|  |  |  |
| --- | --- | --- |
| МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ  Федеральное государственное бюджетное образовательное учреждение высшего образования  МОСКОВСКИЙ АВИАЦИОННЫЙ ИНСТИТУТ  (НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ) | | |
|  |  | |
|  | |
|  |  | |
| КУРСОВАЯ РАБОТА  по дисциплине:  «Анализ данных на Python»  Вариант №18 | | |
|  |
|  |  | |
| Руководитель:  д-р. ф-м. наук, доцент   Ковалевский А.П.  « »   2024 года | | Выполнил: студент группы М3О-312Б-22   Кудряшев Д.А.  « »   2024 года |
|  | | |

Москва 2024

# СОДЕРЖАНИЕ

[СОДЕРЖАНИЕ 2](#_Toc184847645)

[ПЕРЕЧЕНЬ СОКРАЩЕНИЙ И ОБОЗНАЧЕНИЙ 4](#_Toc184847646)

[ЗАДАНИЕ 5](#_Toc184847647)

[ВВЕДЕНИЕ 6](#_Toc184847648)

[1. ТЕОРЕТИЧЕСКАЯ ЧАСТЬ 7](#_Toc184847649)

[1.1. Описание исследуемых данных 7](#_Toc184847650)

[1.2. Описание наиболее подходящей модели 7](#_Toc184847651)

[1.3. Подбор регрессионных параметров 8](#_Toc184847652)

[1.4. Линейная регрессия 9](#_Toc184847653)

[1.5. Регрессия на X2 9](#_Toc184847654)

[1.6. Логарифмическая регрессионная модель 9](#_Toc184847655)

[1.7. Методы оценки регрессионных моделей 9](#_Toc184847656)

[1.8. Предварительная обработка данных 12](#_Toc184847657)

[2. СЧИТЫВАЕМ И ПРЕДВАРИТЕЛЬНО АНАЛИЗИРУЕМ ДАННЫЕ 14](#_Toc184847658)

[3. ПЕРВИЧНОЕ ПОСТРОЕНИЕ РЕГРЕССИОННЫХ МОДЕЛЕЙ 20](#_Toc184847659)

[3.1. Простая линейная регрессия 20](#_Toc184847660)

[3.2. Регрессия на X2 23](#_Toc184847665)

[3.3. Логарифмическая регрессионная модель 25](#_Toc184847666)

[3.4. Интерпретация результатов 28](#_Toc184847667)

[4. ПОСТРОЕНИЕ НА ПРОЛОГАРИФМИРОВАННЫХ ДАННЫХ О МАССЕ ТЕЛА 30](#_Toc184847668)

[4.1. Простая линейная регрессия 33](#_Toc184847669)

[4.2. Регрессия на Х2 35](#_Toc184847670)

[4.3. Логарифмическая регрессионная модель 38](#_Toc184847671)

[4.4. Интерпретация результатов 41](#_Toc184847672)

[5. МОДЕЛИРОВАНИЕ НА ПОЛНОСТЬЮ ПРОЛОГАРИФМИРОВАННЫХ ДАННЫХ 43](#_Toc184847673)

[5.1. Простая линейная регрессия 45](#_Toc184847674)

[5.2. Регрессия на Х2 48](#_Toc184847675)

[5.3. Логарифмическая регрессионная модель 51](#_Toc184847676)

[5.4. Интерпретация результатов 51](#_Toc184847677)

[ВЫВОД 53](#_Toc184847678)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 54](#_Toc184847679)

[ПРИЛОЖЕНИЕ А: ЛИСТИНГ ПРОГРАММЫ 55](#_Toc184847680)

# ПЕРЕЧЕНЬ СОКРАЩЕНИЙ И ОБОЗНАЧЕНИЙ

В настоящей работе применяются следующие сокращения и обозначения:

AFP – Alpha-Fetoprotein (альфа-фетопротеин)

MoM – Multiple of the Median (произведение медиан)

AFPMoMs – произведение медиан значений альфа-фетопротеина

OLS – Ordinary Least Squares (обычный метод наименьших квадратов)

# ЗАДАНИЕ

В соответствии с номером варианта поставлена следующая задача: построить модель зависимости произведения медиан AFPMoMs от массы тела по данным из файла.

При решении задачи необходимо использовать ранее изученные методы анализа данных, предоставляемые языком программирования Python.

# ВВЕДЕНИЕ

Предполагается при решении задачи следовать следующим пунктам:

* Найти и прочитать данные для исследования;
* Построить вероятностную модель;
* Проконсультироваться с лектором о выборе модели;
* Исследовать значимость параметров модели и исключить незначимые;
* Исследовать адекватность модели статистическим тестом и/или стохастическим моделированием, кросс-валидацией;
* На основании этого исследования исправить модель, а затем проверить адекватность исправленной модели.

При этом, в варианте №18 нет необходимости в поиске данных для исследования, а также в исключении незначимых параметров для составляемой вероятностной модели данных.

Пример данных, содержащихся в предоставляемом xlsx файле, продемонстрирован на рисунке 1.

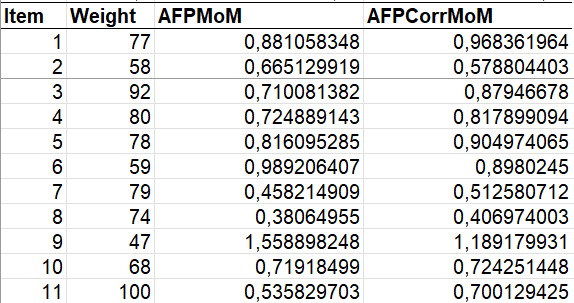


Рисунок 1 — Пример данных из файла

# ТЕОРЕТИЧЕСКАЯ ЧАСТЬ

## Описание исследуемых данных

Альфа-фетопротеин (AFP) — это белок, который вырабатывается печенью плода и попадает в кровь матери. Его уровень может варьироваться в зависимости от срока беременности. Исследование значения данного показателя в крови женщины помогает выявить риск различных аномалий плода (например, дефектов нервной трубки плода или хромосомных аномалий, таких как синдром Дауна).

У взрослых в норме альфа-фетопротеин не синтезируется и лишь содержится в крови с определенным значением. Отклонения данного значения для взрослого также указывает на различные патологии.

MoM (Multiples of the Median) — это способ нормализации уровня AFP, который позволяет сравнивать результаты с медианными значениями для определенного срока беременности.

Значение AFPMoM показывает, насколько уровень AFP в крови матери выше или ниже медианного значения для данного срока. Значение AFPMoMs ниже 1.0 может указывать на низкий риск аномалий, в то время как значение выше 2.0 может сигнализировать о повышенном риске. Важно учитывать, что результаты должны интерпретироваться в контексте других тестов и клинической информации. AFPMoMs является важным инструментом в пренатальной диагностике, позволяющим оценить здоровье плода и риски возможных аномалий.

## Описание наиболее подходящей модели

Имея необходимость проследить зависимость некоторой переменной от одного/нескольких параметров, часто прибегают к использованию регрессионных моделей.

В контексте задачи предположим, что у нас есть некоторое значение массы тела человека (регрессора) и нам необходимо найти значение коэффициента AFPMoM. Имеем типичную задачу регрессии, для которой может хорошо подойти параметрическая регрессионная модель. Построив ее, получим предсказанное значение коэффициента AFPMoM, причем точность полученного значения будет зависеть от корректности выбранной регрессии.

В наиболее общем виде регрессионная модель описывается следующим образом:

где — параметры модели, — случайная ошибка модели.

## Подбор регрессионных параметров

Помимо грамотного подбора регрессионной модели не менее важно подобрать наиболее подходящие параметры для нее. Для этого будем использовать метод OLS (методом наименьших квадратов).

Данный метод позволяет найти такие значения коэффициентов регрессии, которые минимизируют сумму квадратов остатков (разностей между фактическими и предсказанными значениями зависимой переменной).

Учитывая предыдущие формулировки, это означает что мы принимаем случайные ошибки модели , как независимые и имеющие одно и то же распределение, после чего минимизируем сумму их квадратов, принимая те значения параметров модели за наиболее подходящие, которые в конечном итоге установят минимальную сумму квадратов отклонений при фиксированных регрессорах. Формула представлена ниже.

Пример проявления отклонений при построении регрессионной модели проиллюстрирован на рисунке 2.

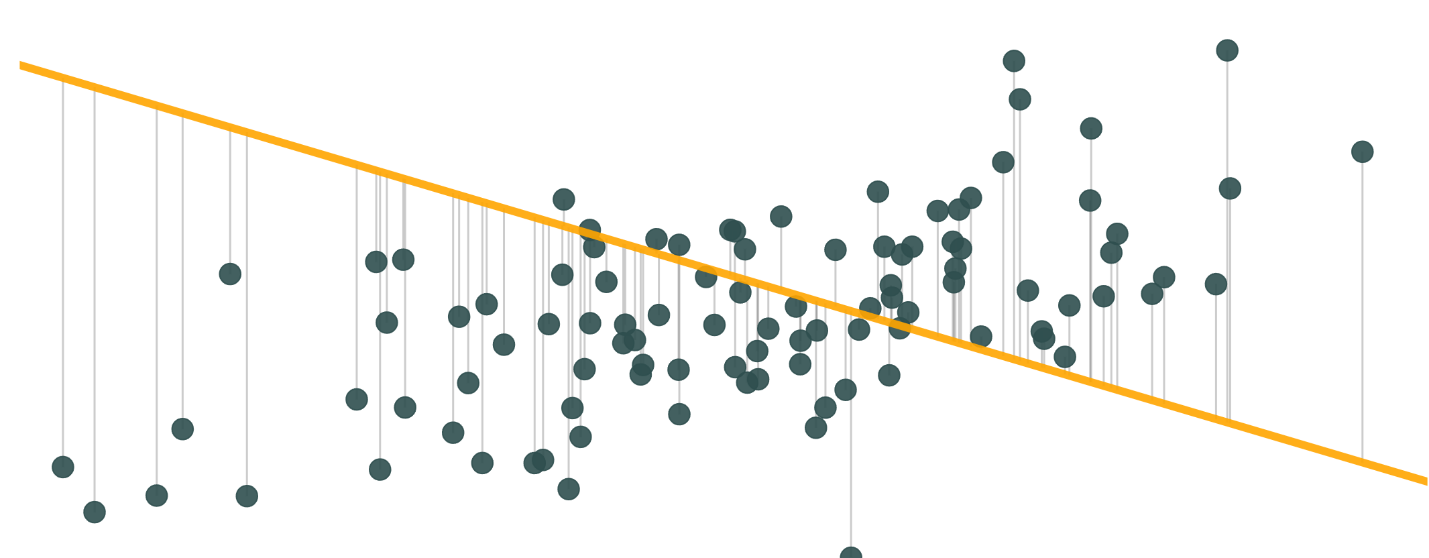


Рисунок 2 — Отклонения от регрессии

Теперь опишем те виды регрессии, которые в дальнейшем будут использованы в работе.

## Линейная регрессия

Регрессия называется линейной, если

где — параметры (коэффициенты) регрессии, — регрессоры (факторы модели), — количество факторов модели.

## Регрессия на X2

Для данного вида регрессионной модели характерна формула:

где — параметры (коэффициенты) регрессии, — регрессоры (факторы модели), — количество факторов модели.

## Логарифмическая регрессионная модель

Данный вид регрессии описывается формулой:

где — параметры (коэффициенты) регрессии, — регрессоры (факторы модели), — количество факторов модели, — случайная ошибка модели.

## Методы оценки регрессионных моделей

После построения нескольких видов регрессионных моделей, необходимо оценить их корректность, а также выбрать ту, которая наиболее точно описывает зависимость между исследуемыми данными. Для этого существует различные методы. Ниже опишем те, которые будут использованы в работе в дальнейшем.

t-статистика — это статистический показатель, который используется для проверки гипотез о значимости параметров в регрессионных моделях. Вычисляется по формуле:

где — t-статистика параметра при i-ом регрессоре модели, — дисперсия i-го параметра по точкам данных (вычисляется при помощи матрицы ковариаций по регрессорам модели).

В свою очередь значимость t-статистики , подскажем нам, стоит ли учитывать тот или иной параметр при рассмотрении модели (если <0.05, то гипотеза о значимости принимается). В случае с единственным регрессором, это даст понять, корректна ли исследуемая модель.

Если речь идет о сравнении моделей между собой, то доля объясненной дисперсии будет примерно схожа для рассматриваемых моделей, поэтому по ней трудно судить о большей или меньшей пригодности конкретной регрессии.

Для анализа получившийся модели, в которой имеется лишь один регрессор, удобно использовать такое понятие, как эмпирический мост. Для его построения необходимо последовательно, по упорядоченным данным суммировать отклонения фактических значений от прогнозируемых. После их нормирования, последовательные суммы придут к нулю (из-за наличия свободного параметра ), однако промежуточные значения помогут оценить качество построенной модели. Пример построения эмпирического моста представлен на рисунке 3.

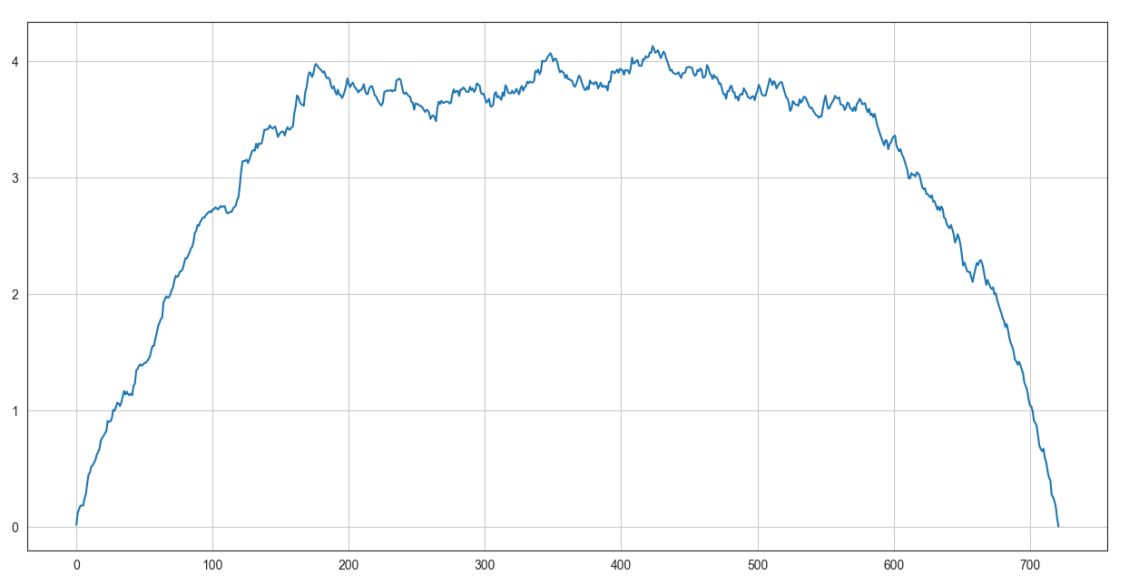


Рисунок 3 — Пример построения эмпирического моста

Имея построенный график эмпирического моста, мы можем обратить внимание на следующие факторы: до каких максимальных и минимальных значений отклоняется график, при каких значениях регрессора проявляются длительные промежутки отклонений, насколько на всем протяжении графика последовательные суммы отклонений от фактических значений отличны от нуля.

