МОСКОВСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
имени М.В.Ломоносова

Факультет вычислительной математики и кибернетики

Учебный курс
«ТЕХНОЛОГИЧЕСКИЙ ПРАКТИКУМ»
Отчет
о выполненном задании

студента 316 учебной группы факультета ВМК МГУ
Рукавица Артёма Кирилловича

Москва
2024

Содержание

1 Описание датасета, краткие характеристики (максимумы, минимумы, средние значения, пропуски и т.п.)

В качестве датасета для выполнения задания я выбрал данные о показателях сна и здоровья. Этот датасет состоит из нескольких столбцов:

1. Heart Rate Variability: Изменения интервалов между ударами сердца.
2. Body Temperature: Температура тела в градусах Цельсия.
3. Movement During Sleep: Количество движений во время сна.
4. Sleep Duration Hours: Длительность сна.
5. Sleep Quality Score: Качество сна.
6. Caffeine Intake (mg): Количество потребления кофеина в миллиграммах.
7. Stress Level: Уровень стресса.
8. Bedtime Consistency: Регулярность режима сна. Принимает значения из отрезка $[0; 1]$, где более низкие значения указывают на большую нерегулярность.
9. Light Exposure Hours: Часы воздействия света в течение дня. Отражает типичное дневное освещение.

Параметр	Count	Mean	Std	Min	25%	50%	75%	Max
Heart Rate Variability	1000	70.39	19.58	5.17	57.05	70.51	82.96	147.05
Body Temperature	1000	36.54	0.50	35.03	36.20	36.53	36.86	38.10
Movement During Sleep	1000	2.01	0.98	-1.02	1.35	2.00	2.66	5.93
Sleep Duration Hours	1000	7.47	1.54	3.11	6.39	7.50	8.50	12.36
Sleep Quality Score	1000	2.59	2.98	1.00	1.00	1.00	2.54	10.00
Caffeine Intake Mg	1000	148.26	94.03	0.00	80.63	145.72	211.24	400.00
Stress Level	1000	4.94	2.03	0.00	3.49	4.89	6.40	10.00
Bedtime Consistency	1000	0.50	0.20	0.00	0.36	0.50	0.64	1.00
Light Exposure Hours	1000	8.04	2.02	0.33	6.73	8.04	9.35	14.75

Таблица 1: **Статистические характеристики данных**

На основе статистических характеристик можно выделить несколько ключевых моментов:

1. Для продолжительности сна среднее значение составляет 7.47 часа, что является нормой для взрослого человека. Диапазон от 3.10 до 12.36 часов показывает возможные проблемы со сном в нижнем диапазоне.
2. Для температуры тела средняя значение составляет 36.54°C с небольшим стандартным отклонением (0.50), что говорит о стабильных данных.
3. Для качества сна средний балл составляет 2.59, что может говорить о сильных проблемах со сном у опрашиваемых людей.

Обратим внимание на то, что пропусков в данных нет.

2 Аппроксимация распределений данных с помощью ядерных оценок

Ядерная оценка плотности (Kernel Density Estimation, KDE) — это метод оценки вероятностной плотности случайной величины. Этот метод позволяет аппроксимировать неизвестную функцию плотности распределения по выборке данных. Визуально KDE позволяет сгладить данные и создать непрерывную кривую, которая описывает, как распределены наблюдения.

Построим KDE для `HeartRateVariability` в двух случаях: без разделения на группы длительности сна и с разделением. Разделим длительность сна на три категории: короткий сон (3-6 часов), сон средней продолжительности (7-9 часов) и долгий сон (10-12 часов). Ниже приведены графики, полученные программой на Python.

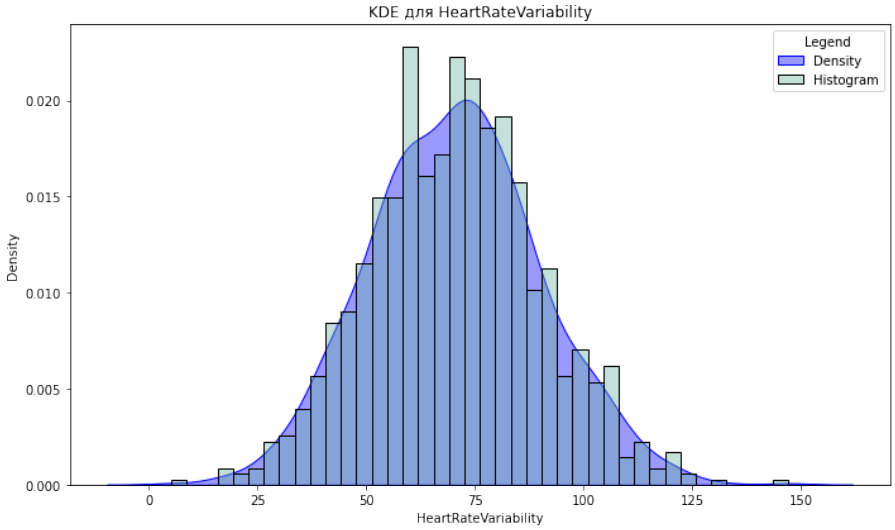

Рис. 1: KDE для HeartRateVariability на Python

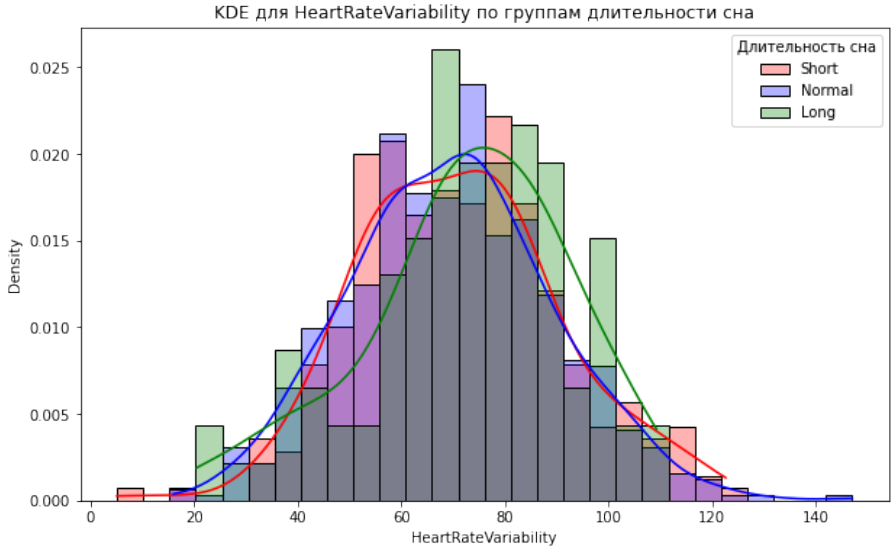

Рис. 2: KDE для HeartRateVariability по группам длительности сна на Python

Выводы по KDE-графикам

График 1: KDE для HeartRateVariability (Общая плотность распределения)

1. Распределение для HeartRateVariability имеет симметричную колоколообразную форму, схожую с нормальным распределением. Центр сосредоточен около среднего значения (порядка 70).
2. Выраженные выбросы отсутствуют.
3. Плотность, отображаемая KDE, согласуется с гистограммой, что подтверждает корректность метода сглаживания.

График 2: KDE для HeartRateVariability, разделённый по группам длительности сна

1. График показывает три группы длительности сна: *Short*, *Normal* и *Long*, каждая из которых имеет свои особенности распределения.
   - Группа *Short* характеризуется узким распределением и сдвигом плотности влево (меньшие значения HeartRateVariability).
   - Группа *Long* имеет плотность, сдвинутую вправо (большие значения HeartRateVariability).
   - Группа *Normal* демонстрирует наиболее симметричное распределение с максимальной плотностью около среднего значения.
2. Значительное перекрытие между группами говорит об отсутствии жёстких границ между состояниями.
3. Группа *Normal* имеет самую выраженную центральную тенденцию, что подтверждает связь нормальной длительности сна с средними значениями HeartRateVariability.

Общие выводы

1. Визуально HeartRateVariability имеет нормальное распределение в общей выборке (в чем мы убедимся ниже применением теста Шапиро-Уилка).
2. Python и R предоставляют возможность построить похожие графики для KDE. Для простоты изложения выше приведены графики только на Python.

3 Анализ данных с помощью cdplot, dotchart, boxplot и stripchart

3.1 cdplot

cdplot (Conditional Density Plot) — это график условной плотности, который показывает распределение одной непрерывной переменной в зависимости от другой категориальной переменной. График помогает визуализировать, как распределение непрерывной переменной изменяется в зависимости от различных категорий.

Построим график зависимости длительности сна от принятого кофеина, предварительно округлив значения часов до целых.

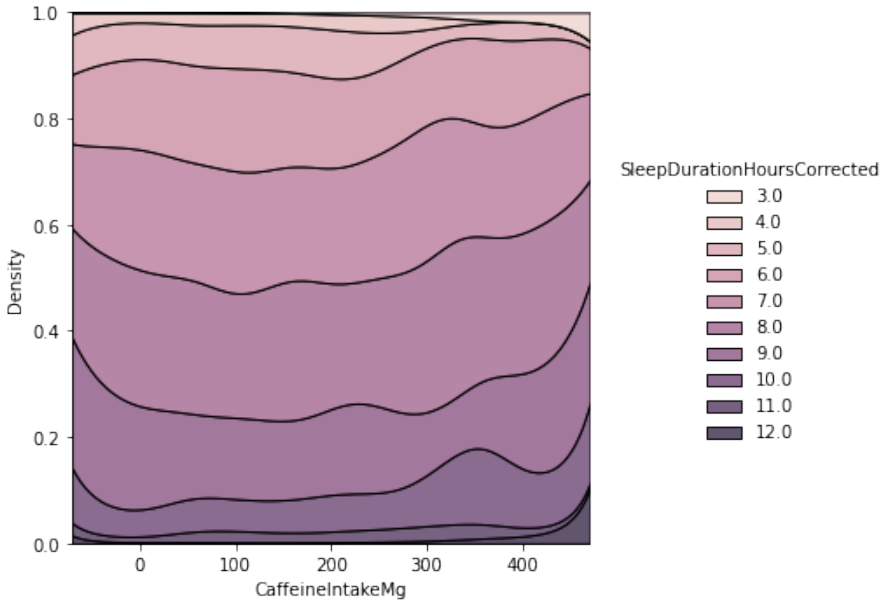

Рис. 3: cdplot для округленных значений длительности сна на Python

На этом графике трудно разобраться, что к чему относится. В параграфе выше было приведено разделение длительности сна по группам: короткий сон, сон средней длины и долгий сон. Построим cdplot для этого случая.

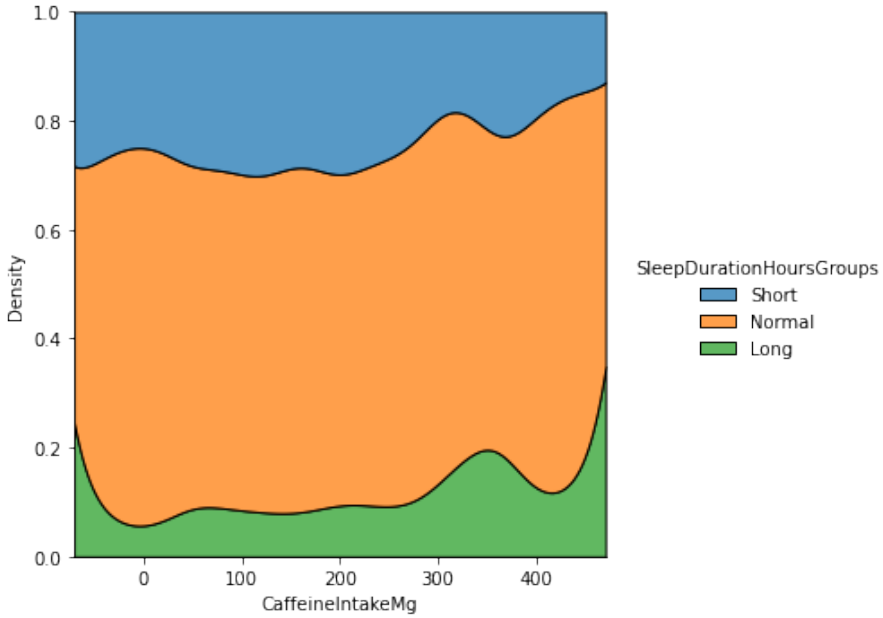

Рис. 4: cdplot для длительности сна по группам на Python

Общие выводы

1. В принципе, во всех категориях длительности сна данные распределены равномерно. Можем заметить, что большинство людей с нормальной продолжительностью сна потребляют разное количество кофеина, и их распределение остается относительно равномерным на всем диапазоне.

2. Люди с длительным сном реже встречаются среди тех, кто потребляет много кофеина, что может свидетельствовать о возможной отрицательной корреляции между

высоким уровнем потребления кофеина и продолжительностью сна.

3. Люди с коротким сном также равномерно распределены по уровню потребления кофеина, однако предпочитают потреблять не так много кофеина (их доля уменьшается после достижения отметки 350mg).

3.2 dotchart

Dotchart (точечная диаграмма) — это вид графика, который отображает значения данных с помощью точек. Точечные диаграммы используются для визуализации числовых данных и сравнений между несколькими категориями. Этот тип графика похож на столбчатую диаграмму, но вместо столбцов используются точки для отображения значений.

Построим dotchart зависимости среднего качества сна от его длительности.

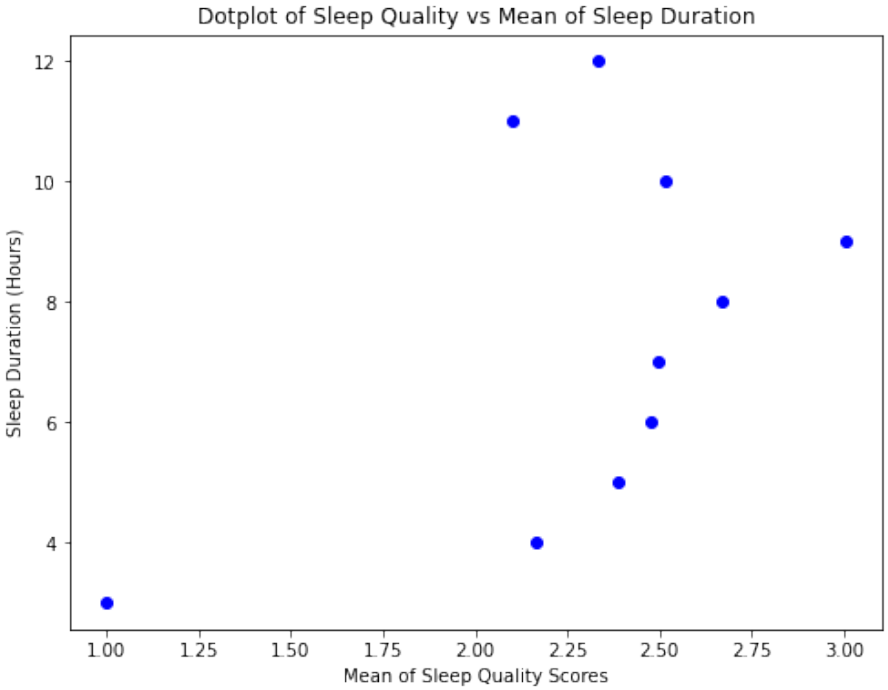

Рис. 5: dotchart для округленных значений длительности сна на Python

Данный график подтверждает гипотезу о том, что качество сна тесно связано с его продолжительностью: чем больше человек спит, тем лучше его сон оценивается в среднем. Однако после определенного порога (8-9 часов) дальнейшее увеличение продолжительности сна уже не приводит к столь значимому улучшению качества сна (а, наоборот, даже ухудшает его: по словам экспертов, после чрезмерно долгого сна человек может испытывать мигрень, позвоночную и мышечную боль и прочие симптомы, поэтому ощущения после подобного сна ухудшаются).

3.3 boxplot

Boxplot (“ящик с усами”) — это графический метод визуализации числовых данных, который отображает пять основных статистических показателей: минимальное значение, первый квартиль (Q1), медиану (Q2), третий квартиль (Q3), и максимальное значение. Это мощный инструмент для выявления распределения данных, межквартильного размаха и наличия выбросов.

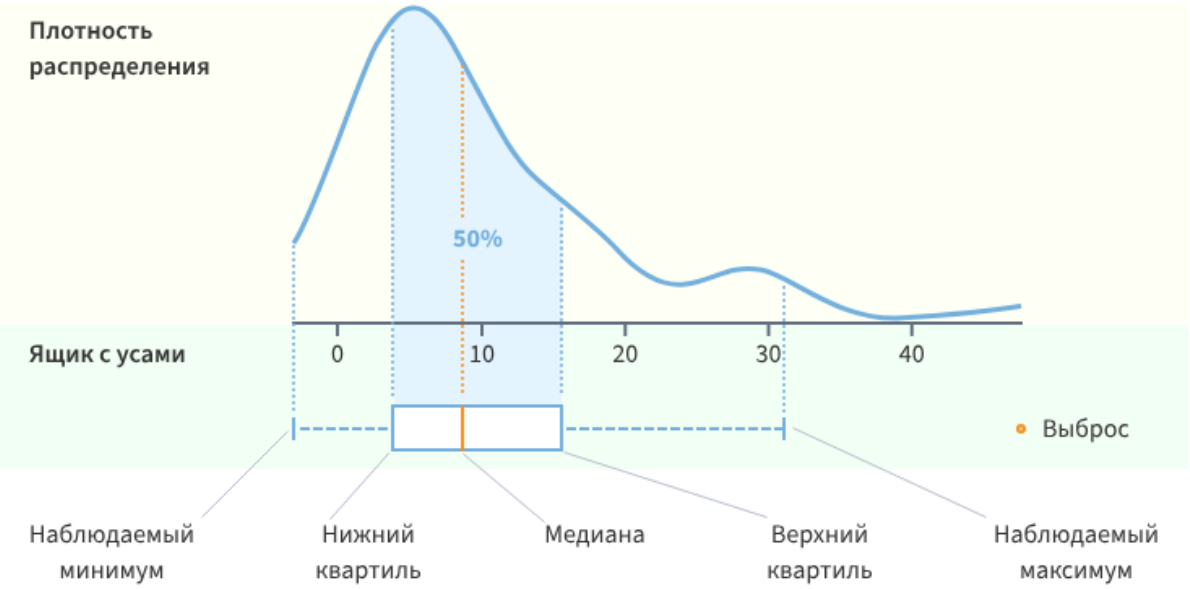

Рис. 6: Общий вид графиков boxplot

Основные элементы boxplot:

1. Медиана (Q2) — линия внутри прямоугольника, представляющая 50-й перцентиль данных, то есть середину распределения. Это центральное значение, которое делит данные пополам.

2. Первый квартиль (Q1) — нижняя граница ящика, представляющая 25-й перцентиль.

3. Третий квартиль (Q3) — верхняя граница ящика, представляющая 75-й перцентиль.

4. Межквартильный размах (IQR) — это расстояние между первым и третьим квартилями: IQR = Q3 - Q1. Этот размах используется для определения разброса данных и определения выбросов.

5. Усы (Whiskers): «Усы» на boxplot простираются от краев ящика (Q1 и Q3) до крайних точек данных, которые находятся в пределах 1.5 IQR от Q1 и Q3. «Усы» помогают определить диапазон, в который попадает основная часть данных.

6. Выбросы (Outliers) — это точки, которые находятся за пределами 1.5 IQR от Q1 и Q3, отображаются как отдельные точки за пределами усов. Эти данные могут быть аномальными или ошибочными значениями, но также могут быть важными для анализа.

Для серии boxplot-ов для удобства разделим уровень стресса на четыре категории: низкий (величина StressLevel меньше 3), средний (от 3 до 6), высокий (от 6 до 8), очень высокий (выше 8).

