ГЛАВА 1.
ВЫПОЛНЕНИЕ РАСЧЕТНО-АНАЛИТИЧЕСКОЙ РАБОТЫ С ПОМОЩЬЮ ЯЗЫКА PYTHON

Прежде чем перейти к выполнению работы с помощью языка программирования Python, необходимо обосновать его актуальность и востребованность для решения данной задачи. В отличие от привычной программы Microsoft Excel, позволяющей выполнять статистические исследования и визуализировать данные при помощи простых инструментов, Python позволяет быстро анализировать большие объемы данных с использованием углубленной аналитической базы, строить модели и визуализировать полученные результаты. Широкий доступ к различным высокоэффективным библиотекам делает Python наиболее доступным и удобным языком программирования, предназначенным для анализа данных. Особенно хорошо Python показывает себя при работе с большими данными (Big Data), анализ которых в среде Excel попросту невозможен ввиду ограничения на число строк (1048576).

1.1. Загрузка программы

Для начала необходимо загрузить утилиту Anaconda, где можно будет использовать окружение Jupiter Notebook. Для этого нужно открыть официальный сайт Anaconda.com, выбрать требующуюся разрядность и запустить установочный файл. (ссылка для скачивания: https://www.anaconda.com/products/distribution).

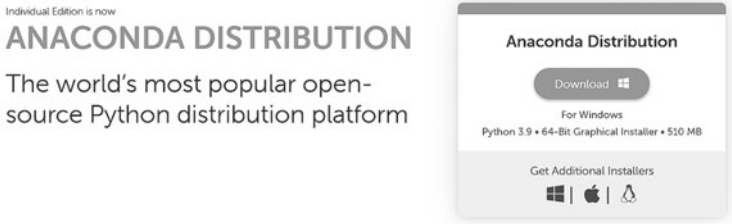

Рис. 1. Загрузка программы Anaconda

В Anaconda.Navigator — графическом интерфейсе рабочего стола — находим окружение Jupiter. Jupyter Notebook — это инструмент для создания аналитических работ, так как он позволяет хранить вместе код, изображения, комментарии, формулы и графики. ПО подготовлено, переходим к парсингу данных. Под парсингом понимается процесс автоматизированного сбора информации, необходимой для анализа.

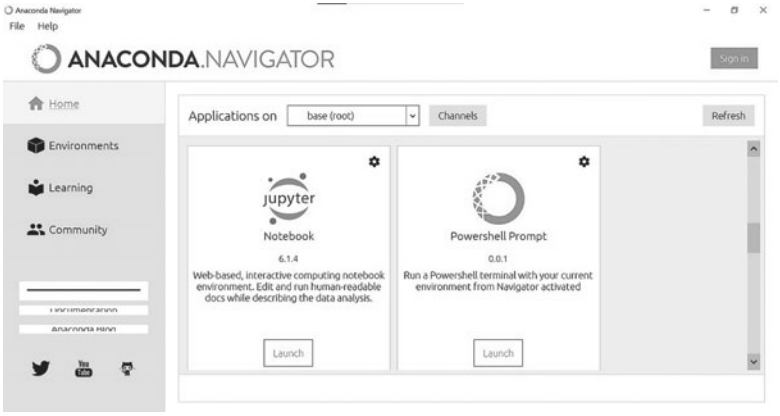

Рис. 2. Главная страница графического интерфейса Anaconda.Navigator

1.2. Загрузка исходных данных

Данные для исследования считываются из базы данных Московской биржи. Для этого следует перейти по адресу: https://mfd.ru/export/

На сайте MFD в разделе «Мосбиржа Акции и ПИФы» найдем тикеры нужных компаний с 2015 года по настоящее время. Далее следует задать формат записей формируемого списка. Для успешной обработки данных об акциях компаний с помощью программы Python установим следующие параметры. Промежуток — неделя, разделитель — точка с запятой, а вот десятичный разделитель лучше выбрать точкой — Python интерпретирует float, то есть числа с плавающей точкой, только если используется именно таковая.

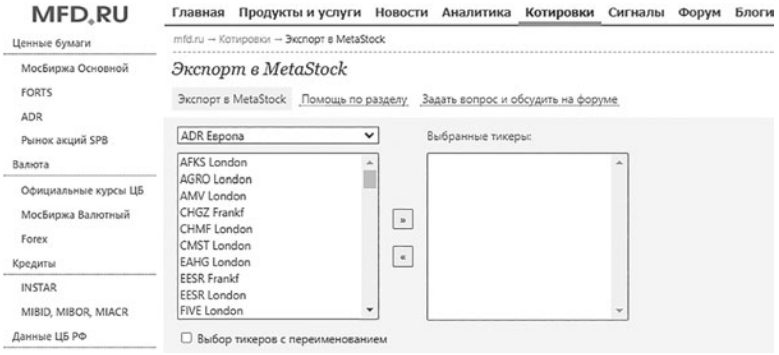

Рис. 3. Выбор акций нужных компаний и формирование формата записей

Вообще формат txt не лучшим образом подходит для обработки библиотекой pandas, но это и не худший вариант. В любом случае, всегда есть возможность использовать формат csv. После нажатия кнопки «получить данные» в памяти компьютера сохранился текстовый файл с необходимыми данными. Теперь перейдем к загрузке этих данных в окружение Jupiter Notebook.

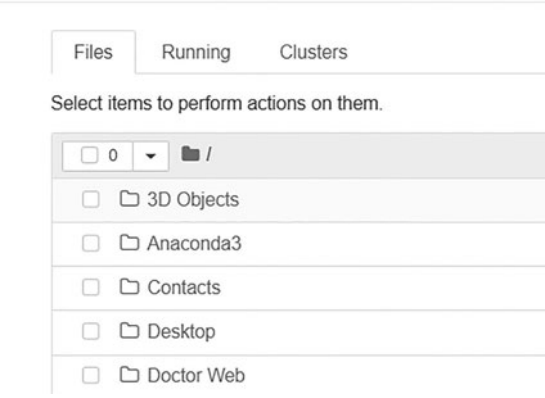

Рис. 4. Поиск исходного файла

Для формирования имени файла следует использовать буквы латинского алфавита, не использовать пробелы. После изменения названия файла осуществим его загрузку. Для этого необходимо нажать кнопку 'Upload'.

Рис. 5. Загрузка исходного файла

Для создания notebook (нового файла) выбираем «New» в верхнем меню, а потом нажимаем «Python 3».

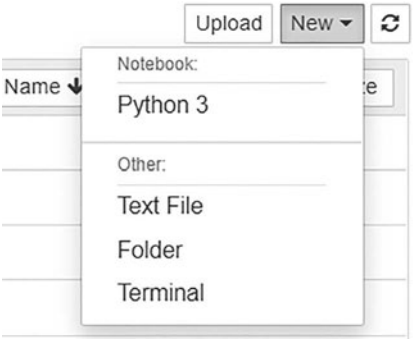

Рис. 6. Создание рабочего файла

Интерфейс Jupiter выглядит следующим образом. Добавление ячейки для написания кода происходит нажатием клавиши «плюс» на панели управления. Запуск кода происходит сочетанием клавиш Cnrl+Enter после того, как код выполнен у ячеек появляется нумерация. В Jupyter Notebook есть несколько инструментов, используемых для добавления описания. С их помощью можно не только оставлять комментарии, но также добавлять заголовки, списки и форматировать текст. Это делается с помощью Markdown. Чтобы поменять тип ячейки, нужно нажать на выпадающее меню с текстом “Code” и выбрать «Markdown» (либо использовать сочетание клавиш Esc+M).

Преступим к загрузке библиотеки. Первая команда — импорт библиотеки pandas. Назовем эту библиотеку pd для удобства командой ‘as pd’ (эта процедура называется, по аналогии с одной известной игрой, alias — присвоение имени).

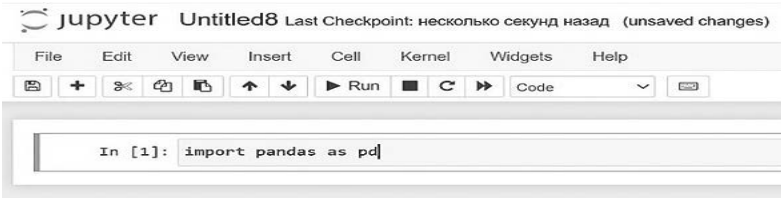

Рис. 7. Загрузка библиотеки pandas

Загрузим данные методом pandas 'read_csv'. И хотя наш файл находится в формате txt, а не csv, метод работает. Укажем в аргументах название файла, кодировку (в нашем случае — utf-8, и разделитель, в нашем случае — точка с запятой. Назовем наш файл df (сокр. data frame — таблица с данными).

```
1 df = pd.read_csv(r"data.txt", encoding='utf-8', sep =';')
2 df
```

Out[2]:

	<TICKER>	<PER>	<DATE>	<TIME>	<CLOSE>	<VOL>
0	АбрауДюрсо	W	20150105	0	90.0	200
1	АбрауДюрсо	W	20150112	0	94.5	2700
2	АбрауДюрсо	W	20150119	0	99.0	800
3	АбрауДюрсо	W	20150126	0	93.0	1000
4	АбрауДюрсо	W	20150202	0	93.0	0
...	...	...	...	...	...	...
970	ЛУКОЙЛ	W	20210222	0	5575.5	6489151
971	ЛУКОЙЛ	W	20210301	0	6236.0	8485835
972	ЛУКОЙЛ	W	20210308	0	6356.0	6099744
973	ЛУКОЙЛ	W	20210315	0	6078.0	10268620
974	ЛУКОЙЛ	W	20210322	0	6162.5	4476623

975 rows × 6 columns

Рис. 8. Загрузка исходных данных в рабочий файл

Первое, что нужно сделать при анализе — посмотреть на структуру исходных данных, для этого существует метод info(). Так как этот метод не требует аргументов, скобки остаются пустыми.

```
1 df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 975 entries, 0 to 974
Data columns (total 6 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   <TICKER>  975 non-null    object
 1   <PER>     975 non-null    object
 2   <DATE>    975 non-null    int64
 3   <TIME>    975 non-null    int64
 4   <CLOSE>   975 non-null    float64
 5   <VOL>     975 non-null    int64
dtypes: float64(1), int64(3), object(2)
memory usage: 45.8+ KB
```

Рис. 9. Анализ структуры исходных данных с помощью метода info()

Исходя из полученных результатов, можно сделать вывод о том, что данные соответствуют своим типам, например, наименование тикера (Ticker) соответствует базовому типу объект (object), а время (Time) соответствует целочисленному типу (int64)[2]. Также можно увидеть, что данные не содержат пропуски, так как количество значений по каждому виду данных соответствует общему числу колонок.

1.3. Оптимизация данных и вычисление дополнительных признаков для каждой компании

Переходим к оптимизации, переименованию столбцов и добавлению новых данных. Удалим лишние и ненужные столбцы методом drop, в аргументах указав те столбцы, которые нужно удалить. В нашем случае аргументами являются столбцы периода и времени. Вообще, обращение к какому-либо столбцу происходит через квадратные скобки и кавычки. Если столбцов несколько, их перечисляют через запятую.

---

[2] *Сузи, Р.А.* Язык программирования Python: Курс лекций. 14 с.

```
1 df = df.drop(columns = ['<PER>' , '<TIME>'])
```

Рис. 10. Удаление ненужных столбцов методом drop()

Для последующей работы переименуем столбцы — передадим колонкам новые названия через метод .columns списком новых названий. В квадратных скобочках указываем наименования столбцов, которые мы хотим получить на выходе.

Следует удалить строки, содержащие нулевые значения цен и объемов.

Также необходимо преобразовать дату. Так как сама дата как элемент данных, получение информации из этого поля не имеет значения для дальнейшего исследования, будет удобнее заменить данные на набор упорядоченных чисел, соответствующих порядку даты. Проще всего это сделать, полностью заменив элемент DataFrame — Series (чаще всего — Series — один столбец или одна строка элемента DataFrame). Так, можно присвоить столбцу Date элемент Series (то есть, присвоить столбцу новый столбец), в которой занесем список из упорядоченного набора чисел от 1 до 325 (всего данных 975, на каждый тикер приходится 975/3 строк). Очень просто такую операция выполнить с помощью простейшей функции range, в качестве аргументов используя начальную позицию (0) и конечную (325). Автоматически шаг значений выставлен в 1, поэтому этот аргумент можно не устанавливать. При этом данному набору значений нужно придать тип списка (функция list), так как списки легко экстраполировать (путем умножения на три — по количеству тикеров — можно получить три одинаковых списка):

```
1 df.columns = ['Ticker', 'Date', 'Price', 'Volume']
2 df['Date'] = pd.Series(list(range(0, 325))*3)
3 display(df)
```

Out[5]:

	Ticker	Date	Price	Volume
0	АбрауДюрсо	0	90.0	200
1	АбрауДюрсо	1	94.5	2700
2	АбрауДюрсо	2	99.0	800
3	АбрауДюрсо	3	93.0	1000
4	АбрауДюрсо	4	93.0	0

Рис. 11. Изменение названий столбцов методом .columns

Добавим столбцы новых параметров: доходностей, логарифмических доходностей (логдоходностей) и логарифмов объемов. Названия столбцов будут соответствовать их «английским» эквивалентам.

Столбец доходностей назовем «Yield». Расчет доходности осуществим с помощью метода pct_change(), считающего относительный рост цены — как раз это и есть доходность. Применим данный метод к столбцу с ценами активов. С технической точки зрения, данный метод позволяет найти разницу между ценами актива соответственно для i–й и i-1 –й, то есть предшествующей недели, и разделить полученный результат на цену актива предшествующей недели

$$r_i = \frac{S_i - S_{i-1}}{S_{i-1}}, \qquad (1)$$

где r_i = доходность,

S_i и S_{i-1} — цены актива соответственно на i-й и $i-1$-й неделях.

Для логдоходностей импортируем библиотеку numpy с названием np. Данная библиотека поддерживает высокоуровневые математические функции и оптимизирует работу с многомерными массивами. Сначала вычислим отношение нынешней цены к предыдущей. Метод shift(1) сдвигает

значения столбца на одно. Далее удалим первую (в python нумерация начинается с нуля), 326 и 651 строку — в них доходности искажены, так как вычислены из цен акций разных эмитентов. После этого выберем натуральный логарифм из библиотеки np и применим к столбцу с логарифмической доходностью. Так мы написали код формулы расчета логарифмической доходности

$$l_i = \ln \frac{s_i}{s_{i-1}}, \tag{2}$$

где l_i = логдоходность,

s_i и s_{i-1} — цены актива соответственно на i-й и $i-1$-й неделях.

Логарифм объёма найдем при помощи лямбда-функции — этот вид функции применим к конкретному значению к ячейке. Методом .apply применим лямбда функцию к значениям столбца с объемами 'Volume', объявим лямбда-функцию при условии 0, если объем равен 0, что логично, иначе логарифм не будет посчитан, и логарифм, если объем отличен от нуля. Вообще лямбда-функция является мощным (наравне с функцией map) инструментом применения собственных функций по какому-либо условию, что бывает полезно для выполнения многих задач анализа данных, поэтому этот функционал можно взять на вооружение для дальнейших исследований.

```
df['Yield'] = df['Price'].pct_change()
import numpy as np
df['Log_Yield'] = df['Price']/df['Price'].shift(1)
df = df.drop([0, 325, 650])
df['Log_Yield'] = np.log(df['Log_Yield'])
df['Log_Volume'] = df['Volume'].apply(lambda x : 0 if (x == 0) else
np.log(x))
display(df.head())
```

Out[6]:

	Ticker	Date	Price	Volume	Yield	Log_Yield	Log_Volume
1	АбрауДюрсо	1	94.5	2700	0.050000	0.048790	7.901007
2	АбрауДюрсо	2	99.0	800	0.047619	0.046520	6.684612
3	АбрауДюрсо	3	93.0	1000	-0.060606	-0.062520	6.907755
4	АбрауДюрсо	4	93.0	0	0.000000	0.000000	0.000000
5	АбрауДюрсо	5	110.0	6800	0.182796	0.167881	8.824678
...	...	...	...	...	...	...	...
970	ЛУКОЙЛ	320	5575.5	6489151	-0.010208	-0.010260	15.685642
971	ЛУКОЙЛ	321	6236.0	8485835	0.118465	0.111957	15.953909
972	ЛУКОЙЛ	322	6356.0	6099744	0.019243	0.019060	15.623757
973	ЛУКОЙЛ	323	6078.0	10268620	-0.043738	-0.044724	16.144603
974	ЛУКОЙЛ	324	6162.5	4476623	0.013903	0.013807	15.314380

972 rows × 7 columns

Рис. 12. Добавление столбцов новых параметров

Как видно, данные успешно добавились в таблицу. Теперь помимо сведений о названии тикера, дате, цене и объеме продаж, мы также знаем величину доходности, логдоходности и логарифмов объемов.

Перейдем к исследованию подготовленных исходных данных.

1.4. Исследование изменения цен и проведение корреляционного анализа

Для получения первоначального представления о поведении исследуемых величин, как правило, используются средства визуализации, например, графики. Поскольку цены акций различных компаний могут отличаться на несколько порядков, изображение их на одном графике невозможно. Для решения отмеченной проблемы используется масштабирование. Соответственно вместо сравнения реальных цен акций разных компаний, приведем их в сопоставимый вид и сравним относительные

цены (t). Их расчет традиционно осуществляется следующим образом:

$$t = \frac{X - X_{min}}{X_{max} - X_{min}}, \quad (3)$$

где $X_{\max}$ и $X_{\min}$ — соответственно минимальное и максимальное значения цены.

Для построения графиков цен в Python сначала находим максимум и минимум цен соответствующих тикеров. Выполняем данное действие с помощью методов .max() и .min(). После этого, для построения графика, нужно загрузить 2 главные библиотеки, использующиеся для визуализации данных, — matplotlib и seaborn. Одна отвечает за построение "композиций" (plot) — полотен графика и осей, вторая — за сами линии графиков. Так, зная это, можно приступить к построению.

Обозначим новую фигуру за fig (сокр. figure) и внесем необходимые параметры: размер (figsize = (16,9)) и разрешение (dpi) — для лучшего отображения всех элементов. После этого полезно объявить оси — ax.add_subplot(111) добавляет на наше полотно новые оси в первой строке, первом столбце, первым номером. Далее происходит построение нужных данных — относительных цен — по формуле, указанной выше. Эта операция повторяется трижды для каждого тикера и удобнее было бы использовать цикл, однако это не входит в программу подготовки РАР; поэтому можно использовать схему, которая уже была использована при работе с датами, а именно, соединение 3 списков в элемент Series и придание соответствующему столбцу этого элемента. Следующий шаг — построение линии графика. Потребуем от библиотеки seaborn (sns) метод lineplot(линейный график). Введем как аргументы наш data frame, ось x и y, color — цвет, а также факультативные аргументы, например style — название столбца, по отношению к которому каждая линия будет иметь уникальный стиль (в нашей работе, это «Ticket»).

С помощью обращения к полотну — ax называем наш общий график и оси, параметром size указываем размер шрифта. ax.legend() выводит на график легенду (можно указать ее положение — loc (location), а также fontsize — размер шрифта). plt.show() — команда показа графика. Стоит отметить, что мы объявили в самом начале только одно полотно и одни оси, значит все линии будут строится именно на нем. Подробную инструкцию по библиотекам matplotlib и seaborn можно прочитать на официальных сайтах: https://matplotlib.org/cheatsheets/ и https://seaborn.pydata.org/tutorial.html.