В данном примере видим, что отклонения происходят только в положительном направлении и большая часть графика достаточно удалена от нуля и колеблется в районе максимального значения. Это означало бы, что построена не самая удачная модель, так как она хорошо работает в серединном диапазоне значений, но сильно искажает данные в начале и в конце этого диапазона.

Помимо всего вышеперечисленного, с ростом числа наблюдений максимальное отклонение эмпирического моста должно сходиться к распределению Колмогорова. Поэтому для оценки регрессионной модели введем следующие значения:

* Максимальное по модулю значение эмпирического моста ;
* Верхняя оценка p-value, которая будет высчитываться построением функции выживания по двустороннему распределению Колмогорова – Смирнова и передаче в нее значения , т.е. получим вероятность, с которой случайным образом сгенерированная выборка превзойдет максимум модуля эмпирического моста . Данное значение считаем не точным, а лишь верхней оценкой p-value, так как при построении модели был использован свободный параметр .

Для того чтобы получить более точную оценку p-value будем использовать стохастическое моделирование для получения эмпирического p-value. Согласно центральной предельной теоремы, 20000 моделирований позволит получить приближение с точностью до двух знаков после запятой.

Так как проведение 20000 моделирований с помощью средств вычислительной техники является довольно долгим процессом, то с целью экономии моделирование будем производить только для тех регрессий, которые имеют удовлетворительную верхнюю оценку p-value относительно других видов регрессионных моделей.

Полученное эмпирическое p-value будем расценивать в качестве финальной оценки для сравнения выведенных моделей.

## Предварительная обработка данных

Регрессионные модели довольно чувствительны к значениям, называемым выбросами, т.е. к тем значениям, точки на графике которых значительно удалены от основного массива данных. Однако важно понимать, что в рассматриваемом варианте приведены данные о медицинских исследованиях, которые являются точными и не подвержены влиянию случайных факторов, поэтому все они являются достаточно важными для модели. С целью улучшения модели будем называть выбросами те измерения, которые можно назвать паталогическими (слишком низкие или высокие значения массы тела, слишком высокие значения AFPMoMs). Для поиска выбросов предлагается использовать следующий алгоритм:

* Находим первую и третью квартиль по определенной величине среди входных данных;
* Находим межквартильное расстояние как разность между третьей и первой квартилями;
* Из первой квартили вычитаем найденное расстояние, умноженное на 1.5, получаем нижнюю границу выбросов;
* К третьей квартили прибавляем найденное расстояния, умноженное на 1.5, получаем верхнюю границу выбросов;
* При необходимости варьируем коэффициент при межквартальном расстоянии для получения оптимальных границ;
* Повторяем алгоритм для тех величин входных данных, в которых необходимо найти выбросы.

С целью упрощения моделей перед построением зависимости прибегают к логарифмированию данных. Этот метод позволяет определенным образом нормировать данные, уменьшив диапазон разброса значений, что позволяет для некоторых моделей найти более точную связь факторов и искомых значений, а также уменьшить влияние выбросов или упростить алгоритм их поиска. Однако важно понимать, что после этой процедуры могут появиться отрицательные значения, что в дальнейшем ограничит выбор моделей для построения. При необходимости прибегнем к предварительному логарифмированию данных.

# СЧИТЫВАЕМ И ПРЕДВАРИТЕЛЬНО АНАЛИЗИРУЕМ ДАННЫЕ

Для работы было принято решение использовать Google Colaboratory для удобной визуализации кода Python в виде блоков и вывода промежуточных результатов.

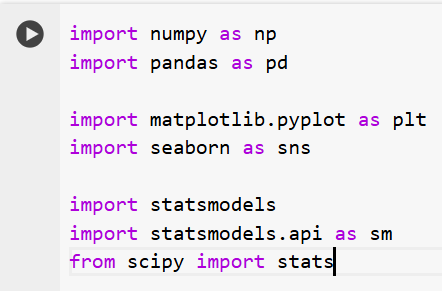
Для начала импортируем необходимые библиотеки. Блок кода с импортируемыми библиотеками представлен на рисунке 4.

Рисунок 4 — Импортируемые библиотеки

Далее при помощи методов, предоставляемых библиотеками NumPy и pandas собираем данные из файла со всех листов в единый DataFrame, очищаем его от строк, содержащих пропуски и убеждаемся в корректности автоматически подобранных типов данных по каждому из столбцов (при необходимости изменяем вручную) и выводим промежуточные результаты. Полученные на этом шаге результаты представлены на рисунке 5.

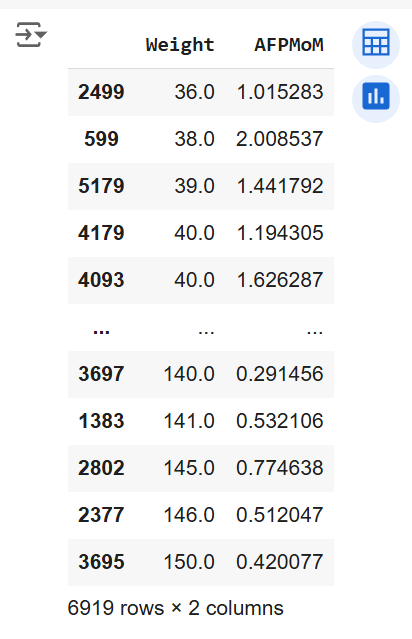
Имеем DataFrame из 6919 измерений, где первый столбец — значения массы тела, второй — значения AFPMoM, соответствующие данной массе. Важно при дальнейших преобразованиях не перемешивать значения, чтобы строки имели исходный вид.

Рисунок 5 — Вывод собранных данных

Для первичной визуализации данных построим эмпирические функции распределения и скрипичные диаграммы по каждой из двух величин. Результаты приведены на рисунке 6.

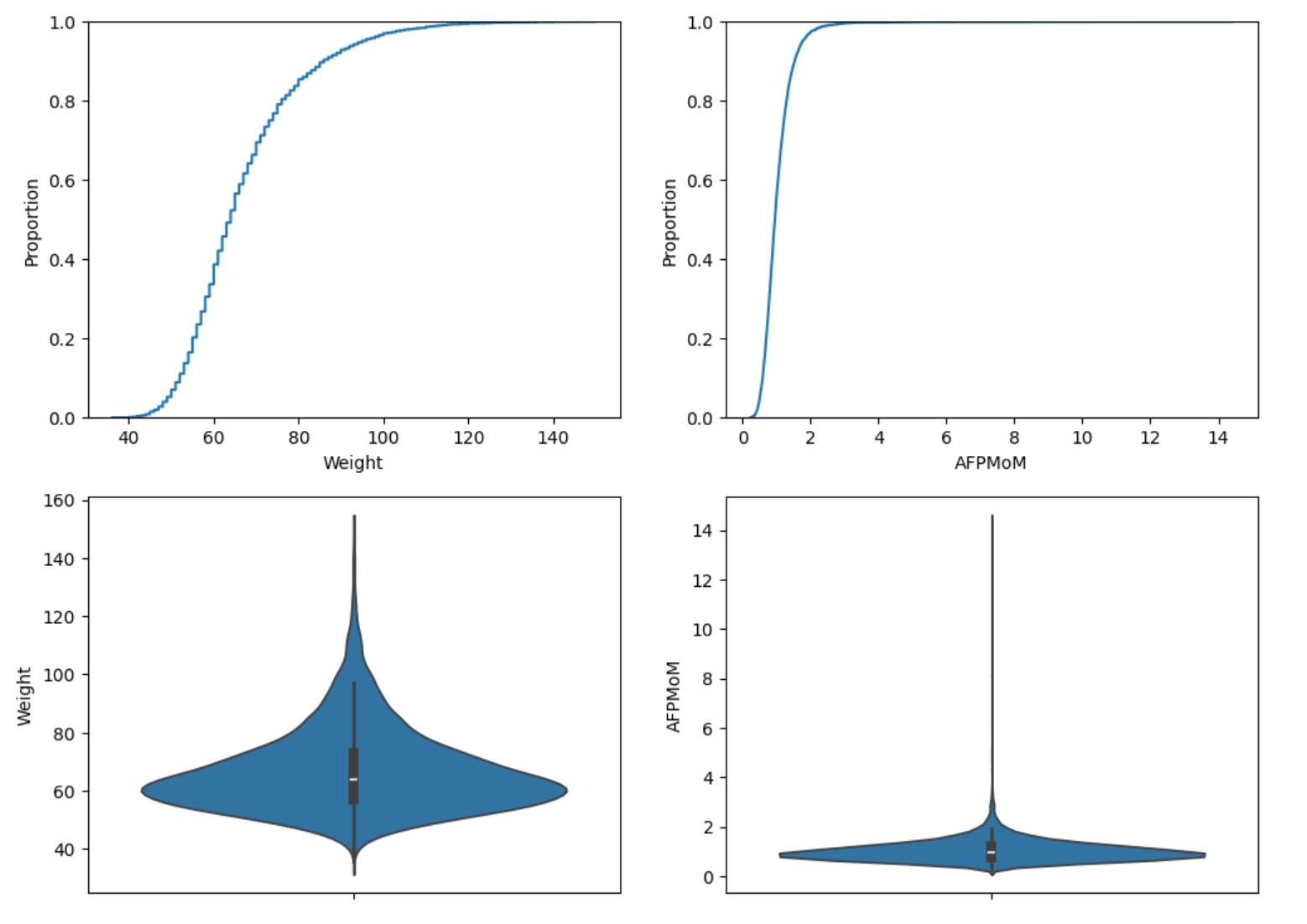


Рисунок 6 — Первичная визуализация данных

Изучив данные и проанализировав графики, приходим к выводу, что среди измерений существуют значения, которым нужно присвоить статус выбросов для корректного формирования моделей в дальнейшем. Чтобы не потерять исходные данные, дальнейшие действия будут производиться с копией DataFrame, полученного на предыдущем этапе.

Блок кода с алгоритмом отбора выбросов, а также полученные граничные значения по каждому из столбцов данных представлен на рисунке 7. Используется алгоритм, описанный в теоретической части.

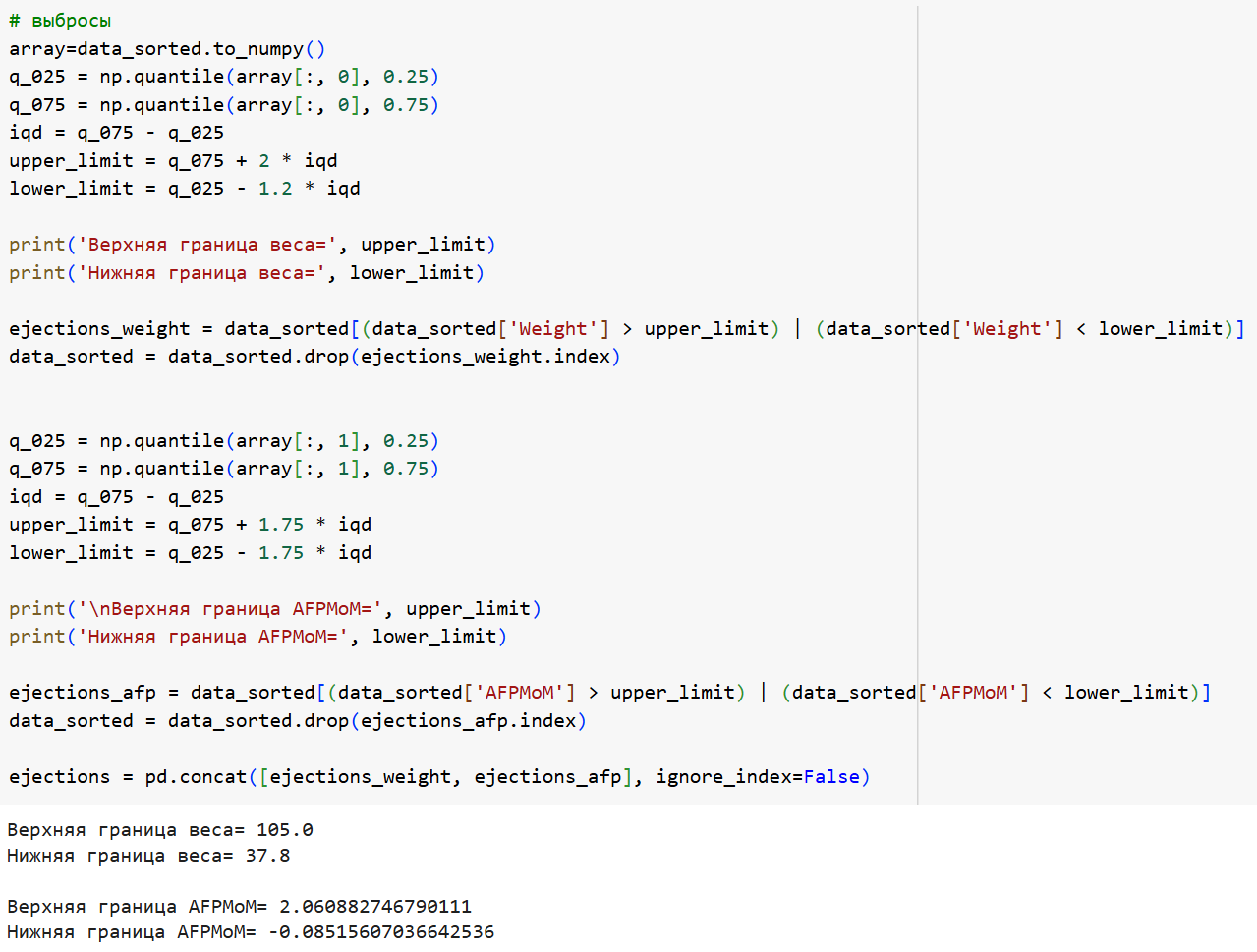


Рисунок 7 — Отбор выбросов

С учетом использованного алгоритма, построим график рассеяния с указанием значений, признанных выбросами. Результат представлен на рисунке 8.