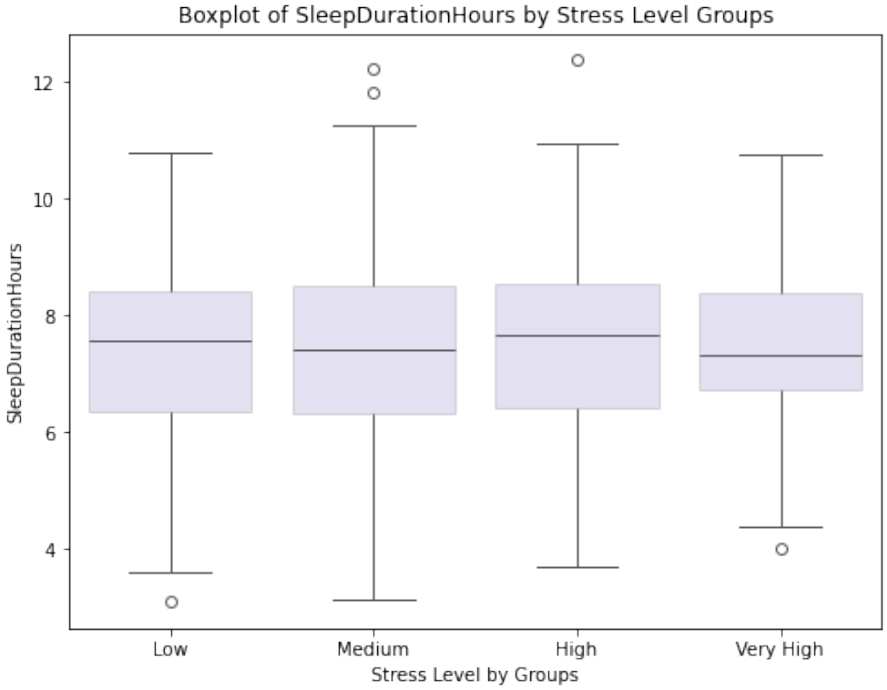

Рис. 7: Boxplot уровня сна от категории стресса на Python

Длительность сна слабо зависит от уровня стресса, однако мы можем обратить внимание на выбросы в группе Medium (вполне возможно, что два человека спят дольше по другим причинам, не из-за среднего уровня стресса).

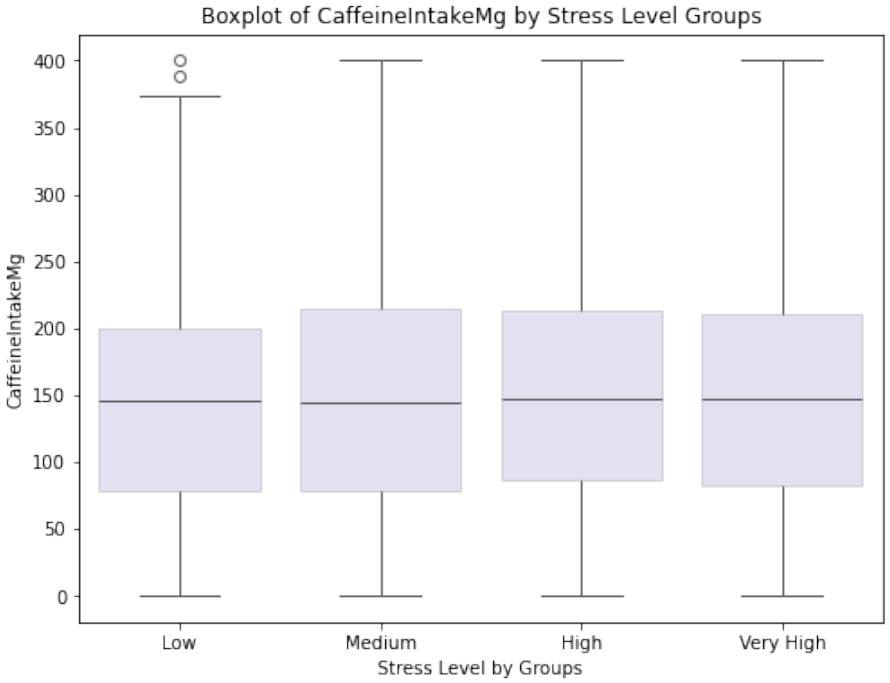

Рис. 8: Boxplot уровня кофеина от категории стресса на Python

Потребление кофеина слабо влияет на уровень стресса.

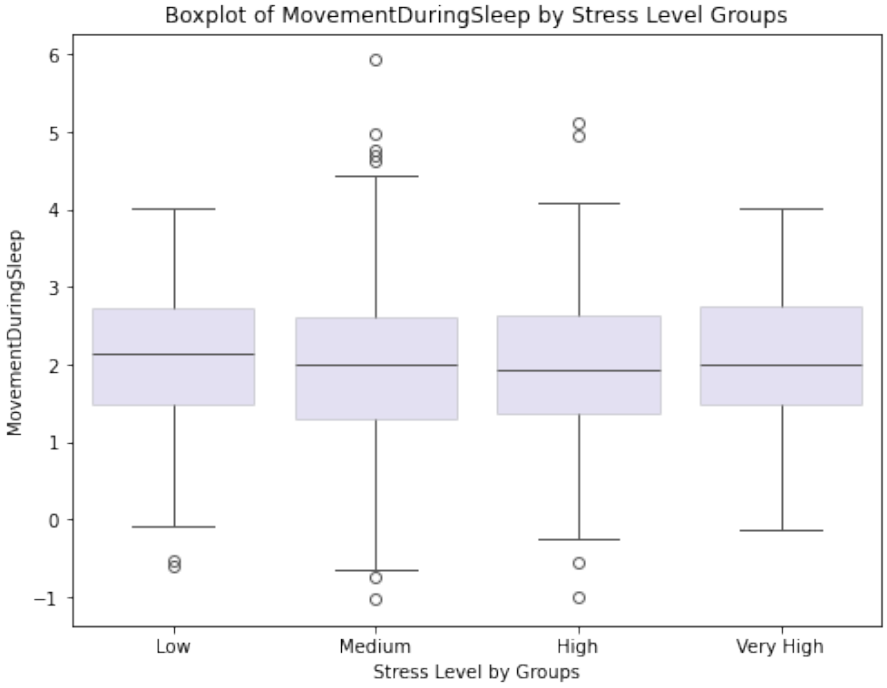

Рис. 9: Boxplot количества движений во время сна от категории стресса на Python

- Медиана почти одинакова для всех групп стресса и находится на уровне около 2 движений во время сна. Это указывает на то, что средний уровень движения во время сна не сильно изменяется в зависимости от уровня стресса.
- В группах Medium, High выбросы особенно видны выше уровня 4 единиц движения, что может свидетельствовать о том, что у некоторых людей с высоким уровнем стресса наблюдается аномально большое количество движения во время сна, однако выбросов в группах Low и Very High почти что нет.

3.4 stripchart

Stripchart — это график, который отображает распределение данных, представляя каждое отдельное наблюдение в виде точки на оси. Он полезен для визуализации небольших наборов данных и позволяет увидеть каждое уникальное значение в одном ряду.

Основные характеристики stripchart:

1. Каждая точка — это одно наблюдение: в отличие от boxplot, который отображает статистическое обобщение данных, stripchart отображает каждое наблюдение на графике.
2. Stripchart может отображать данные для нескольких категорий рядом, что позволяет легко увидеть различия между ними.
3. Когда несколько значений оказываются близко друг к другу, точки могут перекрываться, поэтому иногда используют небольшое смещение по оси (jitter), чтобы разделить их визуально.

Построим stripchart без jitter для сравнения длительности сна с потреблением кофеина.

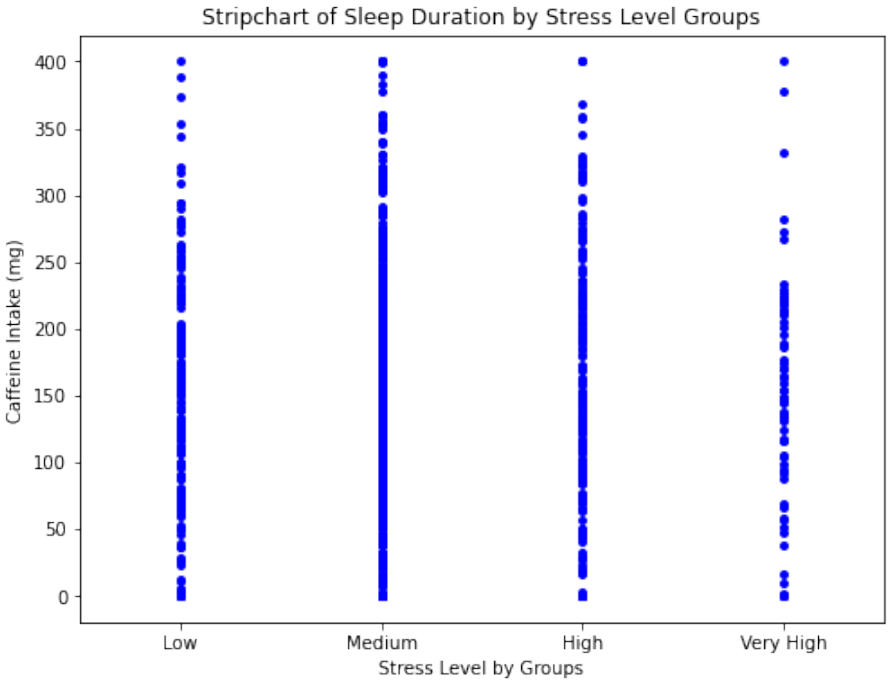

Рис. 10: Stripchart без jitter для длительности сна и потребления кофеина на Python

График тяжело анализировать, поэтому воспользуемся смещением внутри групп.

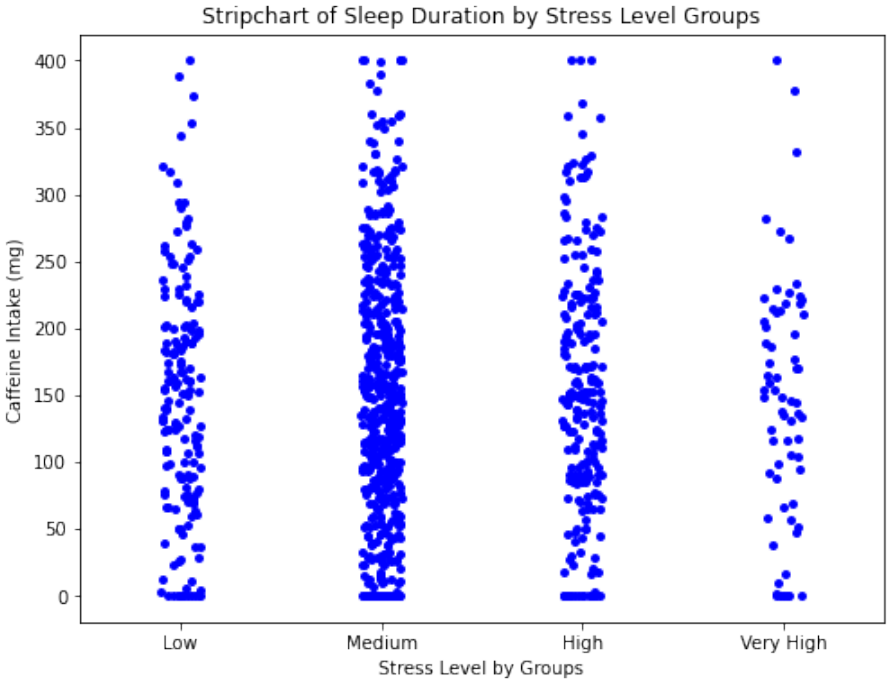

Рис. 11: Stripchart с jitter для длительности сна и потребления кофеина на Python

Выводы

1. Для людей с низким и средним уровнем стресса наблюдается более высокий и равномерный разброс значений потребления кофеина. Это может говорить о том, что

эти группы менее подвержены изменениям в привычках потребления кофеина, либо же кофеин для них играет меньшую роль в уровне стресса.

2. По мере увеличения уровня стресса количество людей, потребляющих большие дозы кофеина (свыше 200 мг), уменьшается, и в группе с очень высоким уровнем стресса наблюдается большее количество людей, которые либо совсем не потребляют кофеин, либо потребляют его в малых дозах.

3. Во всех категориях можно наблюдать точки-выбросы (например, свыше 300 мг). Это может быть вызвано индивидуальными особенностями потребления или другими факторами, влияющими на привычки людей и вызывающих у них больше или меньше стресса (в зависимости от категории).

На обоих языках программирования были построены одинаковые графики каждого типа.

4 Выявление выбросов с точки зрения формальных статистических критериев Граббса и Q-теста Диксона

Для этих критериев требуется нормальность выборки, проверим столбец MovementDuringSleep на нормальность, например, при помощи критерия Шапиро-Уилка. В Python его можно провести при помощи функции `scipy.stats.shapiro()`, в R — при помощи `shapiro.test()`.

`p-value` теста Шапиро-Уилка: $0.717817288266782 > 0.05$, поэтому эти значения действительно распределены нормально, применимы критерий Граббса и Q-тест Диксона. Построим гистограмму для этой выборки, чтобы также визуально убедиться в нормальности данных.

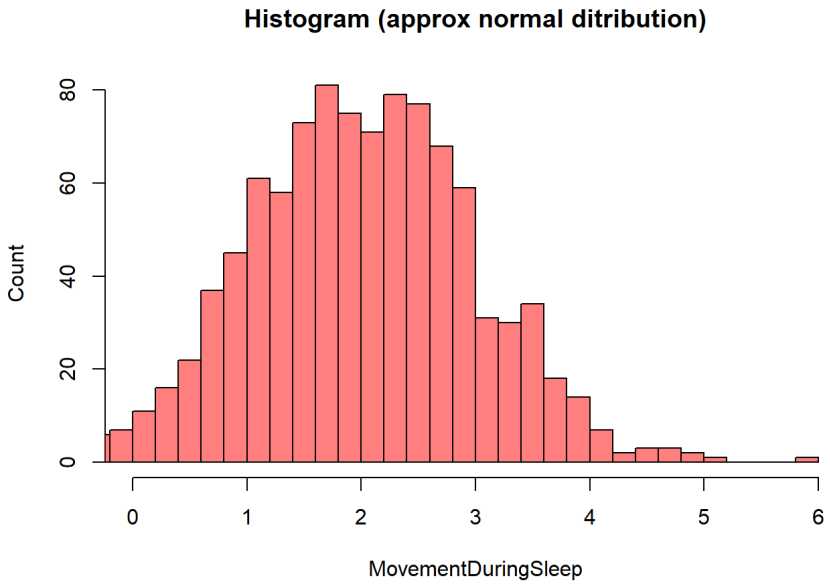

Рис. 12: Гистограмма для выборки MovementDuringSleep на R

4.1 Критерий Граббса

Критерий Граббса — статистический тест, используемый для определения выбросов в одномерном наборе данных, подчиняющихся нормальному закону распределения.

Статистика G для поиска максимального выброса вычисляется следующим образом:

$$G = \frac{X_{max} - \bar{X}}{s},$$

где X_{max} — максимальное значение, которое мы проверяем на принадлежность выбросам, $\bar{X}$; — среднее выборки, s – стандартное отклонение выборки. Если G достигает критического значения из таблицы Граббса для конкретного уровня значимости, то X_{max} — это действительно выброс. Аналогично формулируется критерий Граббса для минимума.

Критерий Граббса можно провести с помощью функции `smirnov_grubbs()` из библиотеки `outliers` на языке Python, на R — с помощью функции `grubbs.test()` из библиотеки `outliers`.

Найдем выбросы в выборке MovementDuringSleep, предварительно округлив значения в этом столбце до двух знаков после запятой. На обоих языках критерий Граббса показал одинаковый индекс выброса — 895, которому соотвествует величина MovementDuringSleep, равная 5.93. Убедимся на boxplot, действительно ли значение, соответствующее этому индексу, — выброс.

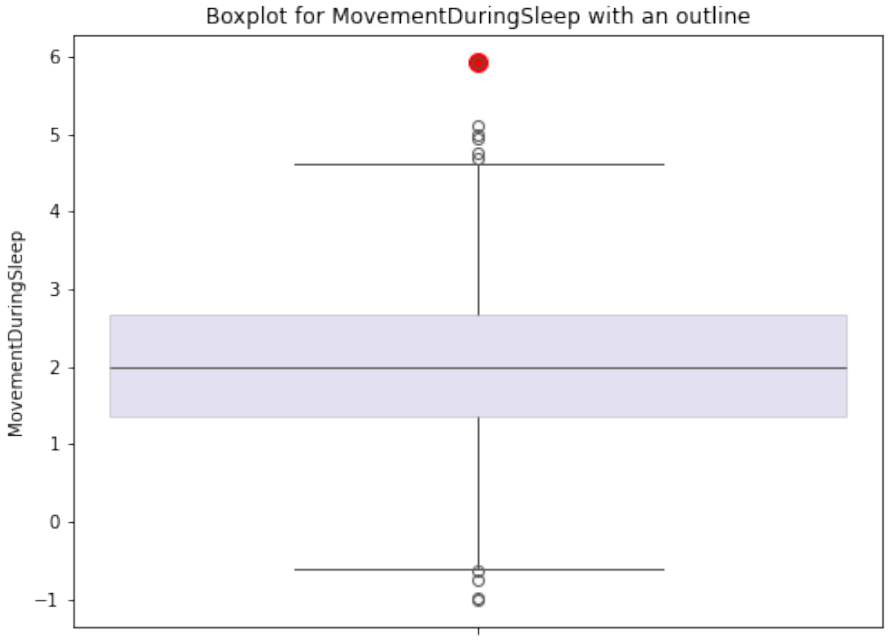

Рис. 13: Boxplot для выборки MovementDuringSleep с отмеченным выбросом на Python

4.2 Q-тест Диксона

Для теста Диксона существуют ограничение: он может корректно анализировать только малые выборки данных (от 3 до 20). По критерию Граббса мы знаем, что выброс находится на 895 позиции датасета, поэтому ограничим поиск выброса на позициях от 890 до 909.

Для проведения q-теста Диксона необходимо отсоритровать массив значений по возрастанию и посчитать число Q по формуле:

$$Q = \frac{|potential\ outlier - nearest\ neighbor|}{maximum\ value - minimum\ value}$$

Далее необходимо сравнить полученное Q с критическим значением из таблицы Q_{table},

соответствующее количеству значений в выборке на определенном уровне значимости. Если $Q > Q_{table}$, то *potential outlier* действительно является выбросом.

На языке Python нет готовой реализации q-теста Диксона, поэтому я его реализовывал сам. Критические значения для теста взяты из специальной таблицы. На языке R q-тест Диксона можно провести с помощью функции `dixon.test()` из библиотеки `outliers`.

Сравним выводы q-теста на Python и R:

- Python: `Q-value: 0.36, Suspected outlier: 5.93, Is outlier: True`.
- R:

Листинг 1: Q-тест Диксона на языке R

```
Dixon test for outliers

data:  subdata
Q = 0.48991, p-value = 0.04092
alternative hypothesis: highest value 5.93 is an outlier
```

Как мы можем заметить, статистика Q немного отличается, однако результат теста одинаковый в обоих случаях: значение 5.93 является выбросом.

5 Воспользоваться инструментами для заполнения пропусков в данных. Пропуски внести вручную и сравнить результаты заполнения с истинными значениями

Заполнение пропусков в данных — это важный этап в анализе данных, необходимым для обеспечения корректности анализа и работы моделей. Пропуски могут:

1. Искажать результаты статистических расчетов.
2. Нарушать работу алгоритмов машинного обучения, которые требуют полных данных.
3. Уменьшать объем доступной информации, если строки с пропусками удаляются.

Заполнение позволяет сохранить как можно больше данных, повысить точность анализа и избежать ошибок в моделях.

В датасете про качество сна нет пропусков, поэтому искусственно создадим их, например, в столбце BodyTemperature. Для этого я сгенерировал случайную выборку из 10 индексов: `894, 2, 198, 753, 186, 78, 45, 187, 952, 886`. Данные по этим индексам будут удалены.

