|  |  |  |
| --- | --- | --- |
| Министерство науки и высшего образования  Российской Федерации | | |
| Федеральное государственное бюджетное  образовательное учреждение высшего образования | | |
| «Новосибирский государственный технический университет» | | |
|  | | |
| Кафедра теоретической и прикладной информатики | | |
|  | | |
| Лабораторная работа № 3 | | |
| по дисциплине «Статистические методы анализа данных» | | |
|  | | |
|  | | |
|  | | |
|  | Факультет: | ПМИ |
| Группа: | ПМИ-02 |
| Вариант: | 6 |
| Студент: | Сидоров Даниил, |
|  | Дюков Богдан |
| Преподаватель: | Попов Александр Александрович. |
|  |  |
|
|  |  |
| Новосибирск | | |
| 2023 | | |

1. **Постановка задачи**
2. Изменить модель регрессии, добавив в нее дополнительный регрессор, ранее не вошедший в состав модели, порождающей данные. Не генерируя новых данных, найти точечные оценки всех параметров расширенной модели. В дальнейшем при рассмотрении этой расширенной модели анализе должно быть показано, что параметр при дополнительном регрессоре незначим.
3. Построить доверительные интервалы для каждого параметра модели регрессии.
4. Проверить гипотезу о незначимости каждого параметра модели.
5. Проверить гипотезу о незначимости самой регрессии.
6. Рассчитать прогнозные значения для математического ожидания функции отклика для всего интервала действия одного из факторов, зафиксировав значения других факторов на границе или в центре области их определения.
7. По полученным в п. 5 прогнозным значениям построить графики прогнозных значений и доверительной полосы для математического ожидания функции отклика и для самого отклика.
8. Заново смоделировать исходные данные, увеличив мощность случайной помехи до 50…70 % от мощности полезного сигнала, и провести оценку параметров. Повторить пункты 3, 4 с новыми данными.
9. **Ход работы**

**Добавление нового регрессора**

Старая модель:

;

= .

Оценки старой модели:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **theta** | **theta\_hat** | **sigma\_squared** | **sigma\_hat\_squared** | **F** |
| 1 | 1,501426421 | 1,957897413 | 2,22151394 | 1,134642666 |
| 2 | 2,336565661 |  |  |  |
| 5 | 5,062817987 |  |  |  |
| 4 | 3,938056574 |  |  |  |
| 1,5 | 1,354423461 |  |  |  |
| 2,5 | 2,27845587 |  |  |  |
| 0,02 | -0,340012046 |  |  |  |
| 0,01 | 0,026997102 |  |  |  |

Модель является адекватной.

После добавления нового регрессора:

;

= .

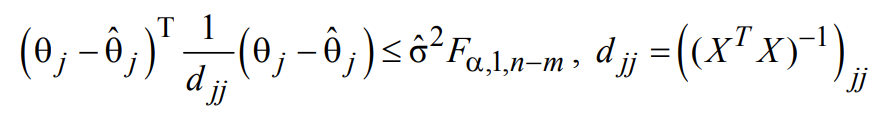
Оценки расширенной модели:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **theta** | **theta\_hat** | **sigma\_squared** | **sigma\_hat\_squared** | **F** |
| 1 | 1,501426421 | 1,957897413 | 2,311921867 | 1,180818694 |
| 2 | 2,336565661 |  |  |  |
| 5 | 4,185311994 |  |  |  |
| 4 | 3,938056574 |  |  |  |
| 1,5 | 1,354423461 |  |  |  |
| 2,5 | 2,27845587 |  |  |  |
| 0,02 | -0,340012046 |  |  |  |
| 0,01 | 0,026997102 |  |  |  |
| 0 | 0,961661846 |  |  |  |

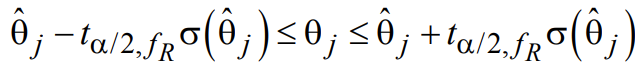
Модель является адекватной.

**Построение доверительных интервалов**

После преобразования формулы:



Получим двустороннее неравенство вида:

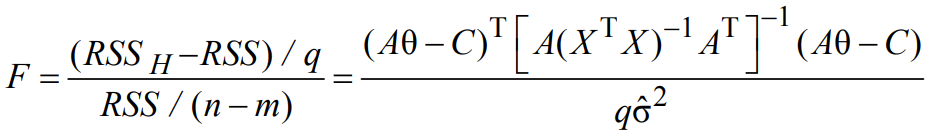
,

где - квантиль распределения Стьюдента, , при уровне значимости 0.05. Квантиль получился равным 2.13.