Чтобы не нагромождать данные, рекомендуется сразу же удалить столбец с относительными ценами, так как он понадобится только для построения графика.

```
AmbauMax = df[df['Ticker'] == 'АбрауДюрсо']['Price'].max()
AmbauMin = df[df['Ticker'] == 'АбрауДюрсо']['Price'].min()
MagnitMax = df[df['Ticker'] == 'Магнит ao']['Price'].max()
MagnitMin = df[df['Ticker'] == 'Магнит ao']['Price'].min()
LukoilMax = df[df['Ticker'] == 'ЛУКОЙЛ']['Price'].max()
LukoilMin = df[df['Ticker'] == 'ЛУКОЙЛ']['Price'].min()

import matplotlib.pyplot as plt
import seaborn as sns

fig = plt.figure(figsize = (16,9), dpi = 300)
ax = fig.add_subplot(111)
df['Relative Price'] = pd.Series(list((df[df['Ticker'] == 
'АбрауДюрсо']['Price'] - AmbauMin)/(AmbauMax - AmbauMin))\
                                 +list((df[df['Ticker'] == 'Магнит 
ao']['Price'] - MagnitMin)/(MagnitMax - MagnitMin))\
                                 +list((df[df['Ticker'] == 'ЛУКОЙЛ'] 
['Price'] - LukoilMin)/(LukoilMax - LukoilMin)))
sns.lineplot(data=df, x = 'Date', y="Relative Price", style = 
'Ticker', color = 'black')
ax.set_xlabel('Дата', size = 14)
ax.set_ylabel('Цена акции', size = 14)
ax.set_title('Динамика относительных цен акций', size = 20)
ax.legend(fontsize = 14, loc = 'lower center')
plt.show()

df = df.drop(columns = ['Relative Price'])
```

Рис. 13. Код для расчета относительных цен

График на рис. 14 позволяет наглядно представить движение цен различных видов акций. Мы видим, что относительные цены акций АО «Магнита» за исследуемый период имеют тренд к снижению. Для акций Лукойла

ситуация обратная — в целом наблюдается уверенный рост относительных цен с допустимыми колебаниями. Акции Абрау-Дюрсо имеют резкие перепады, значительный рост в кратчайшие временные сроки может быть обусловлен особенностями деятельности предприятия или изменениями на рынке.

Рис. 14. График динамики относительных цен акций

Для дальнейшей работы сформируем датафреймы каждого вида акций. То есть создадим отдельные столбцы показателей для каждого тикера. В качестве критерия выделения каждого датафрейма используем название акции. С технической точки зрения, программа производит поиск наименования акции заданного вида в столбце ‘Ticker’.

```
Abrau = df[df['Ticker'] == 'АбрауДюрсо']
Magnit = df[df['Ticker'] == 'Магнит ао']
Lukoil = df[df['Ticker'] == 'ЛУКОЙЛ']
```

Рис. 15. Создание отдельных таблиц по каждому виду акций

Из них сделаем единую таблицу методом merge, по одной «приклеим» друг к другу, по единому неизменяющемуся столбцу (ключу) — дате. В новой таблице напротив каждой даты будут находиться данные о торгах трех видов акций в этот момент времени.

```
full_demo = Abrau.merge(Magnit, on = 'Date')
full = full_demo.merge(Lukoil, on = 'Date')
display(full)
```

	Ticker_x	Date	Price_x	Volume_x	Yield_x	Log_Yield_x	Log_Volume_x	Ticker_y	Price_y	Volume_y	Yield_y	Log_Yield_y	Log_Volume_y	Tick
0	АбрауДюрсо	1	94.5	2700	0.050000	0.048790	7.901007	Магнит во	11350.0	834475	0.076646	0.073850	13.634658	ЛУКОЙ
1	АбрауДюрсо	2	99.0	800	0.047619	0.046520	6.684612	Магнит во	11750.0	738718	0.035242	0.034635	13.512672	ЛУКОЙ
2	АбрауДюрсо	3	93.0	1000	-0.060606	-0.062520	6.907755	Магнит во	10590.0	1010293	-0.098723	-0.103943	13.825751	ЛУКОЙ
3	АбрауДюрсо	4	93.0	0	0.000000	0.000000	0.000000	Магнит во	11075.0	1040878	0.045798	0.044780	13.855575	ЛУКОЙ
4	АбрауДюрсо	5	110.0	6800	0.182796	0.167681	8.824678	Магнит во	12250.0	1291920	0.106095	0.100836	14.071640	ЛУКОЙ

Рис. 16. Соединение таблиц разных тикеров по дате

Следующим этапом работы является исследование корреляции и рассеивания значений признаков. Для этого следует удалить значения тикеров из полной базы данных и переименовать столбцы. Удаление ненужных столбцов осуществляем уже изученным методом drop(). Изменение названий столбцов методом .columns используется для того, чтобы мы могли правильно идентифицировать, к какой компании относится каждая из исследуемых величин.

```
full = full.drop(columns = ['Ticker_x', 'Ticker_y', 'Ticker'])
full.columns = ['Date', 'Abrau_Price', 'Abrau_Volume', 'Abrau_Yield',
'Abrau_Log_Yield', 'Abrau_Log_Volume',
                'Magnit_Price', 'Magnit_Volume', 'Magnit_Yield',
'Magnit_Log_Yield', 'Magnit_Log_Volume',
                'Lukoil_Price', 'Lukoil_Volume', 'Lukoil_Yield',
'Lukoil_Log_Yield', 'Lukoil_Log_Volume']
```

Рис. 17. Подготовка данных для корреляционного анализа

Одно из самых простых средств исследования корреляции или взаимосвязи между двумя величинами — построение диаграммы рассеивания. Она позволяет очень

визуально оценить направление связи и посмотреть на ее характер — тесная она или слабая. Числовой же мерой измерения этого явления является расчет выборочного коэффициента корреляции. Он измеряется от –1 до 1, то есть от абсолютно (функционально) обратной связи до абсолютно прямой связи.

Так как все же главным объектом исследования работы являются случайные величины логдоходностей акций компаний, рассмотрим, есть ли между ними связь. Эти наработки могут быть использованы для прогнозирования — зная, что доходности как-то зависят друг от друга, можно без труда предугадать (с долей вероятности), что какое-то изменение на фондовой бирже приведет к другому. Главное — не попасть в ловушку «ложной корреляции», когда следствие события пытаются объяснить через его само. Это происходит достаточно часто и является одним из минусов данного показателя.

Для построения диаграммы рассеивания в библиотеке seaborn существует функция scatterplot() («график разброса»). Однако хочется снабдить, как уже было сказано, данный график информацией о выборочном коэффициенте корреляции. Добавить эту информацию, как и любой элемент на композиции можно очень просто — через обращение к ax — ax.text(), где можно указать координаты, где будет располагаться надпись (x, y), а также сам текст. Для его введения используем конструкцию '...%.nf:' % text. Это достаточно распространенная конструкция, которая позволяет сделать некоторое неизменяемое начало (вместо ...), добавить любое число (в месте постановки % в кавычках), при этом округлив его до n знаков. После кавычек ставится % и записывается само число. Его можно вычислить с помощью встроенной в pandas методологии .corr(). При этом метод выводит корреляционную матрицу — то есть, все возможные комбинации корреляции между двумя величинами. Для двух величин таких комбинация будет, соответственно, 4, матрица будет 2*2.

```
1 full[['Abrau_Log_Yield', 'Magnit_Log_Yield']].corr()
```

Out[66]:

	Abrau_Log_Yield	Magnit_Log_Yield
Abrau_Log_Yield	1.000000	0.083339
Magnit_Log_Yield	0.083339	1.000000

Рис. 18. Построение корреляционной матрицы

Очевидно, что оттого, какую корреляцию рассматривать — Абрау от Магнита или Магнита от Абрау, ничего не зависит (отсюда и появляется «ложность» корреляции). Поэтому для использования подойдет как значение в ячейке (0,1), так и (1,0). «Достать» ячейку по ее номеру в pandas из DataFrame позволяет метод .iloc.

Зная это, можно приступить к построению диаграмм рассеивания.

Оценка коэффициентов корреляции важна для снижения риска инвестиционного портфеля. Известно, что риском портфеля ценных бумаг, например, акций, называется дисперсия доходности портфеля (***r***). Пусть в исследуемый портфель входят ***n*** ценных бумаг, тогда дисперсия портфеля определяется формулой:

$$\sigma^2 = \sum_{i=1}^{n} \theta_i \sigma_i^2 + \sum_{i=1}^{n} \sum_{j=1}^{n} \theta_i \theta_j k_{ij} \sigma_i \sigma_j ;$$

где $\sigma_i^2 = \sigma^2(r_i)$ – дисперсия доходности *i*-го актива;

k_{ij} – коэффициент корреляции между доходностями *i*-го и *j*-го активов;

θ_i — удельный вес в этом портфеле *i*-го актива;

$$\theta_i = \frac{x_i}{X}$$

x_i — стоимость *i*-го актива, X — стоимость портфеля.

Из приведенной формулы следует, что выбор активов, коэффициенты корреляции которых, имеют наименьшие

значения обеспечивает минимальный риск портфеля. По этой причине задача оценки коэффициентов корреляции весьма актуальна.

```
fig = plt.figure(figsize = (5,5), dpi = 300)
ax = fig.add_subplot(111)
sns.scatterplot(data = full, x = 'Abrau_Log_Yield', y =
'Magnit_Log_Yield', color = 'black')
ax.set_xlabel('Abrau_Log_Yield', size = 14)
ax.set_ylabel('Magnit_Log_Yield', size = 14)
ax.set_title('Диграмма рассеивания между Abrau_Log_Yield и
Magnit_Log_Yield', size = 20)
ax.text(0.6, 0.15, 'r=%.2f' % full[['Abrau_Log_Yield',
'Magnit_Log_Yield']].corr().iloc[0,1], fontsize = 14)
plt.show()

fig = plt.figure(figsize = (5,5), dpi = 300)
ax = fig.add_subplot(111)
sns.scatterplot(data = full, x = 'Lukoil_Log_Yield', y =
'Magnit_Log_Yield', color = 'black')
ax.set_xlabel('Lukoil_Log_Yield', size = 14)
ax.set_ylabel('Magnit_Log_Yield', size = 14)
ax.set_title('Диграмма рассеивания между Lukoil_Log_Yield и
Magnit_Log_Yield', size = 20)
ax.text(-0.2, 0.15, 'r=%.2f' % full[['Lukoil_Log_Yield',
'Magnit_Log_Yield']].corr().iloc[0,1], fontsize = 14)
plt.show()

fig = plt.figure(figsize = (5,5), dpi = 300)
ax = fig.add_subplot(111)
sns.scatterplot(data = full, x = 'Abrau_Log_Yield', y =
'Lukoil_Log_Yield', color = 'black')
ax.set_xlabel('Abrau_Log_Yield', size = 14)
ax.set_ylabel('Lukoil_Log_Yield', size = 14)
ax.set_title('Диграмма рассеивания между Abrau_Log_Yield и
Lukoil_Log_Yield', size = 20)
ax.text(0.5, -0.2, 'r=%.2f' % full[['Abrau_Log_Yield',
'Lukoil_Log_Yield']].corr().iloc[0,1], fontsize = 14)
plt.show()
```

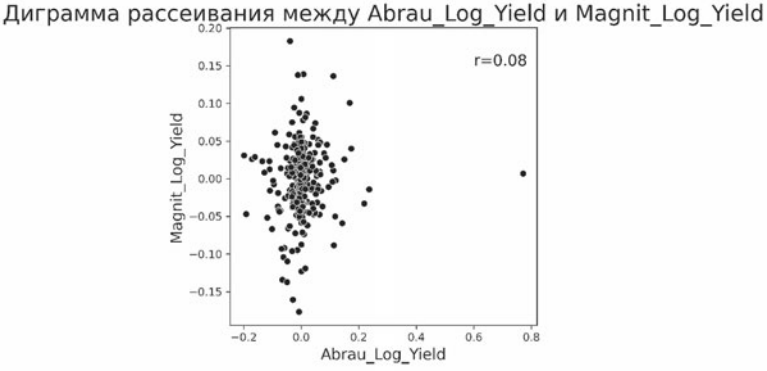

Рис. 19 (начало). Диаграммы рассеивания данных

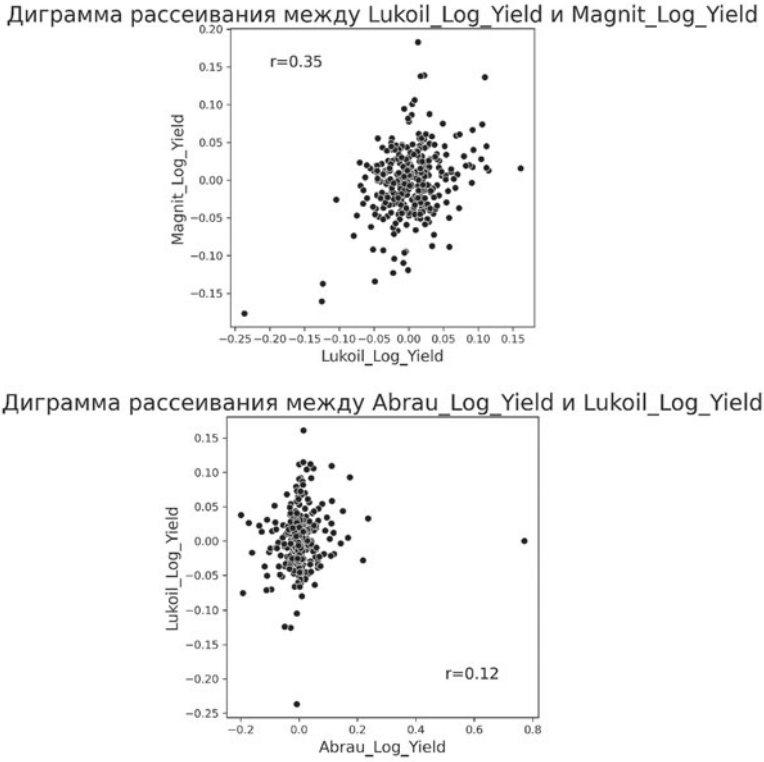

Рис. 19 (окончание). Диаграммы рассеивания данных

Конечно, удобнее это сделать через циклы, ну или на худой конец функции. Однако, циклы не входят в базовые навыки pandas, а скорее являются методом оптимизации рутинных задач. Функции же будут рассмотрены позже в работе и будет хороший повод вернуться к этому материалу для попытки «оптимизировать» код до одной функции с ее множественным применением.

Исходя из увиденного, можно сказать однозначно, что логдоходности акций Абрау-Дюрсо совершенно не подвержены колебаниям других ценных бумаг (коэффициент корреляции низок по модулю, на графике нет

тенденции — все значения сконцентрированы около нуля). Но можно ли сделать вывод что доходности акций Магнита зависят от доходностей акций Лукойла? Конечно, нет. Мало того, что эти компании из разных секторов экономики, они совершенно разные по структуре капитала, масштабу, влиянию на национальную экономику. Это и есть пример ложной корреляции, потому что совершенно понятно, что доходности этих компаний просто одинаково реагируют на рыночные тенденции (например, индекс Мосбиржи или РТС, для этого даже существует специальный показатель — бета-коэффициент — уровень специфического риска). Поэтому сообщать об открытии в своем исследовании — «котировки нефтяного гиганта зависит от FMCG-ритейлера» (или наоборот) — точно не имеет смысла.

1.5. Исследование логарифмических доходностей акций

Для начала построим графики логдоходностей методом lineplot. Этот график рисует линию, которая представляет собой развитие непрерывных или категориальных данных. Исследование логарифмических доходностей позволяет получить прогноз или исследовать какие-либо вероятности движения цен активов. Более того, если логарифмическая доходность распределена нормально, то распределение никогда не приведет к отрицательной цене.

```
fig = plt.figure(figsize = (16,9), dpi = 300)
ax = fig.add_subplot(111)
sns.lineplot(data=df, x = 'Date', y="Log_Yield", color = 'black',
style = 'Ticker')
ax.set_xlabel('Дата', size = 14)
ax.set_ylabel('Логдоходность акции', size = 14)
ax.set_title('Динамика логдоходностей акций', size = 20)
ax.legend(fontsize = 14, loc = 'upper left')
plt.show()
```

Рис. 20. Код для построения графиков логдоходностей

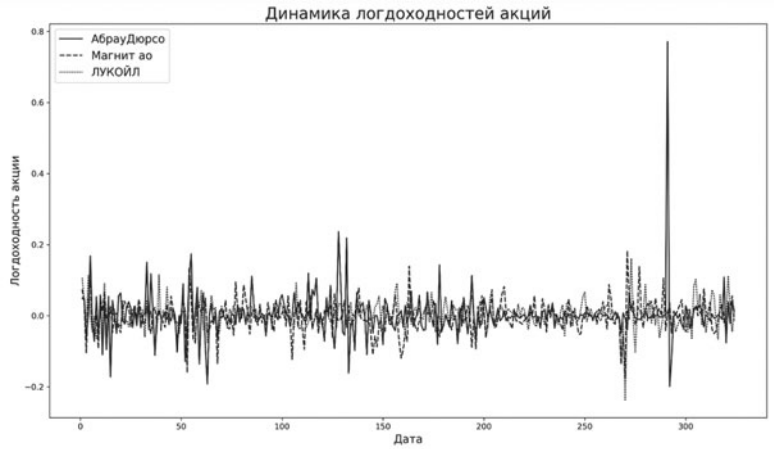

Рис. 21. Динамика логдоходностей акций

Полученные результаты позволяют нам оценить величину и амплитуду изменений логдоходностей каждого тикера. Как мы видим, наибольшие скачки наблюдаются у компании Абрау-Дюрсо. Эти данные требуют большего внимания на последующих этапах анализа акций. Кроме этого, тенденции (зависимости от времени), не наблюдается ни у одного тикера — колебания совершенно непредсказуемы, что еще раз подтверждает тезис о случайности рассматриваемых величин.

1.6. Удаление выбросов логдоходностей акций

Приступим к исследованию и удалению выбросов. Как известно, выбросы являются аномальными значениями, резко отличающимися от других наблюдаемых. Именно поэтому необходимо провести очистку данных от выбросов. Для начала выведем на экран основные статистические данные, такие как процентили, среднее значение, стандартное отклонение и прочие. С этим нам поможет справится атрибут describe().

```
1 full.describe()
```

Out[72]:

	Date	Abrau_Price	Abrau_Volume	Abrau_Yield	Abrau_Log_Yield	Abrau_Log_Volume	Magnit_Price	Magnit_Volume	Magnit_Yield
count	324.00000	324.000000	3.240000e+02	324.000000	324.000000	324.000000	324.000000	3.240000e+02	324.000000
mean	162.50000	132.554012	6.713654e+04	0.005200	0.002731	8.307733	7053.645062	1.362252e+06	-0.001120
std	93.67497	35.779113	6.792588e+05	0.082099	0.065943	2.042701	3196.478644	1.089781e+06	0.044087
min	1.00000	79.000000	0.000000e+00	-0.180357	-0.198887	0.000000	2501.000000	2.019980e+05	-0.161863
25%	81.75000	102.000000	1.300000e+03	-0.014670	-0.014779	7.170120	3860.125000	5.975045e+05	-0.025670
50%	162.50000	136.000000	4.500000e+03	0.000000	0.000000	8.411833	5461.750000	1.010439e+06	0.000000
75%	243.25000	146.000000	1.247000e+04	0.014248	0.014147	9.431080	10350.250000	1.864268e+06	0.022054
max	324.00000	280.000000	1.192961e+07	1.162162	0.771109	16.294534	12396.000000	7.285249e+06	0.200520

Рис. 22. Описательная статистика

С помощью описательной статистики мы можем изучить разрыв между средним и медианным значением (эта информация дает первичное представление и разбросе данных), а также проанализировать, каких значений достигает каждый показатель в точках минимума и максимума.

Кроме того, поближе познакомиться с распределением данных помогут гистограммы — специальные частотные графики, которые демонстрируют, сколько раз в выборке попадается то или иное значение. Как и в случае с диаграммой рассеивания, только лишь графика может быть недостаточно, нужно также рассчитать асимметрию данных и их эксцесс.