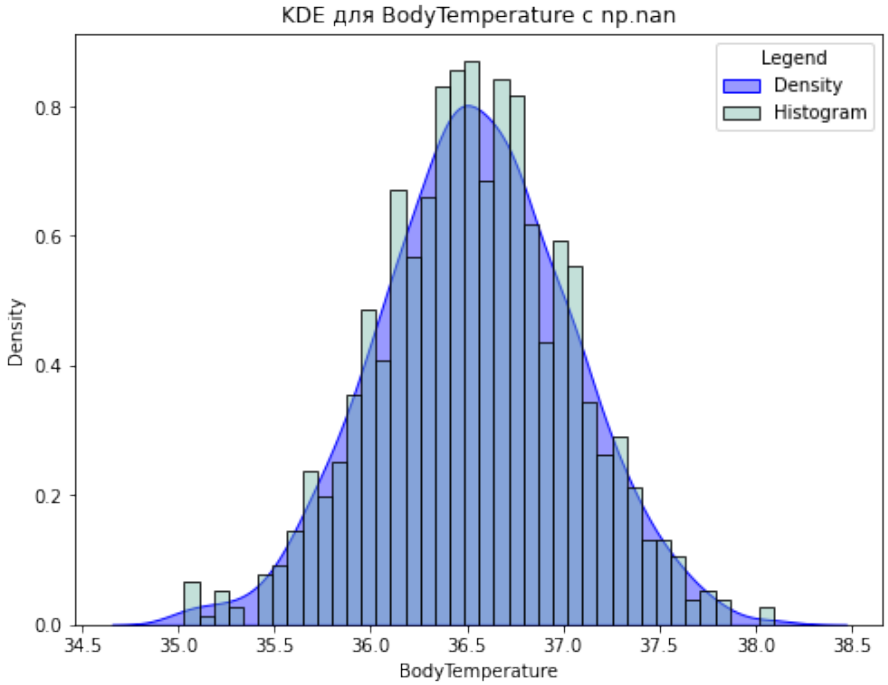

Рис. 14: Гистограмма BodyTemperature с пропусками

Я воспользуюсь двумя способами заполнения пропусков: средним значением и медианой.

1. **Заполнение пропусков средним значением**. Пропуски заменяются средним арифметическим всех известных значений в столбце. Этот метод прост в реализации, сохраняет общее распределение данных, если отсутствуют выбросы, однако есть ряд недостатков: данный способ заполнения пропусков чувствителен к выбросам (сильно завышенные или заниженные значения могут исказить среднее), а также он может уменьшить дисперсию данных, делая их более «гладкими».

2. **Заполнение пропусков медианой**. Пропуски заменяются медианой — центральным значением отсортированных данных. Этот метод устойчив к выбросам, так как медиана не зависит от крайних значений, более того, он лучше сохраняет распределение данных для асимметричных выборок.Однако этот способ может не учитывать важные особенности данных, если распределение сильно изменяется из-за пропусков.

Среднее значение и медиана после удаления данных оказались близки друг к другу (36.5315 и 36.5354 соответственно), поэтому оба способа должны дать похожие результаты.

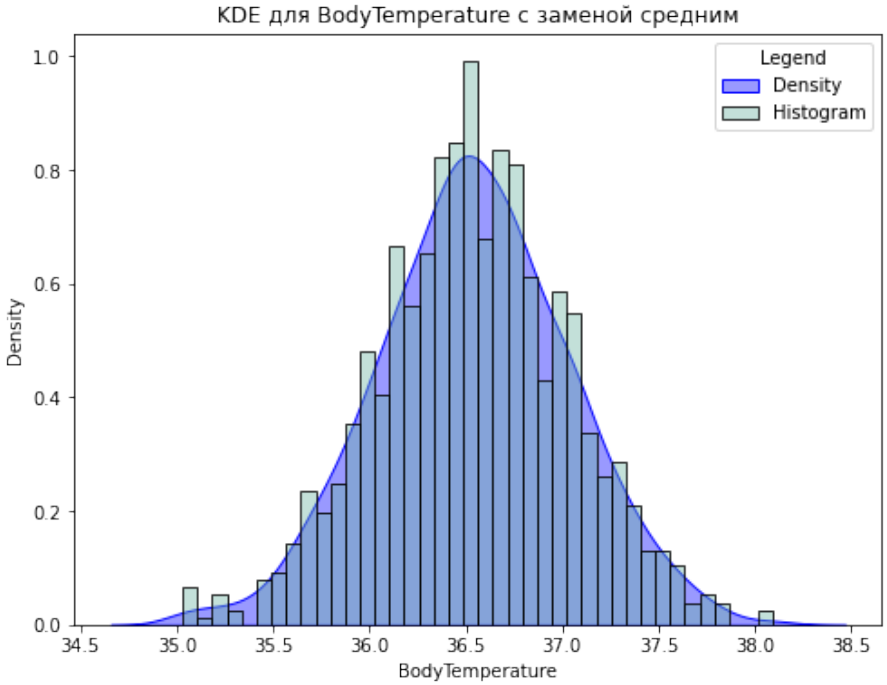

Рис. 15: Гистограмма BodyTemperature с заполнением средним

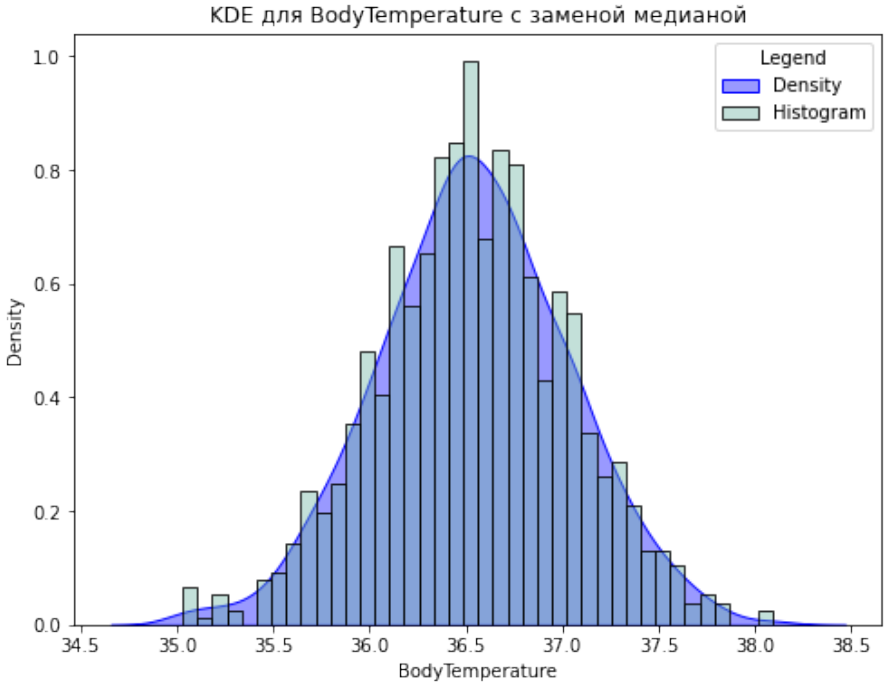

Рис. 16: Гистограмма BodyTemperature с заполнением медианой

Выводы

1. Пропуски в данных приводят к уменьшению общей плотности, так как часть данных отсутствует. Форма распределения сохраняется, но становится менее гладкой из-за отсутствия части данных, что влияет на оценку плотности.

2. При симметричном распределении среднее и медиана совпадают или близки друг к другу, поскольку нет значительных перекосов из-за выбросов.

3. На графиках видно, что заполнение пропусков средним и медианой практически не изменяет форму распределения. Это свидетельствует о том, что оба метода одинаково хорошо подходят для выборки BodyTemperature.

4. На языке R были проведены аналогичные действия по заполнению пропусков, которые дали такие же результаты.

6 Анализ нормального распределения (на малых и больших выборках)

Для выполнения данного пункта было сгенерировано пять выборок из нормального распределения со следующими параметрами:

1. **data1**: математическое ожидание $\mu_1 = 0$, среднеквадратическое отклонение $\sigma_1 = 1$, количество элементов в выборке $n_1 = 50$.

2. **data2**: математическое ожидание $\mu_2 = 2$, среднеквадратическое отклонение $\sigma_2 = 4$, количество элементов в выборке $n_2 = 100$.

3. **data3**: математическое ожидание $\mu_3 = 1$, среднеквадратическое отклонение $\sigma_3 = 4$, количество элементов в выборке $n_3 = 5000$.

4. **data4**: математическое ожидание $\mu_4 = 11$, среднеквадратическое отклонение $\sigma_4 = 3$, количество элементов в выборке $n_4 = 2000$.

5. **data5**: математическое ожидание $\mu_5 = 1$, среднеквадратическое отклонение $\sigma_5 = 1$, количество элементов в выборке $n_5 = 1000$.

6.1 Анализ с помощью графиков эмпирических функций распределений

Эмпирическая функция распределения (выборочная функция распределения) — естественное приближение теоретической функции распределения данной случайной величины, построенное по выборке.

Пусть задана случайная выборка $x^m = (x_1, \ldots, x_m)$ наблюдений $x_i \in X$. Построим по выборке ступенчатую функцию $\hat{F}_m(x)$, возрастающую скачками величины $\frac{1}{m}$ в точках $x_{(i)}$. Построенная функция называется эмпирической функцией распределения. Для задания значений в точках разрыва формально определим её так:

$$\hat{F}_m(x) = \frac{1}{m} \sum_{i=1}^{m} I\{x_i \leq x\}.$$

Замечание: при этом эмпирическая функция непрерывна справа.

Свойства эмпирической функции распределения

1. Эмпирическое распределение для фиксированного x

   Поскольку случайная величина $I\{x_i \leq x\}$ имеет распределение Бернулли с вероятностью успеха $F(x)$ (где $F(x)$ — теоретическая **функция распределения** случайной

величины x), а последовательность $(I\{x_1 \leq x\}, \ldots, I\{x_m \leq x\})$ — схема Бернулли с вероятностью успеха $F(x)$, то по отношению к этой последовательности $\hat{F}_m(x)$ есть частота попаданий левее x.

Из сказанного вытекает, что эмпирическое распределение служит естественным приближением к теоретической функции распределения.

2. Математическое ожидание и дисперсия эмпирического распределения

   Математическое ожидание эмпирической функции распределения:

$$E[\hat{F}_m(x)] = F(x),$$

   таким образом, эмпирическое распределение является **несмещённой оценкой** теоретической функции распределения $F(x)$.

   Дисперсия эмпирического распределения:

$$D[\hat{F}_m(x)] = \frac{F(x)(1 - F(x))}{m}.$$

3. Асимптотические свойства эмпирической функции распределения

   (a) По **усиленному закону больших чисел** $\hat{F}_m(x)$ сходится **почти наверное** к теоретической функции распределения $F(x)$:

$$\hat{F}_m(x) \to F(x) \quad m \to \infty.$$

   (b) 2. Выборочная функция распределения является **асимптотически нормальной** оценкой функции распределения $F(x)$, при условии, что $0 < F(x) < 1, \forall x \in R$:

$$\sqrt{m}(\hat{F}_m(x) - F(x))PN(0, F(x)(1 - F(x))) \quad m \to \infty.$$

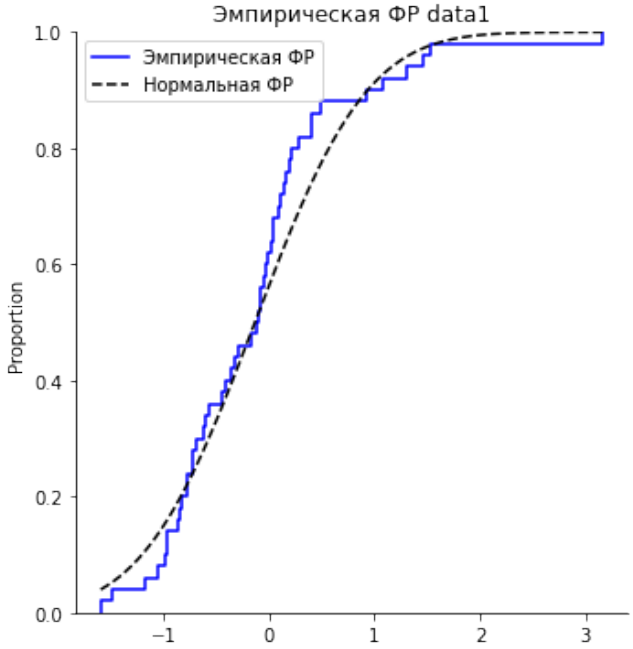

Рис. 17: Эмпирическая ФР для выборки data1

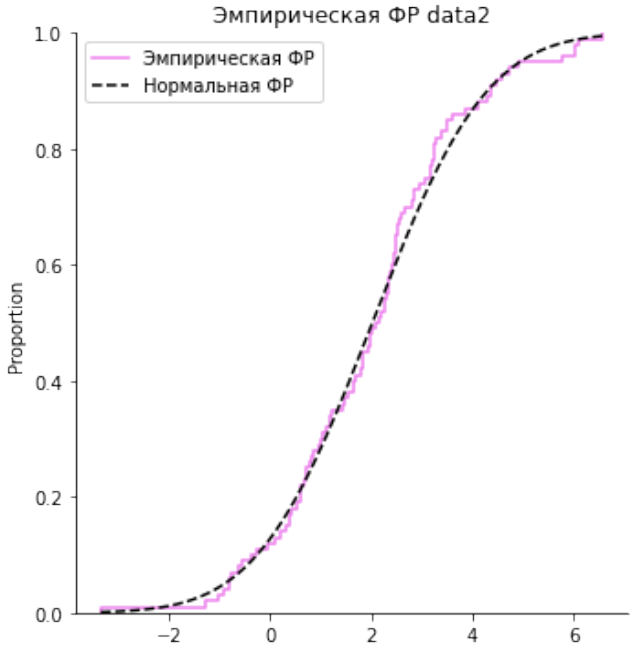

Рис. 18: Эмпирическая ФР для выборки data2

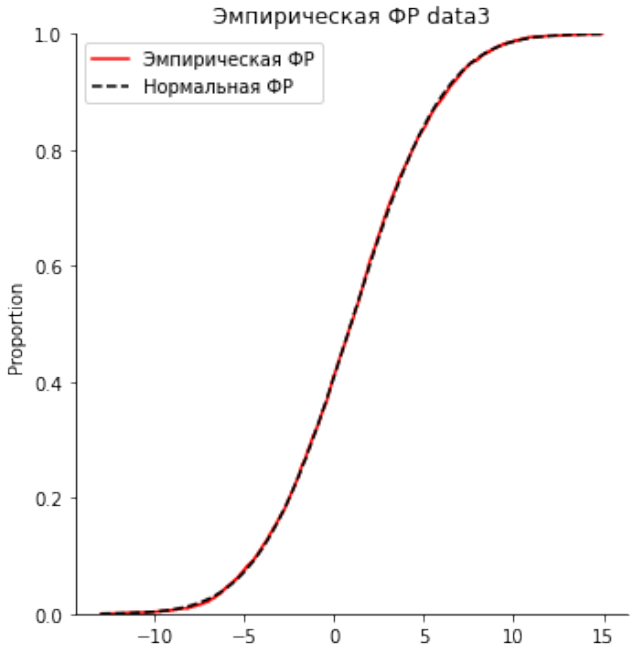

Рис. 19: Эмпирическая ФР для выборки data3

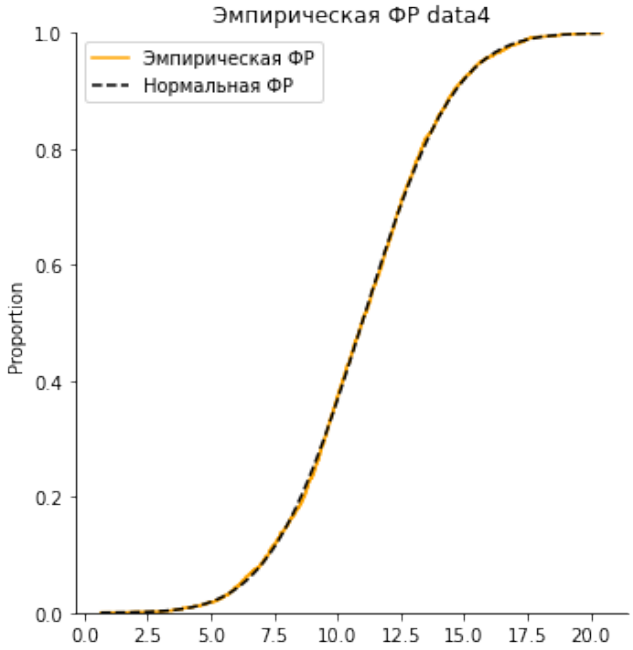

Рис. 20: Эмпирическая ФР для выборки data4

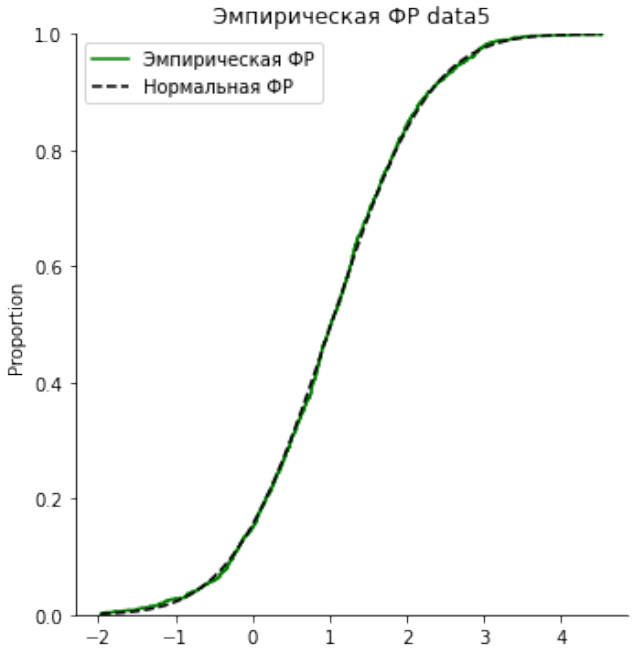

Рис. 21: Эмпирическая ФР для выборки data5

Выводы по выборкам

- **data1** ($\mu = 0, \sigma = 1, n = 50$):
  - Эмпирическая ФР совпадает с нормальной функцией распределения, показывая центр распределения около 0.
  - Небольшой размер выборки ($n = 50$) делает функцию ступенчатой, особенно на крайних значениях.
- **data2** ($\mu = 2, \sigma = 2, n = 100$):

– Эмпирическая ФР также хорошо совпадает с нормальной функцией распределения.

– Параметры ($\mu = 2$) смещают распределение вправо относительно data1, а увеличенная дисперсия ($\sigma = 2$) приводит к более пологому наклону.

- **data3, data4, data5** (выборки большого объема):

  – Эмпирические ФР практически идеально совпадают с теоретической ФР нормального распределения во всех трех случаях, что связано с большим размером выборки.

Общие выводы

На основании представленных графиков эмпирических функций распределения и параметров распределений можно сделать следующие выводы:

- Эмпирические функции распределения хорошо согласуются с теоретическими функциями нормального распределения, что подтверждает нормальный характер данных для каждой выборки.
- Различия в параметрах (математическое ожидание, среднеквадратическое отклонение, количество элементов в выборке) влияют на форму и положение эмпирической ФР.

На языке R были проведены аналогичные действия по построению графиков эмирических функций распределения, которые дали такие же результаты.

6.2 Анализ с помощью графиков квантилей

График квантилей (QQ plot) используется для сравнения распределения данных с теоретическим распределением или с другим набором данных. На осях координат нанесены квантили выборки и квантили теоретического распределения (или другой выборки). Если точки лежат близко к диагональной прямой, распределение выборки соответствует теоретическому.