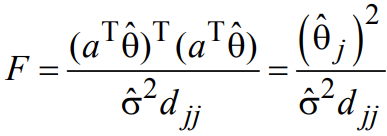
|  |  |  |  |
| --- | --- | --- | --- |
| **left\_interval** | **theta** | **theta\_hat** | **right\_interval** |
| 0,090534878 | 1 | 1,501426421 | 2,912317964 |
| 1,526348119 | 2 | 2,336565661 | 3,146783203 |
| 1,001767791 | 5 | 4,185311994 | 7,368856196 |
| 3,276516721 | 4 | 3,938056574 | 4,599596426 |
| 0,266319358 | 1,5 | 1,354423461 | 2,442527564 |
| 1,468238328 | 2,5 | 2,27845587 | 3,088673412 |
| -1,743349994 | 0,02 | -0,340012046 | 1,063325901 |
| -1,457774198 | 0,01 | 0,026997102 | 1,511768402 |
| -2,388585087 | 0 | 0,961661846 | 4,311908778 |

**Проверка гипотезы о незначимости каждого параметра модели**

Статистика в общем виде:



Частный случай при , где - вектор-строка, в которой на j-м месте стоит единица, на остальных местах – нули. Тогда статистика:

,

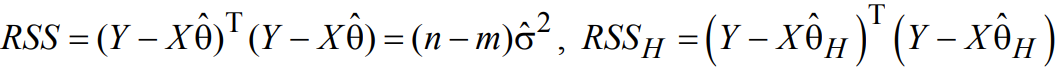
где - j-й диагональный элемент матрицы . .

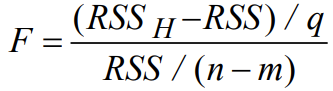
|  |  |  |  |
| --- | --- | --- | --- |
| **F\_values** | **sign** | **F\_quantile** | **Hypothesis** |
| 5,144829196 | > | 4,543077165 | Не принимается |
| 37,7835543 | > | 4,543077165 | Не принимается |
| 7,852067438 | > | 4,543077165 | Не принимается |
| 160,9911456 | > | 4,543077165 | Не принимается |
| 7,039116513 | > | 4,543077165 | Не принимается |
| 35,92758891 | > | 4,543077165 | Не принимается |
| 0,266694576 | < | 4,543077165 | Принимается |
| 0,001501984 | < | 4,543077165 | Принимается |
| 0,374318463 | < | 4,543077165 | Принимается |

В результате имеем то, что незначимы оказались параметры , , .

**Проверка гипотезы о незначимости самой регрессии**

Требуется установить, является ли регрессия с заданными регрессорами значимой, т. е. дает лучшее описание, чем модель среднего. Гипотеза H имеет вид: . Тогда:



,

где – квантиль распределения Фишера, при уровне значимости 0.05.

|  |  |  |  |
| --- | --- | --- | --- |
| **F\_regression** | **sign** | **F\_quantile\_regression** | **Hypothesis** |
| 48,7394423 | > | 2,640796883 | Не принимается |

**Расчет прогнозных значений для математического ожидания функции отклика**

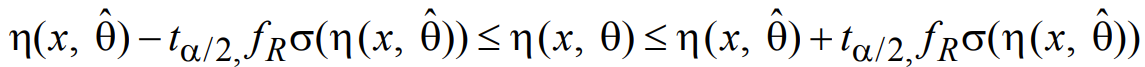
Рассчитаем прогнозные значения для математического ожидания функции отклика для всего интервала действия фактора , зафиксировав значения фактора на границе, а фактора в центре области определения. Формула расчета:



|  |  |  |  |
| --- | --- | --- | --- |
| **x1** | **x2** | **x3** |  |
| 0 | -1 | 1 | 0,319506258 |
| 0 | -0,913043478 | 1 | 0,9086414 |
| 0 | -0,826086957 | 1 | 1,458349395 |
| 0 | -0,739130435 | 1 | 1,972424092 |
| 0 | -0,652173913 | 1 | 2,454659341 |
| 0 | -0,565217391 | 1 | 2,908848992 |
| 0 | -0,47826087 | 1 | 3,338786893 |
| 0 | -0,391304348 | 1 | 3,748266895 |
| 0 | -0,304347826 | 1 | 4,141082848 |
| 0 | -0,217391304 | 1 | 4,5210286 |
| 0 | -0,130434783 | 1 | 4,891898002 |
| 0 | -0,043478261 | 1 | 5,257484904 |
| 0 | 0,043478261 | 1 | 5,621583154 |
| 0 | 0,130434783 | 1 | 5,987986603 |
| 0 | 0,217391304 | 1 | 6,360489101 |
| 0 | 0,304347826 | 1 | 6,742884496 |
| 0 | 0,391304348 | 1 | 7,138966638 |
| 0 | 0,47826087 | 1 | 7,552529378 |
| 0 | 0,565217391 | 1 | 7,987366565 |
| 0 | 0,652173913 | 1 | 8,447272048 |
| 0 | 0,739130435 | 1 | 8,936039677 |
| 0 | 0,826086957 | 1 | 9,457463302 |
| 0 | 0,913043478 | 1 | 10,01533677 |
| 0 | 1 | 1 | 10,61345394 |

**Графики прогнозных значений и доверительной полосы для математического ожидания функции отклика и для самого отклика**

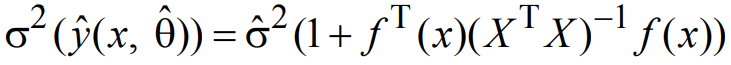
Интервальная оценка для истинного значения математического ожидания функции отклика:

,

где - квантиль распределения Стьюдента, , при уровне значимости 0.05 и .

Для построения доверительного интервала для отклика будем использовать предыдущую формулу за тем исключением, что дисперсия оценки отклика

будет определяться как:

.

**Моделирование с увеличенной мощностью случайной помехи**

Пусть , тогда:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **y** | **y\_hat** | **y-y\_hat** | **theta** | **theta\_hat** | **sigma\_squared** | **sigma\_hat\_squared** |
| -11,4025911 | -8,328386352 | -3,074204752 | 1 | 2,876165873 | 27,41056378 | 32,36690614 |
| 0,138103151 | -4,134044048 | 4,272147199 | 2 | 3,259313392 |  |  |
| 2,139184299 | -4,257471159 | 6,396655458 | 5 | 1,951716604 |  |  |
| -3,315422575 | -0,063128855 | -3,252293721 | 4 | 3,768228922 |  |  |
| -2,7135193 | -3,341220145 | 0,627700846 | 1,5 | 0,955302469 |  |  |
| -6,789761971 | 0,853122159 | -7,64288413 | 2,5 | 1,67105777 |  |  |
| -0,391516789 | 0,860860216 | -1,252377005 | 0,02 | -1,32704173 |  |  |
| 8,980458625 | 5,05520252 | 3,925256105 | 0,01 | 0,073597333 |  |  |
| -11,27262693 | -6,368391469 | -4,904235465 | 0 | 3,598209149 |  |  |
| -1,382369624 | 1,168066375 | -2,550435999 |  |  |  |  |
| -1,050578692 | -1,657423621 | 0,606844929 |  |  |  |  |
| 14,0758054 | 5,879034223 | 8,196771175 |  |  |  |  |
| -3,765407456 | -0,110672978 | -3,654734478 |  |  |  |  |
| 10,25232098 | 7,425784866 | 2,826536119 |  |  |  |  |
| 7,288035958 | 4,731460037 | 2,55657592 |  |  |  |  |
| 9,19059568 | 12,26791788 | -3,077322202 |  |  |  |  |
| -5,257635191 | -7,062480046 | 1,804844856 |  |  |  |  |
| 8,267977499 | 3,816093338 | 4,451884161 |  |  |  |  |
| -2,01991709 | -1,711459544 | -0,308457546 |  |  |  |  |
| -2,472406456 | 9,16711384 | -11,6395203 |  |  |  |  |
| 3,168408391 | 0,465790729 | 2,702617662 |  |  |  |  |
| 16,48512809 | 11,34436411 | 5,140763981 |  |  |  |  |
| 4,446745972 | 5,947976399 | -1,501230426 |  |  |  |  |
| 16,17564739 | 16,82654978 | -0,650902392 |  |  |  |  |

|  |
| --- |
| **F** |
| 1,180818695 |

Гипотеза о незначимости каждого параметра:

|  |  |  |  |
| --- | --- | --- | --- |
| **F\_values** | **sign** | **F\_quantile** | **Hypothesis** |
| 1,348536404 | < | 4,543077165 | Принимается |
| 5,251350867 | > | 4,543077165 | Не принимается |
| 0,121964608 | < | 4,543077165 | Принимается |
| 10,52893902 | > | 4,543077165 | Не принимается |
| 0,250128387 | < | 4,543077165 | Принимается |
| 1,380389739 | < | 4,543077165 | Принимается |
| 0,290179481 | < | 4,543077165 | Принимается |
| 0,00079731 | < | 4,543077165 | Принимается |
| 0,374318462 | < | 4,543077165 | Принимается |