Нормальное колокообразное распределение (Гауссово распределение) абсолютно симметрично, унимодально, то есть гистограмма его будет иметь абсолютно зеркальный вид, асимметрия будет равна 0, а выпуклость будет лишь одна — в точке наибольшего «попадания» значений — среднее, мода и медиана (эти значения у нормального распределения равны). Для вычисления асимметрии и эксцесса загрузим необходимую для этого библиотеку — статистический пакет Python — scipy.stats. Функция skew() возвращает ассиметрию набора чисел, функция kurtosis() — эксцесс.

```
import scipy.stats as st

fig = plt.figure(figsize = (7, 5), dpi = 300)
ax = fig.add_subplot(111)
sns.histplot(full['Abrau_Log_Yield'], color = 'black')
ax.set_title('Гистограмма логдоходностей акций - Abrau_Log_Yield',
size = 20)
ax.text(0.4, 60, 'As=%.2f, Ex=%.2f' %
(st.skew(full['Abrau_Log_Yield']),
st.kurtosis(full['Abrau_Log_Yield'])), fontsize = 14)
plt.show()
```

Рис. 23. Код для построения гистограммы

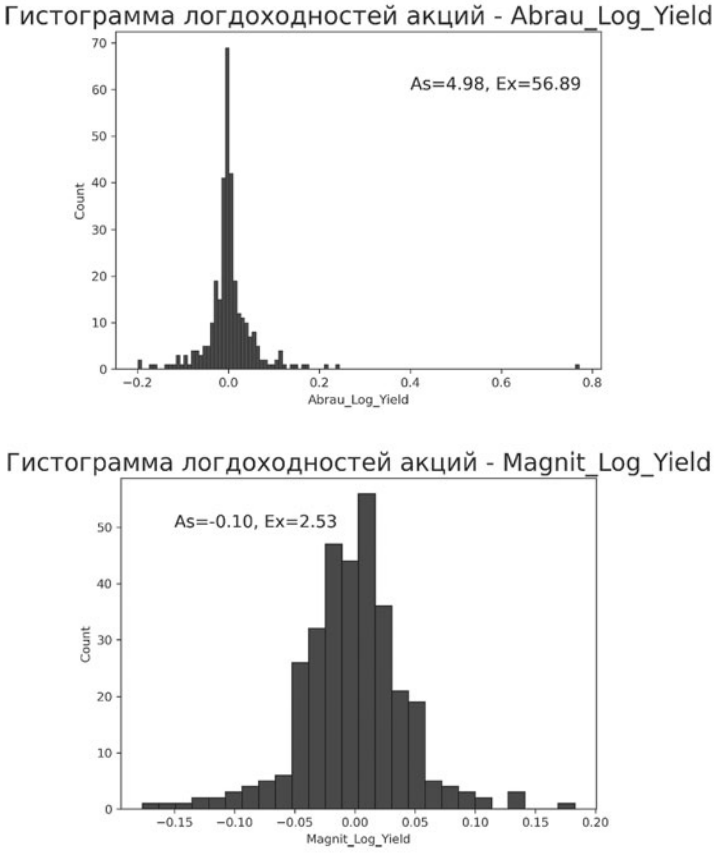

Рис. 24 (начало). Гистограммы логдоходностей

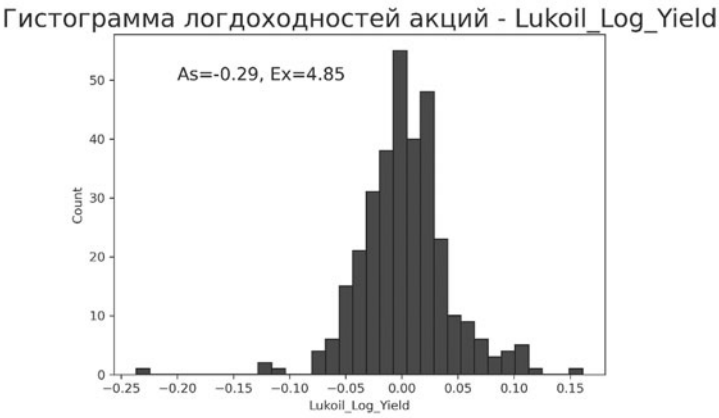

Рис. 24 (окончание). Гистограммы логдоходностей

Человек, начинающий изучать математическую статистику, может захотеть бросить заниматься акциями Абрау-Дюрсо — очевидно, что распределение данных далеко от нормального (асимметрия достигает практически 5!). А если учесть, что многие, особенно параметрические статистические тесты проводятся исключительно с данными, распределенными нормально, исследование усложняется. Кроме того, очевидно, что средняя логдоходность акций равна 0 — и это докажет любой тест — столбец над значением «0» существенно выше, чем все остальные, кроме того, данные стремятся к этому значению (столбцы вокруг нуля много больше, чем в других местах гистограммы). Видны и значительные выбросы — маленькие «пеньки» — в районе 0,8, –0,2 и другие. Их стоит обязательно удалить, потому что они существенно искажают данные (не будь «пенька» в районе 0,8, гистограмма расположилась бы более симметрично, так как сместилась правее, из-за чего асимметрия уменьшилась).

Другие гистограммы выглядят намного более подходящими нормальному закону — логдоходности акций Магнита близки к эталонным для нормального распреде-

ления значениям (As=0, Ex=0), но выбросы также присутствуют, однако существенного влияния, как минимум на гистограмму распределения, они не оказывают. Также близкое к нормальному имеет и распределение данных логдоходностей у Лукойла, однако из-за вогнутости около нуля, эксцесс заметно вырос, что несвойственно нормальному распределению. Выбросы уже более существенны, чем у «Магнита», и чтобы их устранить, удобнее работать с графиком boxplot.

Диаграмма размаха («ящик с усами») является удобным способом визуализации данных через квартили. С помощью диаграммы размаха можно изучить степень разброса данных и выделить выбросы. Для построения диаграмм размаха в Python воспользуемся функцией boxplot(). В качестве аргументов передадим переменные для визуализации: тикеры и логарифмические доходности акций.

```
fig = plt.figure(figsize = (16,9), dpi = 300)
ax = fig.add_subplot(111)
sns.boxplot(x="Ticker", y="Log_Yield", data=df, color = 'black')
ax.set_xlabel('Тикер', size = 14)
ax.set_ylabel('Логдоходность акции', size = 14)
ax.set_title('Рассеивание логдоходностей по тикерам', size = 20)
plt.show()
```

Рис. 25. Код для построения диаграмм размаха логдоходностей

Результаты свидетельствуют о том, что по каждому виду акций есть выбросы — это точки, располагающиеся за приделами верхней и нижней границы нормы. Наибольшее количество и выбросов и их разброс наблюдается по данным Абрау-Дюрсо.

Удалять выбросы следует по правилу полутора межквартильных размахов. То есть выбросами называются элементы вариационного ряда, не принадлежащие отрезку

$$\left[x_{0,25} - 1{,}5 \cdot IQR, x_{0,75} + 1{,}5 \cdot IQR\right], \qquad (4)$$

где $x_{0,25}$ и $x_{0,75}$ — первая и третья квартили соответственно, а IQR — межквартильный размах, характери-

зующий разброс значений признака. Данный показатель рассчитывается по следующей формуле:

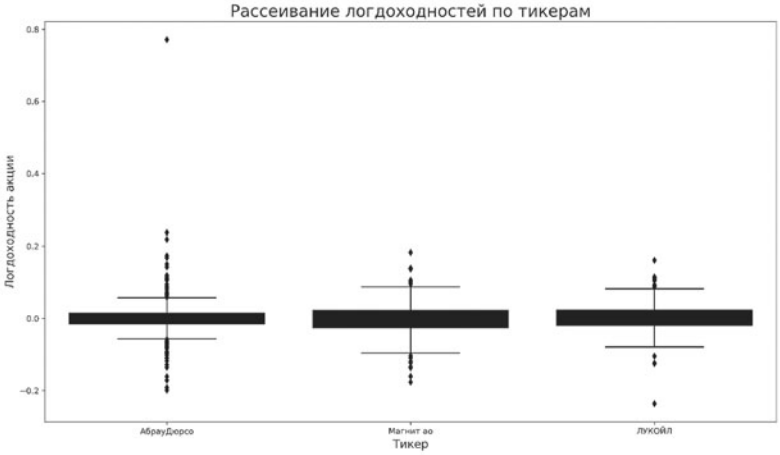

Рис. 26. Рассеивание логдоходностей по тикерам

$$IQR = x_{0,75} - x_{0,25}. \tag{5}$$

Удалять выбросы следует в каждом из имеющихся фреймов данных. Сначала определим межквартильный размах и границы нормы

$$x_0 = x_{0,25} - 1,5 \cdot IQR; \tag{6}$$

$$x_1 = x_{0,75} + 1,5 \cdot IQR. \tag{7}$$

Найдем квартили методом quantile(), где укажем, какие доли нам необходимы (0,25, 0,75). Для удобства транспонируем таблицу методом transpose(). Рассчитаем межквартильный размах, а также верхнюю и нижнюю границы нормы.

```
quartiles = full[['Abrau_Log_Yield', 'Magnit_Log_Yield',
 'Lukoil_Log_Yield']].quantile([0.25, 0.75])
quartiles = quartiles.transpose()
quartiles['IQR'] = quartiles[0.75] - quartiles[0.25]
quartiles['Min'] = quartiles[0.25] - 1.5*quartiles['IQR']
quartiles['Max'] = quartiles[0.75] + 1.5*quartiles['IQR']
quartiles
```

Out[84]:

	0.25	0.75	IQR	Min	Max
Abrau_Log_Yield	-0.014779	0.014147	0.028926	-0.058168	0.057536
Magnit_Log_Yield	-0.026005	0.021814	0.047819	-0.097734	0.093543
Lukoil_Log_Yield	-0.019202	0.022416	0.041618	-0.081630	0.084843

Рис. 27. Расчет числовых характеристик распределения логдоходностей

С помощью метода query() сделаем срез данных — оставим только те данные, у которых логарифмические доходности не выходят за границы нормы. В методе query очень важен правильный синтаксис, так как запрос заключается в кавычки.

```
clear_full = full.query('Abrau_Log_Yield ⩽ 0.057536 & Abrau_Log_Yield
 ⩾ -0.058168 \
& Magnit_Log_Yield ⩽ 0.093543 & Magnit_Log_Yield ⩾ -0.097734 & \
Lukoil_Log_Yield ⩽ 0.084843 & Lukoil_Log_Yield ⩾ -0.081630')
```

Рис. 28. Срез данных, очищенных от выбросов

Проведем проверку данных, очищенных от выбросов. Выведем основную информацию при помощи функции info(). Из 324 строк осталось всего лишь 249 — это результат очистки данных.

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 249 entries, 1 to 323
Data columns (total 16 columns):
 #   Column              Non-Null Count  Dtype
---  ------              --------------  -----
 0   Date                249 non-null    int64
 1   Abrau_Price         249 non-null    float64
 2   Abrau_Volume        249 non-null    int64
 3   Abrau_Yield         249 non-null    float64
 4   Abrau_Log_Yield     249 non-null    float64
 5   Abrau_Log_Volume    249 non-null    float64
 6   Magnit_Price        249 non-null    float64
 7   Magnit_Volume       249 non-null    int64
 8   Magnit_Yield        249 non-null    float64
 9   Magnit_Log_Yield    249 non-null    float64
 10  Magnit_Log_Volume   249 non-null    float64
 11  Lukoil_Price        249 non-null    float64
 12  Lukoil_Volume       249 non-null    int64
 13  Lukoil_Yield        249 non-null    float64
 14  Lukoil_Log_Yield    249 non-null    float64
 15  Lukoil_Log_Volume   249 non-null    float64
dtypes: float64(12), int64(4)
memory usage: 33.1 KB
```

Рис. 29. Срез данных, очищенных от выбросов

Теперь очень полезно будет рассмотреть, как изменилось распределение данных после удаления выбросов. Код для этого уже был рассмотрен раннее, единственное, что нужно изменить — используемый Data Frame (с full на clear_full) и положение (координаты) текста с расчетом асимметрии и эксцесса. В целом задача достаточно тривиальная, приступим к анализу результатов.

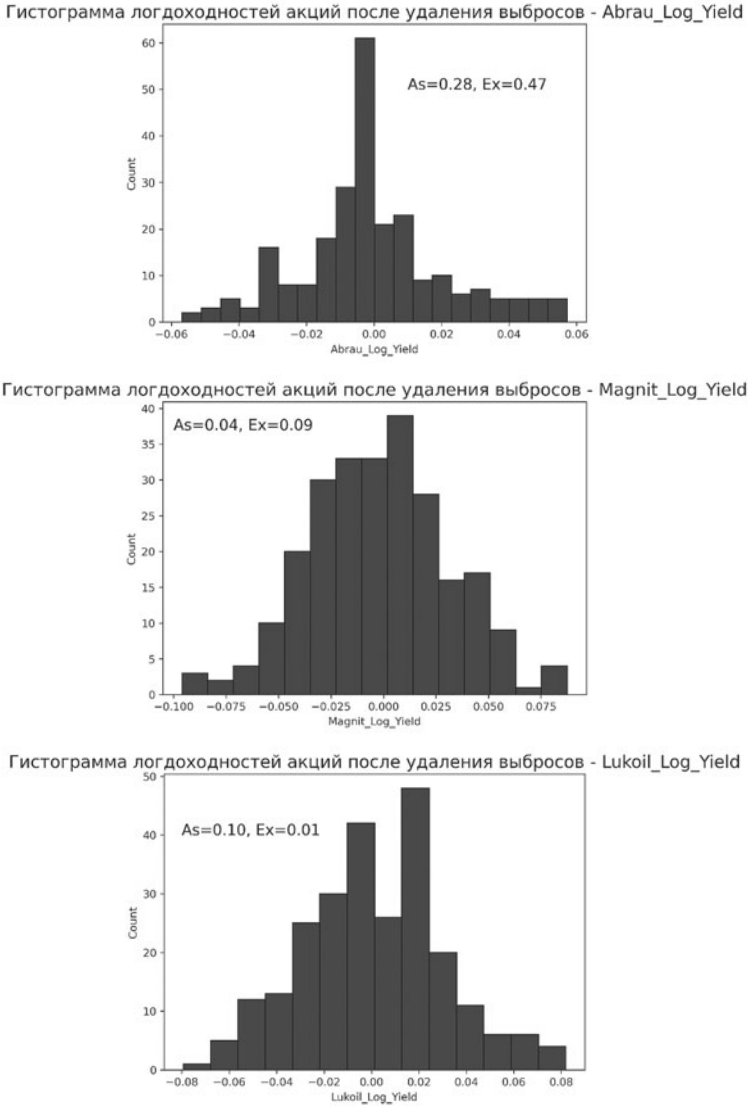

Рис. 30. Гистограммы логдоходностей после удаления выбросов

Наверняка теперь начинающий статистик изменит своё мнение насчет того, нормально ли распределены логдоходности «Абрау-Дюрсо», по крайней мере задумается о проведении тестов на нормальность — распределение заметно «улучшилось», по сравнению с первоначальным набором статистики. Однако сохранился высокий эксцесс, который сошел на нет у остальных распределений (так же как и асимметрия, которая у «Абрау-Дюрсо» также завышена, относительно нормального закона).

Для того, чтобы окончательно понять, какому закону распределения принадлежат исследуемые в работе выборки, необходимо провести строгие статистические тесты. Каждый из тестов, несомненно, имеет свою специфику, и полезно для более надежного принятия или отклонения той или иной гипотезы проводить сразу несколько тестов.

1.7. Проверка гипотез о нормальности логдоходностей для каждой компании

Прежде, чем переходить к столь ответственной работе, как проведение статистических тестов, стоит в принципе определить, что под этим понимается.

Процедура статистических тестов производится для того, чтобы на основании имеющейся выборки сделать некоторые выводы о том, как ведут себя данные, относящееся к этой же величине, в генеральной совокупности (которая, к слову, почти всегда теоретическая). Например, проводится исследование об успеваемости студентов Финансового университета, где нужно вычислить средний балл. Сейчас, с приходом информационных технологий в образовательную среду, эта задача существенно упростилась, однако раньше собрать с каждого студента средний балл, занести его в какую-то таблицу (вручную) и все грамотно высчитать, было задачей не из легких. Когда нет возможности исследовать всю генеральную совокупность (а выводы делать нужно именно по ней)

на помощь приходит выборочный метод и наука, которая его сопровождает — математическая статистика.

Так, статистик выбрал группу (или несколько групп) и собрал информацию о баллах студентов. Посчитав некоторые параметры (отсюда — параметрические тесты) — выборочное среднее (математическое ожидание), выборочную дисперсию, выборочное стандартное отклонение, статистик формирует гипотезу, например, о том, что средний балл студентов Финансового университета равен 80 баллам, то есть на основании части пытается сделать вывод об общем. Это и называется проверкой статистической гипотезы.

В оппозицию к ней обычно ставится альтернативная гипотеза, которая гласит, как правило, противоположное — балл студентов Финансового университета не равен 80 баллам. В дальнейшем выбирается необходимая процедура (в данном случае, например, тест Стьюдента) и рассчитывается, по формуле статистика теста. Далее выбирается уровень значимости-альфа (очень упрощенно — вероятность отвергнуть правильную гипотезу, то есть вероятность фатально ошибиться при проведении теста), который для нестрогих исследований (медицина, биология, физика, химия) обычно принимается за 0,05.

В специальных таблицах, соответствующих методологии теста, содержатся критические значения — те максимальные значения статистики при заданном объеме выборки и уровне значимости, при которых еще можно принять первоначальную — нулевую гипотезу теста. Если такое значение превышено значением рассчитанной статистики, то нулевую гипотезу при заданном уровне значимости необходимо отвергнуть в пользу альтернативной.

Современные программные продукты обычно рассчитывают не только значение статистики, но и p-value — уже готовую вероятность принятия нулевой гипотезы (при условии, что она верна). Процедура становится еще проще — нужно просто сравнить p-value с уровнем значи-

мости, и если он окажется больше, то нулевую гипотезу можно смело принять.

Зная это, можно переходить к самой проверке статистических гипотез.

1.7.1. Проверка гипотезы по критерию Пирсона

Сначала проверим данные на нормальность по критерию согласия Пирсона. Данный критерий был сформулирован в 1900 году. Несмотря на то, что он является «хрестоматийным», критерий широко применяется для решения прикладных статистических задач. Распределение хи-квадрат активно используется в непараметрической статистике, позволяет выявлять согласие между экспериментальными данными и теоретическими законами распределения. Сама идея критерия основана на том, что мы измеряем разницу между наблюдаемой частотой попадания в какой-либо интервал и теоретической вероятностью попадания в этот же интервал[3].

Критерий согласия Пирсона (χ^2) применяют для проверки гипотезы о соответствии эмпирического распределения предполагаемому теоретическому распределению $F(x)$ при относительно большом объеме выборки ($n \geqslant 100$). Критерий применим для любых видов функции $F(x)$, даже при неизвестных значениях их параметров, что обычно имеет место при анализе результатов механических испытаний. В этом заключается его универсальность.

Критерий согласия χ^2 основан на сравнении эмпирической гистограммы распределения случайной величины с ее теоретической плотностью. Диапазон изменения экспериментальных данных разбивается на l интервалов и после этого происходит подсчет статистики. При этом, следует отметить, что на мощность статистического пока-

---

[3] *Кобзарь А.И.* Прикладная математическая статистика. Для инженеров и научных работников: Учебное пособие. 204 с.

зателя χ^2 сильное влияние оказывает число интервалов разбиения гистограммы — l интервалов — и выбор длины интервалов, на которые разбивается исследуемый диапазон.

Использование критерия χ^2 предусматривает разбиение размаха варьирования выборки на интервалы и определения числа наблюдений (частоты) n_j для каждого из l интервалов. Для удобства оценок параметров распределения интервалы выбирают одинаковой длины (h). Число интервалов зависит от объема выборки. Для выбора количества интервалов часто пользуются формулой Стерджеса

$$l \cong 1 + \sqrt{2} \ln n, \tag{8}$$

где n — объём выборки.

Статистика критерия Пирсона имеет вид

$$\chi^2 = \sum_{j=1}^{l} \frac{\left(n_j - np_j\right)^2}{np_j}, \tag{9}$$

где p_j — вероятность попадания изучаемой случайной величины в j-й интервал, вычисляемая в соответствии с гипотетической функцией распределения $F(x)$:

$$p_j = F\left(x_{j+1}\right) - F\left(x_j\right). \tag{10}$$

Согласно теореме Пирсона при большом объеме выборки распределение случайной величины χ^2 можно считать распределением хи-квадрат с $l-1$ степенью свободы.

Нулевую гипотезу о соответствии выборочного распределения теоретическому закону $F(x)$ проверяют путем сравнения вычисленной по формуле (9) величины с критическим значением χ^2 найденным по таблице значений распределения χ^2 для уровня значимости α и числа степеней свободы $k = l_1 - m - 1$. Здесь l_1 — число интервалов после объединения; m — число параметров, оцениваемых по рассматриваемой выборке. Если выполняется

неравенство $\chi^2 \leqslant \chi^2_{kp}$, то нулевую гипотезу не отвергают. При несоблюдении указанного неравенства принимают альтернативную гипотезу о принадлежности выборки неизвестному распределению.

Зная уровень значимости α и число степеней свободы k, можно найти критическое значение статистики χ^2 и сравнить его с эмпирическим. Для расчета критерия Пирсона в Python необходимо задать функцию def pirson(), которая будет принимать значения логарифмических доходностей по каждому типу акций и уровень значимости α. Функция st.normaltest возвращает соответствующее значение статистики и p-значение.