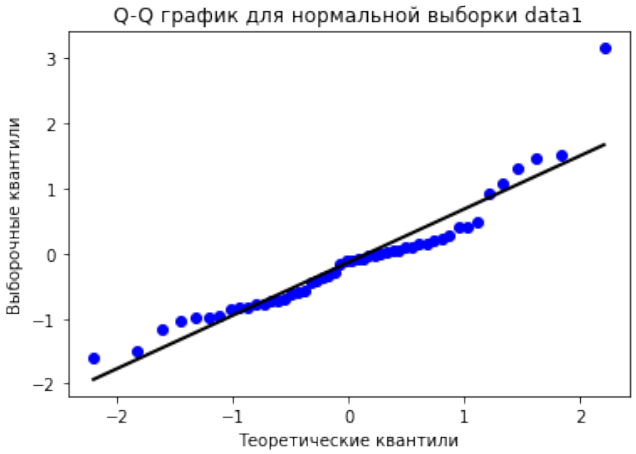

Рис. 22: QQ-plot для выборки data1

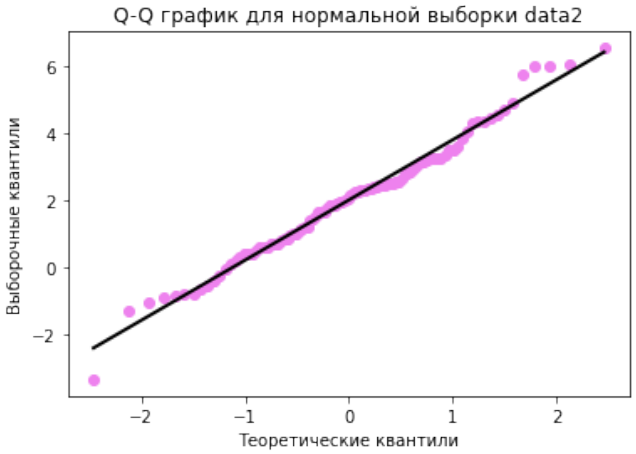

Рис. 23: QQ-plot для выборки data2

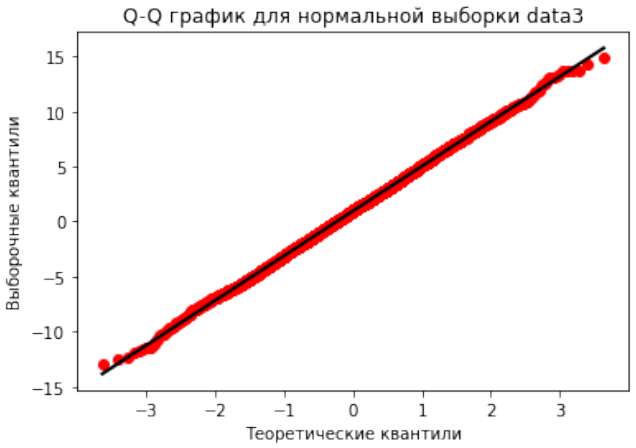

Рис. 24: QQ-plot для выборки data3

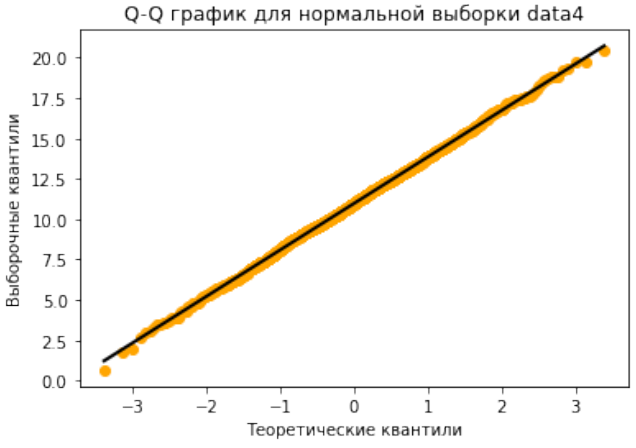

Рис. 25: QQ-plot для выборки data4

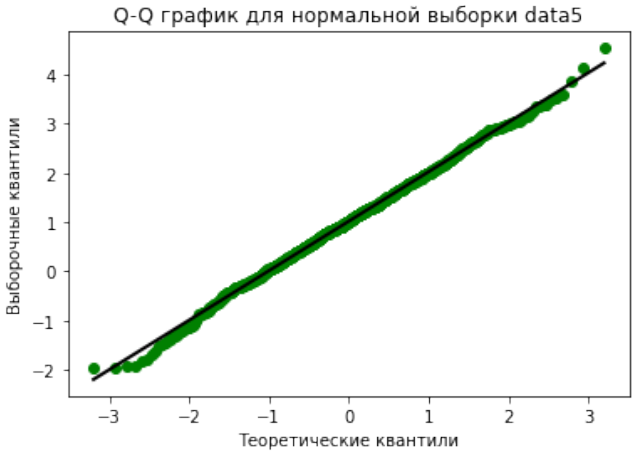

Рис. 26: QQ-plot для выборки data5

Общие выводы

1. На графиках для выборок малого объема (data1, data2) отклонения от теоретической линии более заметны, особенно в хвостах, из-за меньшего числа наблюдений и большей чувствительности к случайному шуму.

2. Графики квантилей для выборок большого объема (data3, data4, data5) демонстрируют почти идеальное соответствие теоретическому нормальному распределению, так как больший объем данных сглаживает влияние шума и случайных отклонений.

3. Графики Q-Q подтверждают нормальный характер всех распределений, с минимальными отклонениями в зависимости от объема и параметров выборок.

4. Результаты программы на Python и R для расчета квантилей не имеют различий.

6.3 Анализ с помощью графиков метода огибающих

Метод огибающих (envelope method) — это статистический метод, используемый для оценки и визуализации соответствия эмпирических данных теоретической модели. Метод огибающих создает области или доверительные интервалы (огибающие) вокруг ожидаемой теоретической зависимости, чтобы учитывать вариации данных. Эти области помогают определить, насколько данные соответствуют теоретической модели, либо служат для анализа ключевых направлений в данных.

На языке Python этот метод я реализовывал самостоятельно.

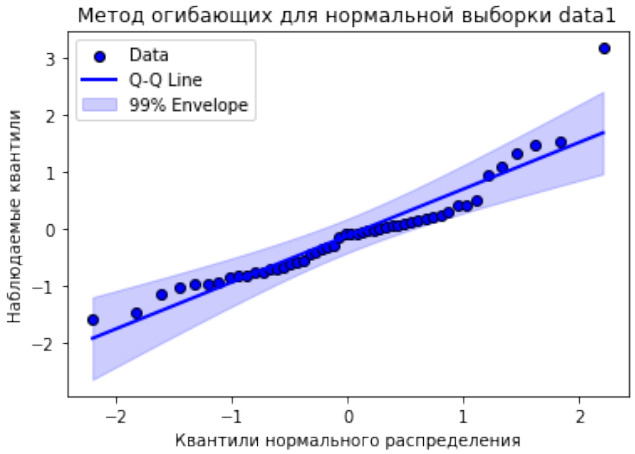

Рис. 27: Метод огибающих для выборки data1

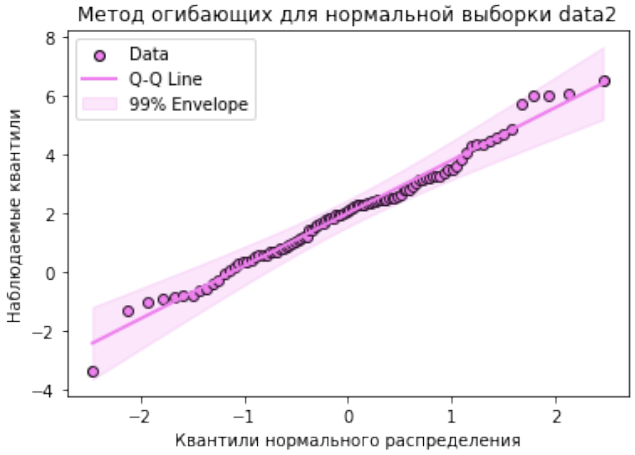

Рис. 28: Метод огибающих для выборки data2

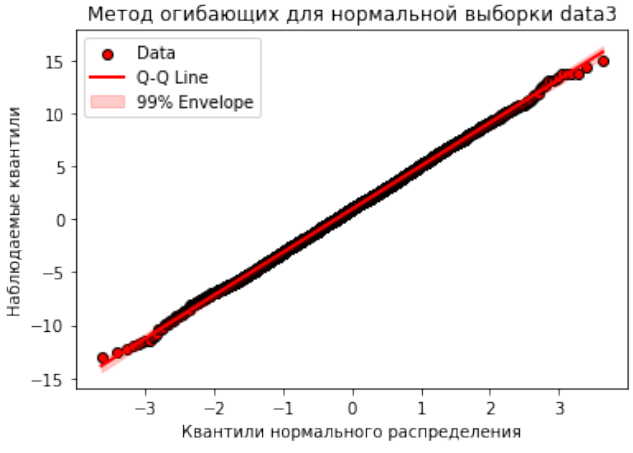

Рис. 29: Метод огибающих для выборки data3

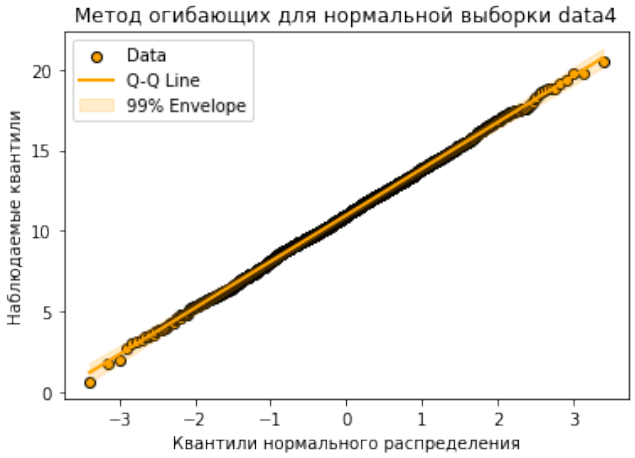

Рис. 30: Метод огибающих для выборки data4

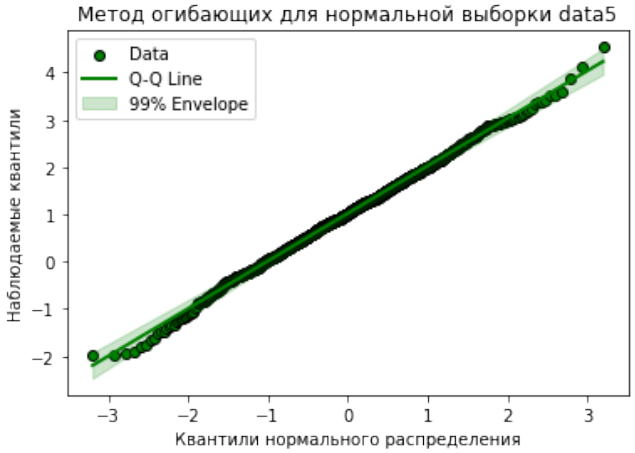

Рис. 31: Метод огибающих для выборки data5

Общие выводы

1. Выборки малого объема (data1 и data2) показывают значительные отклонения от огибающей, что объясняется случайностью данных и недостаточным количеством наблюдений.

2. Выборки большого объема (data3, data4, data5) демонстрируют гораздо лучшее соответствие данным с огибающей, что подтверждает более точное моделирование и уменьшение случайных отклонений, позволяя делать более надежные выводы.

3. С увеличением объема выборки точность метода огибающих возрастает, поэтому результаты становятся более стабильными.

4. Результаты, полученные с использованием метода огибающих, были сходными при применении как языка Python, так и языка R. Оба инструмента показали аналогичные выводы по всем выборкам. Это говорит о надежности полученных результатов в разных программных средах.

6.4 Стандартные процедуры проверки гипотез о нормальности

6.4.1 Критерий Колмогорова-Смирнова

Классический критерий Колмогорова (иногда говорят Колмогорова-Смирнова) предназначен для проверки простых гипотез о принадлежности анализируемой выборки некоторому полностью известному закону распределения.

Пусть X_n — выборка независимых одинаково распределённых случайных величин, $F_n(x)$ — эмпирическая функция распределения, $F(x)$ — некоторая «истинная» функция распределения с известными параметрами. Статистика критерия определяется выражением:

$$D_n = \sup_x |F_n(x) - F(x)| .$$

Обозначим через H_0 гипотезу о том, что выборка подчиняется распределению $F(x) \in \mathrm{C}^1(X)$. Тогда по теореме Колмогорова при справедливости проверяемой гипотезы:

$$\forall t > 0 : \quad \lim_{n\to\infty} P\left(\sqrt{n} D_n \le t\right) = K(t) = \sum_{j=-\infty}^{+\infty} (-1)^j \exp(-2j^2 t^2).$$

Гипотеза H_0 отвергается, если статистика $\sqrt{n}D_n$ превышает квантиль распределения K_α заданного уровня значимости α, и принимается в противном случае.

Критерий Колмогорова-Смирнова можно провести с помощью функции `kstest()` из библиотеки `scipy.stats` на языке Python. На языке R можно воспользоваться функцией `ks.test()` из библиотеки `stats`.

Результаты на языке Python:

1. data1: `KstestResult(statistic=0.19616, pvalue=0.03678)`
2. data2: `KstestResult(statistic=0.05756, pvalue=0.87568)`
3. data3: `KstestResult(statistic=0.01494, pvalue=0.21194)`
4. data4: `KstestResult(statistic=0.01950, pvalue=0.42698)`
5. data5: `data5: KstestResult(statistic=0.03734, pvalue=0.11994)`
6. data1-data3: `KstestResult(statistic=0.42380, pvalue=1.41823e-08)`
7. data4-data5: `KstestResult(statistic=0.99150, pvalue=0.00000)`

Результаты на языке R:

1. data1: статистика D = 0.14864, p-value = 0.1984
2. data2: статистика D = 0.07375, p-value = 0.6483
3. data3: статистика D = 0.00923, p-value = 0.7879
4. data4: статистика D = 0.02668, p-value = 0.1159
5. data5: статистика D = 0.02858, p-value = 0.3877
6. data1-data3: статистика D = 0.40320, p-value = 2.045e-07
7. data4-data5: статистика D = 0.98850, p-value $<$ 2.2e-16

Общие выводы

1. В R для выборки data1 p-value значительно больше 0.05, и гипотеза о нормальности не отвергается, в то время как в Python результат был противоположным, что может говорить о различиях в реализации теста в этих языках, однако для остальных выборок значения похожи. Важно отметить, что выборки были сгенерированы случайно, поэтому расхождения между p-value в рамках одной выборки на разных языках корректны: главное, чтобы не было ситуаций нахождения этих p-value по разные стороны от уровня значимости 0.05.

2. p-value > 0.05 верно для всех выборок, кроме первой, о чем было сказано выше, поэтому гипотеза о нормальности не отвергается.

3. В последних двух критериях p-value < 0.05, поэтому выборки принадлежат разным распределениям.

6.4.2 Критерий Шапиро-Уилка

Критерий Шапиро-Уилка используется для проверки гипотезы H_0: «случайная величина X распределена нормально» и является одним из наиболее эффективных критериев проверки нормальности. Критерии, проверяющие нормальность выборки, являются частным случаем критериев согласия.

Критерий Шапиро-Уилка основан на оптимальной линейной несмещённой оценке дисперсии к её обычной оценке методом максимального правдоподобия. Статистика критерия имеет вид:

$$W = \frac{1}{s^2}\left[\sum_{i=1}^{n} a_{n-i+1}\left(x_{n-i+1} - x_i\right)\right]^2,$$

где

$$s^2 = \sum_{i=1}^{n}\left(x_i - \overline{x}\right)^2, \quad \overline{x} = \frac{1}{n}\sum_{i=1}^{n} x_i.$$

Числитель является квадратом оценки среднеквадратического отклонения Ллойда. Коэффициенты a_{n-i+1} берутся из специальных таблиц.

Критерий Шапиро-Уилка можно провести с помощью функции `shapiro()` из библиотеки `scipy.stats` на языке Python. На языке R можно воспользоваться функцией `shapiro.test()` из библиотеки `stats`.

Результаты критерия на языке Python:

1. data1: `ShapiroResult(statistic = 0.98333, p-value = 0.69817)`
2. data2: `ShapiroResult(statistic = 0.97694, p-value = 0.07659)`
3. data3: `ShapiroResult(statistic = 0.99970, p-value = 0.69642)`
4. data4: `ShapiroResult(statistic = 0.99901, p-value = 0.33996)`
5. data5: `ShapiroResult(statistic = 0.99845, p-value = 0.52223)`

Результаты критерия на языке R:

1. data1: статистика W = 0.97719, p-value = 0.44050
2. data2: статистика W = 0.98332, p-value = 0.23880
3. data3: статистика W = 0.99963, p-value = 0.50280
4. data4: статистика W = 0.99904, p-value = 0.37390
5. data5: статистика W = 0.99829, p-value = 0.42510

Общие выводы

1. Для всех выборок p-value > 0.05, поэтому гипотеза о нормальности не отвергается.
2. Результаты критериев на Python и R идентичны.

6.4.3 Критерий Андерсона-Дарлинга

Классический непараметрический **критерий согласия Андерсона — Дарлинга** предназначен для проверки простых гипотез о принадлежности анализируемой выборки полностью известному закону (о согласии эмпирического распределения $F_n(x)$ и теоретического закона $F(x, \theta)$), то есть для проверки гипотез вида:

$$H_0 : F_n(x) = F(x, \theta)$$

с известным вектором параметров теоретического закона.
В критерии Ω^2 Андерсона — Дарлинга используется статистика вида:

$$S_\Omega = -n - 2\sum_{i=1}^{n}\left\{\frac{2i-1}{2n}\ln(F(x_i,\theta)) + \left(1 - \frac{2i-1}{2n}\right)\ln(1 - F(x_i,\theta))\right\},$$

где n — объём выборки, $x_1, x_2, \ldots, x_n$ — упорядоченные по возрастанию элементы выборки.
Критерий Андерсона-Дарлинга можно провести с помощью функции `anderson()` из библиотеки `scipy.stats` на языке Python. На языке R можно воспользоваться функцией `ad.test()` из библиотеки `nortest`.
Результаты критерия на языке Python:

1. data1: `AndersonResult(statistic=1.08662, p-value>0.15)`
2. data2: `AndersonResult(statistic=0.41841, p-value>0.15)`
3. data3: `AndersonResult(statistic=0.32054, p-value>0.15)`
4. data4: `AndersonResult(statistic=0.30366, p-value>0.15)`
5. data5: `AndersonResult(statistic=0.25660, p-value>0.15)`

Результаты критерия на языке R:

1. data1: статистика A = 0.31449, p-value = 0.53380
2. data2: статистика A = 0.39074, p-value = 0.37490
3. data3: статистика A = 0.44152, p-value = 0.28910
4. data4: статистика A = 0.36998, p-value = 0.42530
5. data5: статистика A = 0.22525, p-value = 0.82030

Общие выводы

1. Для всех выборок p-value > 0.05, поэтому гипотеза о нормальности не отвергается.
2. Результаты критериев на Python и R не противоречат друг другу.

6.4.4 Критерий Крамера фон Мизеса

Критерий Крамера-фон-Мизеса используется для проверки гипотезы «случайная величина X имеет распределение $F(x)$». Пусть $x_1, \ldots, x_n$ — элементы выборки, упорядоченные по возрастанию. Статистика критерия имеет вид

$$n\omega^2 = \frac{1}{12n} + \sum_{i=1}^{n} \left\{ F(x_i) - \frac{2i-1}{2n} \right\}^2,$$

где $F(x)$ — теоретическая функция распределения с известными параметрами. То есть, проверяется простая гипотеза.

При объёме выборки $n > 40$ можно пользоваться квантилями распределения $n\omega^2$, приведенными в следующей таблице:

α	0.900	0.950	0.990	0.995	0.999
$n\omega^2(\alpha)$	0.3473	0.4614	0.7435	0.8694	1.1679

Критерий Крамера фон Мизеса можно провести с помощью функции `cramervonmises()` из библиотеки `scipy.stats` на языке Python. На языке R можно воспользоваться функцией `cvm.test()` из библиотеки `goftest`.