Гипотеза о незначимости самой регрессии:

|  |  |  |  |
| --- | --- | --- | --- |
| **F\_regression** | **sign** | **F\_quantile\_regression** | **Hypothesis** |
| 3,667903206 | > | 2,640796883 | Не принимается |

1. **Код программы**

import numpy as np

import pandas as pd

from scipy.stats import t

from scipy.stats import f

np.random.seed(56)

# Генерация комбинаций факторов

def generate\_combinations(x1\_levels, x2\_levels, x3\_levels):

x1\_list = []

x2\_list = []

x3\_list = []

for i in x1\_levels:

for j in x2\_levels:

for k in x3\_levels:

x1\_list.append(i)

x2\_list.append(j)

x3\_list.append(k)

return map(np.array, [x1\_list, x2\_list, x3\_list])

# Сохранение датафрейма df в файл csv с именем filename

def save\_to\_csv(df, filename):

df.to\_csv(filename, index=False, encoding='utf-8-sig')

# Получение дисперсии шума

def get\_sigma\_squared(u):

# Вычисление мощности сигнала

omega\_squared = np.dot(u - np.mean(u), u - np.mean(u)) / (len(u) - 1)

# Доля от мощности сигнала

rho = 0.7

# Вычисление дисперсии шума

return rho \* omega\_squared

# Получение ошибки

def get\_noise(u, sigma\_squared):

return np.random.normal(0, np.sqrt(sigma\_squared), len(u))

# ------------------ЗАДАНИЕ 1----------------------#

# Определение параметров

theta = np.array([1, 2, 5, 4, 1.5, 2.5, 0.02, 0.01, 0])

# Определение уровней для каждого фактора

x1\_levels = np.array([-1, 0, 1])

x2\_levels = np.array([-1, -0.33, 0.33, 1])

x3\_levels = np.array([-1, 1])

# Генерация комбинаций факторов

x1, x2, x3 = generate\_combinations(x1\_levels, x2\_levels, x3\_levels)

# Вычисление дисперсии шума

sigma\_squared = 1.957897413

# Зашумленный отклик

y = [

-7.421921045,

-2.139324492,

-2.090986381,

-1.350575114,

-1.69529944,

-0.586504844,

0.650083657,

5.35306572,

-8.867318951,

-0.362126435,

-3.687216239,

6.217390024,

-1.994746844,

7.613558733,

3.420614445,

9.791004984,

-8.710567529,

4.429908371,

-4.660703112,

4.743967873,

-0.130615528,

12.95403137,

3.408641288,

16.0689259

]

# Создание матрицы X

X = np.column\_stack((np.ones(len(x1)), x1, x2, x3, x1\*x2, x1\*x3, x1\*\*2, x2\*\*2, x2\*\*3))

# Вычисление вектора параметров модели theta\_hat

theta\_hat = np.linalg.inv(X.T @ X) @ X.T @ y

y\_hat = X @ theta\_hat

# Вычисление вектора остатков

e\_hat = y - y\_hat

# Вычисление несмещенной оценки

sigma\_hat\_squared = e\_hat.T @ e\_hat / (len(y) - len(theta))

# Вычисление F-статистики

F=sigma\_hat\_squared/sigma\_squared

# Сохранение всех датафреймов в файлы csv

save\_to\_csv(pd.concat([

pd.DataFrame({'y': y}),

pd.DataFrame({'y\_hat': y\_hat}),

pd.DataFrame({'y-y\_hat': y-y\_hat}),

pd.DataFrame({'theta': theta}),

pd.DataFrame({'theta\_hat': theta\_hat}),

pd.DataFrame({'sigma\_squared': [sigma\_squared]}),

pd.DataFrame({'sigma\_hat\_squared': [sigma\_hat\_squared]}),

pd.DataFrame({'F': [F]})], axis=1), 'Task1.csv')

