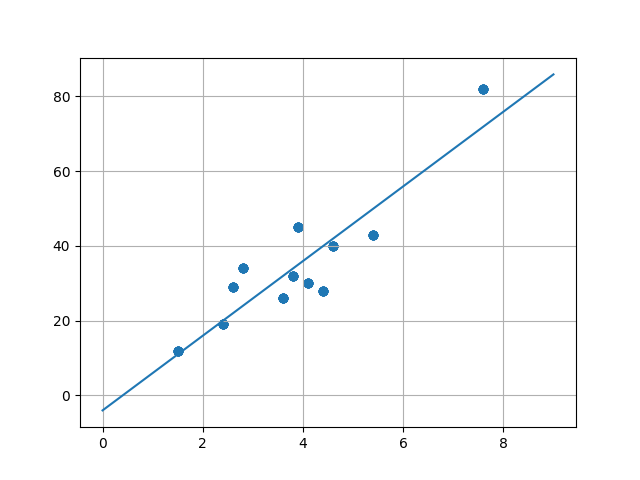
*Результат на консолі та випробування алгоритму*

Для 10 елементів



**

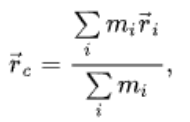
Для 100 елементів



****

*Математична модель*

Центр ваги ми будемо шукати за формулою

Де:

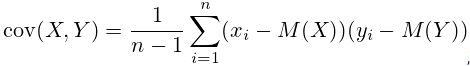
rc – Центр ваги

mi – поточний елемент X

ri – поточний елемент Y

За означенням, коваріація – це певна величина, яка відображає те, наскільки дві випадкові величино спільно змінюються щодо їхніх значень.

Щоб розрахувати її, будемо викорстовувати формулу:



cov(X, Y) – коваріація

xi – поточний елемент x

xi – поточний елемент y

M(X) – Середнє значення x

M(Y) – Середнє значення y

n – кількість елементів

*Псевдокод алгоритму*

def centerofweight(x, y):

chis = 0

znam = 0

for i in range(len(data)):

chis += x[i]\*y[i]

znam += x[i]

centerofweight = chis/znam

print("\nGravitycenter = " + str(centerofweight))

File.write("\n\nGravitycenter = " + str(centerofweight))

def covariance(x, y):

global averagex, averagey

covarience = 0.0

for i in range(len(x)):

averagex += x[i]

averagey += y[i]

averagex = averagex / len(x)

averagey = averagey / len(y)

for i in range(len(x)):

covarience += (x[i] - averagex) \* (y[i] - averagey)

covarience = covarience / (len(x)-1)

print("Covarince: ", covarience)

File.write("Covarince: " + str(covarience))

*Результат на консолі та випробування алгоритму*

Для 10 елементів:

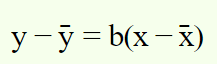


Для 100 елементів:



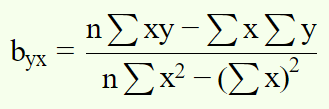
*Математична модель*

Використаємо формулу:



Для знаходження лінії регресії.

Щоб знайти b, використаємо наступну формулу:



*Псевдокод алгоритму*

def lineofregression(X, Y):

global averagex, averagey

byx, sumx, sumy, sumxy, sumx2, sumy2 = 0.0, 0.0, 0.0, 0.0, 0.0, 0.0

for i in range(len(X)):

sumx += X[i]

sumy += Y[i]

sumxy += X[i] \* Y[i]

sumx2 += X[i] \* X[i]

sumy2 += Y[i] \* Y[i]

byx = (len(X) \* sumxy - (sumx \* sumy)) / ((len(X) \* sumx2) - sumx\*\*2)

x, y = sp.symbols("x,y")

line = sp.Eq(y-averagey, byx\*(x-averagex))

linex = sp.solve(line, y)

liney = sp.solve(line, x)

x1 = np.linspace(0, 9)

y1 = byx \* (x1 - averagex) + averagey

plt.plot(x1, y1)

strlinex = str(linex)

strliney = str(liney)

strlinex = strlinex.replace("[", "")

strlinex = strlinex.replace("]", "")

strliney = strliney.replace("[", "")

strliney = strliney.replace("]", "")

print("Line of regression of y on x")

print("x = " + strliney)

print("y = " + strlinex, "\t(y on x)")

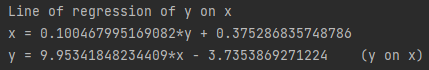
File.write("Line of regression of y on x\n")

File.write("x = " + strliney)

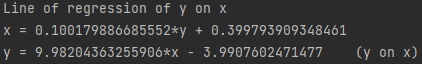
File.write("\ny = " + strlinex + "\t(y on x)")

*Результат на консолі та випробування алгоритму*

Для 10 елементів:

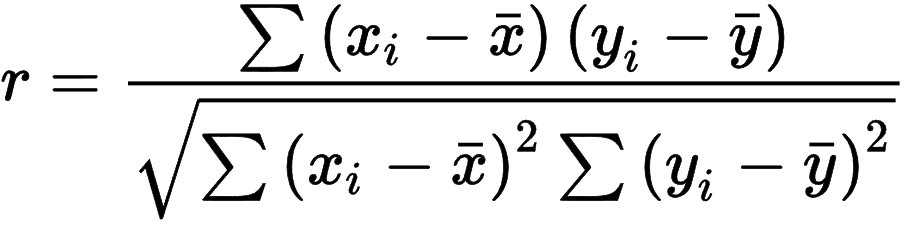


Для 100 елементів:



*Математична модель*

Знаходження кореляції між даними використовуємо формулу:



Де: r - кореляція

*Псевдокод алгоритму*

def correlation(x, y):

global averagex, averagey

corcoef, sum1, sum2, sum3 = 0.0, 0.0, 0.0, 0.0

for i in range(len(x)):

sum1 += (x[i] - averagex) \* (y[i] - averagey)

sum2 += (x[i] - averagex) \* (x[i] - averagex)

sum3 += (y[i] - averagey) \* (y[i] - averagey)

sum2 = sum2 \* sum3

corcoef = sum1/math.sqrt(sum2)

print("Correlation coefficient:", corcoef)

File.write("\nCorrelation coefficient:" + str(corcoef))

*Результат на консолі та випробування алгоритму*

Для 10 елементів:



Для 100 елементів:



**5.** Проаналізувавши всі ми можемо сказати, що залежність y від x є прямопропорційною. Тобто, зі збільшенням значення х, значення у збільшується у відповідь. Відповідно, зі збільшенням у – збільшується х.

**Висновок:**

Під час виконання третьої лабораторної роботи було повторено операції з вхідними даними записаних у txt файл, реалізовано потрібні формули для знаходження інформації за завданнями, також побудовано діаграму розсіювання. Навчились застосовувати набуті знання про міри в двовимірній статистиці, аналізувати дані та діаграму.