ГУАП

КАФЕДРА № 43

|  |  |  |
| --- | --- | --- |
| ОТЧЕТ ЗАЩИЩЕН С ОЦЕНКОЙ: |  |  |

ПРЕПОДАВАТЕЛЬ:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| доцент, к.т.н., доцент | / |  | / |  | / | В. В. Мышко |
| (должность, учёная степень, звание) |  | (подпись) |  | (дата защиты) |  | (инициалы, фамилия) |

ОТЧЕТ О ЛАБОРАТОРНОЙ РАБОТЕ №4

«**Однофакторный**

**регрессионный анализ**»

ПО КУРСУ: «ОБРАБОТКА ЭКСПЕРИМЕНТАЛЬНЫХ ДАННЫХ»

|  |  |  |  |
| --- | --- | --- | --- |
| РАБОТУ ВЫПОЛНИЛ СТУДЕНТ: | 4134к | / | Н.А. Костяков |
|  | (номер группы) |  | (инициалы, фамилия) |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | / |  | / |  |
|  |  | (подпись студента) |  | (дата отчета) |

Санкт-Петербург 2025

**Задание на лабораторную работу**  
  
На основе заданного массива данных:

• построить уравнение регрессии в виде алгебраического полинома второй

степени;

• проверить адекватность уравнения регрессии;

• проверить значимость коэффициентов регрессии.

Расчеты произвести в скалярной и матричной форме.

Порядок выполнения задания:

1. Составить систему нормальных уравнений, используя массив

экспериментальных данных (таблица 4.1);

2. Найти оценки коэффициентов регрессии посредством решения системы

нормальных уравнений;

3. При расчетах в матричной форме составить матричное уравнение с вектором

неизвестных оценок коэффициентов регрессии и найти его решение;

4. Проверить адекватность построенного уравнения регрессии

экспериментальным данным по критерию Фишера при уровне значимости α

= 0,01;

5. Проверить значимость коэффициентов регрессии по критерию Стьюдента

при таком же уровне значимости;

6. Повторно проверить адекватность уравнения регрессии после исключения

незначимых коэффициентов.

Вариант 88



**Ход работы**

**1. Составить систему нормальных уравнений, используя массив**

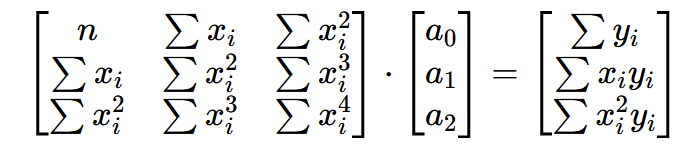
**экспериментальных данных**

Для регрессии второй степени уравнение будет иметь вид:

y=a0​+a1​x+a2​x2

Нужно найти коэффициенты a0, a1, a2

Для этого составим систему нормальных уравнений:



**2. Найти оценки коэффициентов регрессии посредством решения системы**

**нормальных уравнений**

Воспользуюсь пакетом Numpy:

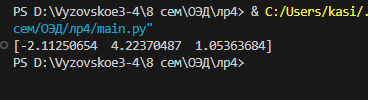


рисунок 1 — результат поиска коэффициентов

Получаем такие коэффициенты:

a1 = **-2.11250654**

a2 = **4.22370487**

a3 = **1.05363684**

Листинг программы для поиска коэффициентов указан в приложении 1

**3. При расчетах в матричной форме составить матричное уравнение с вектором**

**неизвестных оценок коэффициентов регрессии и найти его решение**

Для расчётов в матричной форме, мы можем записать систему нормальных уравнений как матричное уравнение:

**A \* b = Y**

**где:**

A — это матрица коэффициентов, которая зависит от значений x,

b — вектор коэффициентов регрессии ,

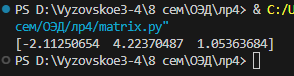
Y — вектор правых частей, который содержит суммы **y, x\*y и x2\*y**

Аналогичные расчеты, только применяя матричный метод показал такие же результаты:

a1 = **-2.11250654**

a2 = **4.22370487**

a3 = **1.05363684**



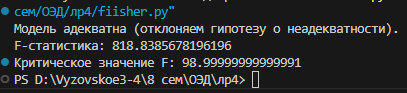
Листинг программы для поиска коэффициентов матричным методом указан в приложении 1

**4. Проверить адекватность построенного уравнения регрессии**

**экспериментальным данным по критерию Фишера при уровне значимости α**

**= 0,01;**

Для проверки адекватности модели с использованием критерия Фишера нужно вычислить значение F-критерия и сравнить его с критическим значением из таблицы распределения Фишера при уровне значимости a=0.01