# ------------------ЗАДАНИЕ 2----------------------#

# Вычисление степеней свободы

f\_R = len(y) - len(theta)

# Вычисление квантиля распределения Стьюдента

t\_quantile = t.ppf(1 - 0.05 / 2, f\_R)

# Вычисление дисперсии оценок параметров

var\_theta\_hat = np.diag(np.linalg.inv(X.T @ X)) \* sigma\_hat\_squared

# Вычисление стандартного отклонения оценок параметров

std\_theta\_hat = np.sqrt(var\_theta\_hat)

# Вычисление доверительных интервалов

left\_interval = theta\_hat - t\_quantile \* std\_theta\_hat

right\_interval = theta\_hat + t\_quantile \* std\_theta\_hat

save\_to\_csv(pd.DataFrame({

'left\_interval': left\_interval,

'theta': theta,

'theta\_hat': theta\_hat,

'right\_interval': right\_interval

}), 'Task2.csv')

# ------------------ЗАДАНИЕ 3----------------------#

# Вычисление квантиля распределения Фишера

F\_quantile = f.ppf(1 - 0.05, 1, f\_R)

# Вычисление F-статистики для каждого параметра

F\_values = (theta\_hat \*\* 2) / (sigma\_hat\_squared \* np.diag(np.linalg.inv(X.T @ X)))

# Проверка гипотезы для каждого параметра

hypothesis\_results = F\_values < F\_quantile

save\_to\_csv(pd.DataFrame({

'F\_values': F\_values,

'sign': ['<' if result else '>' for result in hypothesis\_results],

'F\_quantile': [F\_quantile] \* len(F\_values),

'Hypothesis': ['Принимается' if result else 'Не принимается' for result in hypothesis\_results]

}), 'Task3.csv')

# ------------------ЗАДАНИЕ 4----------------------#

# Вычисление RSS

RSS = e\_hat.T @ e\_hat

# Вычисление RSS\_H

RSS\_H = np.sum((y - np.mean(y)) \*\* 2)

# Вычисление степеней свободы

q = len(theta) - 1

# Вычисление F-статистики

F\_regression = ((RSS\_H - RSS) / q) / (RSS / (len(y) - len(theta)))

# Вычисление квантиля распределения Фишера

F\_quantile\_regression = f.ppf(1 - 0.05, q, len(y) - len(theta))

save\_to\_csv(pd.DataFrame({

'F\_regression': F\_regression,

'sign': ['<' if F\_regression < F\_quantile\_regression else '>'],

'F\_quantile\_regression': F\_quantile\_regression,

'Hypothesis': ['Принимается' if F\_regression < F\_quantile\_regression else 'Не принимается']

}), 'Task4.csv')

# ------------------ЗАДАНИЕ 5----------------------#

# Зафиксированные значения факторов

x1\_fixed = 0

x3\_fixed = 1

# Диапазон значений для x2

x2\_range = np.linspace(-1, 1, 24)

f\_x = np.column\_stack((np.ones(len(x2\_range)), x1\_fixed \* np.ones(len(x2\_range)), x2\_range, x3\_fixed \* np.ones(len(x2\_range)),

x1\_fixed \* x2\_range, x1\_fixed \* x3\_fixed \* np.ones(len(x2\_range)), (x1\_fixed \*\* 2) \* np.ones(len(x2\_range)),

x2\_range \*\* 2, x2\_range \*\* 3))

# Вычисление прогнозных значений

u\_pred = f\_x @ theta\_hat

save\_to\_csv(pd.DataFrame({

'x1': x1\_fixed,

'x2': x2\_range,

'x3': x3\_fixed,

'u\_pred': u\_pred

}), 'Task5.csv')

# ------------------ЗАДАНИЕ 6----------------------#

u\_true = f\_x @ theta

y\_pred = u\_pred

left\_interval\_pred = u\_pred - t\_quantile \* np.sqrt(sigma\_hat\_squared \* np.diag(f\_x @ np.linalg.inv(X.T @ X) @ f\_x.T))

right\_interval\_pred = u\_pred + t\_quantile \* np.sqrt(sigma\_hat\_squared \* np.diag(f\_x @ np.linalg.inv(X.T @ X) @ f\_x.T))

sigma\_hat\_squared\_unbiased = e\_hat.T @ e\_hat / (len(y) - len(theta))

