ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ ОБРАЗОВАТЕЛЬНОЕ БЮДЖЕТНОЕ УЧРЕЖДЕНИЕ
ВЫСШЕГО ОБРАЗОВАНИЯ
«ФИНАНСВЫЙ УНИВЕРСИТЕТ ПРИ ПРАВИТЕЛЬСТВЕ РОССИЙСКОЙ ФЕДЕРАЦИИ»
(ФИНАНСОВЫЙ УНИВЕРСИТЕТ)

Департамент анализа данных и машинного обучения

Дисциплина: «Теория вероятностей и математическая статистика»

Направление подготовки: «Прикладная математика и информатика»

Профиль: «Анализ данных и принятие решений в экономике и финансах»

Факультет информационных технологий и анализа больших данных

Форма обучения очная

Учебный 2020/2021 год, 4 семестр

Курсовая работа

на тему:

«Проверка гипотезы о нормальном распределении дневной логарифмической доходности при условии определенного объема торгов накануне»

Вид исследуемых данных:

Котировки акций компаний, входящих в индекс S&P 100

Допущена к защите
17 мая 2021 г.

Выполнила:
студентка группы ПМ19-2
Гераськина Н.С.

Руководитель:
д-р ф.-м. н. Рябов П.Е.

Москва 2021

Содержание

1. Введение

1.1. Подробное разъяснение темы

Целью данной курсовой работы является проверка гипотезы о нормальном распределении логарифмической доходности при условии определенного объема торгов накануне.

Основным критерием проверки данной гипотезы был выбран критерий χ^2 Пирсона. Как вспомогательный критерий для проверки равномерности распределения будет использован критерий Колмогорова - Смирнова. Помимо нулевой гипотезы в качестве альтернативных гипотез было взято распределение Стьюдента со степенями свободны t(2), t(3), t(5), t(15) и стандартное распределение Коши. Мощность критериев оценивается методом Монте-Карло.

1.2. Описание выборок

В данной курсовой работе используются данные о котировках 18 компаний, входящий в индекс S&P 100. Это компании со следующими тикерами: AAPL, AMZN, CSCO, DIS, FB, FDX, GOOG, IBM, INTC, KO, MCD, MSFT, NFLX, NKE, PFE, PG, TSLA, V. В курсовой работе будут анализироваться данные за период с 1 января 2010 года по 31 декабря 2020 года. В ходе проверки гипотезы часть данных придется убрать. Гипотеза о нормальном распределении дневной логарифмической доходности будет проверяться 10000 раз.

1.3. Планируемая новизна

В качестве новизны в курсовой работе будут рассматриваться другие акции из индекса S&P 100. Будут использованы пять альтернативных гипотез. Также новизна данной работы заключается в том, что квантили и P-значения считаются для различных объемов выборки (году, полугодию, кварталу).

2. Предварительный анализ данных

Для анализа будут использоваться акции компаний, входящих в индекс S&P 100, который включает в себя крупнейшие и наиболее авторитетные, стабильные компании, «голубые шишки» из списка S&P 500. Список компаний, входящих в данный индекс, а также

информация о ценах на акции, взяты с сайта http://finance.yahoo.com . В последний раз данные обновлялись 31.04.21. Ниже приведена таблица компаний и соответствующие им тикеры:

Таблица 1. Список компаний

Тикер	Компания
AAPL	Apple, Inc.
AMZN	Amazom.com, Inc.
CSCO	Cisco Systems, Inc.
DIS	The Walt Disney Company
FB	Facebook, Inc.
FDX	FedEx Corp.
GOOG	Alphabet, Inc.
IBM	International Business Machine
INTC	Intel Corp.
KO	The Coca-Cola Company
MCD	McDonald’s Corp.
MSFT	Microsoft Corp.
NKE	Nike, Inc.
NFLX	Netflix, Inc.
PFE	Pfizer, Inc.
PG	Procter & Gamble Co
TSLA	Tesla, Inc.
V	Visa, Inc.

Для того чтобы удостовериться в пригодности данных по компаниям для исследования в рамках этой работы, рассмотрим таблицу количества торговых дней. Необходимо, чтобы количество дней по каждому тикеру было более 240. Торги должны проводиться на одной бирже, а данные должны быть доступны за один и тот же период времени. Для этого используется программа «Код 01. Количество торговых дней.ipynb» (используемые поля "Date", единица измерения – шт.)

Таблица 2. Количество торговых дней

Тикеры	2010	2011	2012	2013	2014	2015	2016	2017	2018	2019	2020
AAPL	252	252	250	252	252	252	252	251	251	252	252
AMZN	252	252	250	252	252	252	252	251	251	252	252
CSCO	252	252	250	252	252	252	252	251	251	252	252
INTC	252	252	250	252	252	252	252	251	251	252	252
MCD	252	252	250	252	252	252	252	251	251	252	252
NKE	252	252	250	252	252	252	252	251	251	252	252
FDX	252	252	250	252	252	252	252	251	251	252	252
PG	252	252	250	252	252	252	252	251	251	252	252
V	252	252	250	252	252	252	252	251	251	252	252
DIS	252	252	250	252	252	252	252	251	251	252	252
FB	0	0	154	252	252	252	252	251	251	252	252
GOOG	252	252	250	252	252	252	252	251	251	252	252
NFLX	252	252	250	252	252	252	252	251	251	252	252
MSFT	252	252	250	252	252	252	252	251	251	252	252
IBM	252	252	250	252	252	252	252	251	251	252	252
KO	252	252	250	252	252	252	252	251	251	252	252
TSLA	129	252	250	252	252	252	252	251	251	252	252
PFE	252	252	250	252	252	252	252	251	251	252	252

По данным таблицы видно, что не все тикеры удовлетворяют предъявленным требованиям. Из списка исследуемых данных необходимо исключить акции компаний с тикерами FB и TSLA, так как они не торговалась на бирже достаточное количество дней в период с 2010 по 2012 года и в 2010 году соответственно. В остальных компаниях наблюдается одинаковое количество торговых дней.