Результаты критерия на языке Python:

1. data1: `CramerVonMisesResult(statistic=0.03527, p-value=0.95731)`
2. data2: `CramerVonMisesResult(statistic=0.29108, p-value=0.14320)`
3. data3: `CramerVonMisesResult(statistic=0.07334, p-value=0.73124)`
4. data4: `CramerVonMisesResult(statistic=0.09562, p-value=0.60681)`
5. data5: `CramerVonMisesResult(statistic=0.04287, p-value=0.91792)`

Результаты критерия на языке R:

1. data1: статистика $\omega^2 = 0.13752$, p-value $= 0.4311$
2. data2: статистика $\omega^2 = 0.11428$, p-value $= 0.5201$
3. data3: статистика $\omega^2 = 0.21609$, p-value $= 0.2384$
4. data4: статистика $\omega^2 = 0.07895$, p-value $= 0.6980$
5. data5: статистика $\omega^2 = 0.09595$, p-value $= 0.6052$

Общие выводы

1. Все p-value > 0.05, что указывает на то, что гипотеза о нормальности распределения данных не отклоняется для всех выборок.
2. Результаты критериев на Python и R не противоречат друг другу.

6.4.5 Критерий Колмогорова-Смирнова в модификации Лиллиефорса

Критерий Лиллиефорса является модификацией критерия Колмогорова–Смирнова и используется для проверки нулевой гипотезы о том, что выборка распределена по нормальному закону для случая, когда параметры нормального распределения (математическое ожидание и дисперсия) априори неизвестны.

Критерий Колмогорова-Смирнова в модификации Лиллиефорса можно провести с помощью функции `lilliefors()` из библиотеки `statsmodels.stats.diagnostic` на языке Python. На языке R можно воспользоваться функцией `lillie.test()` из библиотеки `nortest`.

Результаты критерия на языке Python:

1. data1: статистика D = 0.08854, p-value = 0.42298
2. data2: статистика D = 0.06219, p-value = 0.45089
3. data3: статистика D = 0.00562, p-value = 0.96538
4. data4: статистика D = 0.01596, p-value = 0.26228
5. data5: статистика D = 0.02839, p-value = 0.07196

Результаты критерия на языке R:

1. data1: статистика D = 0.06610, p-value = 0.84580
2. data2: статистика D = 0.06448, p-value = 0.38810
3. data3: статистика D = 0.00877, p-value = 0.46310
4. data4: статистика D = 0.01469, p-value = 0.37020
5. data5: статистика D = 0.02311, p-value = 0.21910

Общие выводы

1. Все p-value > 0.05, что указывает на то, что гипотеза о нормальности распределения данных не отклоняется для всех выборок.
2. Результаты критериев на Python и R не противоречат друг другу.

6.4.6 Критерий Шапиро-Франсия

Критерий Шапиро-Франсия используется для проверки гипотезы H_0: «случайная величина X распределена нормально». Введен в 1972 году как упрощение теста Шапиро-Уилка.

Пусть $x_{(i)}$ — это i-й порядковый элемент из выборки объёма n. Пусть $m_{i:n}$ — это среднее значение i-го порядкового статистического значения при совершении n независимых выборок из нормального распределения.

Теперь формируем коэффициент корреляции Пирсона между выборочными значениями x и теоретическими значениями m:

$$W' = \frac{cov(x,m)}{\sigma_x \sigma_m} = \frac{\sum_{i=1}^{n}(x_{(i)} - \overline{x})(m_i - \overline{m})}{\sqrt{\left(\sum_{i=1}^{n}(x_{(i)} - \overline{x})^2\right)\left(\sum_{i=1}^{n}(m_i - \overline{m})^2\right)}}$$

При нулевой гипотезе, что данные взяты из нормального распределения, эта корреляция будет сильной, и значения W' будут сгруппированы около 1, при этом пик будет становиться уже и приближаться к 1 с увеличением n. Если данные сильно отклоняются от нормального распределения, то W' будет меньше.

Этот тест является формализацией более старой практики построения графика Q-Q для сравнения двух распределений, где x выполняет роль квантилей эмпирического распределения, а m — роль соответствующих квантилей нормального распределения.

В отличие от статистики теста Шапиро-Уилка W, статистика теста Шапиро-Франсия W' легче вычисляется, поскольку не требует формирования и инверсии матрицы ковариаций между порядковыми статистиками.

Критерий Шапиро-Франсия можно провести с помощью функции `shapiroFrancia()` из библиотеки `sfrancia` на языке Python. На языке R можно воспользоваться функцией `sf.test()` из библиотеки `nortest`.

Результаты критерия на языке Python:

1. data1: статистика W = 0.98280, p-value = 0.58211
2. data2: статистика W = 0.98078, p-value = 0.13454
3. data3: статистика W = 0.99972, p-value = 0.68161
4. data4: статистика W = 0.99956, p-value = 0.91883
5. data5: статистика W = 0.99910, p-value = 0.87633

Результаты критерия на языке R:

1. data1: статистика W = 0.98266, p-value = 0.57620
2. data2: статистика W = 0.98907, p-value = 0.50460
3. data3: статистика W = 0.99976, p-value = 0.82380
4. data4: статистика W = 0.99944, p-value = 0.79680
5. data5: статистика W = 0.99858, p-value = 0.54800

Общие выводы

1. Все p-value > 0.05, что указывает на то, что гипотеза о нормальности распределения данных не отклоняется для всех выборок.
2. Результаты критериев на Python и R не противоречат друг другу.

7 Продемонстрировать пример анализа данных с помощью графиков квантилей, метода огибающих, а также стандартных процедур проверки гипотез о нормальности. Рассмотреть выборки малого и умеренного объемов

Начнем с выборки малого объема. Для этого я воспользуюсь новым датасетом, содержащем информацию о 77 видах хлопьев, в котором я возьму статистику калорийности хлопьев.

Для анализа выборки умеренного объема я возьму данные из датасета sleep — LightExposureHours.

7.1 Анализ данных с помощью графиков квантилей

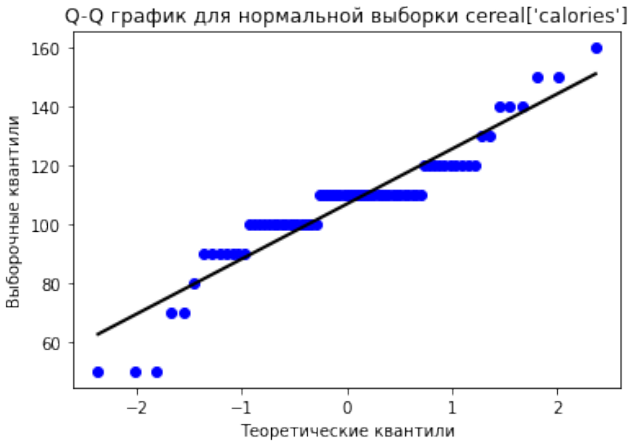

Рис. 32: QQ-plot для выборки calories

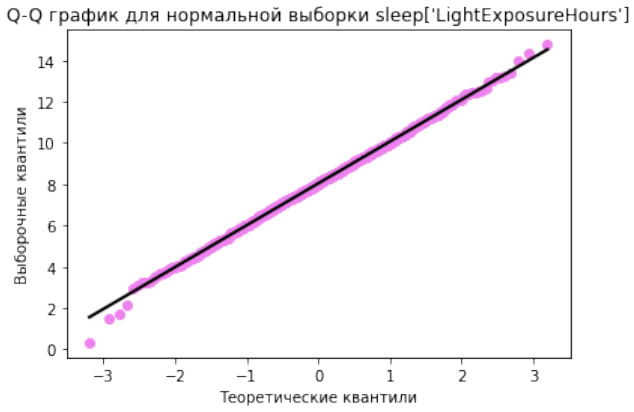

Рис. 33: QQ-plot для выборки LightExposureHours

Общие выводы

1. QQ-plot на малой выборке показывает сильные отклонения от прямой линии, что указывает на то, что данные не являются нормальными. На большой выборке результат противоположный: выборка является нормальной.

2. Результаты анализа данных с использованием графиков квантилей в Python и R показывают одинаковые выводы.

7.2 Анализ с помощью метода огибающих

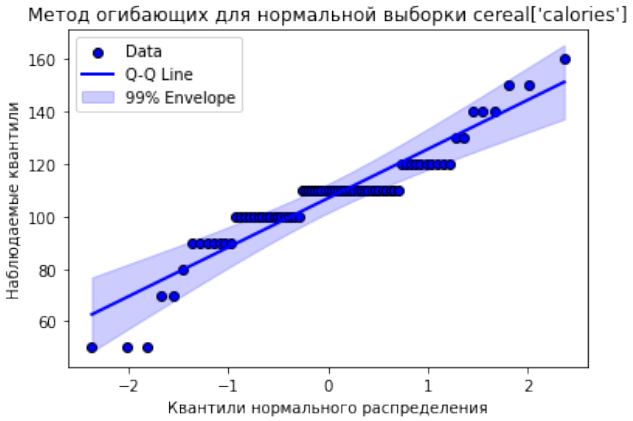

Рис. 34: Метод огибающих для выборки calories

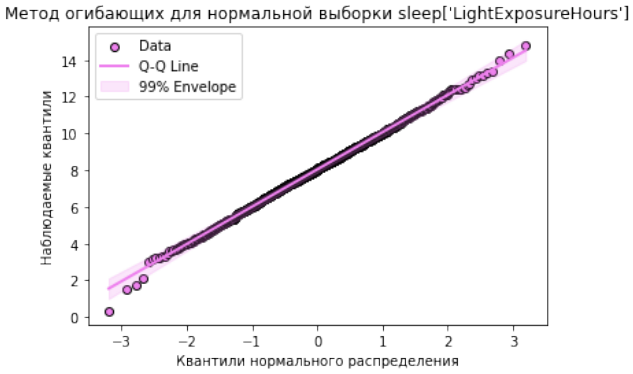

Рис. 35: Метод огибающих для выборки LightExposureHours

Общие выводы

1. Выборка малого объема показывает значительные отклонения от огибающей, что доказывает отсутствие нормальности данных.

2. Выборка большого объема демонстрируют гораздо лучшее со- ответствие данным с огибающей, что подтверждает более точное моделирование и уменьшение случайных отклонений, позволяя делать более надежные выводы.

3. Результаты, полученные с использованием метода огибающих, были сходными при применении как языка Python, так и языка R.

7.3 Стандартные процедуры проверки гипотез о нормальности

7.3.1 Критерий Колмогорова-Смирнова

Результаты на языке Python:

- Малый объем: `KstestResult(statistic=0.20227, pvalue`=0.00308)
- Большой объем: `KstestResult(statistic=0.0128, pvalue=0.99606)`

Результаты на языке R:

- Малый объем: статистика D = 0.20227, p-value=0.00308
- Большой объем: статистика D = 0.0128, p-value=0.99606

7.3.2 Критерий Шапиро-Уилка

Результаты на языке Python:

- Малый объем: `ShapiroResult(statistic=0.95346, pvalue=0.00674)`
- Большой объем: `ShapiroResult(statistic=0.99920, pvalue=0.95743)`

Результаты на языке R:

- Малый объем: статистика W = 0.89398, p-value = 9.73e-06
- Большой объем: статистика W = 0.99921, p-value = 0.9574

7.3.3 Критерий Андерсона-Дарлинга

Результаты на языке Python:

- Малый объем: `AndersonResult(statistic=3.39836, p-value<0.01)`
- Большой объем: `AndersonResult(statistic=0.14420, p-value>0.15)`

Результаты на языке R:

- Малый объем: статистика A = 3.3984, p-value = 1.399e-08
- Большой объем: статистика A = 0.14421, p-value = 0.9698

7.3.4 Критерий Крамера фон Мизеса

Результаты на языке Python:

- Малый объем: `CramerVonMisesResult(statistic=0.65269, pvalue=0.01636)`
- Большой объем: `CramerVonMisesResult(statistic=0.02175, pvalue=0.99505)`

Результаты на языке R:

- Малый объем: статистика W = 0.65269, p-value=0.01636
- Большой объем: статистика W = 0.02175, p-value=0.99505

7.3.5 Критерий Колмогорова-Смирнова в модификации Лиллиефорса и Шапиро-Франсия

Результаты на языке Python:

- Малый объем: статистика D = 0.20268, p-value = 0.00099
- Большой объем: статистика D = 0.012871, p-value = 0.97406

Результаты на языке R:

- Малый объем: статистика D = 0.20269, p-value = 2.39e-08
- Большой объем: статистика D = 0.012872, p-value = 0.9544

7.3.6 Критерий Шапиро-Франсия

Результаты на языке Python:

- Малый объем: статистика W = 0.88825, p-value = 2.1364e-05
- Большой объем: статистика W = 0.99897, p-value = 0.80836

Результаты на языке R:

- Малый объем: статистика W = 0.88825, p-value = 2.136e-05
- Большой объем: статистика W = 0.99898, p-value = 0.8084

Общие выводы по всем критериям

1. Данные о калориях (малая выборка) не распределены нормально, поскольку каждый тест выдал p-value < 0.05. Данные о часах воздействия света в течение дня (большая выборка) распределены нормально, поскольку каждый критерий показал p-value > 0.05.
2. Результаты критериев на Python и R идентичны.

8 Продемонстрировать применение для проверки различных гипотез и различных доверительных уровней (0.9, 0.95, 0.99) некоторых критериев

Для этой части задания я подобрал новый датасет: зависимость уровня счастья в странах мира за 2021 год от различных показателей (ВВП, уровень социальной поддержки, регион, продолжительность жизни и пр.).

Критерий Стьюдента можно провести с помощью функции `ttest()` из библиотеки `pingouin` на языке Python. Данная функция предоставляет возможность использования как для одновыборочных, так и для двухвыборочных критериев; как для односторонних, так и для двухсторонних. На языке R критерий Стьюдента проводится с помощью функции `t.test()` из библиотеки `stats`.

8.1 Одновыборочный критерий Стьюдента

Пусть дана выборка $x^n = (x_1, \ldots, x_n),\ x_i \in R$;

Обозначим через μ и σ^2 математическое ожидание и дисперсию выборки соответственно.

Дополнительное предположение: выборка x^n является нормальной.

Нулевая гипотеза $H_0 : \mu = \mu_0$, где μ_0 - некоторое заданное число.

Возможные альтернативные гипотезы:

1. $H_1 : \mu \neq \mu_0$ (two-sided)
2. $H_1' : \mu > \mu_0$ (greater)
3. $H_1'' : \mu < \mu_0$ (less)

Будем проводить одновыборочный критерий Стьюдента для выборки happiness_score, которая является нормальной по результатам теста Шапиро-Уилка, который выдал значение `p-value`: 0.4893410649623995, что больше пороговой величины 0.05.

Проверим, равен ли уровень счастья величине 5.9 на разных уровнях доверия: $\alpha = 0.90, \alpha = 0.95, \alpha = 0.99$.

8.1.1 Двусторонний критерий Стьюдента

Программы на Python и R выдали следующие результаты:

Таблица 2: Двусторонний тест Стьюдента при $\alpha = 0.9$

T	**dof**	**alternative**	**p-val**	**CI90%**	**power**
-4.173269	148	two-sided	0.000051	[5.3872, 5.6784]	0.985595

Таблица 3: Двусторонний тест Стьюдента при $\alpha = 0.95$

T	**dof**	**alternative**	**p-val**	**CI95%**	**power**
-4.173269	148	two-sided	0.000051	[5.36, 5.71]	0.985595

Таблица 4: Двусторонний тест Стьюдента при $\alpha = 0.99$

T	**dof**	**alternative**	**p-val**	**CI99%**	**power**
-4.173269	148	two-sided	0.000051	[5.3033, 5.7624]	0.985595

8.1.2 Односторонние критерии Стьюдента. Greater.

Программы на Python и R выдали следующие результаты:

Таблица 5: Односторонний тест Стьюдента (greater) при $\alpha = 0.9$

T	**dof**	**alternative**	**p-val**	**CI90%**	**cohen-d**	**power**
-4.173269	148	greater	0.999974	[5.4195, ∞]	0.341888	3.33×10^{-9}

Таблица 6: Односторонний тест Стьюдента (greater) при $\alpha = 0.95$

T	**dof**	**alternative**	**p-val**	**CI95%**	**cohen-d**	**power**
-4.173269	148	greater	0.999974	[5.39, ∞]	0.341888	3.33×10^{-9}

Таблица 7: Односторонний тест Стьюдента (greater) при $\alpha = 0.99$

T	**dof**	**alternative**	**p-val**	**CI99%**	**cohen-d**	**power**
-4.173269	148	greater	0.999974	[5.3259, ∞]	0.341888	3.33×10^{-9}

8.1.3 Односторонние критерии Стьюдента. Less.

Программы на Python и R выдали следующие результаты:

Таблица 8: Односторонний тест Стьюдента (less) при $\alpha = 0.9$

T	**dof**	**alternative**	**p-val**	**CI90%**	**cohen-d**	**power**
-4.173269	148	less	0.000026	[$-\infty$, 5.6460]	0.341888	0.99395

Таблица 9: Односторонний тест Стьюдента (less) при $\alpha = 0.95$

T	**dof**	**alternative**	**p-val**	**CI95%**	**cohen-d**	**power**
-4.173269	148	less	0.000026	[$-\infty$, 5.68]	0.341888	0.99395

Таблица 10: Односторонний тест Стьюдента (less) при $\alpha = 0.99$

T	**dof**	**alternative**	**p-val**	**CI99%**	**cohen-d**	**power**
-4.173269	148	less	0.000026	[$-\infty$, 5.5737]	0.341888	0.99395

Исходя из результатов всех критериев, можно сделать вывод о том, что среднее значение выборки happiness_score < 5.9. Это действительно так, поскольку оно составляет 5.53.

8.1.4 Определение объема выборки для достижения заданной мощности

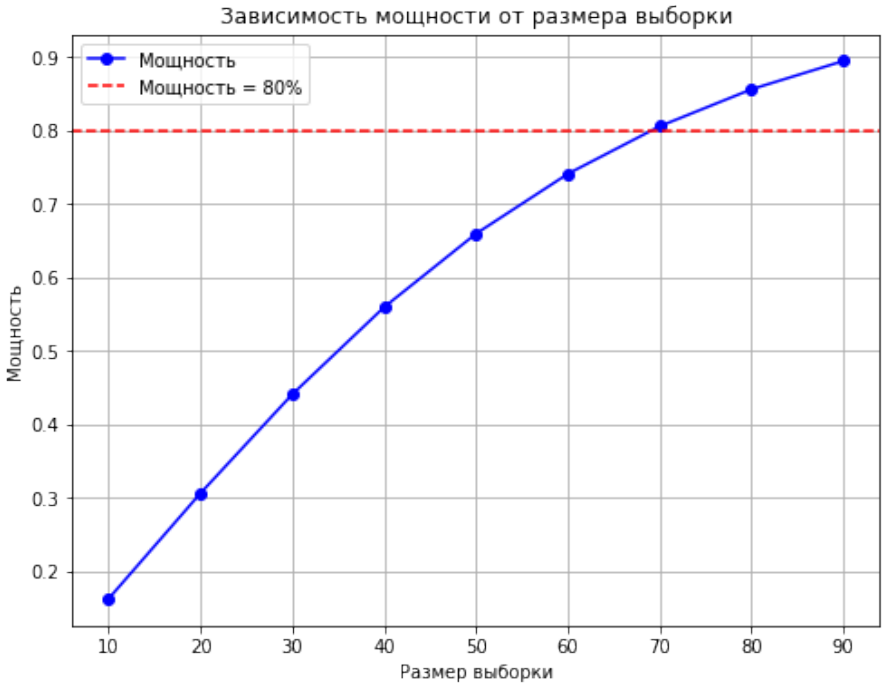

Рис. 36: Определение объема выборки для достижения заданной мощности 0.8 на Python

Видим, что для достижения мощности 0.8 при уровне значимости 0.05 размер выборки должен быть ≈ 70.

8.2 Двухвыборочный критерий Стьюдента

Пусть даны две независимые выборки $x^n = (x_1, \ldots, x_n),\ x_i \in R; \quad y^m = (y_1, \ldots, y_m),\ y_i \in R$.

Обозначим через μ_1 и μ_2 математические ожидания выборок x^n и y^m.

Дополнительное предположение: выборки x^n и y^m являются нормальными.

Нулевая гипотеза $H_0 : \mu_1 = \mu_2$

Альтернативные гипотезы:

- $H_1 : \mu_1 \neq \mu_2$ (two-sided)
- $H_1' : \mu_1 > \mu_2$ (greater)
- $H_1'' : \mu_1 < \mu_2$ (less)

Будем сравнивать математические ожидания выборок, отвечающих за уровень счастья в странах Европы и Азии, которые соответственно равны 6.25116 и 5.21013.