```
def pirson(row, alpha):
    stat, p = st.normaltest(clear_full[row])
    print('Statistics=%.3f, p-value=%.3f' % (stat, p))
    if p > alpha:
        print('Принять гипотезу о нормальности по критерию Пирсона
для', row, ', a =', alpha)
    else:
        print('Отклонить гипотезу о нормальности по критерию Пирсона
для', row, ', a =', alpha)
```

```
Statistics=5.556, p-value=0.062
Принять гипотезу о нормальности по критерию Пирсона для Abrau_Log_Yield , a = 0.05
Statistics=0.335, p-value=0.846
Принять гипотезу о нормальности по критерию Пирсона для Magnit_Log_Yield , a = 0.05
Statistics=0.465, p-value=0.792
Принять гипотезу о нормальности по критерию Пирсона для Lukoil_Log_Yield , a = 0.05
```

Рис. 31. Проверка гипотезы о нормальном распределении с помощью критерия согласия Пирсона

Согласно проведенному тесту, можно сделать вывод о том, что данные о логарифмической доходности по каждому виду акций распределены нормально по критерию Пирсона на уровне значимости 0.05.

1.7.2. Проверка гипотезы по критерию Шапиро-Уилка

Критерий Шапиро-Уилка является параметрическим специальным критерием, использующимся для проверки гипотезы о нормальном распределении. В классическом виде этот критерий надёжен при $8 \leq n \leq 50$ (существует

модифицированный критерий Шапиро-Уилка, применимый при значениях n до 2000). Изучение мощности критерия Шапиро-Уилка показало, что это один из наиболее эффективных критериев проверки нормальности распределения случайных величин. Если выборка нормальна, можно далее применять мощные параметрические критерии, например, критерий Фишера. Критерий Шапиро-Уилка основан на оптимальной линейной несмещённой оценке дисперсии к её обычной оценке методом максимального правдоподобия. Несмотря на то, что критерий Шапиро-Уилка является очень сильным показателем для проверки нормальности, к сожалению, он имеет ограниченную применимость. При больших значениях выборки таблицы коэффициентов становятся неудобными[4]. Однако современные вычислительные мощности и возможности библиотеки scipy.stats позволяют использовать критерий на практически неограниченном диапазоне.

Статистику критерия можно рассчитать по формуле:

$$w = \frac{\sum_{i=1}^{n} a_{n-i-1}\left(x_{n-i-1} - x_i\right)^2}{\sum_{i=1}^{n}\left(x_i - \overline{x}\right)^2}, \qquad (11)$$

где i — номер элемента в вариационном ряду, a_{n-i-1} — табличные параметры, n — объем выборки, $\overline{x}$ — среднее значение.

Учитывая, что параметры a_{n-i-1} — табличные величины, тяжело представить расчет данной статистики вручную. Но с этим отлично справляется библиотека scipy.stats.

Проведем тест Шапиро-Уилка на нормальность распределения выборки исходных данных по торгам изучаемых тикеров. Создадим функцию def shapiro, также

---

[4] *Кобзарь А.И.* Прикладная математическая статистика. Для инженеров и научных работников: Учебное пособие. 238 с.

принимающую значения логарифмических доходностей по каждому типу акций и выберем уровень значимости α. Функция st.shapiro возвращает значение вычисленной статистики и *p*-значение. Пусть уровень значимости равен 0.05. При p-значении меньше 0.05 следует отклонить нулевую гипотезу.

```
def shapiro(row, alpha):
    stat, p = st.shapiro(clear_full[row]) # тест Шапиро-Уилк
    print('Statistics=%.3f, p-value=%.3f' % (stat, p))
    if p > alpha:
        print('Принять гипотезу о нормальности по критерию Шапиро-
Уилка для', row, ', a =', alpha)
    else:
        print('Отклонить гипотезу о нормальности по критерию Шапиро-
Уилка для', row, ', a =', alpha)
```

```
Statistics=0.970, p-value=0.000
Отклонить гипотезу о нормальности по критерию Шапиро-Уилка для Abrau_Log_Yield , a = 0.05
Statistics=0.997, p-value=0.882
Принять гипотезу о нормальности по критерию Шапиро-Уилка для Magnit_Log_Yield , a = 0.05
Statistics=0.994, p-value=0.415
Принять гипотезу о нормальности по критерию Шапиро-Уилка для Lukoil_Log_Yield , a = 0.05
```

Рис. 32. Проверка гипотезы о нормальном распределении с помощью критерия Шапиро-Уилка

Результаты теста Шапиро-Уилка показали, что данные логарифмической доходности компаний Магнит и Лукойл распределены нормально, в то время как логдоходность Абрау-Дюрсо не имеет нормального распределения на уровне значимости 0,05. Более мощный тест, чем тест Пирсона, подтвердил высказанное раннее (еще при анализе гистограмм распределений) допущение о невозможности нормальности распределения логдоходностей тикера «Абрау-Дюрсо». Для того, чтобы подтвердить это предположение, следует провести также и непараметрический тест (тот, который опирается не на какие-то параметры распределения — среднее, дисперсию или стандартное отклонение, а на сами значения выборки и из них формирует значение статистики), первейшим из которых для данной задачи является тест Колмогова-Смирнова.

1.7.3. Проверка гипотезы по критерию Колмогорова-Смирнова

Наиболее известные критерий согласия — критерий Пирсона гибок, легко используется, но имеет элемент произвола в выборе границ группирования экспериментальных данных. Критерий Колмогорова-Смирнова свободен от этого недостатка и имеет хорошую асимптотическую мощность по сравнению с альтернативами, определенными в терминах расстояния между функциями распределения[5]. Данный критерий хорошо применяется на выборках малого объёма. Стоит отметить, что критерий Колмогорова-Смирнова используется для проверки гипотезы о принадлежности наблюдаемой выборки нормальному закону, параметры которого оцениваются по этой самой выборке методом максимального правдоподобия.

Для надежности проверки гипотезы о принадлежности наблюдаемых выборок нормальному закону, используем критерий согласия Колмогорова-Смирнова в нашем исследовании. Как было отмечено, данный непараметрический критерий хорошо подходит для проверок малых выборок[6]. Он позволяет провести качественную проверку в отношении гипотезы о принадлежности рассматриваемой выборки некоторому закону распределения. Расчет данного критерия осуществляется следующим образом:

$$D_n = sup\left|F_n(x) - F_0(x)\right|, \qquad (12)$$

где sup$S$ — точная верхняя грань множества значений функции s,

$F_n(x)$ — функция распределения исследуемой совокупности,

$F_0(x)$ — функция нормального распределения.

---

[5] *Орлов А.И.* Непараметрические критерии согласия Колмогорова, Смирнова, омега-квадрат и ошибки при их применении. 7 с.

[6] *Кобзарь А.И.* Прикладная математическая статистика. Для инженеров и научных работников: Учебное пособие. 224 с.

Если D_n статистика Колмогорова-Смирнова значима ($p < 0{,}05$), то гипотеза о том, что соответствующее распределение нормально, должна быть отвергнута.

Для расчета критерия Колмогорова-Смирнова в Python создадим функцию def ks(), которая будет принимать значения логарифмических доходностей по каждому типу акций и заданному уровню значимости значимости α. Метод st.kstest выполняет проверку распределения наблюдаемой случайной величины по отношению к нормальному распределению.

```
def ks(row, alpha):
    stat, p = st.kstest(clear_full[row], 'norm')
    print('D=%.3f, p-value=%.3f' % (stat, p))
    if p > alpha:
        print('Принять гипотезу о нормальности по критерию Колмогорова-Смирнова для', row, ', a =', alpha)
    else:
        print('Отклонить гипотезу о нормальности по критерию Колмогорова-Смирнова для', row, ', a =', alpha)
ks('Abrau_Log_Yield', 0.05)
ks('Magnit_Log_Yield', 0.05)
ks('Lukoil_Log_Yield', 0.05)
```

```
D=0.477, p-value=0.000
Отклонить гипотезу о нормальности по критерию Колмогорова-Смирнова для Abrau_Log_Yield , a = 0.05
D=0.465, p-value=0.000
Отклонить гипотезу о нормальности по критерию Колмогорова-Смирнова для Magnit_Log_Yield , a = 0.05
D=0.470, p-value=0.000
Отклонить гипотезу о нормальности по критерию Колмогорова-Смирнова для Lukoil_Log_Yield , a = 0.05
```

Рис. 33. Проверка гипотезы о нормальном распределении с помощью критерия Колмогорова-Смирнова

Данная проверка показала, что логарифмические доходности по каждому виду акций отличаются от нормального распределения согласно критерию Колмогорова-Смирнова.

Просуммировав все вышесказанное и приняв во внимание результаты оценки распределения каждого вида тикеров по каждому использованному критерию, можно сделать вывод о том, что тикер 'АбрауДюрсо' имеет распределение, отличное от нормального.

Дальнейшие тесты, касающиеся работы с нормальным распределением, будем проводить с оставшимися тикерами: акциями Магнита и Лукойла.

1.8. Интервальные оценки параметров логарифмических доходностей. Определение доверительного интервала

Для нормально распределённых величин рассчитаем доверительные интервалы логарифмических доходностей. Доверительным называется интервал, в который попадают полученные значения, в нашем случае, с вероятностью 0,95. Для расчета используем функцию st.t.interval, а в качестве аргументов вводим уровень надежности (95%), степень свободы, выборочное среднее, исправленное стандартное отклонение (функция st.sem).

```
print('Доверительный интервал для логдоходности Магнита -',
st.t.interval(0.95, len(clear_full['Magnit_Log_Yield'])-1,
    loc=clear_full['Magnit_Log_Yield'].mean(),
    scale=st.sem(clear_full['Magnit_Log_Yield'])))
```

```
Доверительный интервал для логдоходности Магнита - (-0.006241668501839294, 0.0020182537808302774)
```

```
print('Доверительный интервал для логдоходности Лукойла -',
st.t.interval(0.95, len(clear_full['Lukoil_Log_Yield'])-1,
    loc=clear_full['Lukoil_Log_Yield'].mean(),
    scale=st.sem(clear_full['Lukoil_Log_Yield'])))
```

```
Доверительный интервал для логдоходности Лукойла - (-0.0022761867745069416, 0.0051440519034112765)
```

Рис. 34. Расчет доверительных интервалов для логдоходностей

В результате мы получили границы доверительного интервала по каждому тикеру для последующего анализа логдоходностей.

1.9. Тест Левена для проверки гипотезы о равенстве дисперсий тикеров

Далее проведем тест Левена о равенстве дисперсий двух этих тикеров. Тестом Левена определяется возможность применения критерия Стьюдента. Для применения t-критерия Стьюдента необходимо, чтобы данные (по подгруппам) имели нормальное распределение, но также и важно понимать, одинаковая или разная вариативность этих данных, которая измеряется дисперсией. Действительно, сложно статистически точно определить равенство баллов среди студентов факультетов, если взять в качестве выборки по одной группе с каждого из факультетов. Может оказаться так, что в одной группе студенты учатся абсолютно типично, скажем, у всех по 80 баллов. С высокой долей вероятности средний балл по факультету окажется в районе 80. С другой стороны, есть группа, студенты в которой набирают от 0 баллов до 100, причем совершенно случайно. Можно ли будет с уверенностью сказать, что и во всем факультете студенты учатся со средним баллом, равным 80? Конечно нет, такая вариативность не дает уверенности ни в едином выводе, особенно по сравнению с идеальной, первой группой. Тогда и сравнение факультетов по баллам подвергается определенному сомнению (риску), что будет существенно завышать значение рассчитанной статистики. Поэтому очень важно подойти к вопросу соотношения вариативности (дисперсии) исследуемых выборок, для получения максимально точного вывода.

Чтобы выполнить тест Левена в Python воспользуемся функцией st.levene. В результате проверки гипотеза о равенстве дисперсий для логарифмических доходностей акций компании Магнит и компании Лукойл была принята при уровне значимости 0,05.

```
def levene(row1, row2, alpha):
    stat, p = st.levene(clear_full[row1], clear_full[row2])
    print('W=%.3f, p-value=%.3f' % (stat, p))
    if p > alpha:
        print('Принять гипотезу о равенстве дисперсий', row1, 'и',
row2, ', a =', alpha)
    else:
        print('Отклонить гипотезу о равенстве дисперсий', row1, 'и',
row2, ', a =', alpha)
```

```
W=2.467, p-value=0.117
Принять гипотезу о равенстве дисперсий Magnit_Log_Yield и Lukoil_Log_Yield , a = 0.05
```

Рис. 35. Проведение теста Левена

Тест Левена показал, что при уровне значимости 0,05 можно принять гипотезу о равенстве дисперсией, соответственно мы можем использовать критерий Т-критерий Стьюдента в дальнейшем исследовании с предположением о равенстве дисперсий.

1.10. Проверка гипотезы о равенстве логдоходностей компаний с помощью Т-критерия Стьюдента

Проверим гипотезу о равенстве логдоходностей этих компаний. Функция st.ttest_ind вычисляет Т-критерий для средних значений двух независимых выборок. Это двусторонний тест на нулевую гипотезу о том, что две независимые выборки имеют одинаковые средние (ожидаемые) логдоходности[7]. В качестве аргументов данная функция принимает данные о логарифмических доходностях двух компаний. equal_var=True означает, что данные выборки имеют равные дисперсии (что было доказано с помощью теста Левена).

С помощью Т-критерия Стьюдента удалось определить, что средние логдоходности по компаниям «Магнит» и «Лукойл» равны на уровне значимости 0,05. Гипотеза принимается.

---

[7] *Брюс П.* Б89 Практическая статистика для специалистов Data Science: Пер. с англ. / П. Брюс, Э. Брюс. 86 с.

```
def ttest(row1, row2, alpha):
    stat, p = st.ttest_ind(clear_full[row1], clear_full[row2],
equal_var = True)
    print('Statistics=%.3f, p-value=%.3f' % (stat, p))
    if p > alpha:
        print('Принять гипотезу о равенстве средней логдоходности',
row1, 'и', row2, ', a =', alpha)
    else:
        print('Отклонить гипотезу о равенстве средней логдоходности',
row1, 'и', row2, ', a =', alpha)
```

```
Statistics=-1.258, p-value=0.209
Принять гипотезу о равенстве средней логдоходности Magnit_Log_Yield и Lukoil_Log_Yield , a = 0.05
```

Рис. 36. Проверка гипотезы о равенстве логдоходностей с помощью Т-критерия Стьюдента

1.11. Влияние пандемии на цены акций. Проверка гипотезы об изменении средней после пандемии с помощью Т-критерия Стьюдента

Далее проведем небольшое исследование и узнаем, как пандемия коронавируса повлияла на цены заданных тикеров и объемы продаж. Для этого проведем тест Стьюдента и узнаем, правда ли цены и объемы акций Лукойла и Магнита упали после объявления пандемии. Такие мини-исследования не редкость в бизнес-среде, и они носят название A/B-тестов. Например, для задач маркетинга с целью протестировать введение новой страницы регистрации (как вырастет конверсия в регистрацию?). Некоторые пользователи получают новую страницу, у некоторых она остается старой. С помощью сравнения средней конверсии через, скажем тест Стьюдента или другие тесты, определяется равны ли значения, то есть, дает ли вообще нововведение хоть сколь-либо внушительный экономический эффект. Нечто подобное проводится и в данной работе с целью выяснить, насколько акции крупных компаний устойчивы к таким экономическим изменениям.

Пандемия была объявлена с 272 дня торгов нашей выборки. Разделим данные на две части в зависимости

от даты проведения торгов и применим уже известную функцию st.ttest_ind. Результатом работы этой функции является вычисленное p-value. Если это значение окажется больше заданного уровня значимости, то принимается основная гипотеза, в противном случае — альтернативная. При проверке гипотезы об изменении средней после пандемии p-value, вычисленное с помощью функции st.ttest_ind, оказалось равным 0,000 во всех случаях (см. Рис. 36), следовательно, гипотеза об изменении средних принимается при уровнях значимости $\alpha = 0{,}05$.