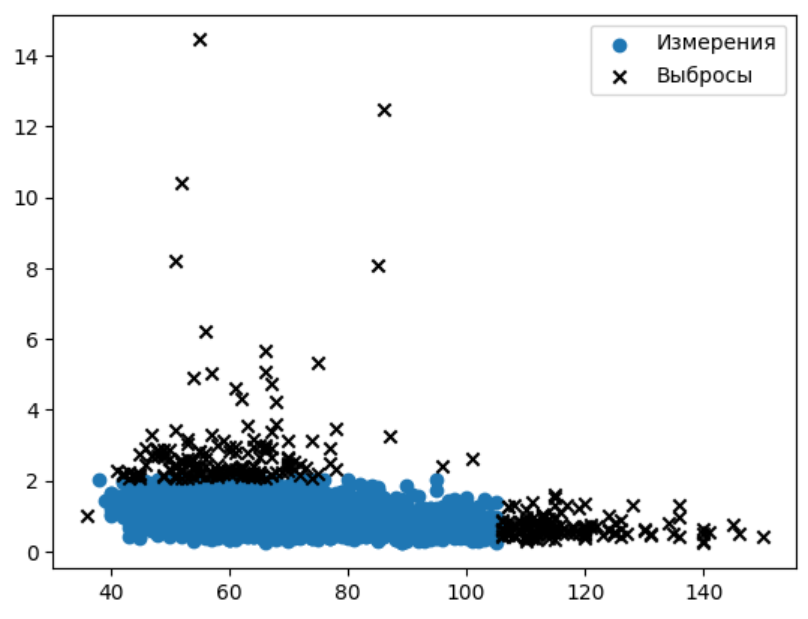


Рисунок 8 — Визуализация выбросов

На рисунке 9 построим диаграмму рассеяния для оставшихся данных без выбросов.

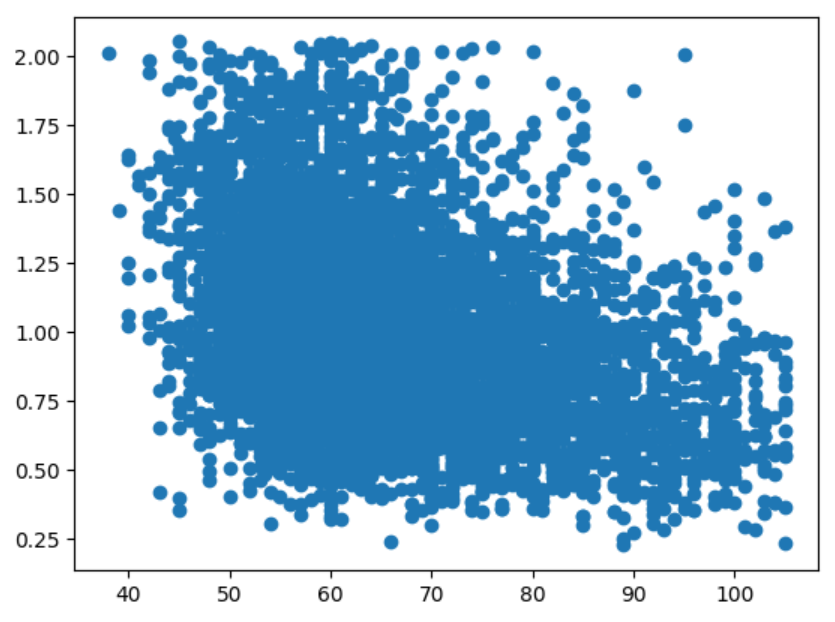


Рисунок 9 — Данные без выбросов

Данные, приведенные на рисунке 9, будут использоваться для дальнейшего построения регрессионных моделей, описанных в теоретической части.

Первоначально, построение будет происходить без предварительного логарифмирования.

# ПЕРВИЧНОЕ ПОСТРОЕНИЕ РЕГРЕССИОННЫХ МОДЕЛЕЙ

## Простая линейная регрессия

Начнем построение с линейной регрессии. За X обозначим значения массы тела, за Y — значения AFPMoMs. Воспользуемся методом OLS, содержащимся в библиотеке statsmodels.api, так как он позволяет быстро и информативно применять к данным регрессионные модели, а также оценивать их качество путем выявления различных значений, предусмотренных методами статистического анализа. В X предварительно добавляется единица, так как в формировании модели используется свободный параметр . Принцип работы метода OLS и рассматриваемые результаты применения метода описаны в теоретической части настоящей работы. Результат построения линейной регрессионной модели представлен на рисунке 10.

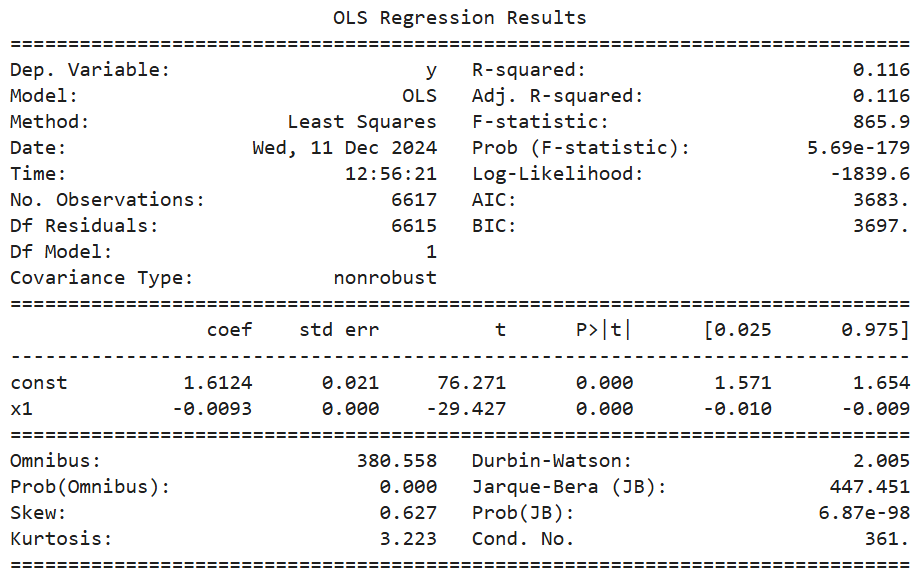


Рисунок 10 — Линейная регрессия

Подобранные параметры указаны в столбце *coef*, а столбец значимости t-статистики своими значениями 0 указывает на то, что единственный регрессор является значимым в данной модели, а значит она имеет право на существование.

Построим полученную зависимость на графике рассеяния значений, а также отложим доверительный интервал. Результат представим на рисунке 11.

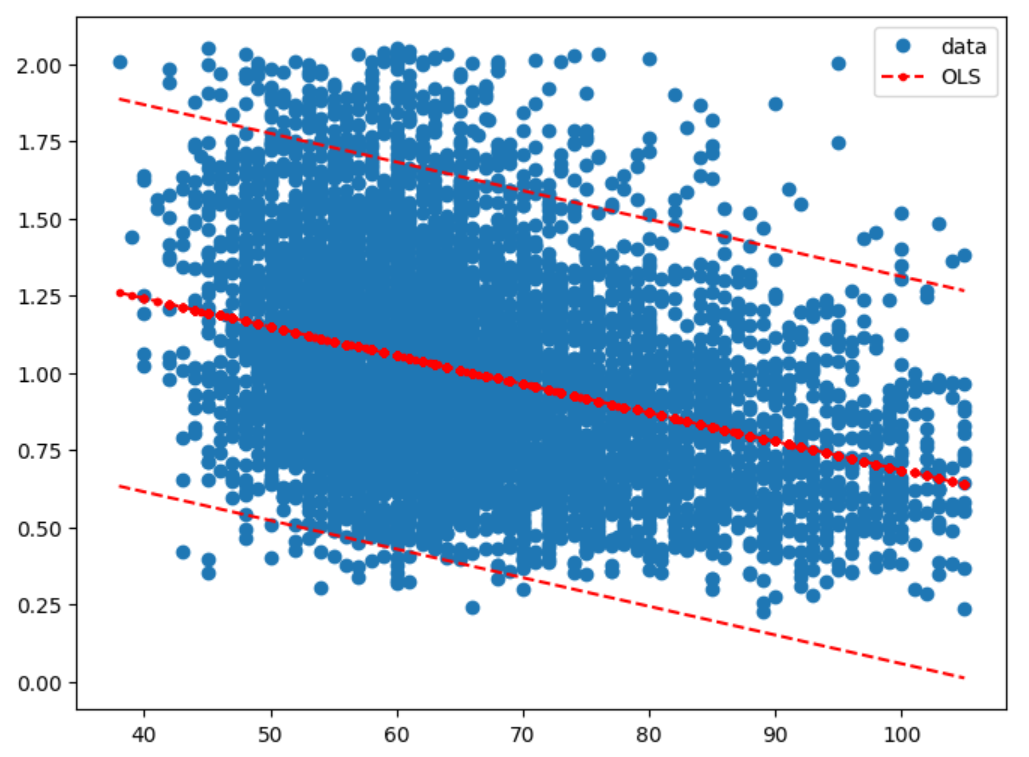


Рисунок 11 — График линейной регрессии

Построим эмпирический мост, а также найдем значение и верхнюю оценку p-value, описанные в теоретической части. Алгоритм данного шага и полученные результаты представлены на рисунке 12.

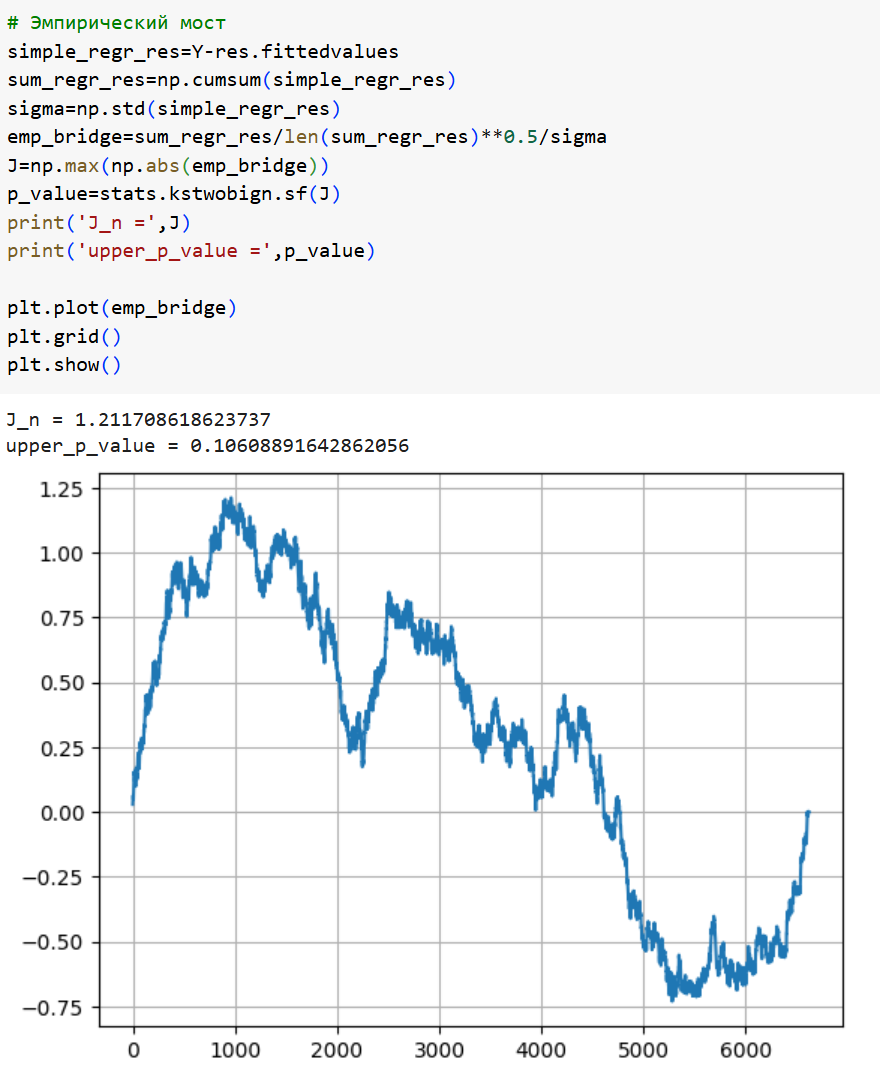


Рисунок 12 — Эмпирический мост для линейной регрессии

Верхняя оценка p-value, достаточно мала, а эмпирический мост отклоняется в достаточно большом диапазоне значений относительно входных данных и имеет короткие промежутки колебания в пределах постоянных значений.



## Регрессия на X2

При построении данного вида используем тот же алгоритм действий, предварительно возведя в квадрат столбец массы тела. Результат построения регрессионной модели представлен на рисунке 13.



Рисунок 13 — Регрессия на Х2

Столбец значимости t-статистики своими значениями 0 указывает на то, что единственный регрессор является значимым в данной модели, а значит она имеет право на существование.

На рисунке 14 построим график полученной зависимости вместе с доверительным интервалом.

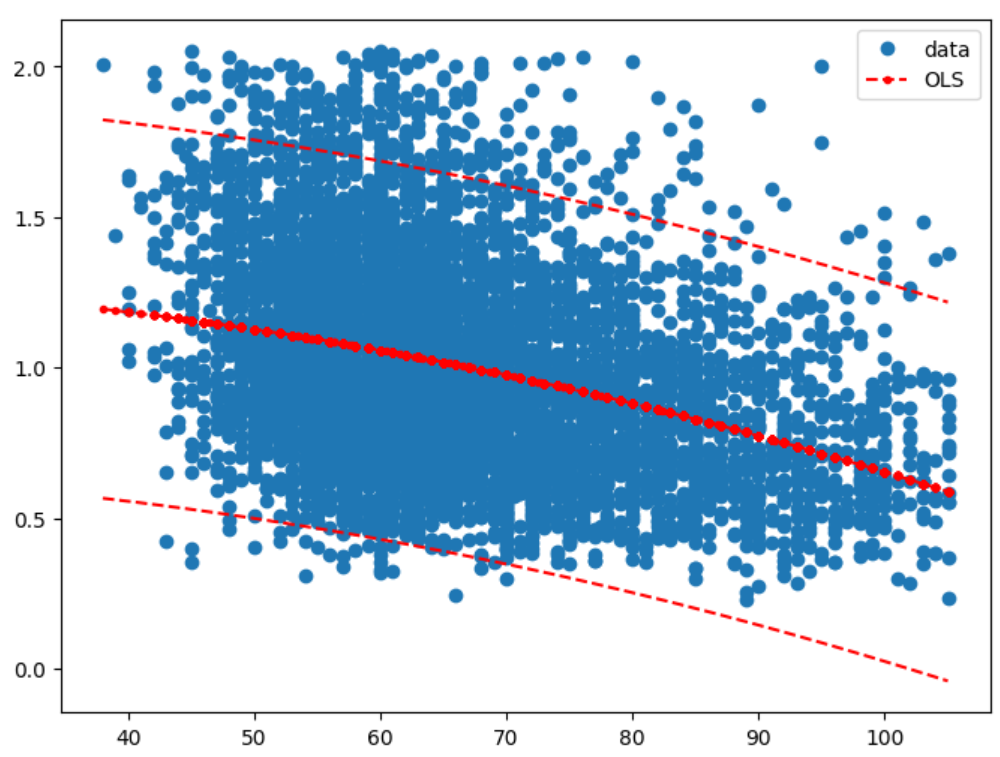


Рисунок 14 — График регрессии на Х2

Теперь повторим построение эмпирического моста и нахождения верхней оценки p-value, используя прежний алгоритм. Результаты представлены на рисунке 15.