8.2.1 Двусторонний критерий Стьюдента

Программы на Python и R выдали следующие результаты:

Таблица 11: Двусторонний критерий Стьюдента при $\alpha = 0.9$

T	**dof**	**alternative**	**p-val**	**CI90%**	**power**
4.707294	37.299201	two-sided	0.000034	[0.6679, 1.4140]	0.997736

Таблица 12: Двусторонний критерий Стьюдента при $\alpha = 0.95$

T	**dof**	**alternative**	**p-val**	**CI95%**	**power**
4.707294	37.299201	two-sided	0.000034	[0.59, 1.49]	0.997736

Таблица 13: Двусторонний критерий Стьюдента при $\alpha = 0.99$

T	**dof**	**alternative**	**p-val**	**CI99%**	**power**
4.707294	37.299201	two-sided	0.000034	[0.4408, 1.6413]	0.997736

8.2.2 Односторонний критерий Стьюдента. Greater

Программы на Python и R выдали следующие результаты:

Таблица 14: Односторонний тест Стьюдента (greater) при $\alpha = 0.9$

T	**dof**	**alternative**	**p-val**	**CI90%**	**power**
4.707294	37.299201	greater	0.000017	[0.7525, ∞]	0.999248

Таблица 15: Односторонний тест Стьюдента (greater) при $\alpha = 0.95$

T	**dof**	**alternative**	**p-val**	**CI95%**	**power**
4.707294	37.299201	greater	0.000017	[0.67, ∞]	0.999248

Таблица 16: Односторонний тест Стьюдента (greater) при $\alpha = 0.99$

T	**dof**	**alternative**	**p-val**	**CI99%**	**power**
4.707294	37.299201	greater	0.000017	[0.5035, ∞]	0.999248

8.2.3 Односторонний критерий Стьюдента. Less

Программы на Python и R выдали следующие результаты:

Таблица 17: Односторонний тест Стьюдента (less) при $\alpha = 0.9$

T	**dof**	**alternative**	**p-val**	**CI90%**	**power**
4.707294	37.299201	less	0.999983	[$-\infty$, 1.3296]	5.048084e-11

Таблица 18: Односторонний тест Стьюдента (less) при $\alpha = 0.95$

T	dof	alternative	p-val	CI95%	power
4.707294	37.299201	less	0.999983	$[-\infty, 1.41]$	5.048084e-11

Таблица 19: Односторонний тест Стьюдента (less) при $\alpha = 0.99$

T	dof	alternative	p-val	CI99%	power
4.707294	37.299201	less	0.999983	$[-\infty, 1.5785]$	5.048084e-11

Исходя из результатов всех критериев, можно сделать вывод о том, что среднее значение выборки happiness_score по Европе выше, чем по Азии.

Выводы

Критерий Стьюдента используется для сравнения средних значений двух групп, чтобы определить, являются ли различия между ними статистически значимыми. Как мы выяснили, существует два вида тестов:

- Двусторонний тест: проверяет, отличаются ли средние значения групп в любом направлении.
- Односторонний тест: проверяет, превышает ли среднее значение одной группы среднее значение другой (или наоборот).

Он эффективен для анализа данных, где выборки соответствуют нормальному распределению, и используется в исследованиях для проверки гипотез.

В данном анализе программы на R и Python выдали похожие результаты, подтверждая надежность вычислений.

8.3 Ранговый критерий Уилкоксона-Манна-Уитни

Пусть даны две независимые выборки $x^n = (x_1, \ldots, x_n),\ x_i \in R; \quad y^m = (y_1, \ldots, y_m),\ y_i \in R$.

Обозначим через μ_1 и μ_2 математические ожидания выборок x^n и y^m.

Нулевая гипотеза $H_0 : \mu_1 = \mu_2$

Альтернативные гипотезы:

1. $H_1 : \mu_1 \neq \mu_2$ (two-sided)
2. $H_1' : \mu_1 > \mu_2$ (greater)
3. $H_1'' : \mu_1 < \mu_2$ (less)

Критерий Уилкоксона-Манна-Уитни можно провести с помощью функции `mannwhitneyu()` из библиотеки `scipy.stats` на языке Python, на R — с помощью функции `wilcox.test()` из библиотеки `stats`.

Сравним при помощи критерия социальную поддержку в странах Европы (Western Europe, Central and Eastern Europe) и в странах Аффрики (Middle East and North Africa, Sub-Saharan Africa).

- Среднее для Европы: 0.90236
- Среднее для Африки: 0.72911

Таблица 20: Критерий Уилкоксона-Манна-Уитни на Python и R

Гипотеза	**Python p-value**	**R p-value**
$EX = EY$ (two-sided)	2.7501435122783994e-12	2.750144e-12
$EX > EY$ (greater)	0.9999999999987016	1
$EX < EY$ (less)	1.3750717561391997e-12	1.375072e-12

На обоих языках программирования критерий показал, что среднее значение социальной поддержки в странах Европы выше, чем в странах Африки.

8.4 Проверка гипотез об однородности дисперсий.

8.4.1 Критерий Фишера

Пусть даны две выборки $x^n = (x_1, \ldots, x_n),\ x_i \in R; \quad y^m = (y_1, \ldots, y_m),\ y_i \in R.$
Обозначим через σ_1^2 и σ_2^2 дисперсии выборок x^n и y^m.
Дополнительное предположение: выборки x^n и y^m являются нормальными.
Нулевая гипотеза $H_0 : \sigma_1^2 = \sigma_2^2$
Альтернативные гипотезы:

1. $H_1 : \sigma_1^2 \neq \sigma_2^2$ (two-sided)
2. $H_1' : \sigma_1^2 > \sigma_2^2$ (greater)
3. $H_1'' : \sigma_1^2 < \sigma_2^2$ (less)

Статистика критерия Фишера: $F = \frac{s_1^2}{s_2^2}$, где s_1^2, s_2^2 - выборочные оценки дисперсий. Статистика F имеет распределение Фишера с $n-1$ и $m-1$ степенями свободы. Обычно в числителе ставится большая из двух сравниваемых дисперсий.

Критерий Фишера на языке Python был реализован мною самостоятельно. В своей программе на R я воспользовался функцией `var.test()` из библиотеки `stats`.

Применим критерий Фишера для выборок LightExposureHours и MovementDuringSleep, у которых дисперсии составляют 4.09402 и 0.96718 соответственно. Выборки являются нормальными, поскольку тест Шапиро-Уилка выдал следующие p-value (0.9574 и 0.7173), что выше порогового значения 0.05.

Выведем p-value для всех возможных альтернатив на Python:

1. two-sided ($DX \neq DY$): 2.08941e-106
2. greater ($DX > DY$): 1.0
3. less ($DX < DY$): 1.04470e-106

Исходя из результатов можем сделать вывод о том, что дисперсия LightExposureHours выше, чем дисперсия MovementDuringSleep. Программа на R показала такой же результат.

8.4.2 Критерий Левене

Пусть даны две выборки $x^n = (x_1, \ldots, x_n),\ x_i \in R; \quad y^m = (y_1, \ldots, y_m),\ y_i \in R.$

Обозначим через σ_1^2 и σ_2^2 дисперсии выборок x^n и y^m.

Нулевая гипотеза $H_0 : \sigma_1^2 = \sigma_2^2$

Альтернативная гипотеза: $H_1 : \sigma_1^2 \neq \sigma_2^2$ (two-sided)

Критерий Левене на языке Python можно провести с помощью функции `levene()` из библиотеки `scipy.stats`. В своей программе на R я воспользовался функцией `leveneTest()` из библиотеки `car`.

Применим критерий Левене для сравнения дисперсии показателя ВВП в странах Восточной и Центральной и Западной Европы. Для этих групп дисперсии составляют 0.09305931 и 0.1569663 соответственно.

Сравним результаты применения критерия на Python и R:

- Python: `LeveneResult(statistic=1.48469, pvalue=0.23096)`
- R:

Листинг 2: Критерий Левене для ВВП разных частей Европы на языке R

```
Levene's Test for Homogeneity of Variance (center = "median")
      Df F value Pr(>F)
group  1  1.4847  0.231
      36
```

Исходя из результатов (p-value > 0.05) нет оснований отвергать гипотезу о равенстве дисперсий. Программа на R показала такой же результат.

8.4.3 Критерий Бартлетта

Пусть даны две выборки $x^n = (x_1, \ldots, x_n),\ x_i \in R; \quad y^m = (y_1, \ldots, y_m),\ y_i \in R.$

Обозначим через σ_1^2 и σ_2^2 дисперсии выборок x^n и y^m.

Дополнительное предположение: выборки x^n и y^m являются нормальными.

Нулевая гипотеза $H_0 : \sigma_1^2 = \sigma_2^2$

Альтернативная гипотеза: $H_1 : \sigma_1^2 \neq \sigma_2^2$ (two-sided)

Критерий Бартлетта на языке Python можно провести с помощью функции `bartlett()` из библиотеки `scipy.stats`. В своей программе на R я воспользовался функцией `bartlett.test()` из библиотеки `stats`.

Применим критерий Бартлетта для выборок LightExposureHours и MovementDuringSleep, которые уже использовались для критерия Фишера.

Сравним результаты проведения критерия на Python и R:

- Python: `BartlettResult(statistic=480.03510, pvalue=2.09944e-106)`
- R:

Листинг 3: Критерий Бартлетта на языке R

```
Bartlett test of homogeneity of variances

data:  list(sleep$MovementDuringSleep, sleep$LightExposureHours)
Bartlett's K-squared = 479.87, df = 1, p-value < 2.2e-16
```

Программы на обоих языках выдали значение p-value < 0.05, поэтому нулевая гипотеза отвергается. Дисперсии между группами значительно различаются.

8.4.4 Критерий Флигнера-Килина

Пусть даны две выборки $x^n = (x_1, \ldots, x_n),\ x_i \in R; \quad y^m = (y_1, \ldots, y_m),\ y_i \in R.$

Обозначим через σ_1^2 и σ_2^2 дисперсии выборок x^n и y^m.

Нулевая гипотеза $H_0 : \sigma_1^2 = \sigma_2^2$.

Альтернативная гипотеза: $H_1 : \sigma_1^2 \neq \sigma_2^2$ (two-sided).

Критерий Флигнера-Килина на языке Python можно провести с помощью функции `fligner()` из библиотеки `scipy.stats`. В своей программе на R я воспользовался функцией `fligner.test()` из библиотеки `stats`.

Применим критерий Флигнера-Килина для сравнения дисперсии показателя ВВП в странах Восточной и Центральной и Западной Европы. Данные выборки уже использовались для критерия Левене.

Сравним результаты проведения критерия на Python и R:

- Python: `FlignerResult(statistic=1.10303, pvalue=0.29360)`
- R:

Листинг 4: Критерий Флингера-Килина на языке R

```
Fligner-Killeen test of homogeneity of variances

data:  c(western, central_eastern) and factor
(c(rep("Western Europe", length(western)), rep("Central and
Eastern Europe", length(central_eastern))))
Fligner-Killeen:med chi-squared = 1.103, df = 1, p-value =
0.2936
```

Программы на обоих языках выдали значение p-value > 0.05, поэтому нет оснований отвергать гипотезу о равенстве дисперсий.

9 Исследовать корреляционные взаимосвязи в данных с помощью коэффициентов корреляции Пирсона, Спирмена и Кендалла.

На языке Python коэффициенты Пирсона, Спирмена и Кендалла можно вычислить, применяя соответственно функции `pearsonr`, `spearmanr`, `kendalltau` из библиотеки `scipy.stats`. На языке R используется одна функция `cor.test()` с различными значениями параметра `method`.

9.1 Коэффициент корреляции Пирсона

Пусть даны две выборки $x^m = (x_1, \ldots, x_m)$, $y^m = (y_1, \ldots, y_m)$; **коэффициент корреляции Пирсона** рассчитывается по формуле:

$$r_{xy} = \frac{\sum_{i=1}^{m}(x_i - \overline{x})(y_i - \overline{y})}{\sqrt{\sum_{i=1}^{m}(x_i - \overline{x})^2 \sum_{i=1}^{m}(y_i - \overline{y})^2}} = \frac{cov(x,y)}{\sqrt{s_x^2 s_y^2}},$$

где $\overline{x}, \overline{y}$ — выборочные средние, x^m и y^m, s_x^2, s_y^2 — выборочные дисперсии, $r_{xy} \in [-1, 1]$.

Коэффициент корреляции Пирсона называют также теснотой линейной связи:

- $|r_{xy}| = 1 \Rightarrow x, y$ линейно зависимы,
- $r_{xy} = 0 \Rightarrow x, y$ линейно независимы.

Для вычисления коэффициентов корреляции Пирсона необходимы две нормально распределенных выборки, которыми являются, например, выборки Light Exposure Hours и Movement During Sleep из датасета про качество сна.

Сравним выводы на языке Python и R:

- Python: `PearsonRResult(statistic=0.00173, pvalue=0.95620)`
- R:

Листинг 5: Коэффициент корреляции Пирсона на языке R

```
Pearson's product-moment correlation

data:  sleep$LightExposureHours and sleep$MovementDuringSleep
t = 0.055217, df = 998, p-value = 0.956
alternative hypothesis: true correlation is not equal to 0
95 percent confidence interval:
-0.06025175  0.06373404
sample estimates:
         cor
0.001747861
```

Результаты на Python и R идентичны. p-value > 0.05, это означает, что нет статистически значимой линейной зависимости между переменными.

9.2 Коэффициент корреляции Спирмена

Коэффициент корреляции Спирмена — мера линейной связи между случайными величинами. Корреляция Спирмена является ранговой, то есть для оценки силы связи используются не численные значения, а соответствующие им ранги. Коэффициент инвариантен по отношению к любому монотонному преобразованию шкалы измерения.

Заданы две выборки $x = (x_1, \ldots, x_n)$, $y = (y_1, \ldots, y_n)$.

Коэффициент корреляции Спирмена вычисляется по формуле:

$$\rho = 1 - \frac{6}{n(n-1)(n+1)} \sum_{i=1}^{n}(R_i - S_i)^2,$$

где R_i — ранг наблюдения x_i в ряду x, S_i — ранг наблюдения y_i в ряду y.

Коэффициент ρ принимает значения из отрезка $[-1; 1]$. Равенство $\rho = 1$ указывает на строгую прямую линейную зависимость, а $\rho = -1$ на обратную.

Вычислим коэффициент корреляции Спирмена между выборками gdp и happiness_score из датасета про счастье.

- Python: `SignificanceResult(statistic=0.80940, pvalue=8.40608e-36)`
- R:

Листинг 6: Коэффициент корреляции Спирмена на языке R

```
Spearman's rank correlation rho

data:  happiness$gdp and happiness$happiness_score
S = 105074, p-value < 2.2e-16
alternative hypothesis: true rho is not equal to 0
sample estimates:
      rho
0.8094072
```

Между переменными существует сильная и значимая положительная монотонная связь. Значение коэффициента Спирмена говорит о том, что с увеличением одной переменной другая также увеличивается, и эта связь достаточно надежна.

Вычислим коэффициент корреляции Спирмена между выборками gdp и corruption_perceptions из датасета про счастье.

- Python: `SignificanceResult(statistic=-0.27692, pvalue=0.00062)`
- R:

Листинг 7: Коэффициент корреляции Спирмена на языке R

```
Spearman's rank correlation rho

data:  happiness$gdp and happiness$corruption_perceptions
S = 703968, p-value = 0.0006287
alternative hypothesis: true rho is not equal to 0
sample estimates:
       rho
-0.2769237
```

Между двумя переменными существует слабая, но значимая отрицательная монотонная связь. Отрицательный коэффициент говорит о том, что по мере увеличения одной переменной другая имеет тенденцию к уменьшению (то есть чем выше уровень коррупции, тем в среднем ниже уровень счастья в стране).

В обоих случаях программы на языке Python и R показали одинаковые результаты.

9.3 Коэффициент корреляции Кендалла

Коэффициент корреляции Кенделла — мера линейной связи между случайными величинами. Корреляция Кенделла является ранговой, то есть для оценки силы связи используются не численные значения, а соответствующие им ранги. Коэффициент инвариантен по отношению к любому монотонному преобразованию шкалы измерения.

Заданы две выборки $x = (x_1, \ldots, x_n)$, $y = (y_1, \ldots, y_n)$.

Коэффициент корреляции Кенделла вычисляется по формуле:

$$\tau = 1 - \frac{4}{n(n-1)}R, \qquad R = \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \left[[x_i < x_j] \neq [y_i < y_j] \right],$$

где R — количество инверсий, образованных величинами y_i, расположенными в порядке возрастания соответствующих x_i.

Коэффициент τ принимает значения из отрезка $[-1; 1]$. Равенство $\tau = 1$ указывает на строгую прямую линейную зависимость, а $\tau = -1$ на обратную.

Вычислим коэффициент корреляции Кендалла между выборками gdp и happiness_score из датасета про счастье.

- Python: SignificanceResult(statistic=0.61653, pvalue=6.69076e-29)
- R:

Листинг 8: Коэффициент корреляции Кендалла на языке R

```
Kendall's rank correlation tau

data:  happiness$gdp and happiness$happiness_score
z = 11.156, p-value < 2.2e-16
alternative hypothesis: true tau is not equal to 0
sample estimates:
      tau
0.6165359
```

Между gdp и happiness_score существует сильная и значимая положительная монотонная связь.

Вычислим коэффициент корреляции Спирмена между выборками gdp и corruption_perceptions из датасета про счастье.

- Python: `SignificanceResult(statistic=-0.17747, pvalue=0.0013)`
- R:

Листинг 9: Коэффициент корреляции Спирмена на языке R

```
Kendall's rank correlation tau

data:  happiness$gdp and happiness$corruption_perceptions
z = -3.2089, p-value = 0.001333
alternative hypothesis: true tau is not equal to 0
sample estimates:
      tau
-0.1774773
```

Между gdp и happiness_score существует сильная и значимая отрицательная монотонная связь.

В обоих случаях программы на языке Python и R показали одинаковые результаты.

10 Продемонстрировать использование методов хи-квадрат, точного теста Фишера, теста МакНемара, Кохрана-Мантеля-Хензеля.

10.1 Метод хи-квадрат

Метод хи-квадрат — это статистический метод для проверки независимости двух категориальных переменных (первая принимает k значений, вторая l значений) в таблице сопряженности k×l.

Пусть A — категориальная выборка, принимающая n значений $a_1, a_2, \ldots, a_n$, а B — категориальная выборка, принимающая k значений $b_1, b_2, \ldots, b_k$.

	$A = a_1$	$A = a_2$	$\ldots$	$A = a_n$
$B = b_1$	f_{11}	f_{12}	$\ldots$	f_{1n}
$B = b_2$	f_{21}	f_{22}	$\ldots$	f_{2n}
$\ldots$	$\ldots$	$\ldots$	$\ldots$	$\ldots$
$B = b_k$	f_{k1}	f_{k2}	$\ldots$	f_{kn}

Нулевая гипотеза H_0 : две переменные независимы.

Метод хи-квадрат можно провести с помощью функции `chi2_contingency()` из библиотеки `scipy.stats` на языке Python, на R — с помощью функции `chisq.test()` из библиотеки `stats`.

Выясним, существует ли зависимость между уровнем ВВП и регионом проживания.

Для это сделаем предварительную обработку данных:

- Разделим показания ВВП стран на три уровня: низкий (ниже 8.6), средний (от 8.6 до 10.1), высокий (более 10.1).
- Для удобства объединим некоторые регионы в бóльшие сущности:
  1. Europe: Western Europe + Central and Eastern Europe + Commonwealth of Independent States;
  2. America: North America and ANZ + Latin America and Caribbean;
  3. Asia: East Asia + Southeast Asia + South Asia;
  4. Africa: Sub-Saharan Africa + Middle East and North Africa

Построим таблицу сопряженности на Python и R для этого случая:

region \gdp_category	**low**	**medium**	**high**
Africa	32	15	6
America	1	18	5
Asia	6	10	6
Europe	2	15	33

Результаты программы на Python и R идентичны.