left\_interval\_pred2 = y\_pred - t\_quantile \* np.sqrt(sigma\_hat\_squared\_unbiased \* (1 + np.diag(f\_x @ np.linalg.inv(X.T @ X) @ f\_x.T)))

right\_interval\_pred2 = y\_pred + t\_quantile \* np.sqrt(sigma\_hat\_squared\_unbiased \* (1 + np.diag(f\_x @ np.linalg.inv(X.T @ X) @ f\_x.T)))

save\_to\_csv(pd.DataFrame({

'left\_interval': left\_interval\_pred,

'u\_true': u\_true,

'right\_interval': right\_interval\_pred

}), 'Task6\_Graphs1.csv')

save\_to\_csv(pd.DataFrame({

'left\_interval': left\_interval\_pred2,

'y\_true': u\_true,

'right\_interval': right\_interval\_pred2

}), 'Task6\_Graphs2.csv')

# ------------------ЗАДАНИЕ 7----------------------#

# Задание 7-1

# Вычисление истинного отклика без шума

u = theta[0] + theta[1]\*x1 + theta[2]\*x2 + theta[3]\*x3 \

+ theta[4]\*x1\*x2 + theta[5]\*x1\*x3 \

+ theta[6]\*x1\*\*2 + theta[7]\*x2\*\*2 + theta[8]\*x2\*\*3

# Вычисление дисперсии шума, учитывая, что мощность случайной помехи = 70% от мощности полезного сигнала

sigma\_squared = get\_sigma\_squared(u)

# Вычисление ошибки

e = get\_noise(u, sigma\_squared)

# Зашумленный отклик

y = u + e

# Вычисление вектора параметров модели theta\_hat

theta\_hat = np.linalg.inv(X.T @ X) @ X.T @ y

y\_hat = X @ theta\_hat

# Вычисление вектора остатков

e\_hat = y - y\_hat

# Вычисление несмещенной оценки

sigma\_hat\_squared = e\_hat.T @ e\_hat / (len(y) - len(theta))

# Вычисление F-статистики

F=sigma\_hat\_squared/sigma\_squared

# Сохранение всех датафреймов в файлы csv

save\_to\_csv(pd.concat([

pd.DataFrame({'y': y}),

pd.DataFrame({'y\_hat': y\_hat}),

pd.DataFrame({'y-y\_hat': y-y\_hat}),

pd.DataFrame({'theta': theta}),

pd.DataFrame({'theta\_hat': theta\_hat}),

pd.DataFrame({'sigma\_squared': [sigma\_squared]}),

pd.DataFrame({'sigma\_hat\_squared': [sigma\_hat\_squared]}),

pd.DataFrame({'F': [F]})], axis=1), 'Task7\_1.csv')

# Задание 7-2

# Вычисление квантиля распределения Фишера

F\_quantile = f.ppf(1 - 0.05, 1, f\_R)

# Вычисление F-статистики для каждого параметра

F\_values = (theta\_hat \*\* 2) / (sigma\_hat\_squared \* np.diag(np.linalg.inv(X.T @ X)))

# Проверка гипотезы для каждого параметра

hypothesis\_results = F\_values < F\_quantile

save\_to\_csv(pd.DataFrame({

'F\_values': F\_values,

'sign': ['<' if result else '>' for result in hypothesis\_results],

'F\_quantile': [F\_quantile] \* len(F\_values),

'Hypothesis': ['Принимается' if result else 'Не принимается' for result in hypothesis\_results]

}), 'Task7\_2.csv')

# Задание 7-3

# Вычисление RSS

RSS = e\_hat.T @ e\_hat

# Вычисление RSS\_H

RSS\_H = np.sum((y - np.mean(y)) \*\* 2)

# Вычисление степеней свободы

q = len(theta) - 1

# Вычисление F-статистики

F\_regression = ((RSS\_H - RSS) / q) / (RSS / (len(y) - len(theta)))

# Вычисление квантиля распределения Фишера

F\_quantile\_regression = f.ppf(1 - 0.05, q, len(y) - len(theta))

save\_to\_csv(pd.DataFrame({

'F\_regression': F\_regression,

'sign': ['<' if F\_regression < F\_quantile\_regression else '>'],

'F\_quantile\_regression': F\_quantile\_regression,

'Hypothesis': ['Принимается' if F\_regression < F\_quantile\_regression else 'Не принимается']

}), 'Task7\_3.csv')