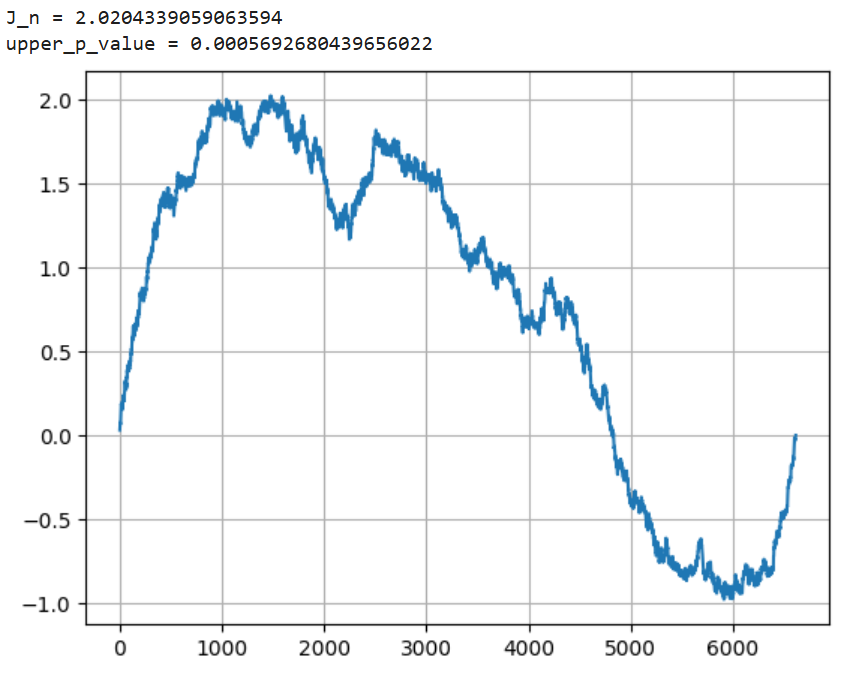


Рисунок 15 — Эмпирический мост для Х2

Видим, что разброс значений больше, а верхняя оценка p-value меньше, чем для линейной регрессии.

## Логарифмическая регрессионная модель

Для построения данной модели повторяем алгоритмы для линейной модели, предварительно прологарифмировав столбцы X и Y. Результаты построения модели изобразим на рисунке 16.

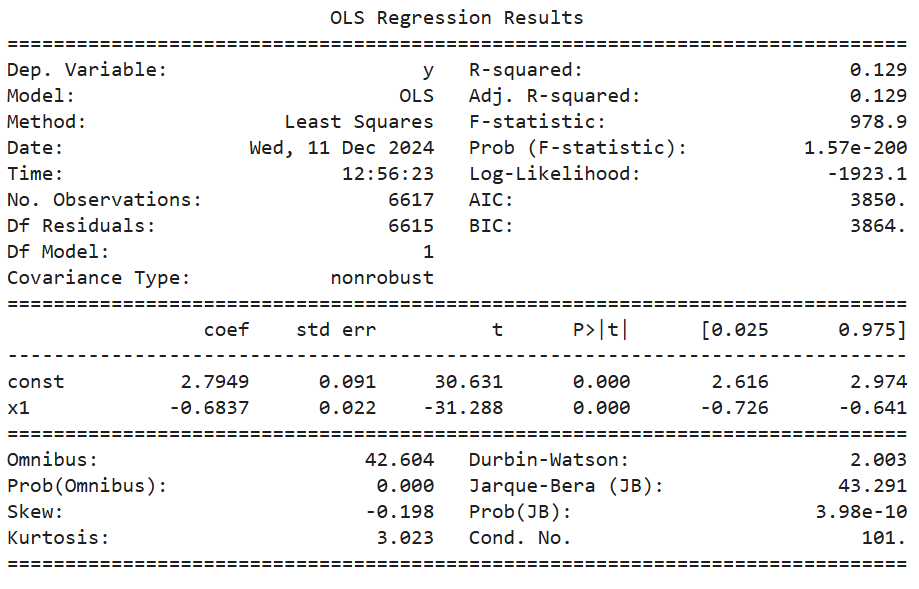


Рисунок 16 — Логарифмическая регрессия

Столбец значимости t-статистики своими значениями указывает на то, что единственный регрессор является значимым для данной модели, а значит она имеет право на существование.

Теперь построим график полученной модели с доверительным интервалом. Результат представлен на рисунке 17.

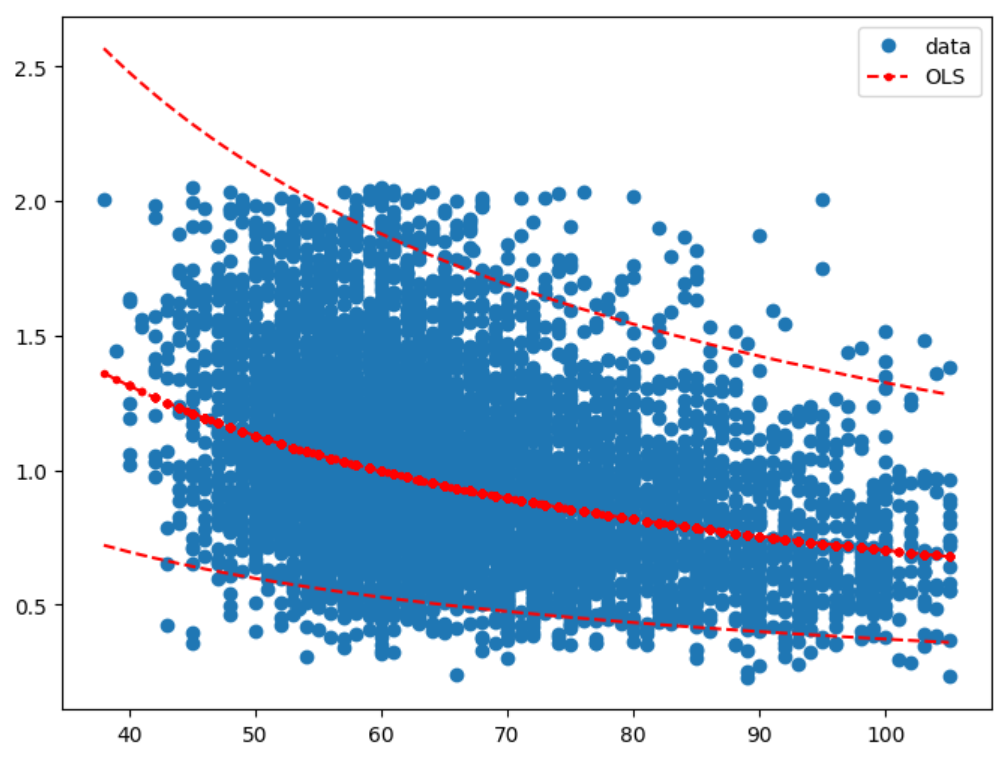


Рисунок 17 — График логарифмической регрессии

Теперь повторим построение эмпирического моста и нахождения верхней оценки p-value, используя прежний алгоритм. Результаты представлены на рисунке 18.

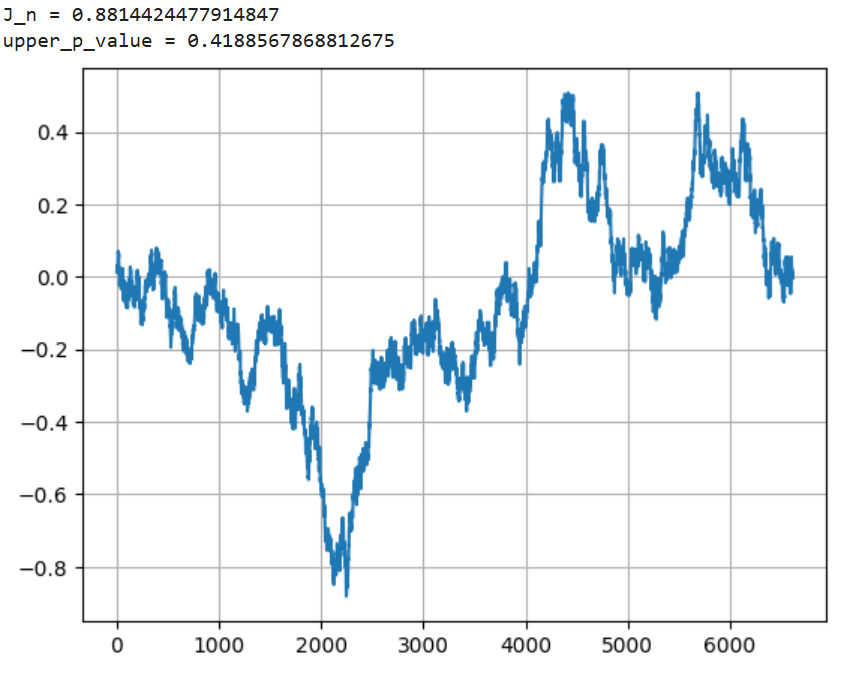


Рисунок 18 — Эмпирический мост для логарифмической регрессии

В результате видим довольно высокое значение верхней оценки p-value, а также наиболее удачный график эмпирического моста.

## Интерпретация результатов

Построив три вида регрессионных моделей и произведя необходимые вычисления, можем сравнить полученные результаты. Наиболее удачной получилась логарифмическая модель с верхней оценкой p-value равной 0.4188567868812675. Выбор этой модели из трех в данном случае очевиден, однако для проверки ее качества проведем стохастическое моделирование для оценки эмпирического p-value, описанное в теоретической части настоящей работы. Использованный алгоритм и результат его выполнения приведен на рисунке 19.

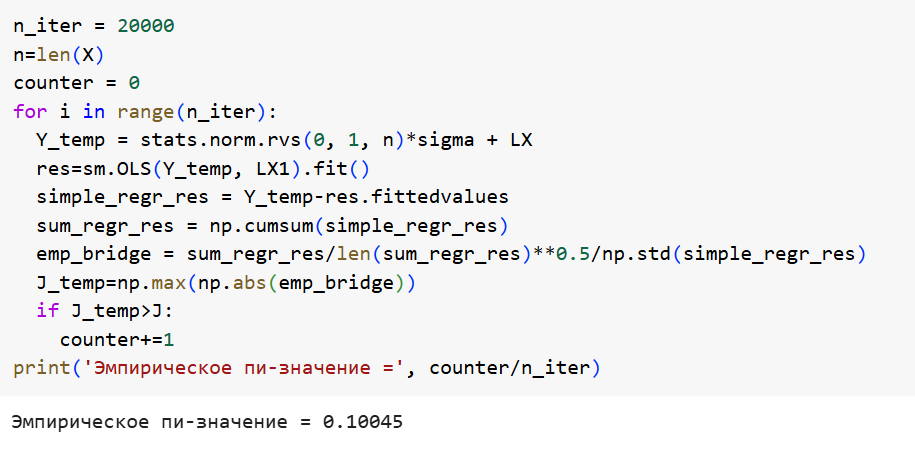


Рисунок 19 — Стохастическое моделирование

Получили значение эмпирического p-value, удовлетворяющее требованиям.

# ПОСТРОЕНИЕ НА ПРОЛОГАРИФМИРОВАННЫХ ДАННЫХ О МАССЕ ТЕЛА

Важно понимать, что несмотря на удачную модель, полученную на предыдущем шаге, она была построена на данных, откуда довольно грубо были удалены выбросы. Есть вероятность, что были исключены важные для построения модели данные. Чтобы это избежать и получить более удачную модель воспользуемся предварительным логарифмированием данных, описанным в теоретической части настоящей работы, и повторим все произведенные выше вычисления.

Для начала попробуем прологарифмировать только данные о массе тела, оставив соответствующие им значения AFPMoMs без изменений. При этом, диапазон значений массы тела позволяет не получить отрицательные значения после логарифмирования, а это значит, что данная процедура никак не повлияет на возможность построения всех трех видов регрессионных моделей.

Вид данных после логарифмирования представлен на рисунке 20.

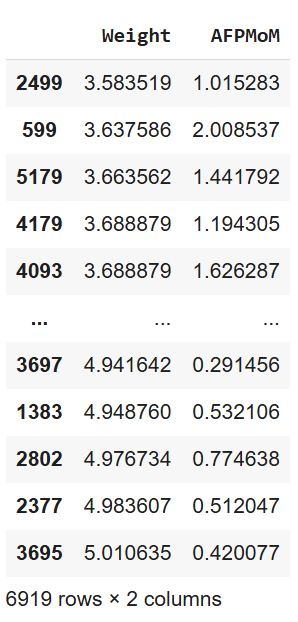
Попробуем найти выбросы согласно описанному выше алгоритму. График рассеяния данных с отмеченными выбросами приведен на рисунке 21.

Рисунок 20 — Данные после логарифмирования массы тела



Рисунок 21 — Визуализация выбросов

На рисунке 22 покажем график рассеяния значений без уже удаленных выбросов.

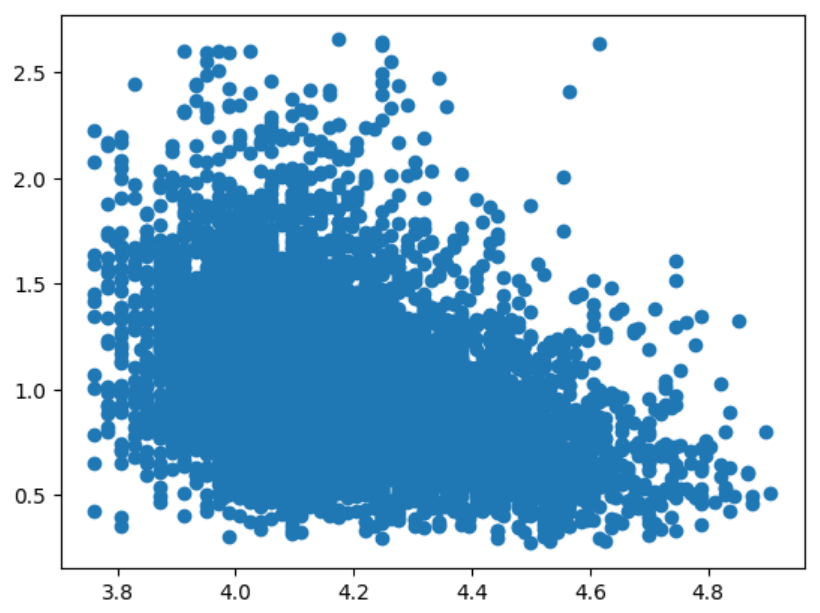


Рисунок 22 — Данные без выбросов

Данные, изображенные на графике выше, будут использоваться для повторного построения моделей.