Листинг 10: Результат метода хи-квадрат

```
chi^2: 71.33695169360263, p-value: 2.1739840843314693e-13,
degree of freedom: 6
```

```
Expected frequencies:
[[14.58389262 20.63087248 17.7852349 ]
 [ 6.60402685  9.34228188  8.05369128]
 [ 6.05369128  8.56375839  7.38255034]
 [13.75838926 19.46308725 16.77852349]]
```

Значение p-value < 0.05, значит между регионом и уровнем ВВП существует статистически значимая зависимость.

10.2 Точный тест Фишера

Точный тест Фишера — это статистический метод для проверки независимости двух категориальных переменных в таблице сопряженности 2×2, особенно эффективный при малых выборках.

Пусть A, B - две категориальные выборки, принимающие два значения: a_1, a_2 и b_1, b_2 соответственно.

	$A = a_1$	$A = a_2$
$B = b_1$	a	b
$B = b_2$	c	c

Нулевая гипотеза H_0 : две переменные независимы.

Альтернативная гипотеза H_1 : между переменными A и B существует статистически значимая зависимость.

Точный критерий Фишера рассчитывается по следующей формуле:

$$P = \frac{(a+b)! \cdot (c+d)! \cdot (a+c)! \cdot (b+d)!}{a! \cdot b! \cdot c! \cdot d! \cdot n!}$$

Точный тест Фишера можно провести с помощью функции `fisher_exact()` из библиотеки `scipy.stats` на языке Python, на R — с помощью функции `fisher.test()` из библиотеки `stats`.

Выясним, существует ли зависимость между уровнем счастья (высокий или низкий) и регионом проживания (Европа или Африка).

Построим таблицу сопряженности на Python и R для этого случая:

region \happiness	happy	unhappy
Africa	6	47
Europe	28	10

Результаты программы на Python и R идентичны.

Листинг 11: Результат точного теста Фишера

```
Odds ratio: 0.04559270516717325
p-value: 1.6941705379426915e-09
```

Значение p-value < 0.05, значит между регионом и уровнем счастья существует статистически значимая зависимость.

10.3 Тест МакНемара

Рассмотренный выше критерий хи-квадрат для анализа таблиц сопряженности размером 2x2 применим только в отношении независимых наблюдений. Если же учет какого-либо бинарного признака выполняется, например, на одних и тех же испытуемых, то вместо критерия хи-квадрат следует использовать **критерий Мак-Немара**. Он особенно полезен при анализе изменений в дихотомических переменных до и после воздействия или лечения.

	После: положительный	**После: отрицательный**	**Итого**
До: положительный	a	b	$a+b$
До: отрицательный	c	d	$c+d$
Итого	$a+c$	$b+d$	n

Обозначения:

- a: количество объектов с положительным результатом до и после воздействия;
- b: количество объектов с положительным результатом до и отрицательным после;
- c: количество объектов с отрицательным результатом до и положительным после;
- d: количество объектов с отрицательным результатом до и после.

Нулевая гипотеза (H_0): пропорции положительных и отрицательных изменений равны, то есть b = c.

Альтернативная гипотеза (H_1): пропорции положительных и отрицательных изменений различаются, то есть $b \neq c$.

Тест МакНемара можно провести с помощью функции `mcnemar()` из библиотеки `statsmodels.stats.contingency_tables` на языке Python, на R — с помощью функции `mcnemar.test()` из библиотеки `stats`.

В дополнение к датасету об уровне счастья в мире за 2021 год я нашел данные за 2023 год. Сравним изменения.

Пусть страна является счастливой, если happiness_score > 6.

Построим таблицу сопряженности 2×2 на Python и R:

2021 \2023	happy	unhappy
happy	47	3
unhappy	7	76

Результаты теста на Python и R идентичны:

- Статистика: 0.9
- p-value: 0.34278171114790873

p-value > 0.05, что свидетельствует о отсутствии статистически значимых измененииий между уровнем счастья в мире в 2021 и 2023 годах.

10.4 Тест Кохрана-Мантеля-Хензеля

Тест Кохрана–Мантеля–Хензеля (КМХ) предназначен для анализа связи между двумя категориальными переменными при контроле влияния третьей категориальной переменной. Для этого создается K таблиц сопряженности 2x2, где K — количество значений в третьей категориальной переменной.

Нулевая гипотеза, проверяемая при помощи CMH-критерия, заключается в том, что между двумя анализируемыми качественными признаками нет никакой связи.

Изучим, как связаны уровень счастья happiness_score и социальная поддержка social_support в зависимости от региона (для этого теста вновь разделим все страны на четыре больших региона, описанных в тесте хи-квадрат).

Напомним, что мы называем страну *счастливой*, если happiness_score > 6. Назовем страну *страной с высоким уровнем социальной поддержки*, если social_support > 0.77.

Для каждого региона программы на Python и R выдали следующие таблицы сопряженности.

Листинг 12: Таблицы сопряженности в КМХ

```
## $Europe
##
##            high low
##   happy      30   0
##   unhappy    18   2
##
## $Africa
##
##            high low
##   happy       6   0
##   unhappy    13  34
##
## $Asia
##
##            high low
##   happy       2   0
##   unhappy    14   6
##
## $America
##
##            high low
##   happy      13   1
##   unhappy     9   1
```

Значения статистик и p-value на Python и R немного отличаются.

- Python:
  - Статистика: 12.964508202240273
  - p-value: 0.00031745183824949397
- R:

Листинг 13: Результаты КМХ на R

```
Mantel-Haenszel chi-squared test with continuity correction

data:  combined_array
Mantel-Haenszel X-squared = 10.812, df = 1, p-value = 0.001009
alternative hypothesis: true common odds ratio is not equal to 1
95 percent confidence interval:
    2.130018 125.704286
sample estimates:
common odds ratio
        16.36314
```

p-value < 0.05, значит связь между уровнем счастья и уровнем социальной поддержки есть.

11 Проверить наличие мультиколлинеарности в данных с помощью корреляционной матрицы и фактора инфляции дисперсии.

Мультиколлинеарность — это явление, при котором одна из входных переменных статистической модели (например, множественной линейной регрессии) линейно зависит от других входных переменных, т.е. между ними наблюдается сильная корреляция. В этой ситуации оценки коэффициентов (параметров) модели могут случайно и значительно изменяться даже при небольших изменениях в исходных данных, т.е. решение становится неустойчивым.

Существует множество методов борьбы с мультиколлинеарностью, на своих датасетах я воспользуюсь первыми двумя из списка ниже:

1. Анализ корреляционной матрицы.
2. Фактор инфляции дисперсии (Variance Inflation Factor, VIF).
3. PCA (Principal Component Analysis): преобразование исходных переменных в набор независимых компонентов.
4. Регуляризация (например, Ridge или Lasso регрессия).

11.1 Корреляционнная матрица

Корреляционная матрица (матрица корреляций) — это квадратная таблица, заголовками строк и столбцов которой являются обрабатываемые переменные, а на пересечении строк и столбцов выводятся коэффициенты корреляции для соответствующей пары признаков.

Корреляционная матрица обладает следующими свойствами:

1. На главной диагонали находятся коэффициенты корреляции, равные единице.
2. Матрица симметрична относительно главной диагонали

Ниже приведены корреляционные матрицы для датасетов о качестве сна и об уровне счастья.

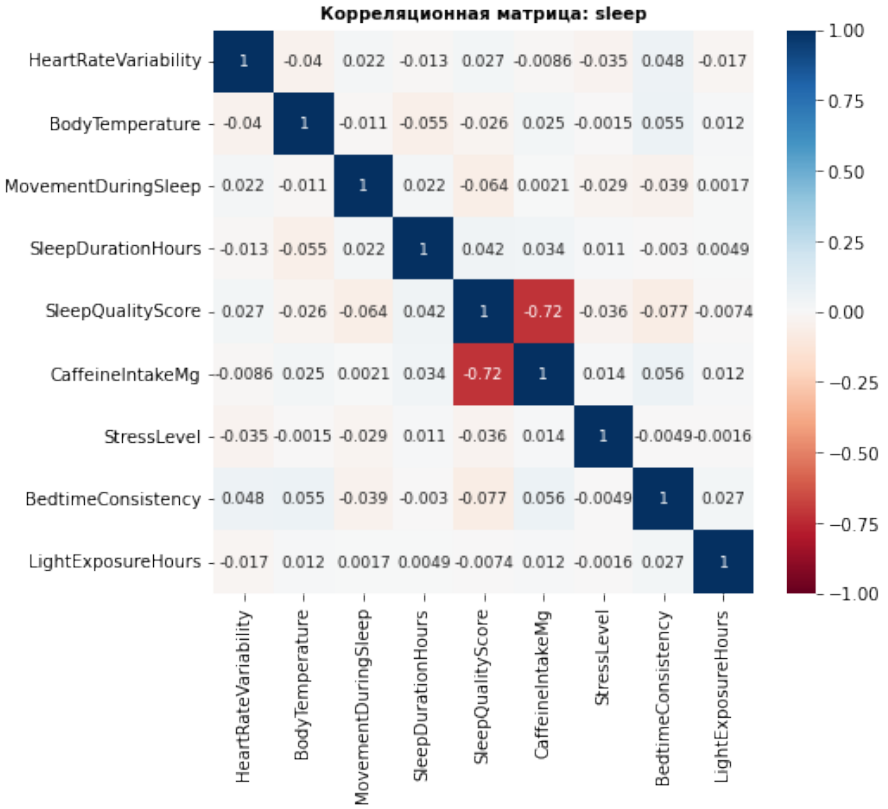

Рис. 37: Корреляционная матрица для датасета о качестве сна на Python

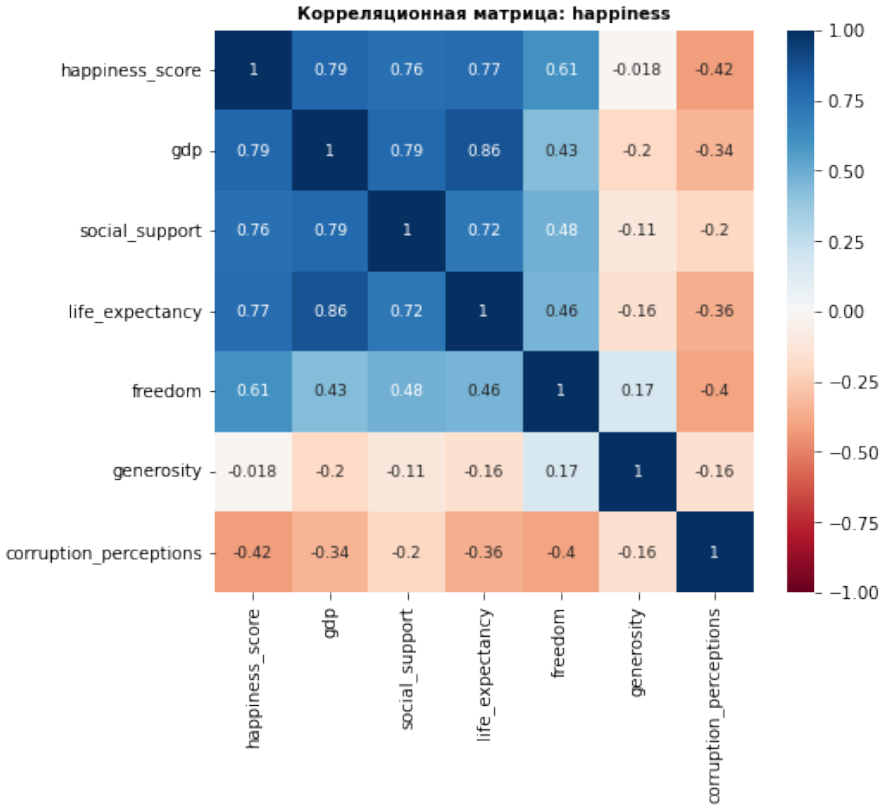

Рис. 38: Корреляционная матрица для датасета об уровне счастья на Python

Выводы

1. В первой матрице матрице для качества сна видна четкая зависимость между оценкой качества сна и количеством принятого кофеина: чем больше человек принял кофеина (например, выпил много кофе), тем хуже он спит.

2. Во второй матрице на уровень счастья в стране положительно влияют высокий показатель ВВП (т.е. развитая экономика), высокая социальная поддержка (пенсии, стипендии, хорошее медобслуживание и прочее), уровень свободы, однако очень много людей беспокоит уровень коррупции, что снижает уровень счастья.

3. На языке Python корреляции можно посчитать при помощи внутреннего метода `corr()` библиотеки `pandas`, для визуализации я воспользовался методом `heatmap()` из библиотеки `seaborn`. На языке R существуют похожие удобные инструменты для построения корреляционной матрицы: метод `cor()` для вычисления корреляций и метод `ggplot()` с дополнительными настройками для визуализации.

4. Программы на обоих языках программирования построили одинаковые корреляционные матрицы.

11.2 Фактор инфляции дисперсии

В задаче восстановления регрессии фактор инфляции дисперсии (VIF) — мера мультиколлинеарности. Он позволяет оценить увеличение дисперсии заданного коэффициента регрессии, происходящее из-за высокой корреляции данных.

Пусть задана выборка $D = \{y_i, x_i\}_{i=1}^n$ откликов и признаков. Рассматривается множество линейных регрессионных моделей вида:

$$y_i = \sum_{j=1}^{m} w_j x_{ij} + \varepsilon_i, \quad i = 1, \ldots, n$$

Предполагается, что вектор регрессионных невязок имеет нулевое математическое ожидание и дисперсию σ^2. В этом случае дисперсия $\hat{w}_j$:

$$D\hat{w}_j = \frac{\sigma^2}{(n-1)Dx_j \cdot (1 - R_j^2)}$$

Первая дробь связана с дисперсией невязок и дисперсией векторов признаков. Вторая — фактор инфляции дисперсии, связанный с корреляцией данного признака с другими:

$$VIF_j = \frac{1}{1 - R_j^2}$$

где R_j^2 — коэффициент детерминации j-го признака относительно остальных:

$$R_j^2 = 1 - \frac{\sum_{i=1}^n (x_{ij} - \hat{x}_{ij})^2}{\sum_{i=1}^n (x_{ij} - \bar{x}_j)^2}$$

Равенство единице фактора инфляции дисперсии говорит об ортогональности вектора значений признака остальным. Если значение VIF_j велико, то $1 - R_j^2$ мало, то есть R_j^2 близко к 1. Большие значения фактора инфляции дисперсии соответствуют почти линейной зависимости j-го столбца от остальных. Высокие значения VIF указывают на потенциальные

проблемы с мультиколлинеарностью. Как правило, значение VIF выше 5 требует внимания, а выше 10 — серьезного рассмотрения изменений в модели.

Посчитаем VIF для датасетов о качестве сна и об уровне счастья, сравним показатели, полученные программами на Python и R.

Variable	**VIF (Python)**	**VIF (R)**
HeartRateVariability	1.007842	1.007952
BodyTemperature	1.008690	1.008979
MovementDuringSleep	1.013106	1.013596
SleepDurationHours	1.015123	1.015059
SleepQualityScore	2.139904	2.125253
CaffeineIntakeMg	2.118646	2.103486
StressLevel	1.004118	1.003903
BedtimeConsistency	1.014450	1.014817
LightExposureHours	1.001291	1.001373

Таблица 21: Сравнение фактора инфляции дисперсии на Python и R для датасета о качестве сна

Variable	**VIF (Python)**	**VIF (R)**
happiness_score	4.095795	4.095795
gdp	5.477390	5.477390
social_support	3.259623	3.259623
life_expectancy	4.248569	4.248569
freedom	1.770176	1.770176
generosity	1.191685	1.191685
corruption_perceptions	1.408890	1.408890

Таблица 22: Сравнение фактора инфляции дисперсии на Python и R для датасета об уровне счастья

В модели об уровне счастья на фоне прочих предикторов сильно выделяются `gdp`, `life_expectancy`, `social_support`. Удалим один из параметров (например, `gdp`) из данных.

Variable	**VIF (Python)**	**VIF (R)**
happiness_score	3.817253	3.817253
social_support	2.837053	2.837053
life_expectancy	2.963900	2.963900
freedom	1.742486	1.742486
generosity	1.154653	1.154653
corruption_perceptions	1.391416	1.391416

Таблица 23: Сравнение фактора инфляции дисперсии на Python и R для датасета об уровне счастья после удаления признака `gdp`

Выводы

1. В модели о качестве сна показатели VIF низкие, поэтому мультиколлинеарности данных нет.

2. В модели об уровне счастья до удаления предиктора `gdp` были высокие показатели VIF; после его удаления они стали ниже, мультиколлинеарности данных нет.

3. На языке Python фактор инфляции дисперсии можно посчитать при помощи функции `variance_inflation_factor()` модуля `statsmodels.stats.outliers_influence`. На языке R существуют похожие удобные инструменты: метод `VIF()` для вычисления VIF линейной модели, которая может быть построена при помощи метода `lm()`.

4. Программы на обоих языках программирования вычислили одинаковые показатели VIF во всех случаях.

12 Исследовать зависимости в данных с помощью дисперсионного анализа.

Дисперсионный анализ (ANOVA) — это статистический метод, который используется для сравнения средних значений двух или более выборок. Он позволяет определить, различаются ли средние значения между группами, или же различия случайны.

12.1 Однофакторный дисперсионный анализ (one-way ANOVA)

Однофакторный ANOVA (однофакторный дисперсионный анализ) —- это метод статистического анализа данных, который используется для определения наличия статистически значимых различий между двумя или более группами по одной независимой переменной.

Пусть A — категориальная переменная (фактор) с k уровнями и пусть $x^1 = (x_{11}, \ldots, x_{1n}), ..., x^k = (x_{k1}, \ldots, x_{kn})$ — подвыборки выборки x, $\mu_1, \ldots, \mu_k$ — математические ожидания подвыборок.

Нулевая гипотеза $H_0 : \mu_1 = \mu_2 = \ldots = \mu_k$

Альтернативная гипотеза: не все средние равны.

Дополнительные предположения для однофакторного ANOVA:

1. **Независимость наблюдений**: данные в каждой группе должны быть независимыми друг от друга.

2. **Нормальность распределения**: распределение зависимой переменной в каждой группе должно быть близким к нормальному.

3. **Гомогенность дисперсий**: дисперсии зависимой переменной в разных группах должны быть примерно равными.

Применим однофакторный дисперсионный анализ для зависимой переменной `happiness_score` по факторной переменной `region`. Ранее было выявлено, что выборка `happiness_score` является нормальной; считаем, что проведение опросов в каждой стране происходило независимо от опроса в других странах.

Теперь проверим гомогенность дисперсий `happiness_score` в разных группах. Для начала посмотрим на данные по регионам при помощи stripchart.