Рассмотрим максимальные отклонения цен акций, построив таблицу максимальных дневных относительных скачков вверх и вниз по годам и тикерам. Эти таблицы были построены при помощи программы «Код 02. Максимальные скачки цены.ipynb» (используемые поля "Date", "Open", "Close", единица измерения – проценты)

Таблица 3. Максимальные скачки цены вверх (в %)

	2010	2011	2012	2013	2014	2015	2016	2017	2018	2019	2020	Max
AAPL	4,08	3,85	5,31	3,96	3,72	8,7	3,08	2,67	5,98	3,95	6,26	8,7
AMZN	12,22	8,18	5,36	4,16	5,02	4,12	4,43	4,05	7,45	5,05	6,55	12,22
CSCO	4,61	4,21	3,46	2,14	3,45	3,45	2,71	2,17	4,8	3,06	9,95	9,95
INTC	3,67	3,84	4,01	3,03	5,3	6,63	2,68	2,67	4,98	4,19	12,78	12,78
MCD	2,25	3	2,98	2,06	3,06	3,95	2,58	2,58	4,31	1,86	10,58	10,58
NKE	4,02	5,43	3,75	4,32	3,39	3,6	3,13	4,24	6,27	2,93	10,09	10,09
FDX	5,49	5,73	4,59	4,56	2,61	2,64	4,34	3,29	3,99	3,62	12,98	12,98
PG	2,41	2,95	3,22	2,09	2,2	2,43	2,9	1,56	3,05	2,16	7,11	7,11
V	4,09	14,03	3,91	3,18	4,56	6,55	5,76	1,89	5,86	3,52	6,69	14,03
DIS	4,68	3,93	3,43	2,65	2,26	3,2	3,8	3,49	5,31	7,32	7,88	7,88
GOOG	4,46	3,04	2,6	3,99	2,49	3,69	2,23	1,81	5,2	4,15	4,84	5,2
NFLX	10,24	9,7	19,32	16,4	8,71	9,16	11,77	4,71	9,38	5,57	11,17	19,32
MSFT	3,76	4,28	3,23	3,72	3,69	4,84	3,16	1,87	5,7	2,79	7,68	7,68
IBM	4,34	3,62	2,23	2,67	3,33	3,03	2,87	2,65	4,21	5	6	6
KO	2,44	3,86	2,02	2,34	2,74	2,1	2,46	2,67	2,86	2,3	7,15	7,15
PFE	3,61	3,95	2,41	2,94	2,23	3,49	4,41	3,02	4,58	3,22	6,22	6,22

Таблица 4. Максимальные скачки цены вниз (в %)

	2010	2011	2012	2013	2014	2015	2016	2017	2018	2019	2020	Min
AAPL	-4,49	-3,51	-5,29	-4,16	-4,03	-6,63	-2,97	-4	-4,04	-3,24	-7,26	-7,26
AMZN	-6,58	-5,01	-3,57	-5,62	-4,08	-8,56	-6,29	-3,4	-7,3	-3,82	-4,52	-8,56
CSCO	-4,69	-4,17	-4	-2,75	-2,45	-5,09	-3,23	-2,03	-5,11	-3,85	-7,07	-7,07
INTC	-4,14	-3,24	-2,92	-2,81	-2,98	-4,33	-3,16	-2,85	-5,59	-3,6	-6,15	-6,15
MCD	-2,77	-1,95	-2,69	-1,74	-2,75	-4,46	-4,35	-2,42	-4,06	-2,84	-5,98	-5,98
NKE	-3,02	-5,3	-4,65	-2,64	-2,72	-5,53	-4,33	-2,04	-5,93	-3,53	-5,68	-5,93
FDX	-4,27	-4,94	-3,72	-3,03	-3,29	-5,04	-3,71	-3,53	-6,2	-4,03	-7,17	-7,17
PG	-1,87	-2,43	-1,57	-2,98	-1,9	-5,21	-2,98	-2,35	-3,49	-3,91	-6,19	-6,19
V	-12,94	-4,38	-4,59	-8,64	-3,65	-6,58	-4,44	-3,5	-4,94	-3,47	-5,23	-12,94
DIS	-3,27	-4,07	-3,97	-3,16	-4,21	-4,14	-3,34	-4,42	-4,66	-2,83	-10,43	-10,43
GOOG	-3,83	-4,34	-8,01	-2,48	-5,5	-5,34	-5,62	-3,52	-5,76	-2,75	-5,58	-8,01
NFLX	-6,09	-12,5	-7,15	-16,84	-6,56	-6,98	-6,88	-5,41	-9,16	-4,86	-7,48	-16,84
MSFT	-5,75	-3,98	-2,84	-3,09	-3,44	-4,93	-4,01	-3,96	-5,76	-3,24	-5,92	-5,92
IBM	-2,69	-3,35	-2,85	-2,93	-3,04	-4,07	-3,85	-2,36	-4,22	-3,04	-6,36	-6,36
KO	-2,95	-3,44	-1,81	-2,83	-2,16	-2,91	-2,67	-1,83	-3,77	-2,38	-8,7	-8,7
PFE	-3,99	-4,27	-3,04	-3,68	-2,72	-5,6	-2,7	-1,62	-4,96	-3,08	-6,46	-6,46

Максимальное отклонение цен как вверх, так и вниз наблюдается у компании с тикером NFLX. С помощью программы «Код 03. Графики цен.ipynb» (используемые поля "Date", "Close", единица измерения – доллары США) построим и рассмотрим графики Netflix, Inc.