## Простая линейная регрессия

Построение линейной регрессии будет полностью повторять построение, примененное ранее. Результаты построения представлены на рисунке 23.

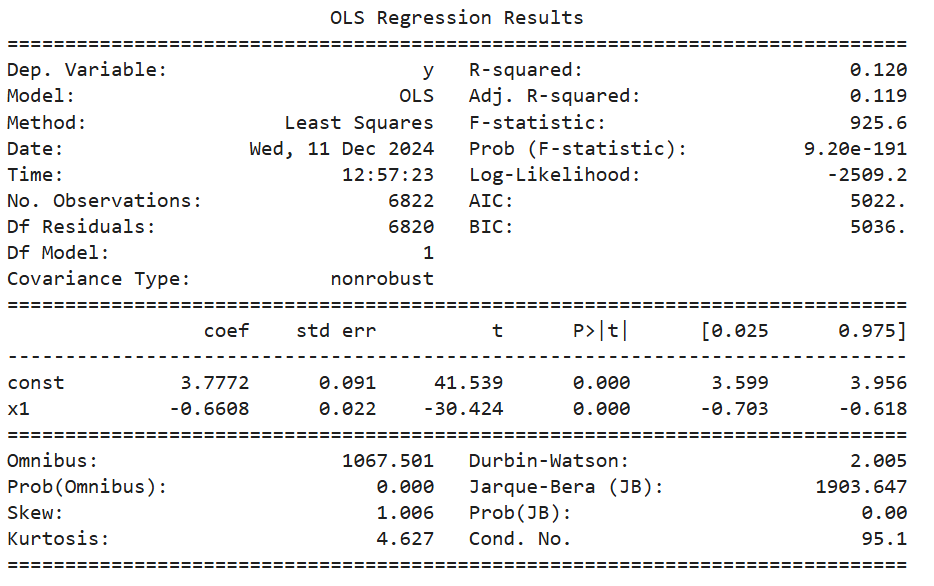


Рисунок 23 — Линейная регрессия

Столбец значимости t-статистики своими значениями указывает на то, что единственный регрессор значим в данной модели, а значит она имеет право на существование.

Теперь построим график полученной модели с доверительным интервалом. Результат представлен на рисунке 24.

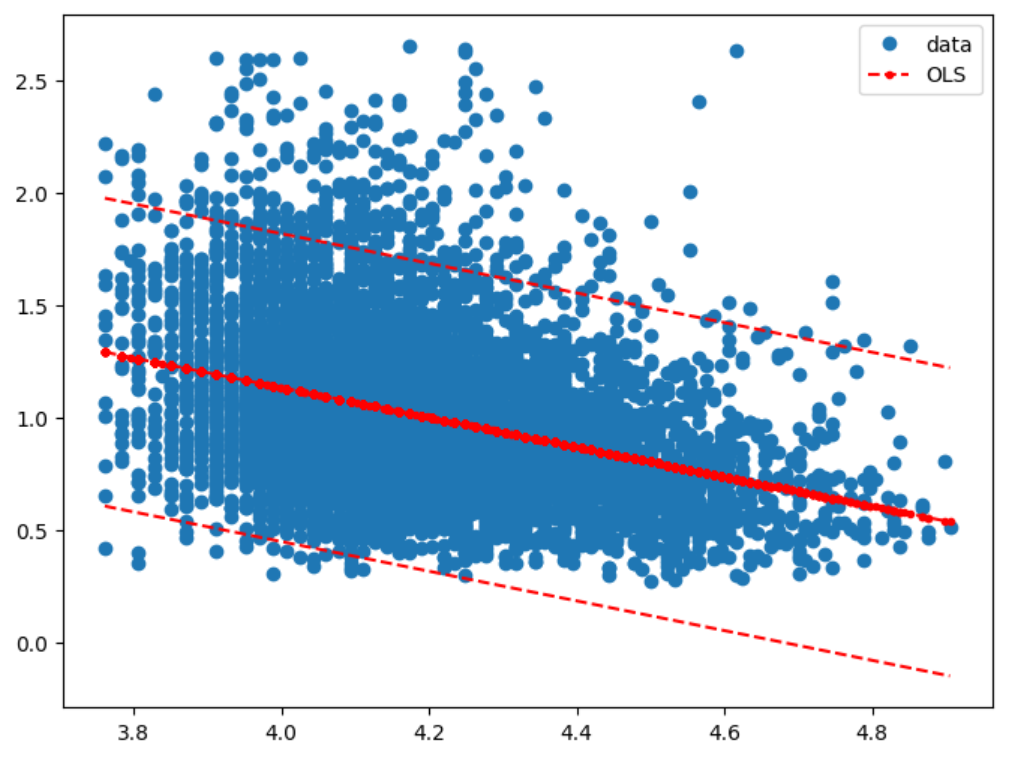


Рисунок 24 — График линейной регрессии

Теперь повторим построение эмпирического моста и нахождения верхней оценки p-value, используя прежний алгоритм. Результаты представлены на рисунке 25.

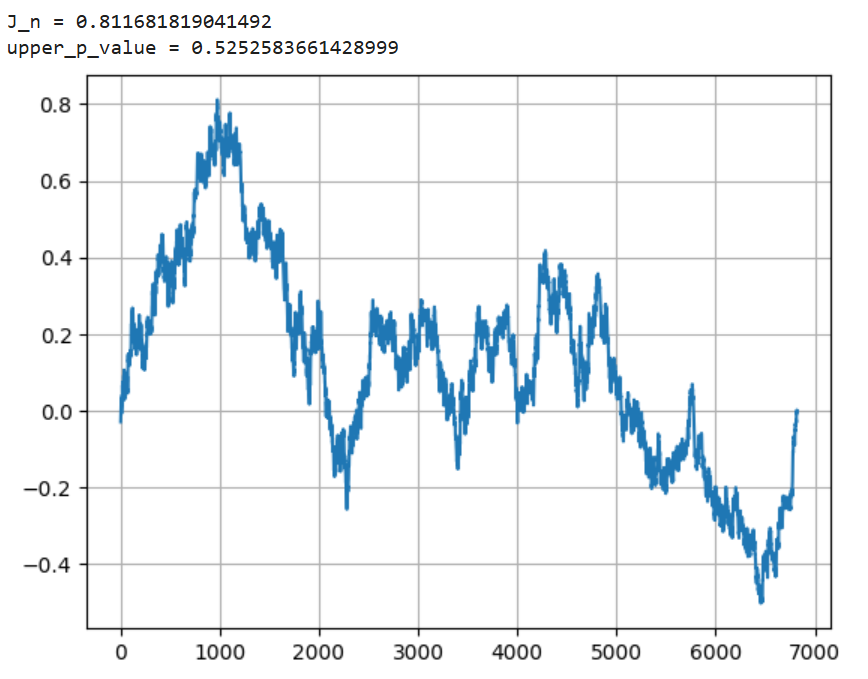


Рисунок 25 — Эмпирический мост линейной регрессии

Имеем вполне удовлетворительную верхнюю оценку p-value и график эмпирического моста, который указывает на относительно небольшие колебания за исключением начальной и конечной части.

## Регрессия на Х2

Теперь повторим построения регрессии на Х2. Результат выполнения алгоритма OLS представлен на рисунке 26.

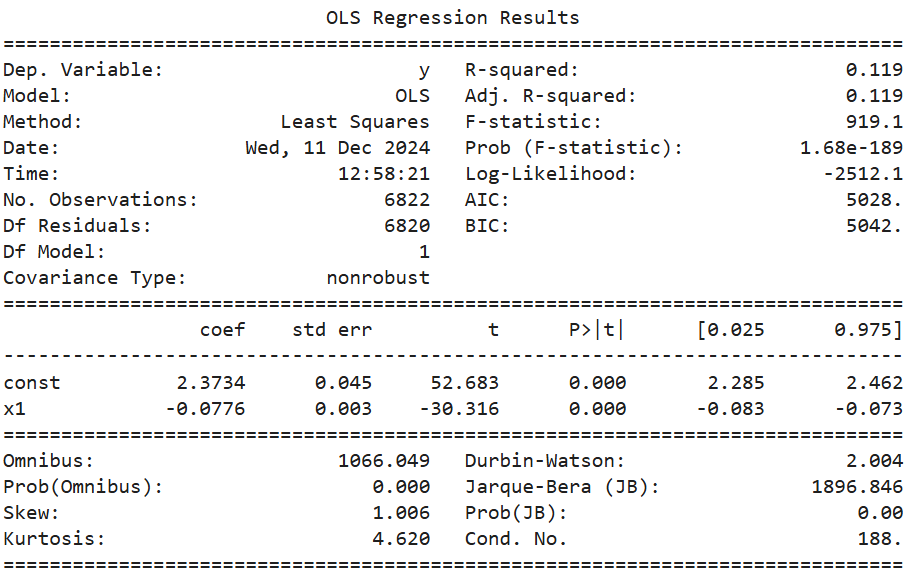


Рисунок 26 — Регрессия на Х2

Столбец значимости t-статистики своими значениями указывает на то, что единственный регрессор является значимым в данной модели, а значит она имеет право на существование.

На рисунке 27 построен график полученной зависимости с указанием доверительного интервала.

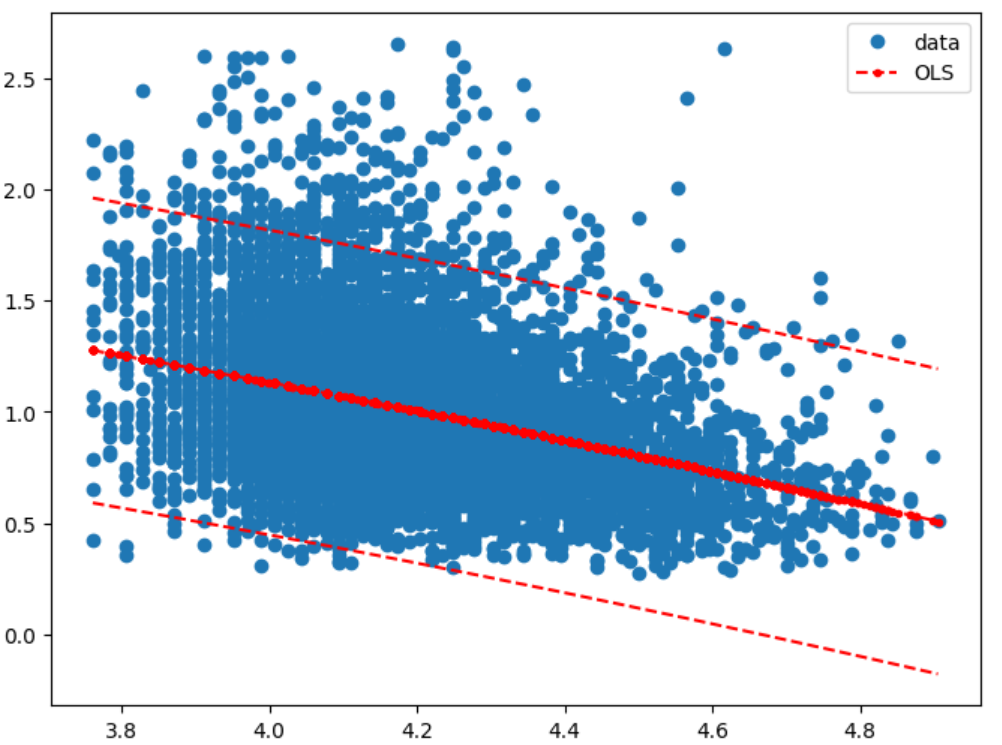


Рисунок 27 — График регрессии на Х2

Теперь повторим построение эмпирического моста и нахождения верхней оценки p-value, используя прежний алгоритм. Результаты представлены на рисунке 28.

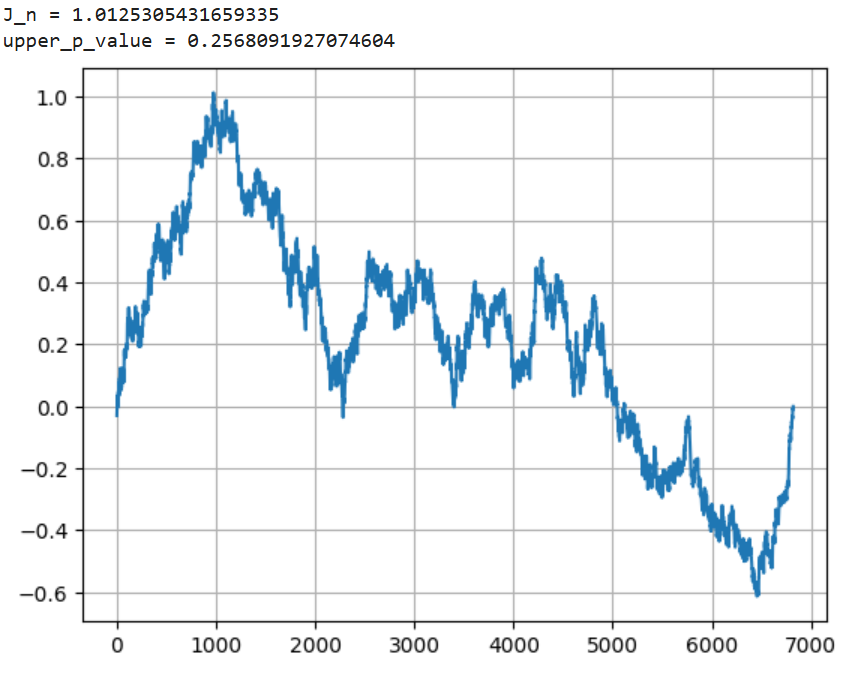


Рисунок 28 — Эмпирический мост регрессии на Х2

Имеем менее удачную верхнюю оценку p-value и больший диапазон разброса значений по сравнению с линейной регрессией.

## Логарифмическая регрессионная модель

Повторим построение логарифмической модели. Результат представлен на рисунке 29.