```
clear_full_before_pandemic = clear_full.query('Date < 272')
clear_full_after_pandemic = clear_full.query('Date ⩾ 272')

def ttest_times(row, alpha):
    stat, p = st.ttest_ind(clear_full_before_pandemic[row],
clear_full_after_pandemic[row],
                           equal_var = True)
    print('Statistics=%.3f, p-value=%.3f' % (stat, p))
    if p > alpha:
        print('Отклонить гипотезу о изменении средней после пандемии
для', row, ', a =', alpha)
    else:
        print('Принять гипотезу о изменении средней после пандемии
для', row, ', a =', alpha)
```

```
Statistics=-11.316, p-value=0.000
Принять гипотезу о изменении средней после пандемии для Abrau_Price , a = 0.05
Statistics=4.596, p-value=0.000
Принять гипотезу о изменении средней после пандемии для Magnit_Price , a = 0.05
Statistics=-5.593, p-value=0.000
Принять гипотезу о изменении средней после пандемии для Lukoil_Price , a = 0.05
Statistics=-9.354, p-value=0.000
Принять гипотезу о изменении средней после пандемии для Abrau_Log_Volume , a = 0.05
Statistics=-8.009, p-value=0.000
Принять гипотезу о изменении средней после пандемии для Magnit_Log_Volume , a = 0.05
Statistics=-8.419, p-value=0.000
Принять гипотезу о изменении средней после пандемии для Lukoil_Log_Volume , a = 0.05
```

Рис. 37. Проверка гипотезы об изменении средней после пандемии с помощью Т-критерия Стьюдента

Для более детального анализа изучим описательную статистику для данных, полученных до и после пандемии коронавируса. Исходя из полученных значений, можно сделать вывод о том, что цены акций Магнита упали после пандемии, а вот у остальных компаний акции, наоборот, взлетели в цене.

	Date	Abrau_Price	Abrau_Volume	Abrau_Yield	Abrau_Log_Yield	Abrau_Log_Volume	Magnit_Price	Magnit_Volume	Magnit_Yield	Magnit_Log_Yiel
count	209.000000	209.000000	209.000000	209.000000	209.000000	209.000000	209.000000	2.090000e+02	209.000000	209.00000
mean	147.531100	124.260766	5947.224880	-0.000122	-0.000383	7.813532	7072.196172	1.154013e+06	-0.003380	-0.00394
std	77.190637	26.519680	8381.043663	0.022957	0.022885	1.755428	3285.680589	8.861447e+05	0.033344	0.03339
min	2.000000	79.000000	0.000000	-0.055249	-0.056833	0.000000	2984.000000	2.019980e+05	-0.091533	-0.09569
25%	82.000000	97.000000	1000.000000	-0.012500	-0.012579	6.907755	5735.000000	5.727940e+05	-0.023810	-0.02409
50%	154.000000	136.500000	3100.000000	0.000000	0.000000	8.039157	6340.000000	9.008040e+05	-0.004950	-0.00496
75%	216.000000	141.000000	7500.000000	0.007752	0.007722	8.922658	10391.000000	1.385733e+06	0.015088	0.01596
max	269.000000	179.500000	71000.000000	0.058824	0.057158	11.170435	12396.000000	5.863352e+06	0.091429	0.08748

	Date	Abrau_Price	Abrau_Volume	Abrau_Yield	Abrau_Log_Yield	Abrau_Log_Volume	Magnit_Price	Magnit_Volume	Magnit_Yield	Magnit_Log_Yiel
count	40.000000	40.00000	4.000000e+01	40.000000	40.000000	40.000000	40.000000	4.000000e+01	40.000000	40.00000
mean	299.675000	178.61250	9.829950e+04	-0.001215	-0.001338	10.512690	4672.537500	2.199926e+06	0.007919	0.00744

Рис. 38. Исследование описательной статистики показателей до и после объявления пандемии коронавируса

```
def mu(row, alpha):
    stat, p = st.mannwhitneyu(clear_full_before_pandemic[row], clear_full_after_pandemic[row])
    print('Statistics=%.3f, p-value=%.3f' % (stat, p))
    if p > alpha:
        print('Принимаем гипотезу о отсутсвии различий между', row, 'до и после пандемии', ', a =', alpha)
    else:
        print('Отвергаем гипотезу о отсутствии различий между', row, 'до и после пандемии', ', a =', alpha)
```

```
Statistics=1672.500, p-value=0.000
Отвергаем гипотезу о отсутствии различий между Abrau_Price до и после пандемии , a = 0.05
Statistics=3333.500, p-value=0.021
Отвергаем гипотезу о отсутствии различий между Magnit_Price до и после пандемии , a = 0.05
Statistics=2122.000, p-value=0.000
Отвергаем гипотезу о отсутствии различий между Lukoil_Price до и после пандемии , a = 0.05
Statistics=415.000, p-value=0.000
Отвергаем гипотезу о отсутствии различий между Abrau_Log_Volume до и после пандемии , a = 0.05
Statistics=1051.000, p-value=0.000
Отвергаем гипотезу о отсутствии различий между Magnit_Log_Volume до и после пандемии , a = 0.05
Statistics=1161.000, p-value=0.000
Отвергаем гипотезу о отсутствии различий между Lukoil_Log_Volume до и после пандемии , a = 0.05
```

Рис. 39. Подтверждение гипотезы об изменениях параметров торгов после пандемии с помощью теста Манна-Уитни

Непараметрическим критерием Манна-Уитни удостоверимся в правильности выводов. U-критерий Манна - Уитни — статистический критерий, используемый для оценки различий между двумя независимыми выборками по уровню какого-либо признака, измеренного коли-

чественно. Для реализации проверки в Python используется функция st.mannwhitneyu, принимающая в качестве аргументов сравниваемые признаки. В нашем случае тест Манна-Уитни подтвердил изменения параметров биржевых торгов после пандемии.

Данный тест также показал наличие изменений ключевых показателей торгов по каждому тикеру до и после наступления пандемии коронавирусной инфекции.

1.12. Однофакторный дисперсионный анализ по периодам

При обработке результатов наблюдений часто возникает вопрос о том насколько существенное влияние оказывает какой-либо фактор или группа факторов на измеряемую величину. Для ответа на этот вопрос может быть использован дисперсионный анализ[8].

Пусть некоторый объект или процесс характеризуется двумя или несколькими факторами, среди которых присутствуют X и Y. Предположим, что Y может принимать значения y_1, y_2, ..., y_i, тогда все множество значений, принимаемых фактором X можно разбить на l классов x_1, x_2, ..., x_i. Каждый из указанных классов x_k содержит все значения, принимаемые фактором X при $Y = Y_k, k = 1,\ldots,l : X_k = \left\{X_{k1}, X_{k2}, \ldots, X_{km_k}\right\}$. Обозначим

$$\mu_k = E\left(X_k\right), k = 1,\ldots,l.$$

Задачей дисперсионного анализа является проверка гипотезы

$$H_0: \mu_1 = \mu_2 = \ldots = \mu_l = \mu.$$

Общая схема дисперсионного анализа:

1) Вычисление средних значений:

---

[8] *Соловьев В.И.* Анализ данных: теория вероятностей и прикладная статистика, обработка и визуализация данных в Microsoft Excel: Учебник. 339 с.

$$\overline{X}_k = \frac{1}{m_k}\sum_{j=1}^{m_k} X_{kj};$$

$$\overline{\overline{X}} = \frac{1}{n}\sum_{k=1}^{l} m_k \overline{X}_k;$$

2) Вычисление межгрупповой дисперсии $\sigma^2_{факт}$:

$$\sigma^2_{факт} = \frac{1}{n}\sum_{k=1}^{l} m_k \left(\overline{X}_k - \overline{\overline{X}}\right)^2;$$

3) Вычисление средней групповой дисперсии $\sigma^2_{ост}$:

$$\sigma^2_{ост} = \frac{1}{n}\sum_{k=1}^{l}\sum_{j=1}^{m_k}\left(X_{kj} - \overline{X}_k\right)^2;$$

4) Вычисление статистики

$$F = \frac{(n-l)\sigma^2_{факт}}{(l-1)\sigma^2_{ост}};$$

5) Определение критического значения $F_\alpha(l-1,\, n-l)$

при заданном уровне значимости α;

6) Сравнение

если $F > F_\alpha(l-1,\, n-l)$ — влияние изучаемого фактора существенно,

если $F \leqslant F_\alpha(l-1,\, n-l)$ — влияние изучаемого фактора недостоверно.

Для проведения однофакторного дисперсионного анализа в Python существует функция st.t_oneway. С помощью созданной функции anova_test_per_period, критическое значение будет сравниваться с заданным уровнем значимости.

```
def anova_test_per_period(row, alpha):
    F, p = st.f_oneway(clear_full[row][clear_full['Date'] < 249/3],
                       clear_full[row][(clear_full['Date'] ⩾ 249/3) &
clear_full['Date']< 249/3],
                       clear_full[row][clear_full['Date'] ⩾ 249*2/3])
    if p>alpha:
        print('Нет основания отвергать нулевую гипотезу о равенстве
средних по периодам для', row, '| alpha =', alpha)
    else:
        print('Есть основания отвергнуть нулевую гипотезу о равенстве
средних по периодам для', row, '| alpha =', alpha)
```

```
Есть основания отвергнуть нулевую гипотезу о равенстве средних по периодам для Abrau_Log_Volume | alpha = 0.05
Нет основания отвергать нулевую гипотезу о равенстве средних по периодам для Magnit_Log_Yield | alpha = 0.05
Нет основания отвергать нулевую гипотезу о равенстве средних по периодам для Lukoil_Log_Yield | alpha = 0.05
Есть основания отвергнуть нулевую гипотезу о равенстве средних по периодам для Abrau_Price | alpha = 0.05
Есть основания отвергнуть нулевую гипотезу о равенстве средних по периодам для Magnit_Price | alpha = 0.05
Есть основания отвергнуть нулевую гипотезу о равенстве средних по периодам для Lukoil_Price | alpha = 0.05
```

Рис. 40. Однофакторный дисперсионный анализ по периодам (1)

```
def anova_test_per_company(start, end, alpha):
    F, p = st.f_oneway(clear_full['Abrau_Log_Volume']
[(clear_full['Date'] ⩾ start) & clear_full['Date'] ⩽ end],
                       clear_full['Magnit_Log_Yield']
[(clear_full['Date'] ⩾ start) & clear_full['Date'] ⩽ end],
                       clear_full['Lukoil_Log_Yield']
[(clear_full['Date'] ⩾ start) & clear_full['Date'] ⩽ end])
    if p>alpha:
        print('Нет основания отвергать нулевую гипотезу о равенстве
средних логдоходностей по компаниям с', start, 'по', end,'| alpha =',
alpha)
    else:
        print('Есть основания отвергнуть нулевую гипотезу о равенстве
средних логдоходностей по компаниям с', start, 'по', end, '| alpha =',
alpha)
```

```
Есть основания отвергнуть нулевую гипотезу о равенстве средних

 логдоходностей по компаниям с 150 по 240 | alpha = 0.05
```

Рис. 41. Однофакторный дисперсионный анализ по периодам (2)

Однофакторный дисперсионный анализ показал, что для логарифмической доходности акции Магнита и Лукойла нет основания отвергнуть гипотезу о равенстве средних по периодам.

1.13. Исследование тесноты связи между логдоходностями компаний

Для исследования тесноты связи между значениями логдоходностей акций компаний «Абрау-Дюрсо», «Лукойл», «Магнит» используем процедуру «Корреляция». Для оценки значимости коэффициента корреляции используем критерий Стьюдента.

Для исследования наличия корреляции между указанными факторами используется следующий алгоритм:

1. Вычисляется выборочные коэффициент корреляции

$$\hat{r} = \frac{\sum_{k=1}^{n}(x_k - \bar{x})(y_k - \bar{y})}{\sqrt{\sum_{k=1}^{n}(x_k - \bar{x})^2 \sum_{k=1}^{n}(y_k - \bar{y})^2}}.$$

2. Выдвигается нулевая гипотеза о равенстве нулю генерального коэффициента корреляции r

$$H_0 : r = 0,$$

при альтернативной гипотезе

$$H_1 : r \neq 0.$$

3. Вычисляется статистика

$$t = \hat{r} \cdot \frac{\sqrt{n-2}}{\sqrt{1-\hat{r}^2}}. \qquad (12)$$

4. При заданном значении уровня значимости α определяется критическое значение случайной величины. Распределенной по закону Стьюдента с $n - 2$ степенями свободы t_{kp}.

Если $|t| > t_{kp}$, то гипотеза H_0 отклоняется, в противном случае — не отклоняется.

Определим значимость коэффициентов корреляции с помощью Python. Зададим функцию t_corrmat, которая будет выполнять расчет по формуле, указанной выше. Следует отметить, что функция abs() преобразует целое число или число с плавающей запятой в его абсолютное значение. Построим тепловую карту уже изученным нами способом: с помощью функции heatmap().

```
t_corrmat = abs(clear_full.corr()*((len(clear_full['Date']) - 2)/(1-(clear_full.corr())**2))**(0.5))
f, ax = plt.subplots(figsize =(16, 9), dpi = 300)
sns.heatmap(t_corrmat, ax = ax, linewidths = 0.1, annot = True, vmax = 1.9696, fmt='.3g', color = 'black')
ax.set_title('Матрица t-статистики коэффициентов корреляции (белые - статистически значимые коэффициенты)', size = 20)
plt.show()
```

Рис. 42. Построение матрицы значимости коэффициентов корреляции

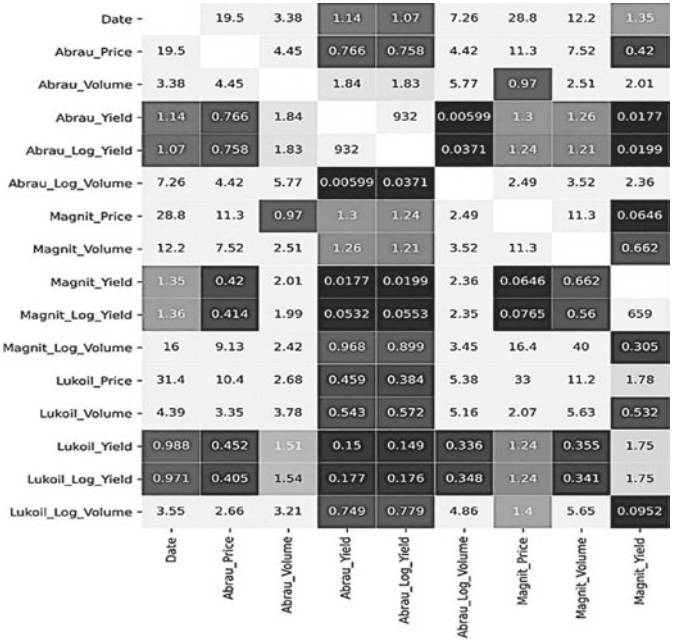

Рис. 43. Фрагмент матрицы значимости коэффициентов корреляции

Во время анализа матрицы значимости коэффициентов корреляции важно обращать внимание на поля, выделенные белым цветом и имеющие значение близкое к 1,75. Чем меньше показатель, тем меньше статистическая значимость коэффициента корреляции. Попробуем интерпретировать результаты, связанные с исследуемыми данными: в частности, с логарифмическими доходностями компаний. Исследование показало, что коэффициенты корреляции логарифмической доходности акций компании Абрау-Дюрсо с логарифмическими доходностями Лукойла и Магнита статистически не значимы и равняются (0,176) и (0,0553) соответственно. Однако матрица t-статистики показывает достаточно сильную статистическую значимость коэффициента корреляции логарифмической доходности акций Магнита и логарифмической доходности акций Лукойла (1,73).

ГЛАВА 2.
ВЫПОЛНЕНИЕ РАСЧЕТНО-АНАЛИТИЧЕСКОЙ РАБОТЫ С ПОМОЩЬЮ ЯЗЫКА R

R — второй по популярности язык для анализа данных, который часто сравнивают с Python. Он был разработан для статистических вычислений и графики, что отлично подходит для анализа данных. В нем есть инструменты для визуализации данных. Он совместим с любыми статистическими приложениями, работает офлайн, а разработчикам предлагаются различные пакеты для управления данными и создания графиков. R в основном используется для статистических вычислений. Он имеет набор алгоритмов, которые углубленно применяются в области машинного обучения. А конкретнее — в анализе временных рядов, классификации, кластеризации, линейном моделировании и т.д.

2.1. Загрузка программы

Первым шагом необходимо установить среду разработки R с официального сайта на устройство: https://cran.r-project.org/bin/windows/base/. После этого загрузим R-Studio. Для этого перейдем на сайт в раздел загрузки: https://www.rstudio.com/products/rstudio/download/

RStudio — это набор интегрированных инструментов, предназначенных для повышения производительности работы с R и Python. Он включает в себя консоль, редактор подсветки синтаксиса, который поддерживает прямое

выполнение кода, а также множество надежных инструментов для построения графиков, просмотра истории, отладки и управления рабочей областью.

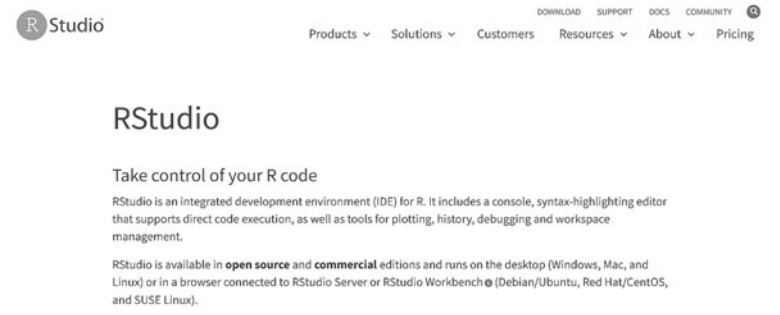

Рис. 44. Загрузка R-Studio с официального сайта

В разделе «продукты» выберем нужный пакет Open Source — Rstudio. Функционала данного пакета достаточно для совершения статистического анализа, предусмотренного пособием.

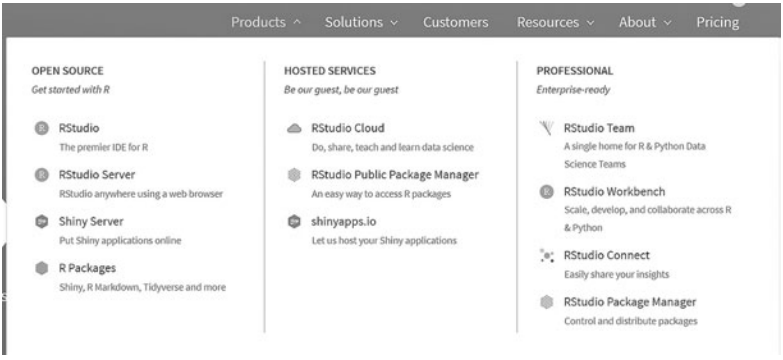

Рис. 45. Выбор пакета Open Source

После этого выберем версию для персональных компьютеров — Desktop (см. Рис. 45).

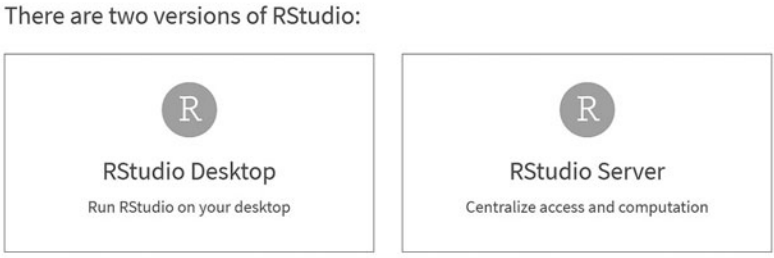

Рис. 46. Выбор версии RStudio Desktop

2.2. Загрузка исходных данных

Данные для исследования считываются из базы данных Московской биржи. Для этого следует перейти по адресу: https://mfd.ru/export/

На сайте MFD в разделе «Мосбиржа Акции и ПИФы» найдем тикеры нужных компаний, например с 2015 года по настоящее время. Далее следует задать формат записей формируемого списка. Для успешной обработки данных об акциях компаний с помощью программы RStudio установим следующие параметры. Промежуток — неделя, разделитель — точка с запятой, десятичный разделитель — точка.

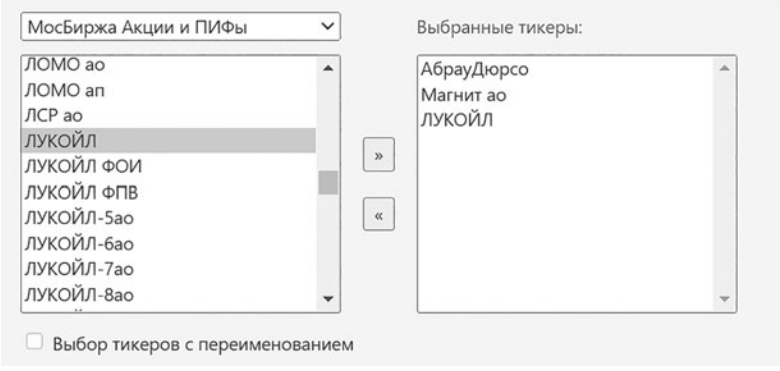

Рис. 47. Выбор акций нужных компаний

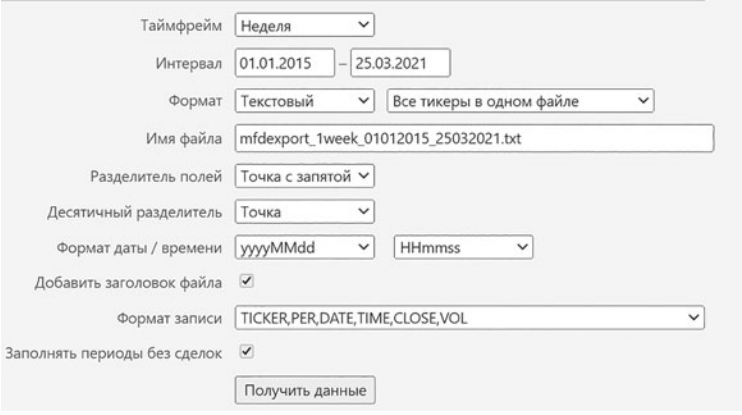

Рис. 48. Формирование формата записей

Теперь перейдем к загрузке необходимых данных в саму среду разработки RStudio. Формат с расширением .txt является часто используемым при работе с R, поэтому в нашем случае файл не требует дополнительной конвертации и готов к импорту. Загрузку текстового файла легче всего сделать через File-Import Dataset.

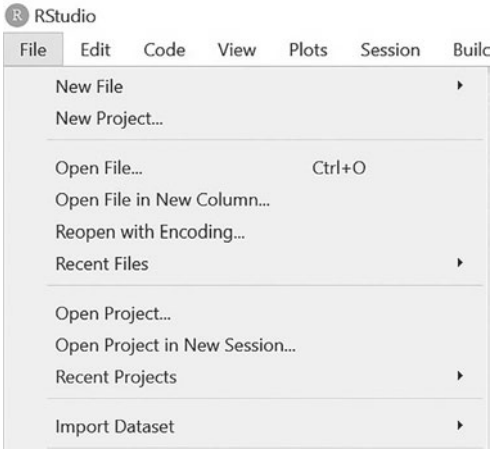

Рис. 49. Загрузка исходных данных (1)

Так как с сайта мы получили текстовый файл (формат .txt) выбираем пункт From text (base), чтобы на выходе получить рабочую базу данных. Возможно, R предложит загрузить необходимые для импорта данных библиотеки, загружаем их.

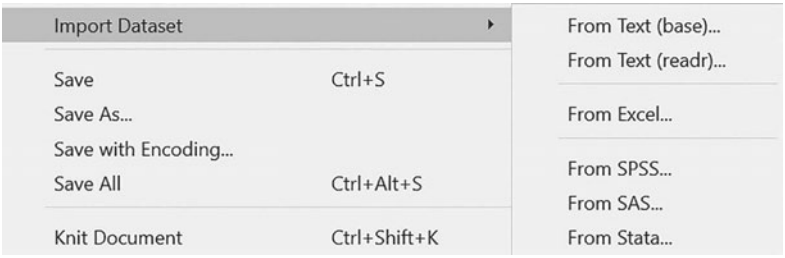

Рис. 50. Загрузка исходных данных (2)

Теперь необходимо озаглавить исходный файл латинскими буквами во избежание ошибок.

После загрузки R предложит изменить некоторые параметры данных. Необходимо изменить кодировку на UTF-8 для читаемости, поставить флажок на Heading, так как у столбцов имеются заголовки, номера строк (row names) удобнее представлять в численном формате, разделителем (separator) служит точка с запятой, десятичный знак — точка, цитат и комментариев в данных нет, можно их импортировать клавишей Import. На рисунке 47 наглядно представлены необходимые изменения данных.

Данные успешно загружены. Перейдем к просмотру и первичному ознакомлению. Выполним данный шаг с помощью функции print(), которая выводит первые 10 строк массива, включая заголовки таблицы[9].

[9] *Дж. Д. Лонг и Пол Титор R*. Книга рецептов: Проверенные рецепты для статистики, анализа и визуализации данных / Пер. с англ. Д.А. Беликова. 105 с.

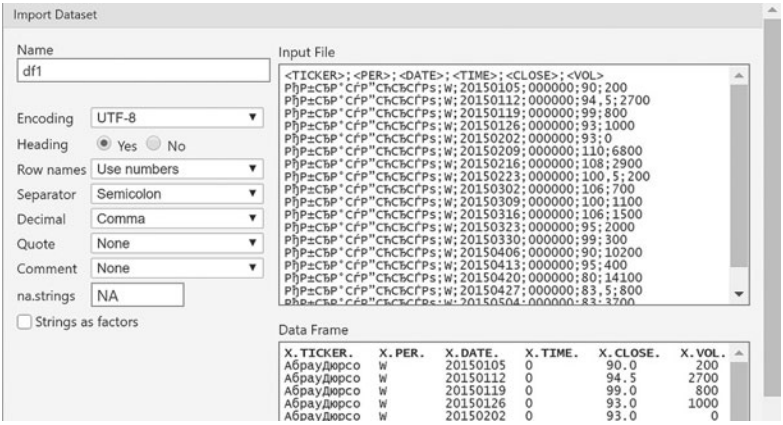

Рис. 51. Загрузка исходных данных (3)