Результат программы проверки критерием Фишера

Если F статистическое больше F критического, значит гипотеза о неадекватности отклоняется

F статистическое = **818.8385**

F критическое = **98.(9)**

**Гипотеза о неадекватности отклоняется**

Листинг программы проверки критерием Фишера описан в приложении 1

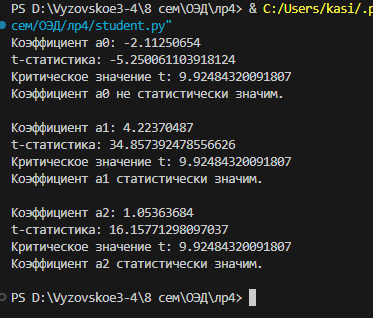
**5. Проверить значимость коэффициентов регрессии по критерию Стьюдента**

**при таком же уровне значимости;**

Критерием Стьюдента проверим, какие критерии значимые, а какие нет. Для этого написал программу на python 3.12, вот результат ее работы и листинг

Вывод программы

|  |
| --- |
| Коэффициент a0: -2.11250654  t-статистика: -5.250061103918124  Критическое значение t: 9.92484320091807  **Коэффициент a0 не статистически значим.**  Коэффициент a1: 4.22370487  t-статистика: 34.857392478556626  Критическое значение t: 9.92484320091807  **Коэффициент a1 статистически значим.**  Коэффициент a2: 1.05363684  t-статистика: 16.15771298097037  Критическое значение t: 9.92484320091807  **Коэффициент a2 статистически значим.** |



**Листинг программы проверки коэффициентом Стьюдента указан в приложении 1**

**6. Повторно проверить адекватность уравнения регрессии после исключения**

**незначимых факторов.**

Из уравнения убираем незначимые коэффициенты и получаем уравнение

b = np.array([4.22370487, 1.05363684])

y\_pred = b[0] \* x + b[1] \* x\*\*2

**Получаем такой результат:**

Модель адекватна (отклоняем гипотезу о неадекватности).

F-статистика: **68.63175410255741**

Критическое значение F: **34.116221564529795**

**Листинг программы проверки адекватности со значимыми коэффициентами указан в приложении 1**

**Приложение 1**

**Листинги программ**

**Листинг программы поиска коэффициентов**

|  |
| --- |
| import numpy as np  # Данные  x = np.array([-3, -2, -1, 0, 3])  y = np.array([-5, -7, -5, -2, 20])  # Вычисление сумм для системы нормальных уравнений  n = len(x)  sum\_x = np.sum(x)  sum\_x2 = np.sum(x\*\*2)  sum\_x3 = np.sum(x\*\*3)  sum\_x4 = np.sum(x\*\*4)  sum\_y = np.sum(y)  sum\_x\_y = np.sum(x \* y)  sum\_x2\_y = np.sum(x\*\*2 \* y)  # Составляем матрицу и вектор правых частей  A = np.array([[n, sum\_x, sum\_x2],                [sum\_x, sum\_x2, sum\_x3],                [sum\_x2, sum\_x3, sum\_x4]])  b = np.array([sum\_y, sum\_x\_y, sum\_x2\_y])  # Решение системы линейных уравнений для коэффициентов регрессии  coefficients = np.linalg.solve(A, b)  # Выводим коэффициенты  print(coefficients) |

**Листинг программы используя матричный метод указан**

|  |
| --- |
| import numpy as np  # Данные  x = np.array([-3, -2, -1, 0, 3])  y = np.array([-5, -7, -5, -2, 20])  # Вычисление сумм для системы нормальных уравнений  n = len(x)  sum\_x = np.sum(x)  sum\_x2 = np.sum(x\*\*2)  sum\_x3 = np.sum(x\*\*3)  sum\_x4 = np.sum(x\*\*4)  sum\_y = np.sum(y)  sum\_x\_y = np.sum(x \* y)  sum\_x2\_y = np.sum(x\*\*2 \* y)  # Составляем матрицу A и вектор Y  A = np.array([[n, sum\_x, sum\_x2],                [sum\_x, sum\_x2, sum\_x3],                [sum\_x2, sum\_x3, sum\_x4]])  Y = np.array([sum\_y, sum\_x\_y, sum\_x2\_y])  # Решение матричного уравнения для коэффициентов регрессии  b = np.linalg.inv(A).dot(Y)  # Выводим коэффициенты  print(b) |