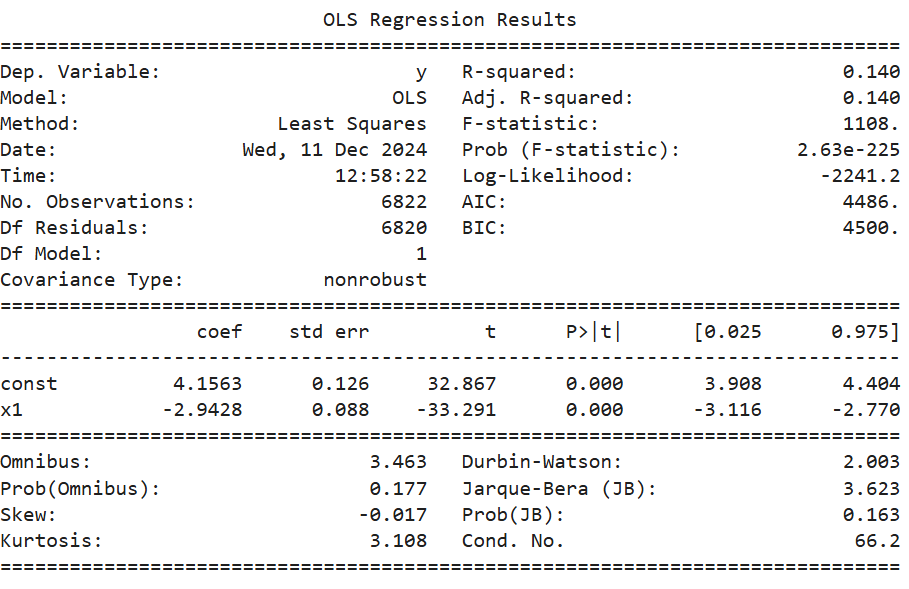


Рисунок 29 — Логарифмическая регрессия

Столбец значимости t-статистики своими значениями указывает на то, что единственный регрессор является значимым в данной модели, а значит она имеет право на существование.

На рисунке 30 изображен график полученной зависимости с доверительным интервалом.

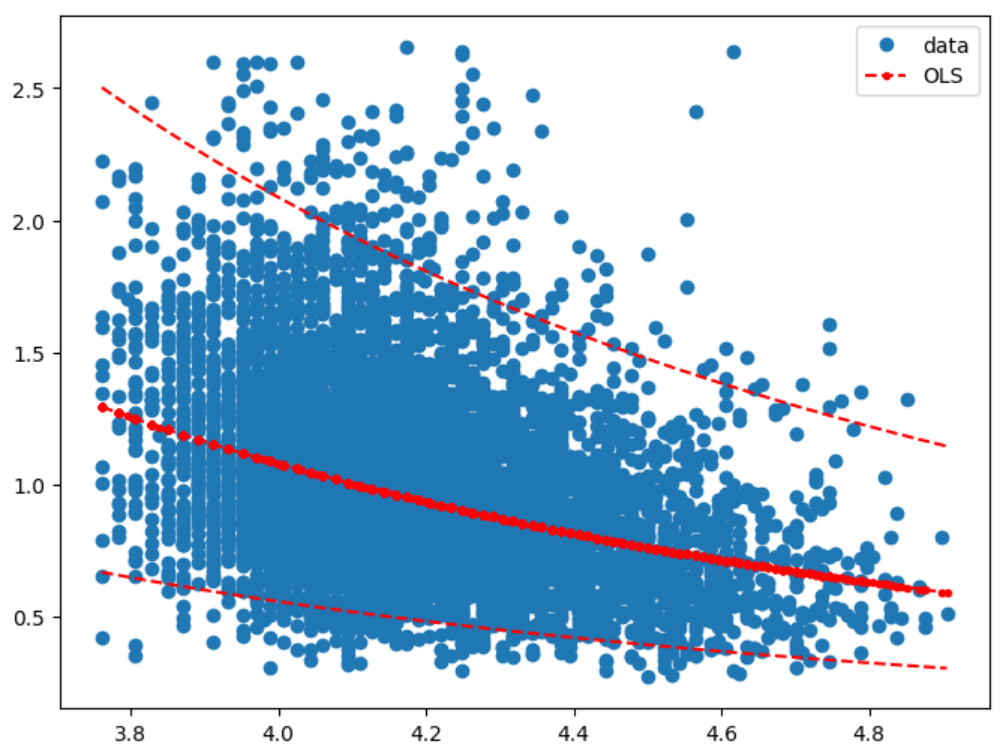


Рисунок 30 — График логарифмической регрессии

Теперь построим эмпирический мост и рассчитаем верхнюю оценку p-value. Результат приведен на рисунке 31.

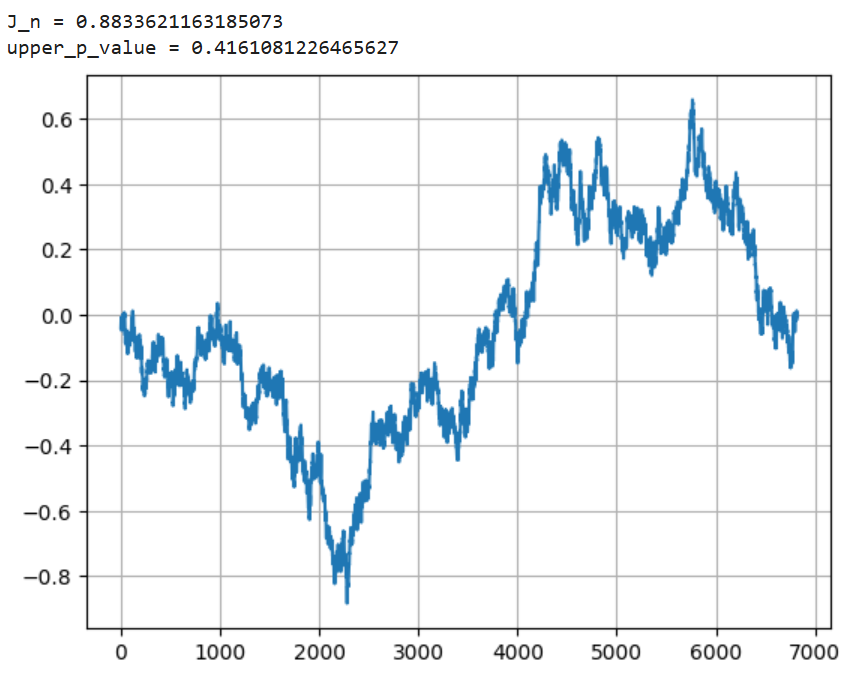


Рисунок 31 — Эмпирический мост для логарифмической регрессии

Здесь наблюдаем неплохую оценку p-value и довольно частые промежутки возрастания или убывания графика эмпирического моста, результаты чуть хуже, чем для линейной регрессии.

## Интерпретация результатов

Наименее удачной получилась регрессия на Х2, будем рассматривать линейную и логарифмическую модели. По графику и верхней оценке p-value линейная регрессия получилась более удачной, но так как логарифмическая модель также показала неплохой результат, проведем стохастическое моделирование для обеих моделей, чтобы точно определиться с наилучшей на данном этапе.

Используя алгоритмы, описанные выше в предыдущих разделах, получим, что для линейной регрессии эмпирическое p-value равно 0.1999, а для логарифмической — 0.10925.

Учитывая, что мы имеем право считать определенными точно до двух знаков после запятой, наилучшей моделью при логарифмировании данных о массе тела является линейная регрессионная модель с наивысшим и удовлетворительным эмпирическим значением p-value равным 0.1999.

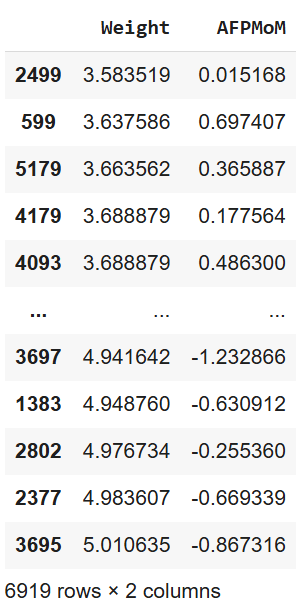
# МОДЕЛИРОВАНИЕ НА ПОЛНОСТЬЮ ПРОЛОГАРИФМИРОВАННЫХ ДАННЫХ

Теперь для проверки наибольшего количества моделей попробуем построить их, предварительно прологарифмировав не только данные о массе тела, но и соответствующие значения AFPMoMs.

Важно понимать, что в результате логарифмирования данных о значениях AFPMoMs появятся отрицательные значения, что повлияет на возможность построения логарифмической модели.

Вид данных после полного логарифмирования представлен на рисунке 32.

Рисунок 32 — Данные после полного логарифмирования

В соответствии с алгоритмами из предыдущих разделов отчистим данные от выбросов. Результат представлен на рисунке 33.

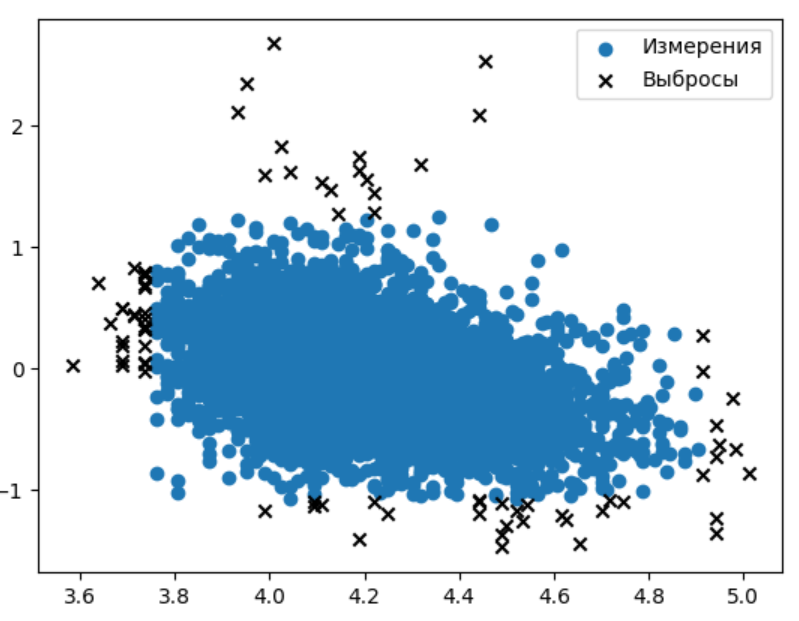


Рисунок 33 — Выбросы после полного логарифмирования

В конечном итоге данные, на которых будут производиться следующие построения регрессионных моделей на графике рассеяния принимают вид, изображенный на рисунке 34.

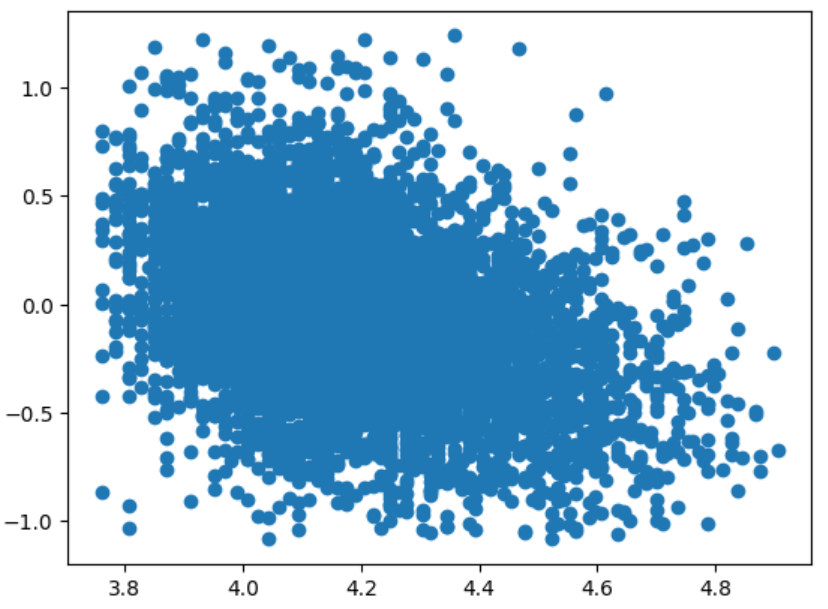


Рисунок 34 — Данные без выбросов

## Простая линейная регрессия

Построим линейную регрессию на полностью прологарифмированных данных. Результат изображен на рисунке 35.



Рисунок 35 — Линейная регрессия

Столбец значимости t-статистики вновь позволяет сделать вывод, что модель имеет право на существование.

Построим теперь полученную зависимость на графике вместе с доверительным интервалом. Результат показан на рисунке 36.

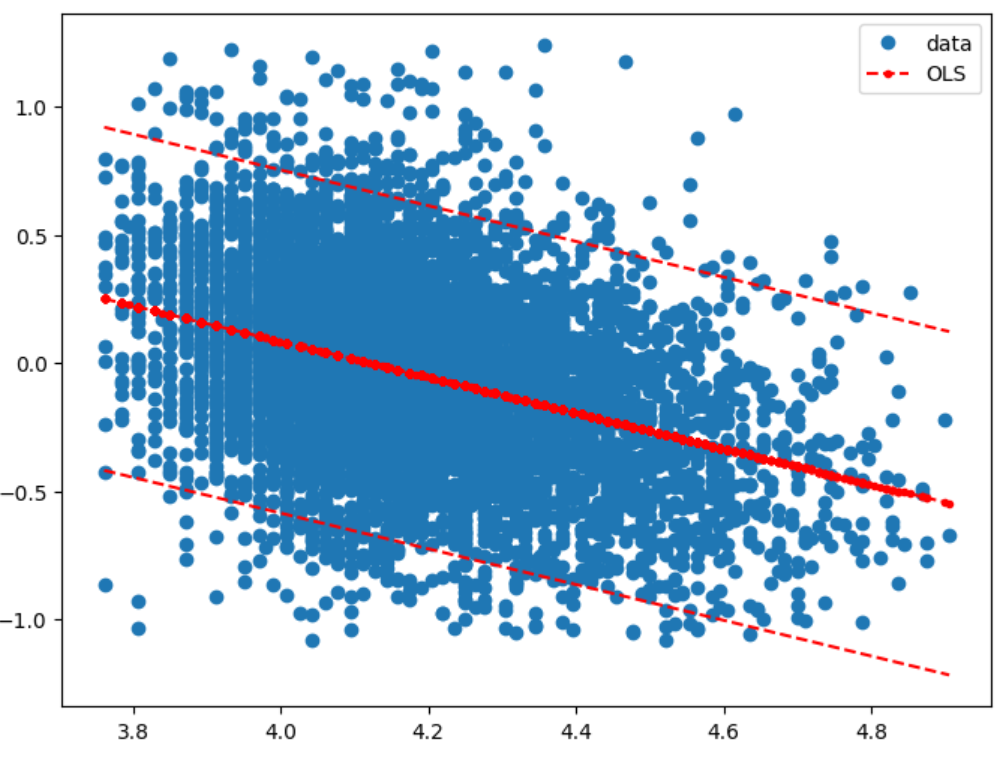


Рисунок 36 — График линейной регрессии

Теперь вновь построим эмпирический мост и найдем верхнюю оценку p-value, используя алгоритмы из предыдущих разделов. Результат изображен на рисунке 37.