```
1 print(df1)
```

	X.TICKER.	X.PER.	X.DATE.	X.TIME.	X.CLOSE.	X.VOL.
1	АбрауДюрсо	W	20150105	0	90	200
2	АбрауДюрсо	W	20150112	0	94.5	2700
3	АбрауДюрсо	W	20150119	0	99	800
4	АбрауДюрсо	W	20150126	0	93	1000
5	АбрауДюрсо	W	20150202	0	93	0
6	АбрауДюрсо	W	20150209	0	110	6800
7	АбрауДюрсо	W	20150216	0	108	2900
8	АбрауДюрсо	W	20150223	0	100.5	200
9	АбрауДюрсо	W	20150302	0	106	700

Рис. 52. Предпросмотр данных

В целом процедура получения данных схожа с загрузкой в Jupiter Notebook с той лишь разницей, что R не использует собственную папку, а загружает данные непосредственно с жесткого диска, что бывает неудобно, однако сам процесс получения данных аналогичен. Удобнее же сделана настройка импорта данных — в Python она прописывается через строку, тогда как здесь можно настроить параметры через графический интерфейс.

С помощью функции str() произведем описание данных. Функция помогает компактно представить внутреннюю структуру данных. Аналогом метода str() в Python является функция .info(), изученная в первой главе.

```
  1 str(df1)
'data.frame':   975 obs. of  6 variables:
 $ X.TICKER.: chr  "АбрауДюрсо" "АбрауДюрсо" "АбрауДюрсо" "АбрауДюрсо" ...
 $ X.PER.   : chr  "W" "W" "W" "W" ...
 $ X.DATE.  : int  20150105 20150112 20150119 20150126 20150202 20150209 2015
50223 20150302 20150309 ...
 $ X.TIME.  : int  0 0 0 0 0 0 0 0 0 0 ...
 $ X.CLOSE. : chr  "90" "94.5" "99" "93" ...
 $ X.VOL.   : int  200 2700 800 1000 0 6800 2900 200 700 1100 ...
```

Рис. 53. Анализ структуры исходных данных с помощью метода str()

Как видно, исходные данные необходимо либо представить в виде числового ряда, либо изменить тип данных на отображение даты. Остальные типы данных соответствуют признакам.

Перейдем к удалению ненужных столбцов. Столбец «время» и «период» не содержат важных значений, которые бы пригодились в дальнейшем исследовании. Поэтому в нашем случае их уместнее удалить. Если в Python мы использовали метод drop(), то здесь легче всего просто передать столбцу пустые данные (NULL)[10].

```
1 df1$X.PER. ← NULL
2 df1$X.TIME. ← NULL
```

Рис. 54. Удаление столбцов времени и периода

Следует удалить строки, содержащие нулевые значения цен и объемов.

Следующим шагом будет изменение название столбцов на более удобное. Это необходимо для облегчения

[10] *Дж. Д. Лонг и Пол Титор R*. Книга рецептов: Проверенные рецепты для статистики, анализа и визуализации данных / Пер. с англ. Д.А. Беликова. 46 с.

дальнейшей работы с данными. Поскольку мы неоднократно будем обращаться к столбцам по их названиям, лучше всего оставить только буквенные символы. Для переименования столбцов в Python мы использовали метод columns и передавали ему список с именами, здесь же используется функция columns() с аргументом dataframe, которому и передается заданный вектор имен. На языке R обозначение вектора осуществляется буквой «c».

```
1 colnames(df1) ← c("Ticker", "Date", "Price", "Volume")
```

Рис. 55. Переименование столбцов

Переведем дату в привычный формат с помощью функции as.Date. Аргумент character помогает задать нужный нам формат. В нашем случае после обработки дата будет иметь следующий вид: ГГГГ-ММ-ДД. Также переведем столбец цен в численный формат. Как мы могли видеть на рисунке 51 сейчас он имеет строковый формат («chr»). Для этого существует функция as.numeric (по аналогии, as.Date и as.character — «в качестве даты», «в качестве строки»).

```
1 df1$Date ← as.Date(as.character(df1$Date), "%Y%m%d")
2 df1$Price ← as.numeric(df1$Price)
```

Рис. 56. Изменение формата столбцов

Добавим столбец дня, опять же, передав конкретным строкам числовой вектор с теми номерами, которые мы хотим видеть. Это, безусловно, хардкодинг, но в данных целях его целесообразнее использовать, чем цикл, что усложнит понимание нашей работы. Так, получаем полностью преобразованные данные. В отличие от Python, который перезаписывает столбец полностью при попытке обратиться к строкам, R избирательно меняет значения строк.

```
df1$Day[1:325] ← c(1:325)
df1$Day[326:650] ← c(1:325)
df1$Day[651:975] ← c(1:325)
df1
```

	Ticker	Date	Price	Volume	Day
1	АбрауДюрсо	2015-01-05	90	200	1
2	АбрауДюрсо	2015-01-12	94.5	2700	2
3	АбрауДюрсо	2015-01-19	99	800	3
4	АбрауДюрсо	2015-01-26	93	1000	4
5	АбрауДюрсо	2015-02-02	93	0	5

Рис. 57. Добавление столбца дня

Наши данные полностью готовы к работе. Перейдем к расчетам синтетических признаков, необходимых для дальнейшего статистического анализа.

2.3 Оптимизация данных и вычисление дополнительных признаков для каждой компании

Теперь создадим новые признаки — доходность, логдоходность, логарифм объема. Для этого загрузим библиотеку DescTools, которая позволяет создавать циклические сдвиги в числовом векторе. Это достаточно неудобно — в Python для вычисления темпа роста, чем по своей сути и является доходность, существует специальный метод — pot_change(). Далее вычисляется логарифмическая доходность или логдоходность по уже знакомой формуле. Заметим, что функция натурального логарифма уже есть в стандартном пакете R, тогда как в Python приходилось загружать библиотеку numpy.

Код для расчета доходности акций содержит функцию, позволяющую найти разницу между ценами актива соответственно для i–й и i-1 –й, то есть предшествующей недели, и разделить полученный результат на цену актива предшествующей недели.

$$r_i = \frac{S_i - S_{i-1}}{S_{i-1}}, \qquad (14)$$

где r_i = доходность,

s_i и s_{i-1} — цены актива соответственно на i-й и $i-1$-й неделях.

Так мы написали код формулы расчета логарифмической доходности:

$$l_i = \ln \frac{s_i}{s_{i-1}}, \tag{15}$$

где l_i = логдоходность,

s_i и s_{i-1}– цены актива соответственно на i-й и $i-1$-й неделях.

Далее следует найти значения логарифмов объемов торгов

$$Q_i = \ln V_i, \tag{16}$$

где Q_i = логарифм объема,

V_i — объем продаж актива.

Все из вышеперечисленных переменных загрузим в отдельные столбцы (передадим значения этим столбцам), удалим значения строк 1, 326, 651 с помощью среза. Функцией map заменим Inf значения в столбце логарифма объема — они получаются, когда объем равен 0. В Python эта процедура была более автоматизирована с помощью lambda-функции, что недоступно в R.

```
require(DescTools)
Prices ← as.vector(df1$Price)
df1$Yield ← (Prices - VecRot(Prices, 1))/VecRot(Prices,1)
df ← df1[-c(1,326,651), ]
df$Log_Yield ← log(1+df$Yield)
df$Log_Volume ← log(df$Volume)
df$Log_Volume ← Map(function(x) replace(x, is.infinite(x), 0),
df$Log_Volume)
df$Log_Volume ← as.numeric(df$Log_Volume)
```

Рис. 58. Добавление новых параметров

Теперь создадим одну таблицу, выставив новые значения столбцов по компаниям по порядку.

Первым шагом будет создание отдельных срезов по тикерам. Для этого мы воспользуемся функцией sebset(), принимающей в качестве аргумента условие для выбора тех или иных данных. На рисунке 55 можно наглядно увидеть формирование трех отдельных таблиц по названию анализируемой компании. Данная операция тождественна выполненной выше в программе Python.

```
Abrau ← subset(df,df$Ticker == 'АбрауДюрсо')
Magnit ← subset(df,df$Ticker == 'Магнит ао')
Lukoil ← subset(df,df$Ticker == 'ЛУКОЙЛ')
```

Рис. 59. Формирование трех срезов по компаниям

После того, как мы сформировали три отдельные таблицы, приступим к их соединению по столбцам даты и дня торгов. Вид объединения в данном случае не играет роли, так как формы таблиц тождественны. Аналогично операции в Jupiter, методом (а в данном случае, функцией) merge объединим поэтапно по парам все таблицы, в качестве объединяющих столбцов передадим дату и день торгов. Отличие от Python — Python нужно в любых операциях множественных элементов frame передавать списком, а в R — вектором.

```
full_demo ← merge(Abrau, Magnit, by = c('Date', 'Day'))
full ← merge(full_demo, Lukoil, by = c('Date', 'Day'))
```

	Date	Day	Ticker.x	Price.x	Volume.x	Yield.x	Log_Yield.x
1	2015-01-12	2	АбрауДюрсо	94.5	2700	0.050000000	0.048790164
2	2015-01-19	3	АбрауДюрсо	99.0	800	0.047619048	0.046520016
3	2015-01-26	4	АбрауДюрсо	93.0	1000	-0.060606061	-0.062520357
4	2015-02-02	5	АбрауДюрсо	93.0	0	0.000000000	0.000000000
5	2015-02-09	6	АбрауДюрсо	110.0	6800	0.182795699	0.167880873

Рис. 60. Объединение данных по трем тикерам в одну таблицу

Теперь удалим значения в столбцах с названиями тикеров и по-новому их озаглавим. Добавим название компании («Абрау», «Магнит», «Лукойл») к относящемуся признаку. Это упростит понимание данных и увеличит легкость прочтения кода.

```
full$Ticker.x ← NULL
full$Ticker.y ← NULL
full$Ticker ← NULL
colnames(full) ← c('Date','Day', 'Abrau_Price', 'Abrau_Volume',
'Abrau_Yield', 'Abrau_Log_Yield', 'Abrau_Log_Volume',
                   'Magnit_Price', 'Magnit_Volume','Magnit_Yield',
'Magnit_Log_Yield', 'Magnit_Log_Volume',
                   'Lukoil_Price', 'Lukoil_Volume', 'Lukoil_Yield',
'Lukoil_Log_Yield', 'Lukoil_Log_Volume')
full
```

	Date	Day	Abrau_Price	Abrau_Volume	Abrau_Yield	Abrau_Log_Yield
1	2015-01-12	2	94.5	2700	0.050000000	0.048790164
2	2015-01-19	3	99.0	800	0.047619048	0.046520016
3	2015-01-26	4	93.0	1000	-0.060606061	-0.062520357
4	2015-02-02	5	93.0	0	0.000000000	0.000000000
5	2015-02-09	6	110.0	6800	0.182795699	0.167880873

Рис. 61. Изменения заголовков столбцов

2.4 Исследование изменения цен и проведение корреляционного анализа

Для дальнейшего исследования нормализуем цены активов методом 'min-max'. Приведем стоимость активов в сопоставимый вид и сравним относительные цены (t). Для этого используем уже известную нам формулу:

$$t = \frac{X - X_{min}}{X_{max} - X_{min}}, \qquad (17)$$

где X_{max} и X_{min} — соответственно минимальное и максимальное значения цены.

Для этого, вычислим максимальные и минимальные цены каждого актива функциями max, min. Далее функцией Map применим функцию нормализации к конкретному столбцу (аналогично тому, как происходило применение функции с логарифмом объема и lambda-функции).

```
AbrauMax ← max(full$Abrau_Price)
AbrauMin ← min(full$Abrau_Price)
MagnitMax ← max(full$Magnit_Price)
MagnitMin ← min(full$Magnit_Price)
LukoilMax ← max(full$Lukoil_Price)
LukoilMin ← min(full$Lukoil_Price)
full$Abrau_Normalize_Price ← Map(function(x) ((x-AbrauMin)/(AbrauMax-
AbrauMin)), full$Abrau_Price)
full$Magnit_Normalize_Price ← Map(function(x) ((x-
MagnitMin)/(MagnitMax-MagnitMin)), full$Magnit_Price)
full$Lukoil_Normalize_Price ← Map(function(x) ((x-
LukoilMin)/(LukoilMax-LukoilMin)), full$Lukoil_Price)
```

Рис. 62. Вычисление относительных цен

В стандартном пакете R есть визуализация с помощью функции plot. Она уже устарела и не такая красивая, как визуализация с помощью пакета ggplot2. Поэтому для построения графиков мы используем знаменитую библиотеку R — ggplot2[11]. ее функционал поистине огромен, и она подход для очень многих типов данных, в том числе и временных рядов, чем и по сути является цена. Так, необходимо сначала установить данную библиотеку через меню Tool-Install packages… или с помощью функции require (как правило, эта библиотека устанавливается достаточно быстро). По аналогии с Python, нужно задать композицию (ggplot), где необходимо указать, с какими данными будем работать. В дальнейшем принцип отличается — если в Python все исходит от осей (axis) и их расположения на композиции, то в R это лишено смысла, там как каждый объект — от линии, графика или оси добавляется простым прибавлением к общей композиции. Так,

[11] *Дж. Д. Лонг и Пол Титор R*. Книга рецептов: Проверенные рецепты для статистики, анализа и визуализации данных / Пер. с англ. Д.А. Беликова. 258 с.

можно задать общий «сюжет» — data — данные, aes — оси, linetype — стиль какого признака будут принимать линии, нейминг — labs — title — название всего графика, x — название оси x, y — название оси y, настроить пространство вокруг графика — theme — legend.position — положение легенды (в данном случае — «bottom» — внизу). Дополнительно писать show() не требуется, как это было в Python — график в любом случае отобразиться в боковом окне.

```
require(ggplot2)
df$Normalize_Price ← c(full$Abrau_Normalize_Price,
full$Magnit_Normalize_Price, full$Lukoil_Normalize_Price)
ggplot(data = df, aes(x = Day, y = as.numeric(Normalize_Price), linetype
= Ticker)) +
geom_line()+
  labs(title = 'График нормализованных цен акций',
       x = 'Дата',
       y = 'Нормализованные цены акций')+
  theme(legend.position = 'bottom')
```

Рис. 63. Код для построения графика нормализованных цен

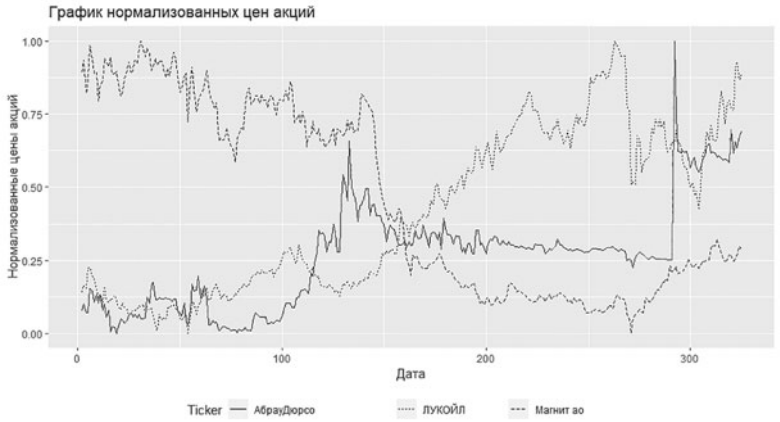

Рис. 64. График динамики относительных цен акций

Также, как и в первом исследовании с помощью Python, график, полученный с помощью R, демонстрирует движение нормализованных цен представленных акций за определенный промежуток времени. Мы видим,

что относительные цены акций АО «Магнита» за исследуемый период имеют тренд к снижению. Для акций Лукойла ситуация обратная — в целом наблюдается уверенный рост относительных цен с допустимыми колебаниями. Акции Абрау-Дюрсо имеют резкие перепады, значительный рост в кратчайшие временные срок может быть обусловлен особенностями деятельности предприятия или изменениям на рынке.

Для исследования корреляции построим корреляционную матрицу с диаграммами рассеивания. Корреляционная матрица строится очень просто с помощью функции cor().

```
1 cor(full[2:17])
```

Рис. 65. Построение корреляционной матрицы

Как это было в опыте с Python (п. 1.4) проведем корреляционный анализ с помощью коэффициента корреляции, который можно взять из корреляционной матрицы аналогично тому, как это было реализовано на Python. Что касается построения диаграммы рассеивания, это можно легко сделать с помощью диаграммы geom_point() библиотеки ggplot2, схема построения графика такая же, как была в предыдущем пункте. С помощью функции annotate(), можно добавить на систему координат любой текст, а функция cat() поможет в одном предложении соединить строковый формат и числовой. Теперь ограничиваем фрейм сверху и снизу теми столбцами, взаимосвязь с которыми хотим продемонстрировать.