**Листинг программы проверки методом Фишера**

|  |
| --- |
| import numpy as np  from scipy.stats import f  # Данные  x = np.array([-3, -2, -1, 0, 3])  y = np.array([-5, -7, -5, -2, 20])  # Предсказанные значения y по уравнению регрессии  b = np.array([-2.11250654, 4.22370487, 1.05363684])  y\_pred = b[0] + b[1] \* x + b[2] \* x\*\*2  # Сумма квадратов отклонений от регрессии (SQR\_res)  SQR\_res = np.sum((y - y\_pred)\*\*2)  # Сумма квадратов регрессии (SQR\_reg)  SQR\_reg = np.sum((y\_pred - np.mean(y))\*\*2)  # Степени свободы  k = 3  # количество коэффициентов (a0, a1, a2)  n = len(x)  # количество наблюдений  # Средняя квадратичная ошибка для регрессии и для ошибок  MSE\_reg = SQR\_reg / (k - 1)  MSE\_res = SQR\_res / (n - k)  # Расчет F-статистики  F\_stat = MSE\_reg / MSE\_res  # Критическое значение F для заданных степеней свободы  F\_critical = f.ppf(1 - 0.01, k - 1, n - k)  # Сравнение с критическим значением  if F\_stat > F\_critical:      print("Модель адекватна (отклоняем гипотезу о неадекватности).")  else:      print("Модель неадекватна (не отклоняем гипотезу о неадекватности).")  # Вывод результатов  print(f"F-статистика: {F\_stat}")  print(f"Критическое значение F: {F\_critical}") |

**Листинг программы для проверки критерием Стьюдента**

|  |
| --- |
| import numpy as np  from scipy.stats import t  # Данные  x = np.array([-3, -2, -1, 0, 3])  y = np.array([-5, -7, -5, -2, 20])  # Степени свободы  n = len(x)  k = 3  # Количество коэффициентов (a0, a1, a2)  # Матрица X для полиномиальной регрессии второй степени  X = np.vstack([np.ones(n), x, x\*\*2]).T  # Оценки коэффициентов регрессии (a0, a1, a2)  b = np.array([-2.11250654, 4.22370487, 1.05363684])  # Расчет остатков (ошибок)  y\_pred = X.dot(b)  # Предсказанные значения  residuals = y - y\_pred  # Средняя квадратичная ошибка  SQR\_res = np.sum(residuals\*\*2)  MSE\_res = SQR\_res / (n - k)  # Обратная матрица (X^T X)^(-1)  XTX\_inv = np.linalg.inv(X.T @ X)  # Стандартные ошибки для коэффициентов  standard\_errors = np.sqrt(MSE\_res \* np.diagonal(XTX\_inv))  # t-статистики для коэффициентов  t\_stats = b / standard\_errors  # Степени свободы для критического значения t  df = n - k  # Критическое значение t для уровня значимости α = 0.01  t\_critical = t.ppf(1 - 0.01 / 2, df)  # Проверка значимости коэффициентов  for i in range(k):      print(f"Коэффициент a{i}: {b[i]}")      print(f"t-статистика: {t\_stats[i]}")      print(f"Критическое значение t: {t\_critical}")        if abs(t\_stats[i]) > t\_critical:          print(f"Коэффициент a{i} статистически значим.")      else:          print(f"Коэффициент a{i} не статистически значим.")      print() |

**Повторная оценка на адекватность**

|  |
| --- |
| import numpy as np  from scipy.stats import f  # Данные  x = np.array([-3, -2, -1, 0, 3])  y = np.array([-5, -7, -5, -2, 20])  # Предсказанные значения y по уравнению регрессии  b = np.array([4.22370487, 1.05363684])  y\_pred = b[0] \* x + b[1] \* x\*\*2  # Сумма квадратов отклонений от регрессии (SQR\_res)  SQR\_res = np.sum((y - y\_pred)\*\*2)  # Сумма квадратов регрессии (SQR\_reg)  SQR\_reg = np.sum((y\_pred - np.mean(y))\*\*2)  # Степени свободы  k = 2  # количество коэффициентов (a0, a1, a2)  n = len(x)  # количество наблюдений  # Средняя квадратичная ошибка для регрессии и для ошибок  MSE\_reg = SQR\_reg / (k - 1)  MSE\_res = SQR\_res / (n - k)  # Расчет F-статистики  F\_stat = MSE\_reg / MSE\_res  # Критическое значение F для заданных степеней свободы  F\_critical = f.ppf(1 - 0.01, k - 1, n - k)  # Сравнение с критическим значением  if F\_stat > F\_critical:      print("Модель адекватна (отклоняем гипотезу о неадекватности).")  else:      print("Модель неадекватна (не отклоняем гипотезу о неадекватности).")  # Вывод результатов  print(f"F-статистика: {F\_stat}")  print(f"Критическое значение F: {F\_critical}") |