Рисунок 1. График цен NFLX за 2012 год

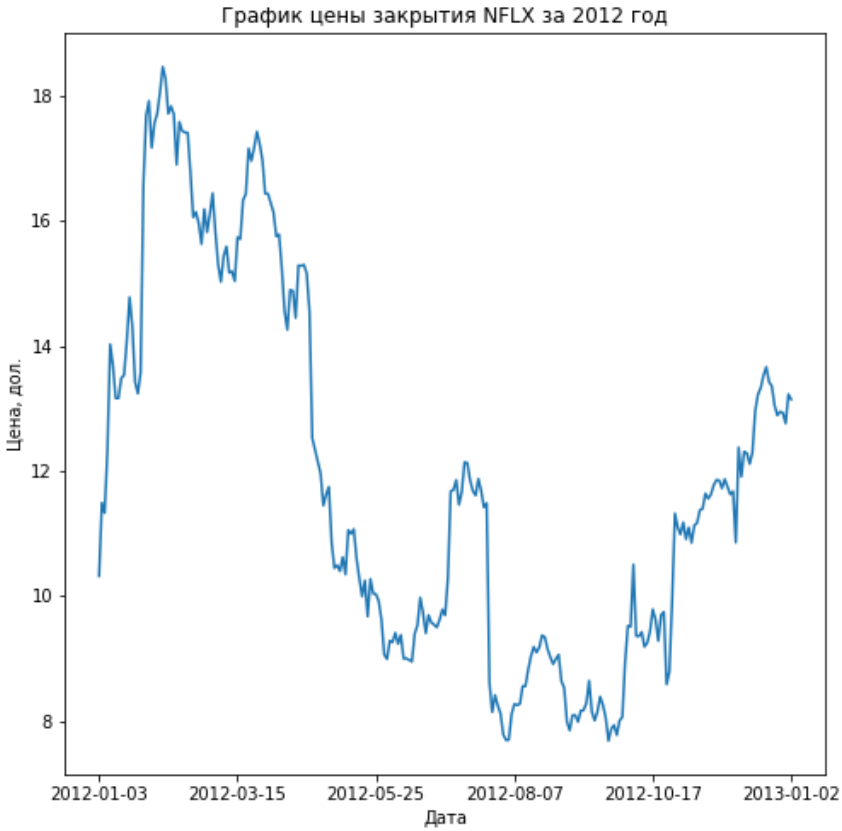
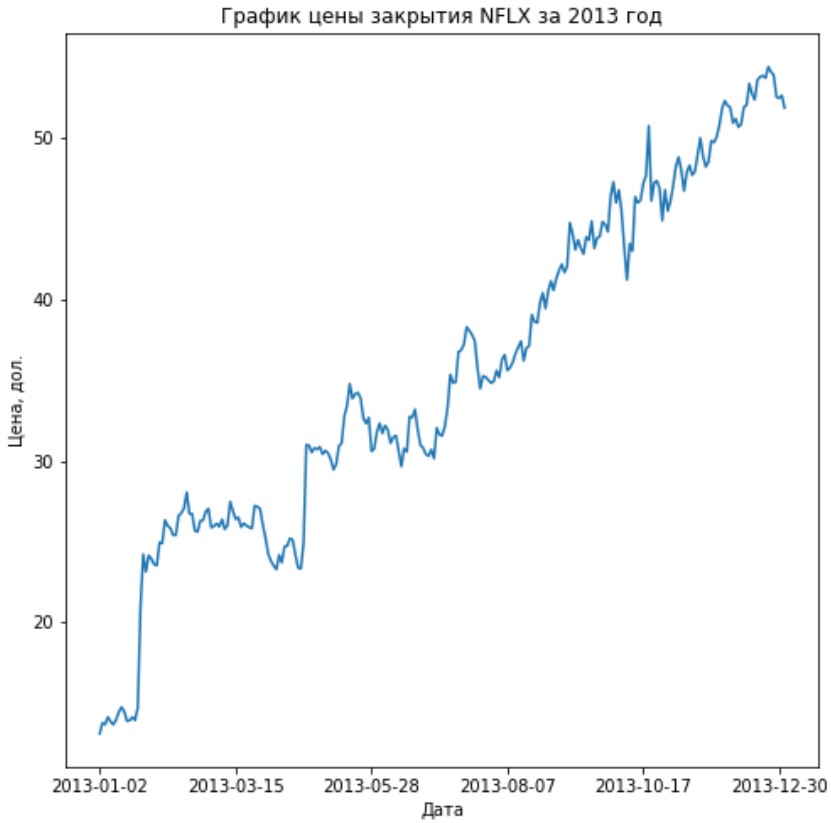


Рисунок 2. График цен NFLX за 2013 год



Судя по данным Таблицы 3 и Таблицы 4 скачки цен, которые можно увидеть на

графика на Рисунке 1 и Рисунке 2, тикер NFLX можно оставить для дальнейшего анализа, так как изменение цены не превышало 50 %.

3. Теоретическая справка по проверке гипотез

3.1. Критерий Пирсона для сложной гипотезы

В нашем случае мы оцениваем оба параметра выборки (математическое ожидание и дисперсию).

Пусть наша выборка разбита на r групп так, что i-я группа, содержащая v_i значений, принадлежит интервалу $\left(\varepsilon_i - \frac{1}{2}h; \varepsilon_i + \frac{1}{2}h\right)$, где $\varepsilon_i = \varepsilon_1 + (\mathrm{i} - 1)h$, а $h = \frac{x_{max} - x_{min}}{m}$, где $m = 1 + [\log_2 n], n -$ объем выборки. Для двух крайних групп (i = 1 и i = r) за интервалы разбиения следует принять соответственно $\left(-\infty; \varepsilon_1 + \frac{1}{2}h\right)$ и $\left(\varepsilon_r - \frac{1}{2}h; +\infty\right)$. Тогда выражения для нахождения мат. ожидания и дисперсии будут выглядеть следующим образом:

$$m = \frac{1}{n}\sum_i v_i \frac{\int x g(x)\,\mathrm{d}x}{\int g(x)\,\mathrm{d}x}$$

$$\sigma^2 = \frac{1}{n}\sum_i v_i \frac{\int (x-m)^2 g(x)\,\mathrm{d}x}{\int g(x)\,\mathrm{d}x}$$