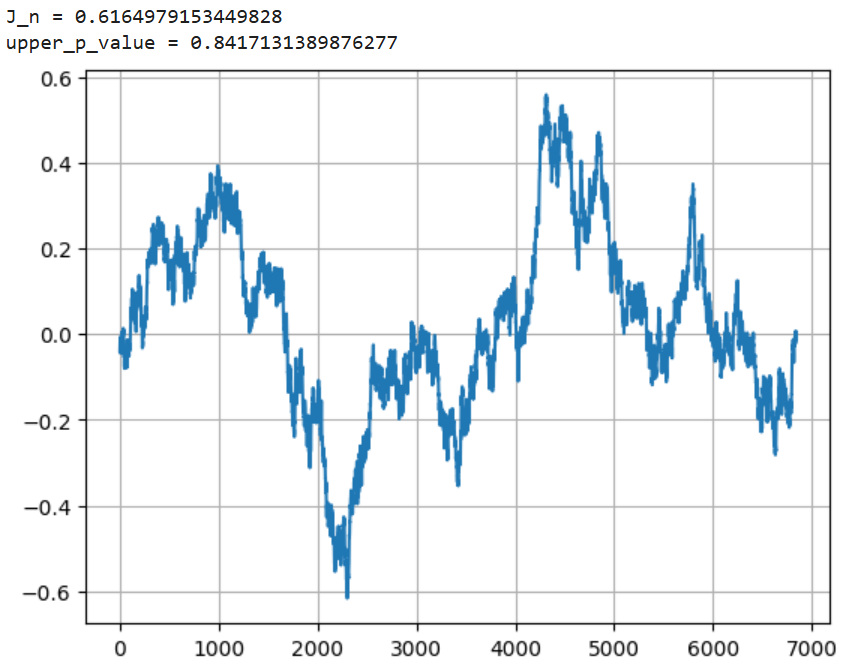


Рисунок 37 — Эмпирический мост для линейной регрессии

Имеем большую верхнюю оценку p-value и наиболее удачный на данный момент график эмпирического моста.

## Регрессия на Х2

Теперь повторим все те же действия для регрессии на Х2. Результат построения модели показан на рисунке 38.



Рисунок 38 — Регрессия на Х2

Столбец значимости t-статистики вновь позволяет сделать вывод, что модель имеет право на существование.

Покажем на графике полученную зависимость и доверительные интервалы. Результат представлен на рисунке 39.

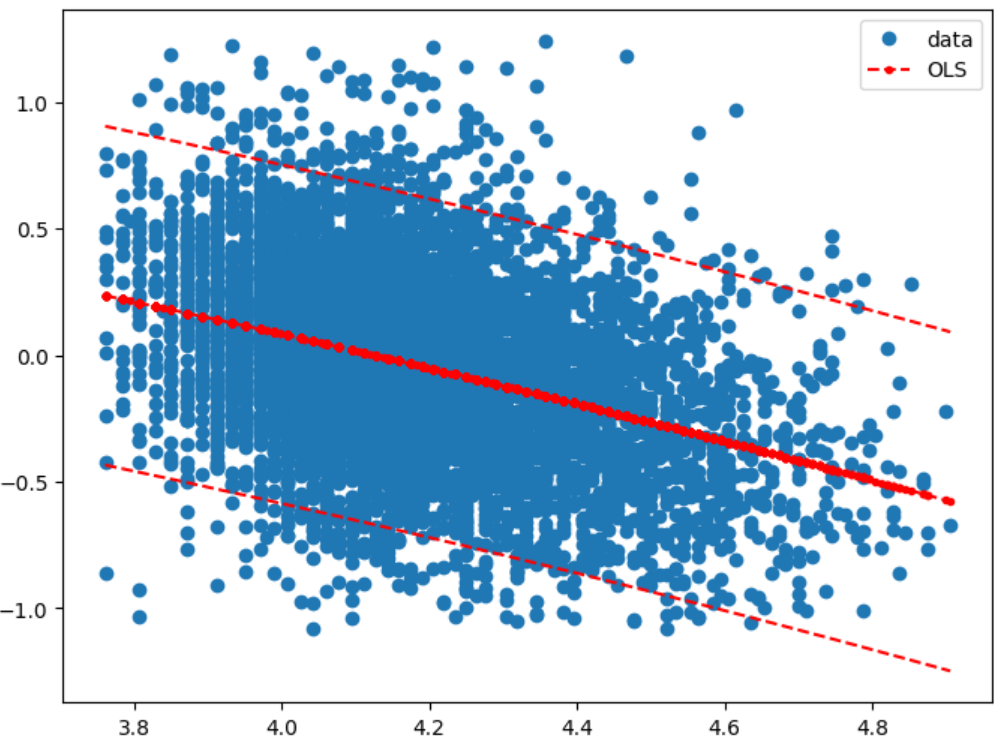


Рисунок 39 — График регрессии на Х2

Найдем верхнюю оценку p-value и построим график эмпирического моста. Результат изображен на рисунке 40.

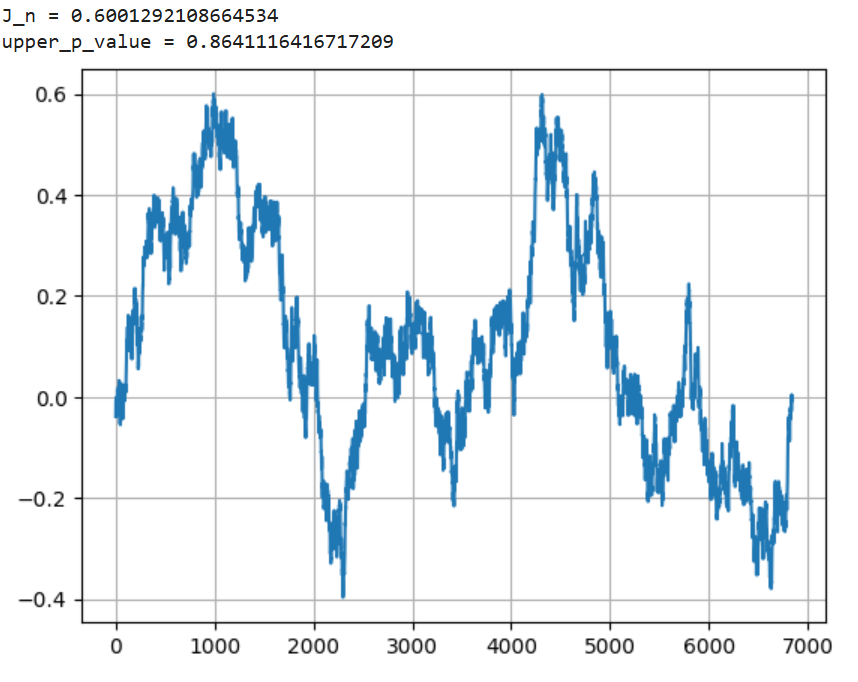


Рисунок 40 — Эмпирический мост для регрессии на Х2

На графике эмпирического моста видим картину, похожую на случай с линейной регрессией, но с меньшим интервалом колебаний графика, а также близкое немного более высокое эмпирическое значение p-value.

## Логарифмическая регрессионная модель

Построение логарифмической модели невозможно в связи с появлением отрицательных значений в столбце AFPMoMs после предварительного логарифмирования данных.

## Интерпретация результатов

Имеем линейную регрессию и регрессию на Х2 со похожими верхними оценками p-value. Для более детального рассмотрения и точного выбора одной из моделей воспользуемся стохастическим моделированием согласно алгоритмам, представленным в более ранних разделах.

После проведения моделирования имеем эмпирическое p-value для линейной регрессии равное 0.60155, для регрессии на Х2 — 0.64875. Это говорит нам о том, что в конечном итоге наиболее удачной моделью, отражающей зависимость величины AFPMoM от массы тела человека, является модель регрессии на Х2 построенная на прологарифмированных данных. Данная модель является наиболее удачной также и с точки зрения грамотного очищения от аномальных значений входных данных, благодаря чему наиболее полно отражает исследуемую зависимость.

# ВЫВОД

Построили модель зависимости произведения медиан AFPMoMs от массы тела по данным из файла, в качестве наиболее подходящей модели выбрали регрессию на Х2, сравнили ее с другими регрессионными моделями, построив эмпирический мост и проведя стохастическое моделирование.

# СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Регрессионный анализ // ArcGIS URL: https://doc.arcgis.com/ru/insights/latest/analyze/regression-analysis.htm (дата обращения: 01.12.2024).
2. Интерпретация summary из statsmodels для линейной регрессии // Хабр URL: https://habr.com/ru/articles/681218/ (дата обращения: 08.12.2024).
3. Alpha-Fetoprotein (AFP) Test // MedlinePlus URL: https://medlineplus.gov/lab-tests/alpha-fetoprotein-afp-test/ (дата обращения: 10.12.2024).
4. Alpha-Fetoprotein (AFP), Single Marker Screen, Maternal, Serum // MAYO CLINIC LABORATORIES URL: https://www.mayocliniclabs.com/test-catalog/overview/113382#Specimen (дата обращения: 10.12.2024).

# ПРИЛОЖЕНИЕ А: ЛИСТИНГ ПРОГРАММЫ

import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

import statsmodels

import statsmodels.api as sm

from scipy import stats

data021 = pd.read\_excel('/content/drive/MyDrive/Colab Notebooks/Анализ данных на Python/AFPMoMs (1).xls', sheet\_name='FRA021', header=1).drop(['Item', 'AFPCorrMoM'], axis=1)

data021

data021.info()

data004 = pd.read\_excel('/content/drive/MyDrive/Colab Notebooks/Анализ данных на Python/AFPMoMs (1).xls', sheet\_name='FRA004', header=1).drop(['Item', 'AFPCorrMoM'], axis=1)

data004

data004.info()

data011 = pd.read\_excel('/content/drive/MyDrive/Colab Notebooks/Анализ данных на Python/AFPMoMs (1).xls', sheet\_name='FRA011', header=1).drop(['Item', 'AFPCorrMoM'], axis=1)

data011

data011.info()

data011['Weight'] = data011['Weight'].astype(float)

data011.info()

data = pd.concat([data021, data004, data011], ignore\_index=True)

data = data.dropna(how='any', axis=0)

data\_sorted = data.sort\_values(by='Weight')

data\_sorted

fig, axes = plt.subplots(nrows=2, ncols=2, figsize=(12, 9))

p1 = sns.ecdfplot(data\_sorted, x=data\_sorted['Weight'], ax=axes[0, 0])

p2 = sns.ecdfplot(data\_sorted, x=data\_sorted['AFPMoM'], ax=axes[0, 1])

p3 = sns.violinplot(data\_sorted['Weight'], ax=axes[1, 0])

p4 = sns.violinplot(data\_sorted['AFPMoM'], ax=axes[1, 1])

plt.show()

data\_sorted.describe()

data = data\_sorted.copy()

data1 = data\_sorted.copy()

# выбросы

array=data\_sorted.to\_numpy()

q\_025 = np.quantile(array[:, 0], 0.25)

q\_075 = np.quantile(array[:, 0], 0.75)

iqd = q\_075 - q\_025

upper\_limit = q\_075 + 2 \* iqd

lower\_limit = q\_025 - 1.2 \* iqd

print('Верхняя граница веса=', upper\_limit)

print('Нижняя граница веса=', lower\_limit)

ejections\_weight = data\_sorted[(data\_sorted['Weight'] > upper\_limit) | (data\_sorted['Weight'] < lower\_limit)]

data\_sorted = data\_sorted.drop(ejections\_weight.index)

q\_025 = np.quantile(array[:, 1], 0.25)

q\_075 = np.quantile(array[:, 1], 0.75)

iqd = q\_075 - q\_025

upper\_limit = q\_075 + 1.75 \* iqd

lower\_limit = q\_025 - 1.75 \* iqd

print('\nВерхняя граница AFPMoM=', upper\_limit)

print('Нижняя граница AFPMoM=', lower\_limit)

ejections\_afp = data\_sorted[(data\_sorted['AFPMoM'] > upper\_limit) | (data\_sorted['AFPMoM'] < lower\_limit)]

data\_sorted = data\_sorted.drop(ejections\_afp.index)

ejections = pd.concat([ejections\_weight, ejections\_afp], ignore\_index=False)

plt.scatter(x=data\_sorted['Weight'], y=data\_sorted['AFPMoM'], label='Измерения')

plt.scatter(x=ejections['Weight'], y=ejections['AFPMoM'], c='black', marker='x', label='Выбросы')

plt.legend()

plt.show()

plt.scatter(x=data\_sorted['Weight'], y=data\_sorted['AFPMoM'])

plt.show()

arr=data\_sorted.to\_numpy()

X=arr[:,0]

Y=arr[:,1]

X1 = sm.add\_constant(X)

model = sm.OLS(Y, X1)

results = model.fit()

print(results.summary())

res=sm.OLS(Y, X1).fit()

pred\_ols = res.get\_prediction()

iv\_l = pred\_ols.summary\_frame()["obs\_ci\_lower"]

iv\_u = pred\_ols.summary\_frame()["obs\_ci\_upper"]

fig, ax = plt.subplots(figsize=(8, 6))

ax.plot(X, Y, "o", label="data")

ax.plot(X, res.fittedvalues, "r--.", label="OLS")

ax.plot(X, iv\_u, "r--")

ax.plot(X, iv\_l, "r--")

ax.legend(loc="best")

plt.show()

# Эмпирический мост

simple\_regr\_res=Y-res.fittedvalues

sum\_regr\_res=np.cumsum(simple\_regr\_res)

sigma=np.std(simple\_regr\_res)

emp\_bridge=sum\_regr\_res/len(sum\_regr\_res)\*\*0.5/sigma

J=np.max(np.abs(emp\_bridge))

p\_value=stats.kstwobign.sf(J)

print('J\_n =',J)

print('upper\_p\_value =',p\_value)

plt.plot(emp\_bridge)

plt.grid()

plt.show()

arr=data\_sorted.to\_numpy()

X=arr[:,0]

Y=arr[:,1]

X2=np.power(X,2)

X2\_1 = sm.add\_constant(X2)

results=sm.OLS(Y, X2\_1).fit()

pred\_ols = results.get\_prediction()

iv\_l = pred\_ols.summary\_frame()["obs\_ci\_lower"]

iv\_u = pred\_ols.summary\_frame()["obs\_ci\_upper"]

print(results.summary())

fig, ax = plt.subplots(figsize=(8, 6))

ax.plot(X, Y, "o", label="data")

ax.plot(X, results.fittedvalues, "r--.", label="OLS")

ax.plot(X, iv\_u, "r--")

ax.plot(X, iv\_l, "r--")

ax.legend(loc="best")

plt.show()

# Эмпирический мост

simple\_regr\_res=Y-results.fittedvalues

sum\_regr\_res=np.cumsum(simple\_regr\_res)

sigma=np.std(simple\_regr\_res)

emp\_bridge=sum\_regr\_res/len(sum\_regr\_res)\*\*0.5/sigma

J=np.max(np.abs(emp\_bridge))

p\_value=stats.kstwobign.sf(J)

print('J\_n =', J)

print('upper\_p\_value =', p\_value)

plt.plot(emp\_bridge)

plt.grid()

plt.show()

arr=data\_sorted.to\_numpy()

X=arr[:,0]