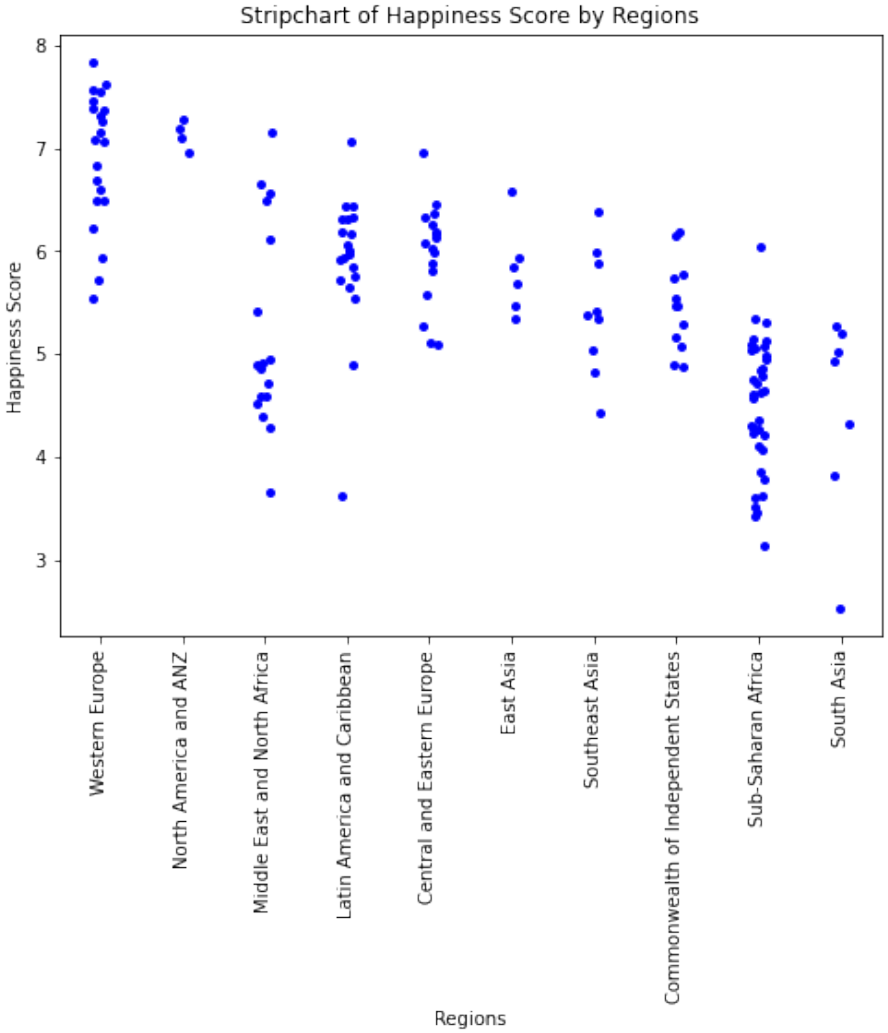

Рис. 39: Stripchart зависимости уровня счастья от региона на Python

Заметим, что регионы `Latin America and Caribbean`, `Sub-Saharan Africa`, `Middle East and North Africa` сильно выделяются на фоне других по дисперсии. Проведем тест Левене для них.

- Python: `LeveneResult(statistic=1.7279520537516624, pvalue=0.1851384416472439)`
- R:

Листинг 14: Результаты теста Левене для трех регионов на R

```
Levene's Test for Homogeneity of Variance (center = median)
       Df F value Pr(>F)
group   2   1.728 0.1851
       70
```

p-value > 0.05, поэтому можем принять гипотезу о равенстве дисперсий. **One-Way ANOVA применим**.

Сравним результаты однофакторного дисперсионного анализа на Python и R:

- Python:
  - F-статистика: 22.962959574332046
  - p-value: 2.1483855397171614e-08
- R:

Листинг 15: Результаты One-Way ANOVA на R

```
             Df  Sum  Sq  Mean  Sq F  value    Pr(>F)
region        2   26.32   13.162    22.96 2.15e-08 ***
Residuals    70   40.12    0.573
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

p-value < 0.05, поэтому мы не можем принять гипотезу о равенстве средних, значит, регион проживания действительно влияет на уровень счастья.

Выводы

1. Для применения однофакторного дисперсионного анализа необходимо проверить независимость наблюдений, нормальность распределения зависимой переменной и гомогенность дисперсий зависимой переменной по группам.
2. На языке Python однофакторный дисперсионный анализ можно провести при помощи функции `f_oneway()` библиотеки `scipy.stats`. На языке R существует аналогичная функция `aov()`.
3. Программы на обоих языках продемонстрировали одинаковые результаты анализа.

12.2 Двухфакторный дисперсионный анализ (two-way ANOVA)

Двухфакторный дисперсионный анализ используется для оценки влияния двух независимых факторов на зависимую переменную, а также для анализа взаимодействия между этими факторами.

В качестве зависимой переменной вновь возьмем `happiness_score`, в качестве факторных — регион и уровень свободы. Предварительно разделим уровень свободы на три фактора: низкий (значение `freedom`меньше 0.7), средний (от 0.7 до 0.85), высокий (больше 0.85).

Ниже приведем результаты two-way ANOVA на языке Python и R.

Source	**Sum Sq**	**df**	**F**	**PR(>F)**
freedom_level	8.830030	2	11.514710	0.000054
region	11.585382	2	15.107799	0.000004
freedom_level:region	6.755099	4	4.404459	0.003301
Residual	24.539128	64	NaN	NaN

Таблица 24: Результаты two-way ANOVA на языке Python

Листинг 16: Результаты two-way ANOVA на R

```
Anova Table (Type II tests)
Response: happiness_score
                       Sum Sq Df F value    Pr(>F)
freedom_level          8.8300  2 11.5147 5.351e-05 ***
region                11.5854  2 15.1078 4.225e-06 ***
freedom_level:region   6.7551  4  4.4045  0.003301 **
Residuals             24.5391 64
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Выводы

1. На языке Python двухфакторный дисперсионный анализ можно провести следующим образом: для начала необходимо построить модель при помощи функции `ols()` библиотеки `statsmodels.formula.api`, а затем использовать метод `anova_lm()` библиотеки `statsmodels.api.stats` для вывода результатов анализа. На языке R подход аналогичен: нужно построить линейную модель, используя функцию `lm()`, а затем провести дисперсионный анализ при помощи функции `Anova()`.

2. Программы на обоих языках продемонстрировали одинаковые результаты анализа: вышеперечисленные факторы и их сочетания влияют на уровень счастья.

13 Подогнать регрессионные модели (в том числе, нелинейные) к данным, а также оценить качество подобной аппроксимации.

13.1 Линейная регрессия для датасета об уровне счастья

Линейная регрессионная модель используется для предсказания значения зависимой переменной на основе одной или нескольких независимых переменных, выявляя линейные зависимости между ними.

Для начала построим модель зависимости уровня счастья (HS) от остальных признаков: gdp (GDP), social_support (S), life_expectancy (LE), freedom (F), generosity (G). Уравнение регрессии в этом случае будет иметь вид:

$$HS = \beta_0 + \beta_1 \cdot GDP + \beta_2 \cdot S + \beta_3 \cdot LE + \beta_4 \cdot F + \beta_5 \cdot G,$$

где β_0 — свободный член, а $\beta_i, i = 1, ..., 5$ — коэффициенты при соответствующих признаках.

Результаты регрессии:

- R^2: 0.780 (Python), 0.7798 (R)
- Уравнение регрессии (на обоих языках):

$$HS = -3.3409 + 0.2849 \cdot GDP + 2.8889 \cdot S + 0.0292 \cdot LE + 2.4813 \cdot F + 0.4784 \cdot G$$

- MSE (mean square error, на обоих языках): 0.2675

- RMSE (root mean square error, на обоих языках): 0.5172

Построим модель зависимости уровня счастья (HS) от одного признака (например, gdp (GDP)) и сравним с результатами модели от нескольких переменных.

- R^2: 0.619 (Python), 0.6194 (R)
- Уравнение регрессии (на обоих языках):

$$HS = -1.4149 + 0.7363 \cdot GDP$$

- MSE (mean square error, на обоих языках): 0.46235
- RMSE (root mean square error, на обоих языках): 0.67996

Рис. 40: Линейная регрессия зависимости уровня счастья от ВВП

Выводы

1. Линейная регрессионная модель от нескольких предикторов дает лучшие результаты в сравнении с моделью от одного предиктора gdp.
2. Оба языка программирования дают одинаковое качество моделей.

13.2 Полиномиальная регрессия для датасета об уровне счастья

Полиномиальная регрессионная модель используется для моделирования нелинейных зависимостей между переменными, добавляя полиномиальные степени независимых переменных, чтобы лучше описывать сложные отношения.

Для начала построим полиномиальную модель степени 2 зависимости уровня счастья (HS) от остальных признаков и их квадратов: gdp (GDP), social_support (S), life_expectancy (LE), freedom (F), generosity (G). Уравнение регрессии в этом случае будет иметь вид:

$$HS = \beta_0 + \beta_1 \cdot GDP + \beta_2 \cdot S + \beta_3 \cdot LE + \beta_4 \cdot F + \beta_5 \cdot G + \beta_6 \cdot GDP^2 + \beta_7 \cdot S^2 + \beta_8 \cdot LE^2 + \beta_9 \cdot F^2 + \beta_{10} \cdot G^2$$

где β_0 — свободный член, а $\beta_i, i = 1, ..., 10$ — коэффициенты при соответствующих признаках.

Результаты регрессии:

- R^2: 0.839 (Python), 0.8388 (R)
- Уравнение регрессии (на обоих языках): $\text{HS} = 9.782 + 0.621 \cdot GDP - 20.444 \cdot S - 0.156 \cdot LE + 4.214 \cdot F + 6.266 \cdot G + 0.118 \cdot GDP^2 - 7.835 \cdot S^2 - 0.0012 \cdot LE^2 - 2.330 \cdot F^2 - 2.009 \cdot G^2$
- MSE (mean square error, на обоих языках): 0.1958
- RMSE (root mean square error, на обоих языках): 0.4425

Построим полиномиальную модель степени 2 зависимости уровня счастья (HS) от одного признака (например, gdp (GDP)) и сравним с результатами модели от нескольких переменных.

- R^2: 0.642 (Python), 0.6425 (R)
- Уравнение регрессии (на обоих языках):

$$HS = 7.5689 - 1.2518 \cdot GDP + 0.1081 \cdot GDP^2$$

- MSE (mean square error, на обоих языках): 0.4343
- RMSE (root mean square error, на обоих языках): 0.659

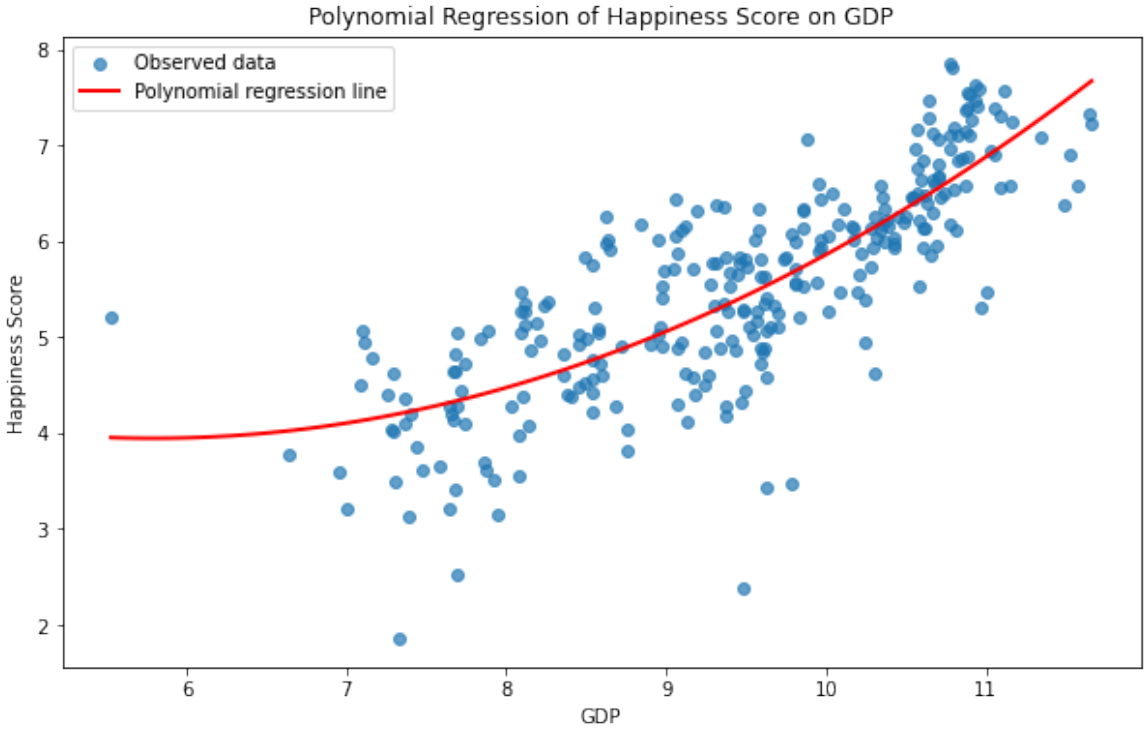

Рис. 41: Полиномиальная регрессия степени 2 зависимости уровня счастья от ВВП

Выводы

1. Полиномиальная регрессионная модель степени 2 от всех предикторов дает лучшие результаты в сравнении с моделью от одного предиктора gdp.
2. Оба языка программирования дают одинаковое качество моделей.

13.3 Логарифмическая регрессия для датасета об уровне счастья

Попробуем построить модель не от самих предикторов, а от их логарифмов.

Для начала построим полиномиальную модель зависимости уровня счастья (HS) от остальных признаков: gdp (GDP), social_support (S), life_expectancy (LE), freedom (F), generosity (G). Уравнение регрессии в этом случае будет иметь вид:

$$HS = \beta_0 + \beta_1 \cdot \log(GDP) + \beta_2 \cdot \log(S) + \beta_3 \cdot \log(LE) + \beta_4 \cdot \log(F) + \beta_5 \cdot \log(G)$$

где β_0 — свободный член, а $\beta_i, i = 1, ..., 5$ — коэффициенты при соответствующих признаках.

Результаты регрессии:

- R^2: 0.771 (Python), 0.7714 (R)
- Уравнение регрессии (на обоих языках):

  $$HS = -14.8488 + 2.7811 \cdot \log(GDP) + 4.9545 \cdot \log(S) + 2.0335 \cdot \log(LE) + 4.2328 \cdot \log(F) + 0.6212 \cdot \log(G)$$

- MSE (mean square error, на обоих языках): 0.2777
- RMSE (root mean square error, на обоих языках): 0.5269

Построим модель зависимости уровня счастья (HS) от логарифма признака ВВП (GDP) и сравним с результатами модели от нескольких переменных.

- R^2: 0.597 (Python), 0.5966 (R)
- Уравнение регрессии (на обоих языках):

  $$HS = -9.0838 + 6.5361 \cdot \log(GDP)$$

- MSE (mean square error, на обоих языках): 0.49
- RMSE (root mean square error, на обоих языках): 0.7

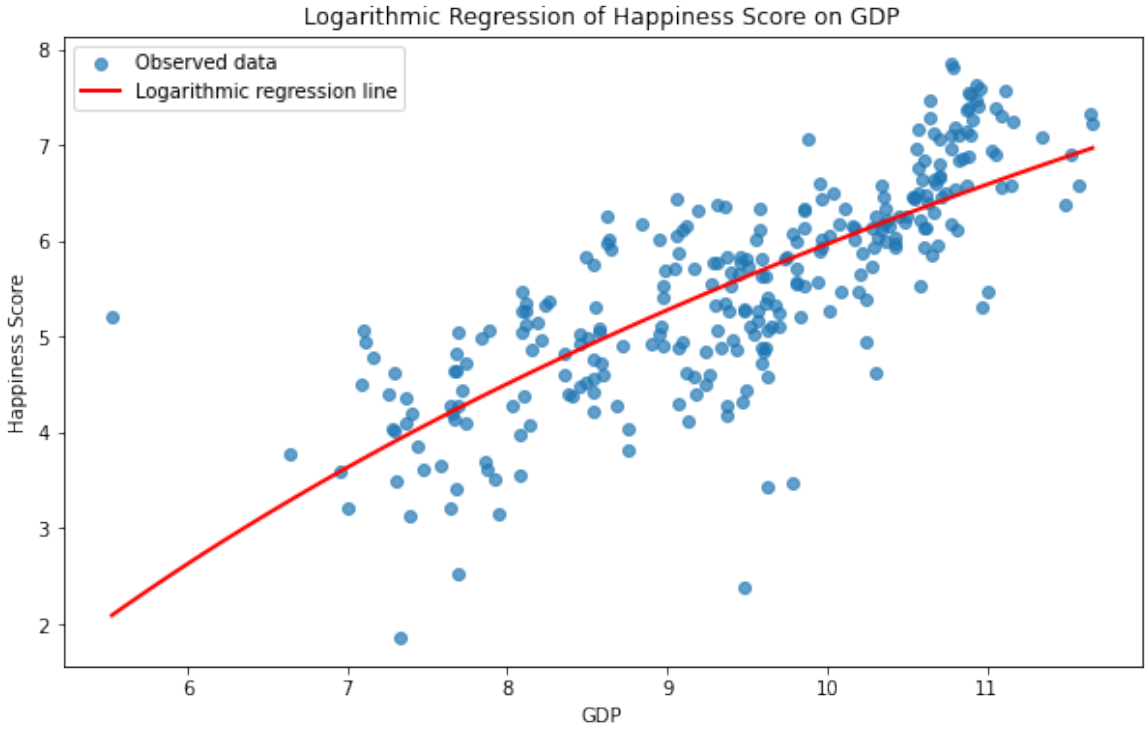

Рис. 42: Регрессия зависимости уровня счастья от логарифма ВВП

Выводы

1. Регрессионная модель от логарифмов всех предикторов дает лучшие результаты в сравнении с моделью от логарифма предиктора gdp.

2. Оба языка программирования дают одинаковое качество моделей.

Общие выводы

1. Линейная модель дает среднее качество, логарифмирование признаков лишь ухудшило качество, а полином степени 2 лучше аппроксимирует данные.

2. Лучшее качество показала полиномиальная модель степени 2 от всех предикторов, худшее — модель от логарифма предиктора ВВП.

3. Оба языка программирования дают одинаковое качество моделей.

14 Выводы

В процессе выполнения задания по курсу «Технологический практикум» я освоил применение инструментов статистического анализа данных на языках программирования R и Python. Были изучены свойства различных датасетов, включая данные о качестве сна, уровне счастья в мире за 2021 и 2023 годы, а также информацию о различных видах хлопьев.

Для анализа данных мной были использованы ключевые статистические методы и техники:

- **Визуализация данных**: построены графики `cdplot`, `dotchart`, `boxplot` и `stripchart`, что позволило наглядно представить распределение и вариативность данных.
- **Выявление выбросов**: применены критерии Граббса и Q-тест Диксона для обнаружения аномальных значений в выборках.
- **Исследование нормальности распределения**: проанализированы свойства нормально распределённых выборок с помощью графиков эмпирических функций распределения, квантилей и метода огибающих, а также проведены стандартные процедуры проверки гипотез о нормальности.
- **Аппроксимация распределений**: выполнена оценка плотности распределения данных с помощью ядерных оценок, что позволило более точно охарактеризовать поведение случайных величин.
- **Обработка пропусков**: изучены и применены инструменты заполнения пропусков в данных, обеспечивающие корректный анализ даже при неполноте исходной информации.

Кроме того, я научился формулировать и проверять статистические гипотезы, используя следующие техники:

- **Параметрические и непараметрические тесты**: критерии Стьюдента, Уилкоксона-Манна-Уитни, Фишера и другие.
- **Корреляционный анализ**: исследованы взаимосвязи между переменными с помощью коэффициентов корреляции Пирсона, Спирмена и Кендалла.
- **Критерии независимости**: применены методы хи-квадрат, Фишера, МакНемара и Кохрана-Мантеля-Хензеля для проверки независимости распределений.

Важным этапом работы стала проверка наличия мультиколлинеарности в данных с использованием корреляционной матрицы и фактора инфляции дисперсии (VIF). Также был проведён дисперсионный анализ (однофакторный и двухфакторный) и подгонка регрессионных моделей под имеющиеся данные, что позволило выявить основные влияющие факторы и построить предсказательные модели.

Все применённые методы были успешно реализованы на выбранных мной датасетах, что позволило глубоко проанализировать представленные данные и сделать обоснованные выводы. В результате выполнения данного задания я значительно повысил свои навыки в области статистического анализа данных и их интерпретации.

Кроме того, результаты, полученные с использованием языков программирования Python и R, оказались практически идентичными, что подтверждает возможность применения обоих языков программирования для статистического анализа данных.

Список литературы

[1] Брюс П., Брюс Э., Гедек П. *Практическая статистика для специалистов Data Science.* – 2-е изд., перераб. и доп. — СПб.: БХВ-Петербург, 2021. – 352 с.

[2] Мастицкий С.Э., Шитиков В.К. *Статистический анализ и визуализация данных с помощью R.* – М.: ДМК Пресс, 2015. – 496 с.

[3] Нильсен Э. *Практический анализ временных рядов: прогнозирование со статистикой и машинное обучение.* – СПб.: ООО «Диалектика», 2021. – 544 с.