При первом приближении получаются следующие формулы:

$$m^* = \frac{1}{n}\sum_i v_i \varepsilon_i$$

$$\sigma^{*2} = \frac{1}{n}\sum_i v_i (\varepsilon_i - m^*)^2$$

Критерий Пирсона гласит, что его статистика критерия равна:

$$\chi^2 = \sum_i^r \frac{(v_i - np_i)^2}{np_i}$$

где v_i - групповые вероятности выборки, $p_i -$ соответствующие значения заданной вероятностной функции, такое что для любой части разбиения S_i верно:

$$p_i = P(S_i)$$

$$\sum_1^r p_i = 1$$

Гипотеза H_0 отвергается, если при заданном уровне значимости α выполняется данное неравенство:

$$X^2 > X^2_{1-\alpha,r-m-1}$$

где r – количество разбиений выборки на отрезки, m – количество оцениваемых параметров, X^2 – вычисленная статистика критерия.

3.2. Критерий Колмогорова – Смирнова

В качестве вспомогательного критерия по проверке равномерности распределения P-значения основного критерия был взят критерий Колмогорова - Смирнова. Он проверяется справедливость гипотезы, позволяет произвести проверку согласия эмпирической функции распределения $\hat{F}_n(x)$ с теоретической $F(x)$.

H_0: $\hat{F}_n(x) = F(x)$. Статистика критерия Колмогорова определяется как наибольший модуль разности между указанными двумя функциями распределения $\hat{F}_n(x)$и $F(x)$:

$$D = \max |\hat{F}_n(x) - F(x)|.$$

При неограниченных объемах данных случайная величина $\lambda = D\sqrt{n}$ начинает стремиться к случайной величине Q, имеющей распределение Колмогорова:

$$p(\lambda < x) \to p(\theta \leq x) = 1 + 2\sum_{k=1}^{+\infty}(-1)^{k-1} \cdot \mathrm{e}^{2k^2x^2}, n \to \infty$$

При заданном уровне значимости α представляется возможным найти из соотношения $P(\lambda\alpha) = \alpha$ соответствующее критическое значение $\lambda\alpha$, которое как раз таки даст ответ о справедливости гипотезы H_0:

- если $\lambda < \lambda\alpha$, то считается, что предполагаемая функция распределения согласуется с полученными данными, то есть H_0 верна;
- если $\lambda > \lambda\alpha$, то гипотеза H_0 отклоняется в пользу конкурирующей H_1.

4. Проверка гипотезы на модельных данных

Перед тем как начинать работать с реальными данными, необходимо убедиться, что программы осуществляют верные действия. Поэтому практическая часть начинается с исследования на нормальность модельных данных.

Программа «Код 04. Модельные данные.ipynb» случайным образом генерирует

выборку, распределенную по нормальному закону, и высчитывает для нее значение статистики критерия Пирсона. Объем выборки равен количеству торговых дней одной компании за один год (т.е. n = 250). Затем это действие повторяется 10000 раз. Методом Монте-Карло формируется таблица 999 квантилей для распределения статистики.

Таблица 5. Квантили основной статистики

Квантиль	квартал	полугодие	год
0.1	0.9098207	1.2383337	1.7403321
0.2	1.4182825	1.9078275	2.487463
0.3	1.9362421	2.4971749	3.1974014
0.4	2.4535453	3.0766045	3.9176977
0.5	3.0407026	3.7587325	4.6915171
0.6	3.6537802	4.5116414	5.5466956
0.7	4.4850679	5.455063	6.5517714
0.8	5.6601941	6.7745567	7.932561
0.9	7.6018455	9.0408337	10.2334618

В работе приведены только 9 квантилей (0,1; 0,2; …; 0,9), остальные будут представлены в отдельном файле «Таблица 06. 999 квантилей модельных данных.csv».

Далее, с помощью программы «Код 04. Модельные данные.ipynb» проверяется равномерность, с которой Р-значения распределяются на отрезке [0;1]. Для этого в начале создается эмпирический закон из ранее полученных 999 квантилей. Далее следует вычислить Р-значения критерия Пирсона χ^2 и проверить на равномерность по критерию Колмогорова - Смирнова.