Y=arr[:,1]

LX=np.log(X)

LY=np.log(Y)

LX1 = sm.add\_constant(LX)

results=sm.OLS(LY, LX1).fit()

print(results.summary())

pred\_ols = results.get\_prediction()

iv\_l = pred\_ols.summary\_frame()["obs\_ci\_lower"]

iv\_u = pred\_ols.summary\_frame()["obs\_ci\_upper"]

fig, ax = plt.subplots(figsize=(8, 6))

ax.plot(X, Y, "o", label="data")

ax.plot(X, np.exp(results.fittedvalues), "r--.", label="OLS")

ax.plot(X, np.exp(iv\_u), "r--")

ax.plot(X, np.exp(iv\_l), "r--")

ax.legend(loc="best")

plt.show()

# Эмпирический мост

simple\_regr\_res=LY-results.fittedvalues

sum\_regr\_res=np.cumsum(simple\_regr\_res)

sigma=np.std(simple\_regr\_res)

emp\_bridge=sum\_regr\_res/len(sum\_regr\_res)\*\*0.5/sigma

J=np.max(np.abs(emp\_bridge))

p\_value=stats.kstwobign.sf(J)

print('J\_n =', J)

print('upper\_p\_value =', p\_value)

plt.plot(emp\_bridge)

plt.grid()

plt.show()

n\_iter = 20000

n=len(X)

counter = 0

for i in range(n\_iter):

  Y\_temp = stats.norm.rvs(0, 1, n)\*sigma + LX

  res=sm.OLS(Y\_temp, LX1).fit()

  simple\_regr\_res = Y\_temp-res.fittedvalues

  sum\_regr\_res = np.cumsum(simple\_regr\_res)

  emp\_bridge = sum\_regr\_res/len(sum\_regr\_res)\*\*0.5/np.std(simple\_regr\_res)

  J\_temp=np.max(np.abs(emp\_bridge))

  if J\_temp>J:

    counter+=1

print('Эмпирическое пи-значение =', counter/n\_iter)

data

data['Weight'] = np.log(data['Weight'])

data

# построим график

plt.scatter(x=data['Weight'], y=data['AFPMoM'])

plt.show()

# выбросы

array1 = data.to\_numpy()

q\_025 = np.quantile(array1[:,0], 0.25)

q\_075 = np.quantile(array1[:,0], 0.75)

iqd = q\_075 - q\_025

upper\_limit = q\_075 + 2.5 \* iqd

lower\_limit = q\_025 - 1.2 \* iqd

print('Верхняя граница веса=', upper\_limit)

print('Нижняя граница веса=', lower\_limit)

ejections\_weight1 = data[(data['Weight'] > upper\_limit) | (data['Weight'] < lower\_limit)]

data = data.drop(ejections\_weight1.index)

q\_025 = np.quantile(array1[:,1], 0.25)

q\_075 = np.quantile(array1[:,1], 0.75)

iqd = q\_075 - q\_025

upper\_limit = q\_075 + 3 \* iqd

lower\_limit = q\_025 - iqd

print('\nВерхняя граница AFPMoM=', upper\_limit)

print('Нижняя граница AFPMoM=', lower\_limit)

ejections\_afp1 = data[(data['AFPMoM'] > upper\_limit) | (data['AFPMoM'] < lower\_limit)]

data = data.drop(ejections\_afp1.index)

ejections1 = pd.concat([ejections\_weight1, ejections\_afp1], ignore\_index=False)

ejections1.info()

plt.scatter(x=data['Weight'], y=data['AFPMoM'], label='Измерения')

plt.scatter(x=ejections1['Weight'], y=ejections1['AFPMoM'], c='black', marker='x', label='Выбросы')

plt.legend()

plt.show()

plt.scatter(x=data['Weight'], y=data['AFPMoM'])

plt.show()

arr=data.to\_numpy()

X=arr[:,0]

Y=arr[:,1]

X1 = sm.add\_constant(X)

model = sm.OLS(Y, X1)

results = model.fit()

print(results.summary())

res=sm.OLS(Y, X1).fit()

pred\_ols = res.get\_prediction()

iv\_l = pred\_ols.summary\_frame()["obs\_ci\_lower"]

iv\_u = pred\_ols.summary\_frame()["obs\_ci\_upper"]

fig, ax = plt.subplots(figsize=(8, 6))

ax.plot(X, Y, "o", label="data")

ax.plot(X, res.fittedvalues, "r--.", label="OLS")

ax.plot(X, iv\_u, "r--")

ax.plot(X, iv\_l, "r--")

ax.legend(loc="best")

plt.show()

# Эмпирический мост

simple\_regr\_res=Y-res.fittedvalues

sum\_regr\_res=np.cumsum(simple\_regr\_res)

sigma=np.std(simple\_regr\_res)

emp\_bridge=sum\_regr\_res/len(sum\_regr\_res)\*\*0.5/sigma

J=np.max(np.abs(emp\_bridge))

p\_value=stats.kstwobign.sf(J)

print('J\_n =',J)

print('upper\_p\_value =',p\_value)

plt.plot(emp\_bridge)

plt.grid()

plt.show()

# стохастическое моделирование

n\_iter=20000

n=len(X)

counter=0

for i in range(n\_iter):

  Y\_temp=stats.norm.rvs(0,1,n)\*sigma+X

  res=sm.OLS(Y\_temp, X1).fit()

  simple\_regr\_res=Y\_temp-res.fittedvalues

  sum\_regr\_res=np.cumsum(simple\_regr\_res)

  emp\_bridge=sum\_regr\_res/len(sum\_regr\_res)\*\*0.5/np.std(simple\_regr\_res)

  J\_temp=np.max(np.abs(emp\_bridge))

  if J\_temp>J:

    counter+=1

print('Эмпирическое пи-значение =',counter/n\_iter)

arr=data.to\_numpy()

X=arr[:,0]

Y=arr[:,1]

X2=np.power(X,2)

X2\_1 = sm.add\_constant(X2)

results=sm.OLS(Y, X2\_1).fit()

pred\_ols = results.get\_prediction()

iv\_l = pred\_ols.summary\_frame()["obs\_ci\_lower"]

iv\_u = pred\_ols.summary\_frame()["obs\_ci\_upper"]

print(results.summary())

fig, ax = plt.subplots(figsize=(8, 6))

ax.plot(X, Y, "o", label="data")

ax.plot(X, results.fittedvalues, "r--.", label="OLS")

ax.plot(X, iv\_u, "r--")

ax.plot(X, iv\_l, "r--")

ax.legend(loc="best")

plt.show()

# Эмпирический мост

simple\_regr\_res=Y-results.fittedvalues

sum\_regr\_res=np.cumsum(simple\_regr\_res)

sigma=np.std(simple\_regr\_res)

emp\_bridge=sum\_regr\_res/len(sum\_regr\_res)\*\*0.5/sigma

J=np.max(np.abs(emp\_bridge))

p\_value=stats.kstwobign.sf(J)

print('J\_n =', J)

print('upper\_p\_value =', p\_value)

plt.plot(emp\_bridge)

plt.grid()

plt.show()

arr=data.to\_numpy()

X=arr[:,0]

Y=arr[:,1]

LX=np.log(X)

LY=np.log(Y)

LX1 = sm.add\_constant(LX)

results=sm.OLS(LY, LX1).fit()

print(results.summary())

pred\_ols = results.get\_prediction()

iv\_l = pred\_ols.summary\_frame()["obs\_ci\_lower"]

iv\_u = pred\_ols.summary\_frame()["obs\_ci\_upper"]

fig, ax = plt.subplots(figsize=(8, 6))

ax.plot(X, Y, "o", label="data")

ax.plot(X, np.exp(results.fittedvalues), "r--.", label="OLS")

ax.plot(X, np.exp(iv\_u), "r--")

ax.plot(X, np.exp(iv\_l), "r--")

ax.legend(loc="best")

plt.show()

# Эмпирический мост

simple\_regr\_res=LY-results.fittedvalues

sum\_regr\_res=np.cumsum(simple\_regr\_res)

sigma=np.std(simple\_regr\_res)

emp\_bridge=sum\_regr\_res/len(sum\_regr\_res)\*\*0.5/sigma

J=np.max(np.abs(emp\_bridge))

p\_value=stats.kstwobign.sf(J)

print('J\_n =', J)

print('upper\_p\_value =', p\_value)

plt.plot(emp\_bridge)

plt.grid()

plt.show()

# стохастическое моделирование

n\_iter = 20000

n=len(X)

counter = 0

for i in range(n\_iter):

  Y\_temp = stats.norm.rvs(0, 1, n)\*sigma + LX

  res=sm.OLS(Y\_temp, LX1).fit()

  simple\_regr\_res = Y\_temp-res.fittedvalues

  sum\_regr\_res = np.cumsum(simple\_regr\_res)

  emp\_bridge = sum\_regr\_res/len(sum\_regr\_res)\*\*0.5/np.std(simple\_regr\_res)

  J\_temp=np.max(np.abs(emp\_bridge))

  if J\_temp>J:

    counter+=1

print('Эмпирическое пи-значение =', counter/n\_iter)

data1

data1['Weight'] = np.log(data1['Weight'])

data1['AFPMoM'] = np.log(data1['AFPMoM'])

data1

# построим график

plt.scatter(x=data1['Weight'], y=data1['AFPMoM'])

plt.show()

# выбросы

array2 = data.to\_numpy()

q\_025 = np.quantile(array2[:,0], 0.25)

q\_075 = np.quantile(array2[:,0], 0.75)

iqd = q\_075 - q\_025

upper\_limit = q\_075 + 2.5 \* iqd

lower\_limit = q\_025 - 1.2 \* iqd

print('Верхняя граница веса=', upper\_limit)

print('Нижняя граница веса=', lower\_limit)

ejections\_weight2 = data1[(data1['Weight'] > upper\_limit) | (data1['Weight'] < lower\_limit)]

data1 = data1.drop(ejections\_weight2.index)

q\_025 = np.quantile(array2[:,1], 0.25)

q\_075 = np.quantile(array2[:,1], 0.75)

iqd = q\_075 - q\_025

upper\_limit = q\_075 + 0.1 \* iqd

lower\_limit = q\_025 - 3.9 \* iqd

print('\nВерхняя граница AFPMoM=', upper\_limit)

print('Нижняя граница AFPMoM=', lower\_limit)

ejections\_afp2 = data1[(data1['AFPMoM'] > upper\_limit) | (data1['AFPMoM'] < lower\_limit)]

data1 = data1.drop(ejections\_afp2.index)

ejections2 = pd.concat([ejections\_weight2, ejections\_afp2], ignore\_index=False)

plt.scatter(x=data1['Weight'], y=data1['AFPMoM'], label='Измерения')

plt.scatter(x=ejections2['Weight'], y=ejections2['AFPMoM'], c='black', marker='x', label='Выбросы')

plt.legend()

plt.show()

plt.scatter(x=data1['Weight'], y=data1['AFPMoM'])

plt.show()

arr=data1.to\_numpy()

X=arr[:,0]

Y=arr[:,1]

X1 = sm.add\_constant(X)

model = sm.OLS(Y, X1)

results = model.fit()

print(results.summary())

res=sm.OLS(Y, X1).fit()

pred\_ols = res.get\_prediction()

iv\_l = pred\_ols.summary\_frame()["obs\_ci\_lower"]

iv\_u = pred\_ols.summary\_frame()["obs\_ci\_upper"]

fig, ax = plt.subplots(figsize=(8, 6))

ax.plot(X, Y, "o", label="data")

ax.plot(X, res.fittedvalues, "r--.", label="OLS")

ax.plot(X, iv\_u, "r--")

ax.plot(X, iv\_l, "r--")

ax.legend(loc="best")

plt.show()

# Эмпирический мост

simple\_regr\_res=Y-res.fittedvalues

sum\_regr\_res=np.cumsum(simple\_regr\_res)

sigma=np.std(simple\_regr\_res)

emp\_bridge=sum\_regr\_res/len(sum\_regr\_res)\*\*0.5/sigma

J=np.max(np.abs(emp\_bridge))

p\_value=stats.kstwobign.sf(J)

print('J\_n =',J)

print('upper\_p\_value =',p\_value)

plt.plot(emp\_bridge)

plt.grid()

plt.show()

# стохастическое моделирование

n\_iter=20000

n=len(X)

counter=0

for i in range(n\_iter):

  Y\_temp=stats.norm.rvs(0,1,n)\*sigma+X

  res=sm.OLS(Y\_temp, X1).fit()

  simple\_regr\_res=Y\_temp-res.fittedvalues

  sum\_regr\_res=np.cumsum(simple\_regr\_res)

  emp\_bridge=sum\_regr\_res/len(sum\_regr\_res)\*\*0.5/np.std(simple\_regr\_res)

  J\_temp=np.max(np.abs(emp\_bridge))

  if J\_temp>J:

    counter+=1

print('Эмпирическое пи-значение =',counter/n\_iter)

arr=data1.to\_numpy()

X=arr[:,0]

Y=arr[:,1]

X2=np.power(X,2)

X2\_1 = sm.add\_constant(X2)

results=sm.OLS(Y, X2\_1).fit()

pred\_ols = results.get\_prediction()

iv\_l = pred\_ols.summary\_frame()["obs\_ci\_lower"]

iv\_u = pred\_ols.summary\_frame()["obs\_ci\_upper"]

print(results.summary())

fig, ax = plt.subplots(figsize=(8, 6))

ax.plot(X, Y, "o", label="data")

ax.plot(X, results.fittedvalues, "r--.", label="OLS")

ax.plot(X, iv\_u, "r--")

ax.plot(X, iv\_l, "r--")

ax.legend(loc="best")

plt.show()

# Эмпирический мост

simple\_regr\_res=Y-results.fittedvalues

sum\_regr\_res=np.cumsum(simple\_regr\_res)

sigma=np.std(simple\_regr\_res)

emp\_bridge=sum\_regr\_res/len(sum\_regr\_res)\*\*0.5/sigma

J=np.max(np.abs(emp\_bridge))

p\_value=stats.kstwobign.sf(J)

print('J\_n =', J)

print('upper\_p\_value =', p\_value)

plt.plot(emp\_bridge)

plt.grid()

plt.show()

# стохастическое моделирование

n\_iter=20000

n=len(X)

counter=0

for i in range(n\_iter):

  Y\_temp=stats.norm.rvs(0,1,n)\*sigma+X

  res=sm.OLS(Y\_temp, X1).fit()

  simple\_regr\_res=Y\_temp-res.fittedvalues

  sum\_regr\_res=np.cumsum(simple\_regr\_res)

  emp\_bridge=sum\_regr\_res/len(sum\_regr\_res)\*\*0.5/np.std(simple\_regr\_res)

  J\_temp=np.max(np.abs(emp\_bridge))

  if J\_temp>J:

    counter+=1

print('Эмпирическое пи-значение =',counter/n\_iter)