```
1 ggplot(data = full) +
2   geom_point(mapping = aes(x = Abrau_Log_Yield, y = Magnit_Log_Yield))+
3   annotate('text', x = 0.5, y = 0.1, label = cat(sprintf('r=%.2f',
  cor(full[c('Abrau_Log_Yield','Magnit_Log_Yield')])[2])))+
4   labs(title = 'Диаграмма рассеивания Abrau_Log_Yield и
  Magnit_Log_Yield')
```

Рис. 66 (начало). Диаграммы рассеивания ggplot2

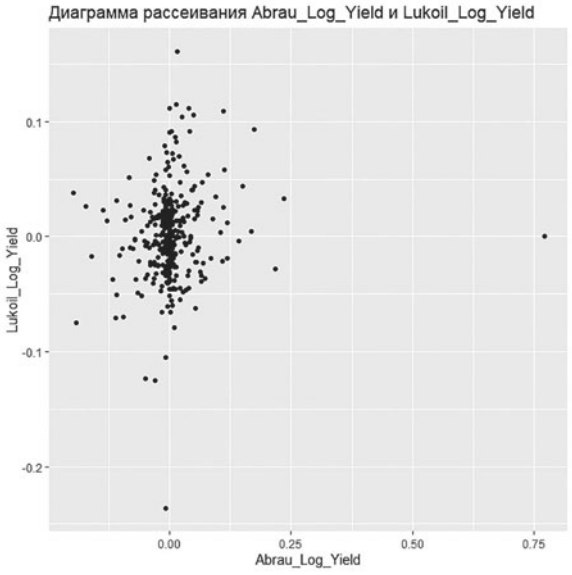

Рис. 66 (окончание). Диаграммы рассеивания ggplot2

Аналогично тому, как был построен график нормализованных цен, построим график динамики логдоходностей. С его помощью мы попытаемся визуально оценить изменения логарифмических доходностей для разных акций.

```
ggplot(data = df, aes(x = Day, y = as.numeric(Log_Yield), linetype =
Ticker)) +
  geom_line()+
  labs(title = 'Динамика логдоходности акций',
       x = 'Дата',
       y = 'Логдоходность акций')+
  theme(legend.position = 'bottom')
```

Рис. 67. Построение графика динамики логдоходностей

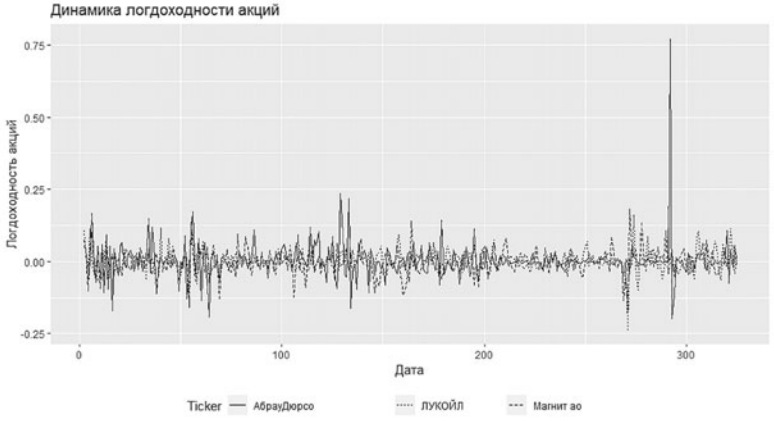

Рис. 68. График динамики логдоходности акций

График демонстрируют сильную динамику изменение логарифмических доходностей компания Абрау-Дюрсо, что может свидетельствовать о наличии значительного количества выбросов в исходных данных.

Перейдем к изучению описательной статистики. Если в Python для ее вывода необходимо лишь задействовать метод describe() из стандартного пакета pandas, в случае R для получения полноценной статистики по каждому признаку необходимо задействовать дополнительные библиотеки, например библиотеку pastecs. Pastecs расшифровывается как пакет для анализа пространственно-временных экологических рядов. Загрузим данную библиотеку через меню Tools. С помощью встроенной в библиотеку функции stat.desc(), выводим на экран описательную статистику по каждому столбцу.

```
1 library('pastecs')
2 stat.desc(full)
```

Рис. 69. Запуск описательной статистики

	Abrau_Volume	Abrau_Yield
nbr.val	3.240000e+02	324.000000000
nbr.null	6.000000e+00	40.000000000
nbr.na	0.000000e+00	0.000000000
min	0.000000e+00	-0.180357143
max	1.192961e+07	1.162162162
range	1.192961e+07	1.342519305
sum	2.175224e+07	1.684800525
median	4.500000e+03	0.000000000
mean	6.713654e+04	0.005200002
SE.mean	3.773660e+04	0.004561034
CI.mean.0.95	7.424056e+04	0.008973084
var	4.613925e+11	0.006740181
std.dev	6.792588e+05	0.082098606
coef.var	1.011757e+01	15.788188448

Рис. 70. Вывод описательной статистики

Теперь, аналогично функции sns.histplot() в Python, выведем гистограммы распределения признаков. Сделать это можно с помощью функции geom_histogram() библиотеки ggplot2.

```
1
2 ggplot(data = full) +
3   geom_histogram(mapping = aes(x = Abrau_Log_Yield))+
4   labs(title = 'Гистограмма логдоходностей Abrau_Log_Yield')
```

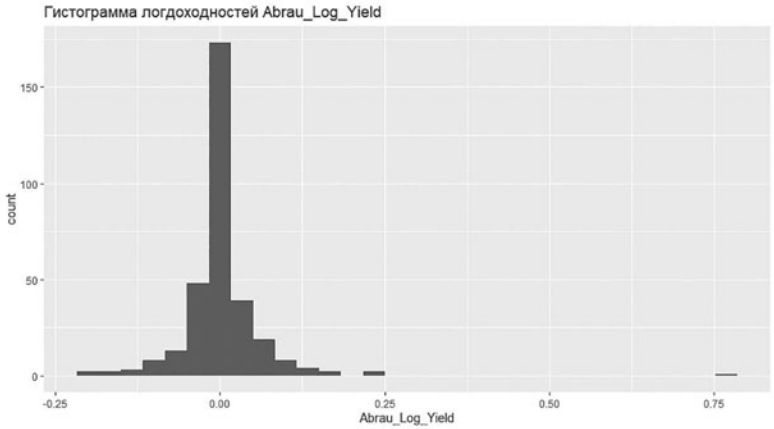

Рис. 71 (начало). Построение гистограмм

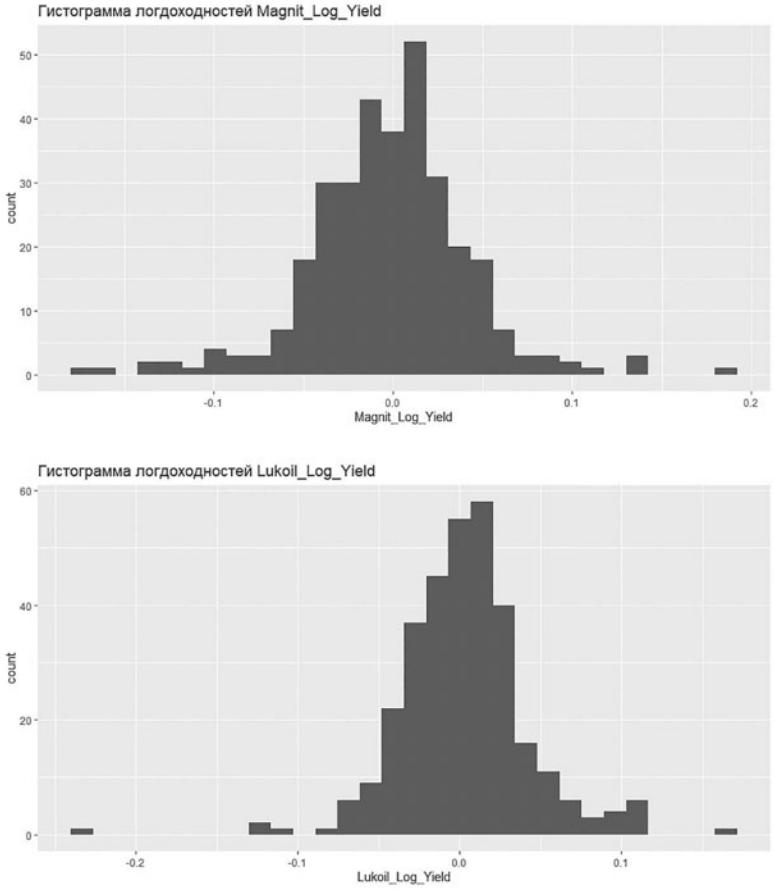

Рис. 71 (окончание). Построение гистограмм

Иногда на выходе получается изображение, которое невозможно прочитать. Например, при использовании функции pairs (на самом деле практически при любом выводе нужна индивидуальная настройка, этот кейс очень показателен). Это обуславливается большим объемом выходных данных, а также малым масштабом выве-

денных графиков. Справиться с возникшей проблемой поможет функция Export.

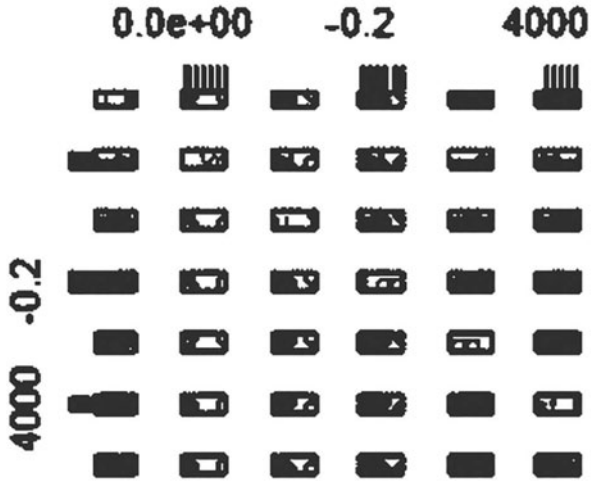

Рис. 72. Тяжело читаемая матрица диаграмм рассеивания

В этом окне можно выставить необходимое разрешение изображения. Используем формат JPEG для удобства хранения изображений. Так как матрица рассеиваний квадратная, выберем разрешение 2000*2000.

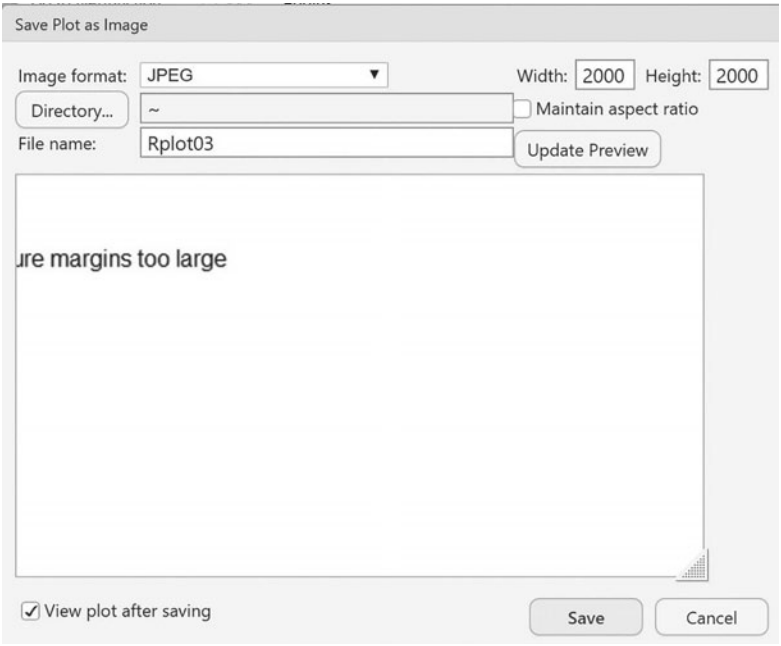

Рис. 73. Сохранение диаграмм рассеивания с помощью функции Export

После сохранения выведем графики на экран. На выходе получаются читаемые диаграммы. При увеличении масштаба можно с легкостью прочитать подписи осей и проанализировать зависимость между основными показателями.

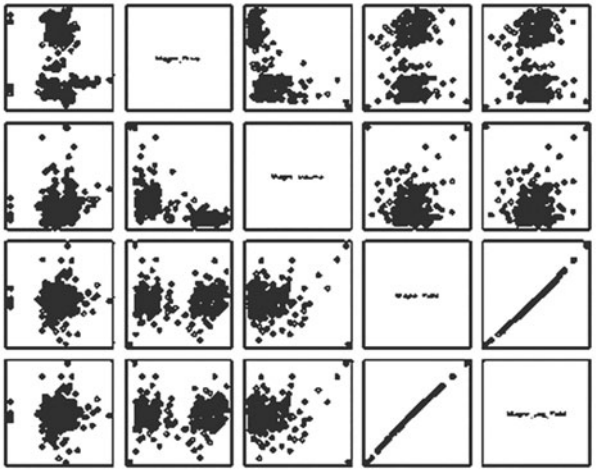

Рис. 74. Обновленная матрица диаграмм рассеивания

2.5. Исследование логарифмических доходностей акций

Перейдем к детальному исследованию логарифмических доходностей акций. Для начала посмотрим на то, как логдоходности распределены. В этом нам поможет функция boxplot. В качестве аргументов зададим исходные данных для обработки, названия осей, заголовок, а также цвет диаграмм размаха[12].

Для использования встроенной в R функции построения таких графиков, ставим логдоходность акций в зависимость от выбранного тикера для автоматического разбиения с помощью тильды, остальные аргументы очень похожи на аргументы схожих функций библиотеки matplotlib. С помощью Export масштабируем вывод графика.

---

[12] *Дж. Д. Лонг и Пол Титор R*. Книга рецептов: Проверенные рецепты для статистики, анализа и визуализации данных / Пер. с англ. Д.А. Беликова. 288 с.

```
boxplot(Log_Yield ~ Ticker,
        xlab = "Тикер",
        ylab = "Значение логдоходностей",
        main = "Boxplot-графики логдоходностей по тикерам",
        col = "black", data = df)
```

Рис. 75. Исследование логарифмических доходностей акций

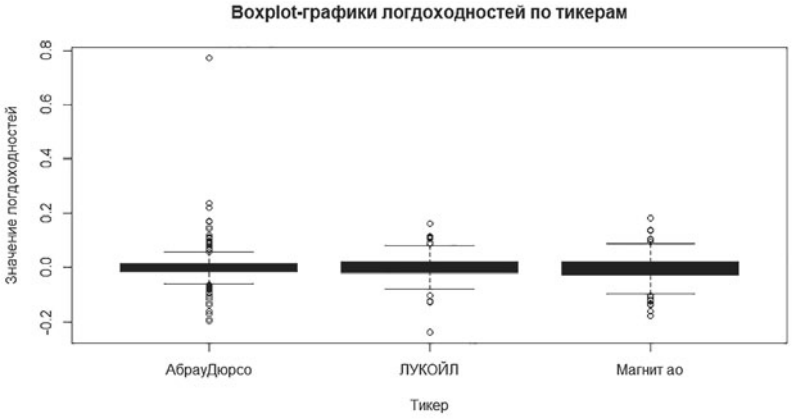

Рис. 76. Диаграммы размаха («ящики с усами») для логдоходностей

На диаграммах четко видно, что данные содержат большое количество выбросов. Конечно, лидером по количеству выбросов являются акции Абрау-Дюрсо. В следующем пункте проведем работу по их удалению.

2.6. Удаление выбросов логдоходностей акций

Теперь очистим данные от выбросов, использовав правило 1,5 межквартальных размаха. Интересно, что в R данную процедуру можно организовать буквально в одну строку кода, что, безусловно, очень удобно. Используя функцию sebset(), можно передать ей условия, что столбцы логдоходности отдельных акций компаний больше или равны нижней границе и меньше или равны

верхней границе. Для выведения квартилей Q_1 и Q_3 используем функцию quartile() с индексом 2 для Q_1 и 4 для Q_3 (1 – Q_0 или минимальное значения, 3 — медиана, 5 — максимальное значение), для выведения межквартального размаха — функцию IQR(Interquartile Range). Получается довольно компактная формула удаления выбросов, что также реализуемо и на Python с той лишь разницей, что IQR необходимо было вычислять вручную, как и границы. Для описания данных используем уже знакомую функцию str() — как оказалось, осталось лишь 249 строк.

```
full_clear ← subset(full, full$Abrau_Log_Yield ⩾
(as.numeric(quantile(full$Abrau_Log_Yield)[2]) -
1.5*IQR(full$Abrau_Log_Yield)) & full$Abrau_Log_Yield ⩽
(as.numeric(quantile(full$Abrau_Log_Yield)[4]) +
1.5*IQR(full$Abrau_Log_Yield)) & full$Magnit_Log_Yield ⩾
(as.numeric(quantile(full$Magnit_Log_Yield)[2]) -
1.5*IQR(full$Magnit_Log_Yield)) & full$Magnit_Log_Yield ⩽
(as.numeric(quantile(full$Magnit_Log_Yield)[4]) +
1.5*IQR(full$Magnit_Log_Yield)) & full$Lukoil_Log_Yield ⩾
(as.numeric(quantile(full$Lukoil_Log_Yield)[2]) -
1.5*IQR(full$Lukoil_Log_Yield)) & full$Lukoil_Log_Yield ⩽
(as.numeric(quantile(full$Lukoil_Log_Yield)[4]) +
1.5*IQR(full$Lukoil_Log_Yield)))
```

Рис. 77. Удаление выбросов в данных

Как и в задаче на Python, построим гистограммы распределения логдоходностей акций компаний после удаления выбросов (с заменой data frame на full_clear).

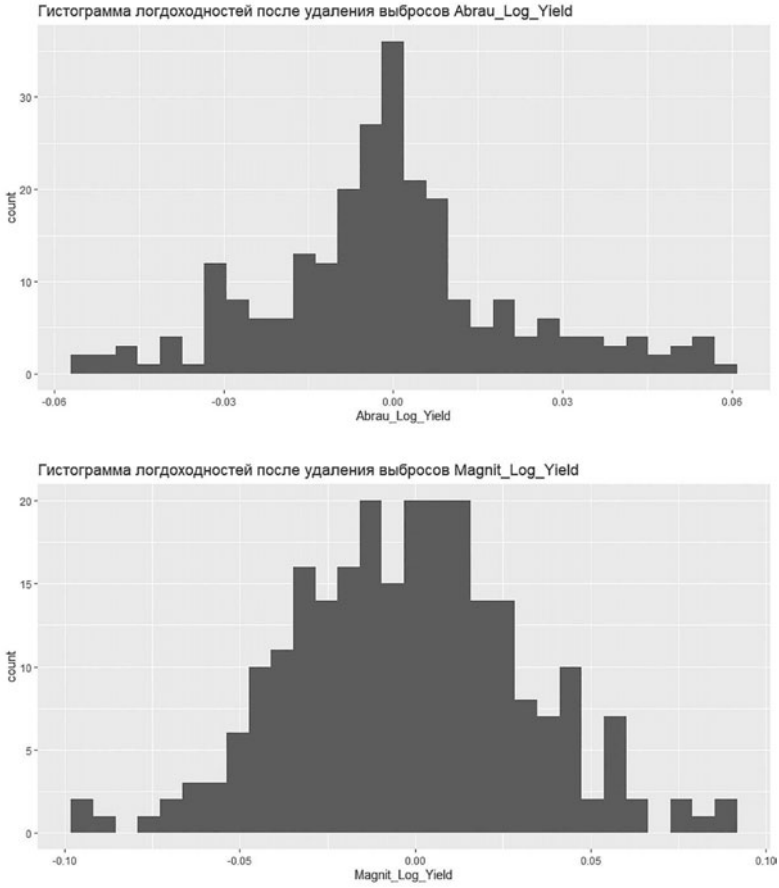

Рис. 78 (начало). Частотные гистограммы «очищенных» данных логдоходностей

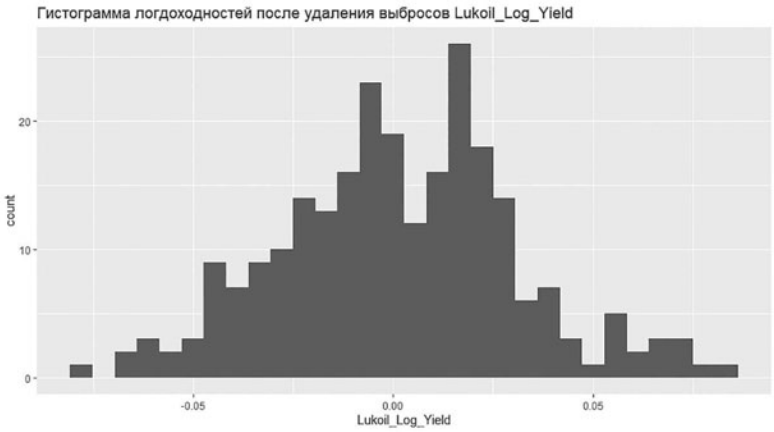

Рис. 78 (окончание). Частотные гистограммы «очищенных» данных логдоходностей

Распределения стали заметно более походить на нормальный закон, из чего можно сделать вывод о том, что удаление выбросов существенно влияет на качество данных и исследования в целом, поэтому эта процедура является неотъемлемой частью любого статистического исследования, работы с данными, потому что практически любой совокупности свойственны аномалии, мешающие работе.

2.7. Проверка гипотез о нормальности логдоходностей для каждой компании

Приступим к проверке гипотез. Как мы помним, в Python для проверки гипотез мы использовали функции, чтобы можно было менять значение alpha на необходимый в каждом отдельном тесте, при этом не создавая лишние коды. Здесь попробуем реализовать похожую схему, но для начала разберемся, что из себя представляет функция в R.

Для того, чтобы уменьшить количество строк кода, сделать код более читабельным и удобным для редактирования используют функции. Сначала нужно написать имя функции. Имя должно отражать сущность операции, выполняемой в ходе реализации кода. Затем прописываем знак присвоения и слово function, которое создает объект класса функция. В скобках мы указываем аргументы, то есть входные данные, использующиеся в процессе анализа. Тело функции находится внутри фигурных скобок. Оно содержит определенные операции, выполняемые напрямую с входными данными. Важно отметить, что для использования функции требуется ввести только имя функции и ее аргументы.

В принципе, все аналогично тому, как это работает в Python, только определяется функция по-другому.