Рисунок 3. Гистограмма p-value критерия Пирсона, посчитанных вручную

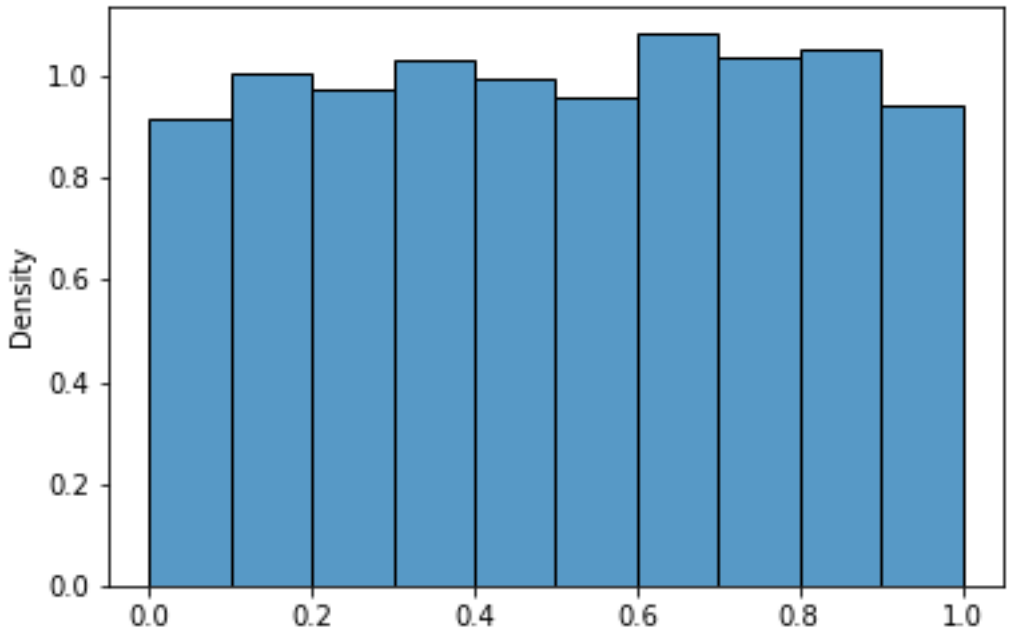
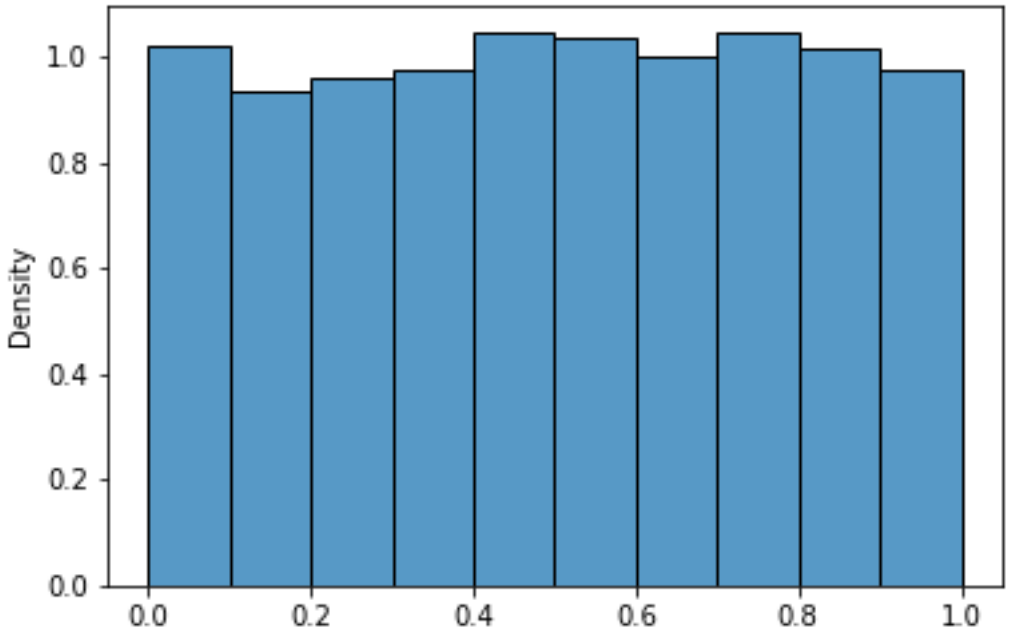


Рисунок 4. Гистограмма p-value критерия Колмогорова - Смирнова



Как мы видим, равномерность подтверждается (Рисунок 3, 4), также p-value критерия Колмогорова – Смирнова составляет 0.74454. Следовательно, что и следовало ожидать, гипотеза о нормальном распределении логарифмической доходности на модельных данных

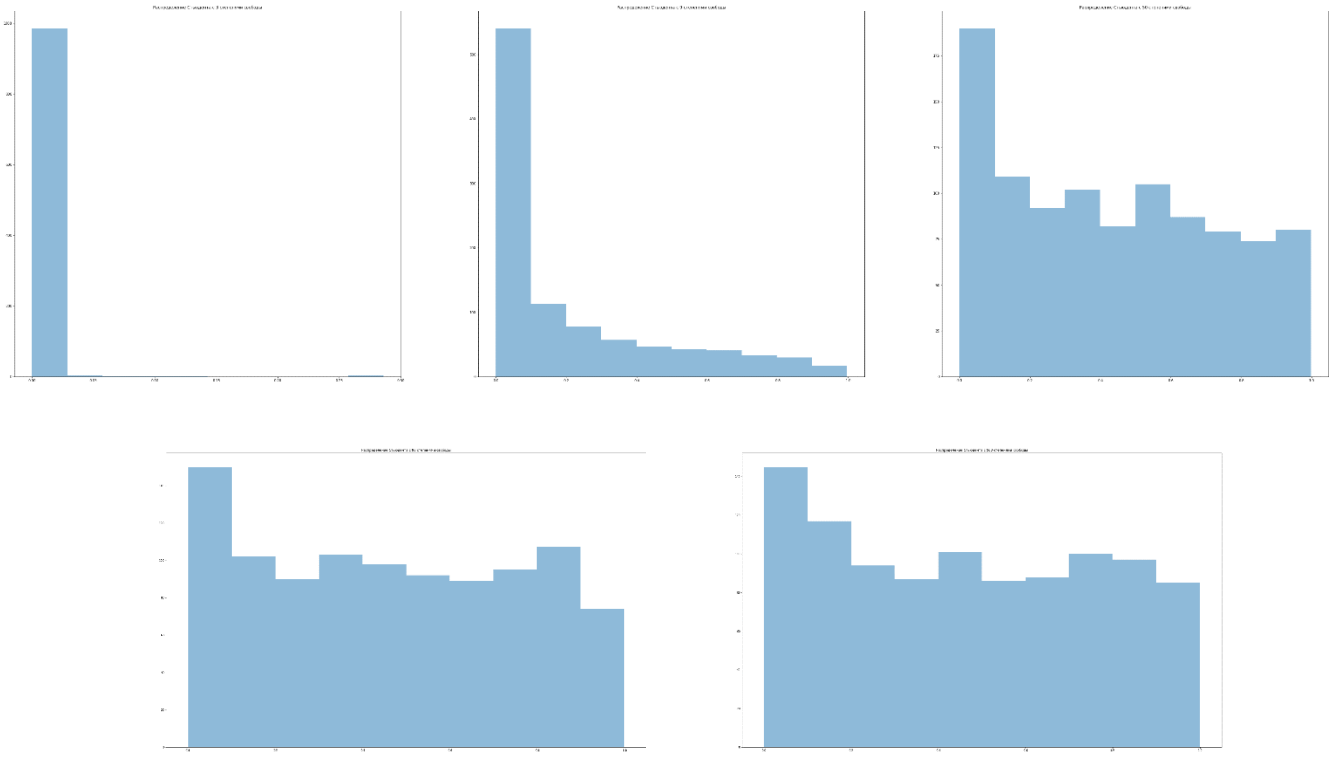
принимается.

5. Выбор альтернативной гипотезы и оценка мощности критерия

Исследуем работу программ на выборках, распределенных не по нормальному закону. В качестве альтернативных гипотез были выбраны следующие: то что логарифмическая доходность имеет распределение Стьюдента со степенями свободны t(2), t(3), t(5), t(15) и стандартное распределение Коши (частный случай распределения Стьюдента со степенями свободы t(1)). Стоит отметить, что распределение Стьюдента используется в статистике для точечного оценивания, тестирования гипотез, касающихся неизвестного среднего статистической выборки из нормального распределения, и построения доверительных интервалов.

Также эти гипотезы выбраны мной не случайно. Распределение Стьюдента сходится к стандартному нормальному при больших значениях статистик. Убедимся в этом при помощи программы «Код 06. Гистограммы p-value распределения Стьюдента.ipynb» 10000 раз вычислим статистику Пирсона χ^2 и p-value, а затем строим гистограммы p-value для распределения Стьюдента со степенями свободы t(1), t(2), t(3), t(5), t(15).

Рисунок 5–6. Гистограммы p-value критерия Пирсона для распределения Стьюдента с разными степенями свободы



С помощью программы «Код 05. Мощность критерия и распределение Стьюдента.ipynb», 10000 раз вычислим статистику Пирсона χ^2. Также следует вычислить Р-значения для этой статистики, а также мощность, поделив количество Р-значений меньше уровня 0,05 на 10000. Найдем таким способом мощность критерия стандартного распределения Коши (Таблица 6) и распределения Стьюдента с 2 (Таблица 7), 3 (Таблица 8), 5 (Таблица 9), 15 (Таблица 10) степенями свободы:

Таблица 6. Мощность критерия для стандартного распределения Коши

квартал	полугодие	год
0.9967	1.0	1.0

Таблица 7. Мощность критерия для распределения Стьюдента t(2)

квартал	полугодие	год
0.8727	0.9892	0.9999

Таблица 8. Мощность критерия для распределения Стьюдента t(3)

квартал	полугодие	год
0.6633	0.8903	0.992

Таблица 9. Мощность критерия для распределения Стьюдента t(5)

квартал	полугодие	год
0.4012	0.5958	0.8311

Таблица 10. Мощность критерия для распределения Стьюдента t(15)

квартал	полугодие	год
0.155	0.1946	0.2477

При увеличении временного интервала мощность критерия увеличивается. Для года при распределении Стьюдента со степенью свободы 2 и 3 и при стандартном распределении Коши мощность критерия достаточно высока. Это подтверждает малую вероятность ошибки второго рода и высокую мощность критерия Пирсона χ^2. Однако для остальных временных интервалов распределения Стьюдента мощность критерия получена низкая. Следствием этого является то, что вероятность ошибки второго рода для малых объемов выборки данного распределения высока.

6. Проверка гипотезы на реальных данных

Теперь нашу гипотезу необходимо проверить на реальных данных, которые мы отобрали. Программа «Код 07. Реальные данные и равномерность.ipynb» во многом повторяет «Код 04. Модельные данные.ipynb.ipynb». Высчитывается таблица p-значений для основного критерия по годам для каждой компании и строится гистограмма распределения частот p-value. Результаты работы программы представлены ниже:

Таблица 11. P-value критерия Пирсона для реальных данных (период – год)

	2010	2011	2012	2013	2014	2015	2016	2017	2018	2019	2020
AAPL	0.006	0.318	0.0	0.878	0.002	0.0	0.0	0.0	0.0	0.275	0.022
AMZN	0.0	0.127	0.282	0.0	0.005	0.0	0.0	0.0	0.0	0.003	0.672
CSCO	0.023	0.071	0.0	0.347	0.28	0.0	0.011	0.832	0.0	0.001	0.0
INTC	0.126	0.848	0.975	0.407	0.0	0.0	0.006	0.0	0.107	0.46	0.0
MCD	0.123	0.084	0.046	0.26	0.0	0.0	0.0	0.0	0.001	0.014	0.0
NKE	0.011	0.107	0.0	0.01	0.481	0.0	0.216	0.001	0.0	0.232	0.0
FDX	0.455	0.24	0.418	0.009	0.019	0.0	0.0	0.0	0.0	0.467	0.0
PG	0.558	0.152	0.041	0.0	0.977	0.0	0.016	0.0	0.0	0.0	0.0
V	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.017	0.0
DIS	0.551	0.173	0.0	0.333	0.0	0.0	0.002	0.0	0.006	0.0	0.0
GOOG	0.013	0.0	0.0	0.001	0.0	0.0	0.0	0.0	0.0	0.072	0.001
NFLX	0.025	0.0	0.0	0.0	0.003	0.002	0.0	0.0	0.004	0.25	0.389
MSFT	0.0	0.002	0.011	0.9	0.005	0.0	0.002	0.0	0.0	0.002	0.0
IBM	0.004	0.129	0.004	0.004	0.03	0.0	0.0	0.241	0.002	0.0	0.0
KO	0.001	0.0	0.321	0.025	0.1	0.01	0.001	0.001	0.0	0.74	0.0
PFE	0.059	0.077	0.0	0.0	0.048	0.0	0.005	0.0	0.0	0.294	0.0

Рисунок 7. Гистограмма p-value критерия Пирсона для реальных данных (период – год)