```
имя функции <- function(аргумент 1, аргумент 2){
  тело функции: код, который использует
  аргумент 1 и аргумент 2 для получения
  результата вычислений
  }
```

Рис. 79. Алгоритм написания функции в среде R

Для начала через меню Tools установим пакет nortest. С его помощью реализуются самые разнообразные тесты на нормальность распределения. Наиболее распространенные приведены ниже:

- ✓ ad.test() – тест Андерсона-Дарлинга
- ✓ cvm.test() — тест Крамера фон Мизеса
- ✓ lillie.test() — тест Колмогорова-Смирнова в модификации Лиллиефорса
- ✓ pearson.test() — критерий хи-квадрат Пирсона
- ✓ shapiro.test() — тест Шапиро-Уилка

```
1 library(nortest)
```

Рис. 80. Загрузка пакета nortest

Теперь мы готовы к проведению проверок гипотез по заданным критериям.

2.7.1. Проверка гипотезы по критерию Пирсона

Сначала проверим данные на нормальность по критерию согласия Пирсона. Критерий согласия Пирсона (χ^2) применяют для проверки гипотезы о соответствии эмпирического распределения предполагаемому теоретическому распределению *F(x)* при большом объеме выборки ($n \geqslant 100$). Критерий применим для любых видов функции *F(x)*, даже при неизвестных значениях их параметров, что обычно имеет место при анализе результатов механических испытаний. В этом заключается его универсальность. Подробно алгоритм расчета данного критерия приведен в пункте 1.7.1.

По схеме, указанной выше (см Рис. 76), расписываем функцию с условиями if … else … Функция проверки столбца на нормальность — pearson.test(), потребуем через $ вывод p.value, и сравним его с уровнем значимости alpha. На выходе, с помощью функции cat, выведем результат проверки гипотезы.

```
pirson ← function(x, alpha) {
  if (pearson.test(x)$p.value > alpha) {
    cat('Нет оснований отвергнуть нулевую гипотезу о нормальности
  распределения по критерию согласия Пирсона, p-value =',
  pearson.test(x)$p.value, 'alpha =', alpha)
  }
  else{
    cat('Есть основания отвергнуть нулевую гипотезу о нормальности
  распределения по критерию согласия Пирсона, p-value =',
  pearson.test(x)$p.value, '| alpha =', alpha)
  }
}

```

Рис. 81. Проверка гипотезы о нормальном распределении с помощью критерия согласия Пирсона

```
> pirson(full_clear$Abrau_Log_Yield, 0.05)
Есть основания отвергнуть нулевую гипотезу о нормальност
и распределения по критерию согласия Пирсона, p-value =
 6.549117e-11 | alpha = 0.05
> pirson(full_clear$Magnit_Log_Yield, 0.05)
Нет оснований отвергнуть нулевую гипотезу о нормальности
 распределения по критерию согласия Пирсона, p-value =
 0.9368251 alpha = 0.05
> pirson(full_clear$Lukoil_Log_Yield, 0.05)
Нет оснований отвергнуть нулевую гипотезу о нормальности
 распределения по критерию согласия Пирсона, p-value =
 0.1089881 alpha = 0.05
```

Рис. 82. Результат проверки гипотез о нормальном распределении с помощью критерия согласия Пирсона

В результате проверки мы видим, что распределение логдоходности по акциям Абрау-Дюрсо отличается от нормального. Для акций Магнита и Лукойла характерно нормальное распределение данных о логарифмических доходностях.

2.7.2. Проверка гипотезы по критерию Шапиро-Уилка

Аналогично предыдущей проверке расписывается и функция критерия Шапиро-Уилка. Критерий Шапиро-Уилка является хорошей альтернативой критерию Пирсона для формулировки выводов о нормальности распределения в малых и средних выборках. Используем функцию критерия Шапиро-Уилка shapiro.test().

```
1 shapiro ← function(x, alpha) {
2   if (shapiro.test(x)$p.value > alpha) {
3     cat('Нет оснований отвергнуть нулевую гипотезу о нормальности
  распределения по критерию Шапиро-Уилка, p-value =', sf.test(x)$p.value,
  'alpha =', alpha)
4   }
5   else{
6     cat('Есть основания отвергнуть нулевую гипотезу о нормальности
  распределения по критерию Шапиро-Уилка, p-value =', sf.test(x)$p.value,
  '| alpha =', alpha)
7   }
8 }
```

Рис. 83. Проверка гипотезы о нормальном распределении с помощью критерия Шапиро-Уилка

```
> shapiro(full_clear$Abrau_Log_Yield, 0.05)
Есть основания отвергнуть нулевую гипотезу о нормальности распределен
ия по критерию Шапиро-Уилка, p-value = 0.0001408923 | alpha = 0.05> s
hapiro(full_clear$Magnit_Log_Yield, 0.05)
Нет оснований отвергнуть нулевую гипотезу о нормальности распределени
я по критерию Шапиро-Уилка, p-value = 0.8764107 alpha = 0.05> shapiro
(full_clear$Lukoil_Log_Yield, 0.05)
Нет оснований отвергнуть нулевую гипотезу о нормальности распределени
я по критерию Шапиро-Уилка, p-value = 0.4620071 alpha = 0.05
```

Рис. 84. Результат проверки гипотез о нормальном распределении с помощью критерия Шапиро-Уилка

Результаты проверок с помощью критерия Пирсона и Шапиро-Уилка показали схожие результаты. Только акции компании Абрау-Дюрсо не являются нормально распределенными.

2.7.3. Проверка гипотезы по критерию Колмогорова-Смирнова

Аналогично, проверим данные на нормальность по критерию Колмогорова-Смирнова. Использования критерия Колмогорова — Смирнова позволяет определить подчинятся ли распределение доходностей актива нормальному закону распределения[13]. Если доходность актив не подчиняется нормальному закону распределения, то к такому активу применяются альтернативные финансовые методы и модели. Если говорить о написании функции Колмогорова-Смирнова в R, то существует модифицированный вариант критерия — критерий Лиллиефорса. Выглядит он следующим образом: lillie.test()

[13] *Орлов А.И.* Непараметрические критерии согласия Колмогорова, Смирнова, омега-квадрат и ошибки при их применении. 10 с.

```
lf ← function(x, alpha) {
  if (lillie.test(x)$p.value > alpha) {
    cat('Нет оснований отвергнуть нулевую гипотезу о нормальности
распределения по критерию Лиллиефорса, p-value =',
lillie.test(x)$p.value, 'alpha =', alpha)
  }
  else{
    cat('Есть основания отвергнуть нулевую гипотезу о нормальности
распределения по критерию Лиллиефорса, p-value =',
lillie.test(x)$p.value, '| alpha =', alpha)
  }
}
```

Рис. 85. Проверка гипотезы о нормальном распределении с помощью критерия Колмогорова-Смирнова

```
> lf(full_clear$Abrau_Log_Yield, 0.05)
Есть основания отвергнуть нулевую гипотезу о нормальност
и распределения по критерию Лиллиефорса, p-value = 5.168
415e-08 | alpha = 0.05
> lf(full_clear$Magnit_Log_Yield, 0.05)
Нет оснований отвергнуть нулевую гипотезу о нормальности
 распределения по критерию Лиллиефорса, p-value = 0.8548
743 alpha = 0.05
> lf(full_clear$Lukoil_Log_Yield, 0.05)
Нет оснований отвергнуть нулевую гипотезу о нормальности
 распределения по критерию Лиллиефорса, p-value = 0.4784
1 alpha = 0.05
```

Рис. 86. Результат проверки гипотез о нормальном распределении с помощью критерия Колмогорова-Смирнова

После проведения проверок гипотез по трем разным критериям, можно сделать вывод о том, что тикер 'АбрауДюрсо' имеет распределение, отличное от нормального.

Дальнейшие тесты, касающиеся работы с нормальным распределением, будем проводить с оставшимися тикерами: акциями Магнита и Лукойла.

2.8. Интервальные оценки параметров логарифмических доходностей. Определение доверительного интервала

Перейдем к интервальной оценке параметров логдоходностей. Для начала найдем доверительный интервал для генеральной средней логдоходности акций, исполь-

зуя формулы (mean()-t*sd()/sqrt(), mean()+t*sd()/sqrt()). В этом случае Python удобнее, так как имеет встроенную функцию для построения доверительных интервалов: функцию st.t.interval. В среде R требуется прописывать данные формулы расчета вручную.

```
> cat('Доверительный интервал для логдоходности Магнита
 -', c(mean(full_clear$Magnit_Log_Yield)-1.96*sd(full_cl
ear$Magnit_Log_Yield)/sqrt(249), mean(full_clear$Magnit_
Log_Yield)+1.96*sd(full_clear$Magnit_Log_Yield)/sqrt(24
9)))
Доверительный интервал для логдоходности Магнита - -0.00
622159 0.001998175
> cat('Доверительный интервал для логдоходности Лукойла
 -', c(mean(full_clear$Lukoil_Log_Yield)-1.96*sd(full_cl
ear$Lukoil_Log_Yield)/sqrt(249), mean(full_clear$Lukoil_
Log_Yield)+1.96*sd(full_clear$Lukoil_Log_Yield)/sqrt(24
9)))
Доверительный интервал для логдоходности Лукойла - -0.00
2258149 0.005126014
```

Рис. 87. Расчет доверительного интервала по каждому тикеру

2.9. Тест Фишера для проверки гипотезы о равенстве дисперсий тикеров

Используем тест Фишера для определения равенства или неравенства генеральных дисперсий. Он основан на переборе всех возможных вариантов заполнения таблицы сопряженности при имеющейся численности групп, и поэтому чем эта численность меньше, тем проще выполнить соответствующие вычисления[14]. Это нужно, чтобы в дальнейшем проводить тест Стьюдента на равенство средних. Для того, чтобы провести тест Фишера, используем функцию var.test().

[14] *Соловьев В.И.* Анализ данных: теория вероятностей и прикладная статистика, обработка и визуализация данных в Microsoft Excel: Учебник. 320 с.

```
fisher ← function(x, y, alpha) {
  if (var.test(x, y)$p.value > alpha) {
    cat('Нет оснований отвергнуть нулевую гипотезу о равенстве дисперсий
по критерию Фишера, p-value =', var.test(x, y)$p.value, 'alpha =',
alpha)
  }
  else{
    cat('Есть основания отвергнуть нулевую гипотезу о равенстве
дисперсий по критерию Фишера, p-value =', var.test(x, y)$p.value, '|
alpha =', alpha)
  }
}
fisher(full_clear$Magnit_Log_Yield, full_clear$Lukoil_Log_Yield, 0.05)
```

```
Нет оснований отвергнуть нулевую гипотезу о равенстве дисперсий по кр
итерию Фишера, p-value = 0.09200032 alpha = 0.05
```

Рис. 88. Подтверждение гипотезы о равенстве дисперсий по критерию Фишера

Как видим, полученное P-значение оказалось равным 0,092, что больше уровня значимости альфа. Таким образом, у нас нет оснований отклонить нулевую гипотезу о том, что дисперсии равны. В ходе работы мы доказали, что дисперсии выборок равны с достоверностью 95%.

2.10. Проверка гипотезы о равенстве средних логдоходностей компаний с помощью T-критерия Стьюдента

Проверим гипотезу о равенстве средних значений логдоходностей этих компаний. Используем функцию t.test для проведения теста Стьюдента. Она вычисляет T-критерий для средних значений двух независимых выборок. Это двусторонний тест на нулевую гипотезу о том, что две независимые выборки имеют одинаковые средние (ожидаемые) логдоходности. В качестве аргументов данная функция принимает данные о логарифмических доходностях двух компаний. Не стоит забывать указать в функции аргумент var.equal = TRUE о равенстве дисперсий, что было доказано с помощью теста Фишера. В результате получаем, что средние логдоходности акций двух компаний равны.

```
ttest ← function(x, y, alpha) {
  if (t.test(x, y, var.equal = TRUE)$p.value > alpha) {
    cat('Нет оснований отвергнуть нулевую гипотезу о равенстве средних
по критерию Стьюдента, p-value =', t.test(x, y)$p.value, 'alpha =',
alpha)
  }
  else{
    cat('Есть основания отвергнуть нулевую гипотезу о равенстве средних
по критерию Стьюдента, p-value =', t.test(x, y)$p.value, '| alpha =',
alpha)
  }
}
ttest(full_clear$Magnit_Log_Yield, full_clear$Lukoil_Log_Yield, 0.05)
```

```
Нет оснований отвергнуть нулевую гипотезу о равенстве средних по крит
ерию Стьюдента, p-value = 0.209033 alpha = 0.05
```

Рис. 89. Подтверждение гипотезы о равенстве средних логдоходностей по критерию Стьюдента

2.11. Влияние пандемии на цены акций. Проверка гипотезы об изменении средней после пандемии с помощью Т-критерия Стьюдента

Теперь проверим, изменилась ли средняя логдоходность до и после пандемии. Это нужно для формулировки выводов о влиянии кризисных периодов на цены акции разных компаний. Создадим 2 среза данных — clear_full_before_pandemic, clear_full_after_pandemic. Основанием для деления исходной таблицы на две станет день проведения торгов. Пандемия началась с 272 торгов нашей выборке. Для формирования новых дата сетов воспользуемся функцией subset().

```
clear_full_before_pandemic ← subset(full_clear, full_clear$Day < 272)
clear_full_after_pandemic ← subset(full_clear, full_clear$Day ⩾ 272)
```

Рис. 90. Разделение данных на две части по дате проведения торгов

Проведем тесты для двух наших нормально распределенных логдоходностей. Проверку гипотез осуществим с помощью функции ttest(). Так, при уровне надежности 95% средняя логдоходность акций Магнит изменилась после пандемии, Лукойла же — не изменилась.

```
> ttest(clear_full_before_pandemic$Magnit_Log_Yield, cle
ar_full_after_pandemic$Magnit_Log_Yield, 0.05)
Есть основания отвергнуть нулевую гипотезу о равенстве с
редних по критерию Стьюдента, p-value = 0.03520958 | alp
ha = 0.05
> ttest(clear_full_before_pandemic$Lukoil_Log_Yield, cle
ar_full_after_pandemic$Lukoil_Log_Yield, 0.05)
Нет оснований отвергнуть нулевую гипотезу о равенстве ср
едних по критерию Стьюдента, p-value = 0.7069252 alpha =
 0.05
```

Рис. 91. Проверка гипотезы об изменении средней логдоходности акций после пандемии с помощью критерия Стьюдента

Теперь проведем непараметрический тест Манна-Уитни, использующийся для оценки различий между двумя независимыми выборками по уровню какого-либо признака, измеренного количественно. В нашем случае используем данный критерий для оценки влияния периода совершения транзакции на изменение цен и логдоходностей.

Сначала пропишем функцию теста wilcox.test(). Опять же, в отличие от Python, данный тест имеется только в модификации Вилкоксона.

```
1 wcn ← function(x, y, alpha) {
2   if (wilcox.test(x, y)$p.value > alpha) {
3     cat('Нет оснований отвергнуть нулевую гипотезу о равенстве средних
  по критерию Вилкоксона, p-value =', wilcox.test(x, y)$p.value, 'alpha
  =', alpha)
4   }
5   else{
6     cat('Есть основания отвергнуть нулевую гипотезу о равенстве средних
  по критерию Вилкоксона, p-value =', wilcox.test(x, y)$p.value, '| alpha
  =', alpha)
7   }
8 }
```

Рис. 92. Функция для проведения теста Манна-Уитни

Влияние периода проведения торгов на изменение цены акций очевидно. Цены акций всех трех компаний отличаются до и после пандемии.

```
> wcn(clear_full_before_pandemic$Abrau_Price, clear_full
_after_pandemic$Abrau_Price, 0.05)
Есть основания отвергнуть нулевую гипотезу о равенстве с
редних по критерию Вилкоксона, p-value = 1.876899e-09 |
 alpha = 0.05
> wcn(clear_full_before_pandemic$Magnit_Price, clear_ful
l_after_pandemic$Magnit_Price, 0.05)
Есть основания отвергнуть нулевую гипотезу о равенстве с
редних по критерию Вилкоксона, p-value = 0.04264558 | al
pha = 0.05
> wcn(clear_full_before_pandemic$Lukoil_Price, clear_ful
l_after_pandemic$Lukoil_Price, 0.05)
Есть основания отвергнуть нулевую гипотезу о равенстве с
редних по критерию Вилкоксона, p-value = 8.217629e-07 |
 alpha = 0.05
```

Рис. 93. Проверка гипотезы об изменении цены акций до и после пандемии

Можно сделать вывод о том, что для каждого из трех видов тикеров характерно наличие изменений цены до и после пандемии.

Проверим изменения доходностей акций в разные периоды. Судя по результатам теста, представленным ниже, доходность отличается до и после пандемии только у Магнита. В отношении остальных компаний гипотеза об изменении доходности не подтвердилась.

```
> wcn(clear_full_before_pandemic$Abrau_Log_Yield, clear_
full_after_pandemic$Abrau_Log_Yield, 0.05)
Нет оснований отвергнуть нулевую гипотезу о равенстве ср
едних по критерию Вилкоксона, p-value = 0.8261964 alpha
 = 0.05
> wcn(clear_full_before_pandemic$Magnit_Log_Yield, clear
_full_after_pandemic$Magnit_Log_Yield, 0.05)
Есть основания отвергнуть нулевую гипотезу о равенстве с
редних по критерию Вилкоксона, p-value = 0.02895427 | al
pha = 0.05
> wcn(clear_full_before_pandemic$Lukoil_Log_Yield, clear
_full_after_pandemic$Lukoil_Log_Yield, 0.05)
Нет оснований отвергнуть нулевую гипотезу о равенстве ср
едних по критерию Вилкоксона, p-value = 0.4833781 alpha
 = 0.05
```

Рис. 94. Проверка гипотезы об изменении доходности акций до и после пандемии