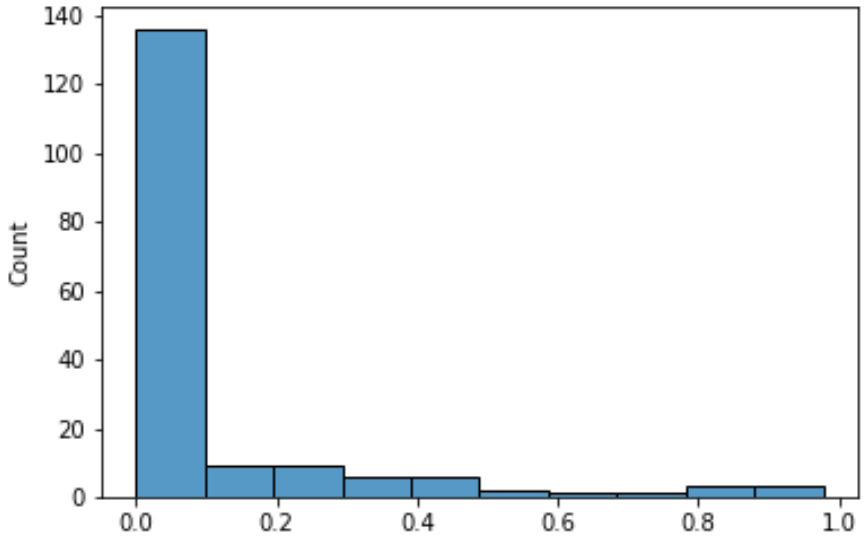
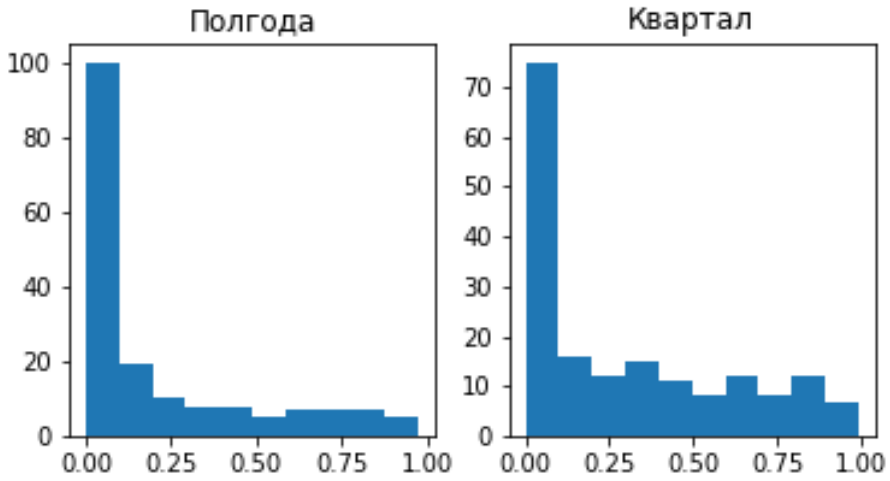


Таблица 12. Доля проверок, для которых гипотеза принималась

1%	5%	10%
0.347	0.256	0.222

По гистограмме видно, что распределение p-value неравномерно и близко к нулю. Это же подтверждает и p-value критерия Колмогорова: она равна 8.12402234523001e-22. При уменьшении рассматриваемого периода гистограмма p-value становится более равномерной:

Рисунок 8. Гистограмма p-value критерия Пирсона для реальных данных для других периодов



6.1. Проверка гипотезы о нормальном распределении дневной логарифмической доходности при условии определенного объема торгов накануне.

Проверим гипотезу при разных объемах торгов. Для этого разобьем данные на 3 интервала: с малым, средним и большим объемом торгов. Проверим основную гипотезу с помощью критерия Пирсона χ^2, используя программу «Код 08. Реальные данные при разных объемах торгов и равномерность.ipynb». Объем торгов в денежном выражении считается как произведение среднего арифметического цены открытия (поле «Open») и цены закрытия (поле «Adj Close») на объем торгов в штуках (поле «Volume»). Далее вычисляется логарифмическая доходность по средней цене. Следом вычисляется статистика Пирсона χ^2 и Р-значения. По полученным данным происходит принятие или опровержение

основной гипотезы.

Рассмотрим результаты программы за 2016 год (Рисунок 8, 9, 10). По гистограмме видно, что при увеличении объема торгов гистограмма стремится к равномерной, лучше всего при средних объемах торгов. Сократим рассматриваемый временной интервал – возьмем данные за второе полугодие и за последний квартал 2016 года. Получаем аналогичные результаты. Для всех других годов данная закономерность подтверждается. Также я построила гистограмму p-value для всех годов и всех акций (Рисунок 11). По гистограммам видно, что при средних объемах продаж гистограмма стремится к равномерной в большей степени, чем при других объемах.

Рисунок 9. Гистограммы p-value критерия Пирсона для реальных данных за 2016 год

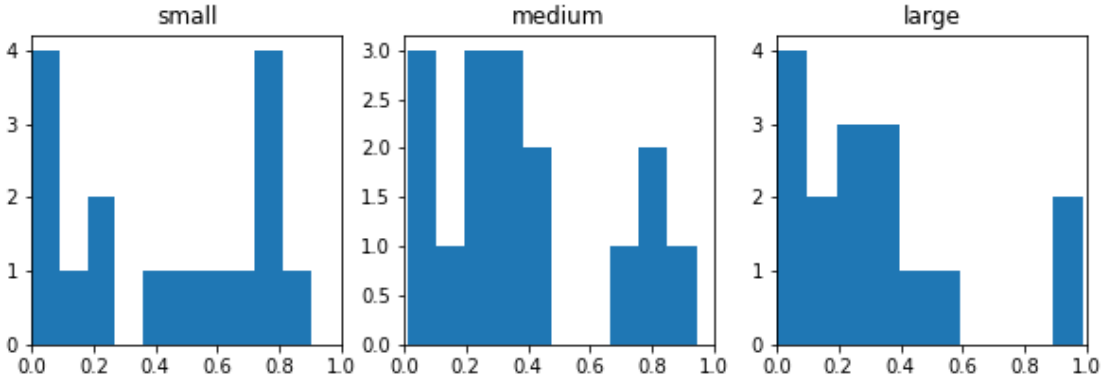


Рисунок 10. Гистограммы p-value критерия Пирсона для реальных данных за вторую половину 2016 года

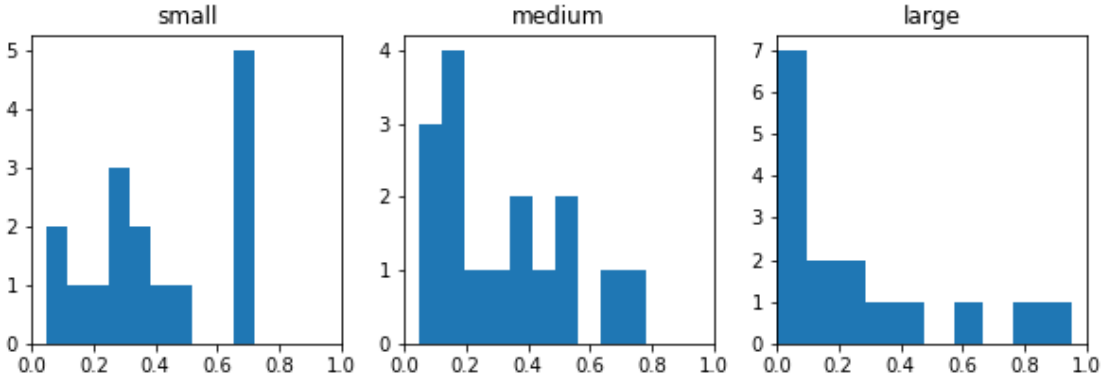


Рисунок 11. Гистограммы p-value критерия Пирсона для реальных данных за четвертый квартал 2016 года

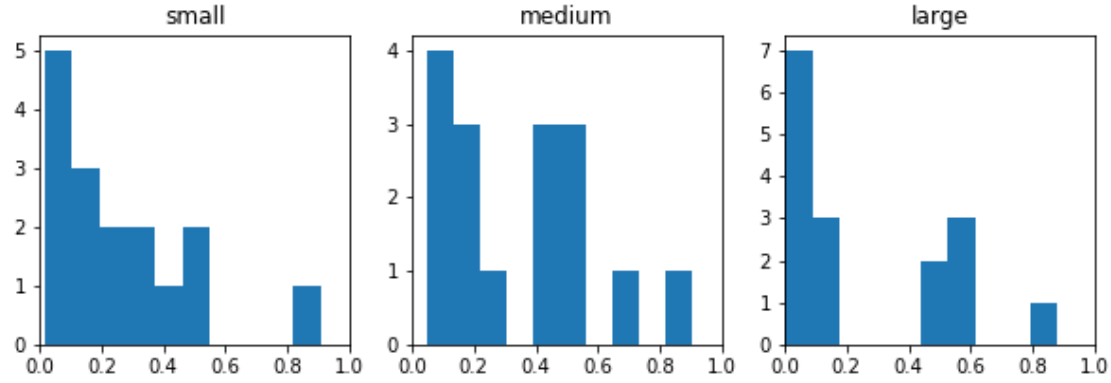
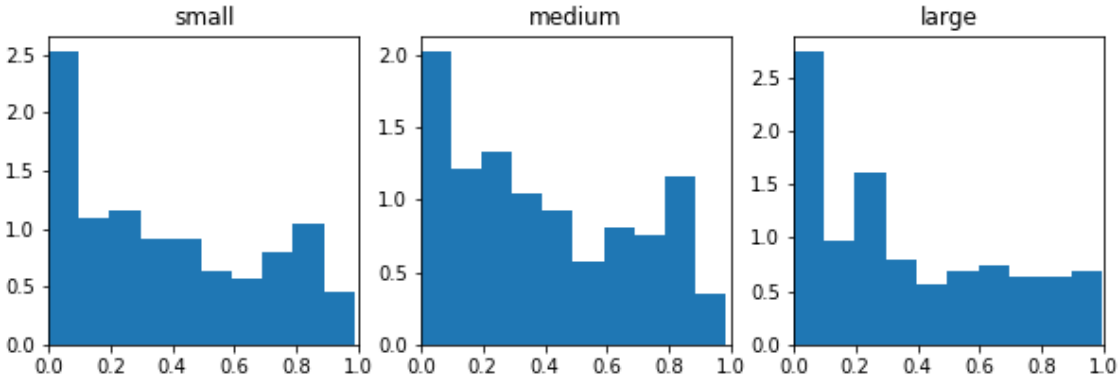


Рисунок 12. Гистограммы p-value критерия Пирсона для реальных данных за все года (2010–2020)



7. Заключение

В данной курсовой работе с помощью средств Jupyter Notebook была проведена проверка гипотезы о нормальном распределении логарифмической доходности акций компаний, входящих в листинг индекса S&P 100, при условии определенного объема торгов накануне. По результатам проведенной работы подтверждаются результаты прошлых курсовых работ, а именно: опровержение гипотезы о нормальном распределении логарифмической доходности в большинстве случаев. Однако в данном исследовании наблюдается интересная закономерность: при среднем уровне объема торгов гипотеза чаще подтверждается.

Новизна данной работы частично обусловлена новыми требованиями: мною дорабатывались программы, используемые в курсовых работах прошлых лет, а также создавались новые, проводилась оценка параметров нормального распределения.

8. Литература

1. Г. Крамер «Математические методы статистики» // Издательство «Мир». 1975.– С. 453–477;
2. Г. И. Ивченко, Ю. И. Медведев «Введение в математическую статистику» // Издательство «ЛКИ». 2009.– С. 320–332;
3. Состав индекса S&P 100// ru.tradingview.com URL: https://ru.tradingview.com/symbols/SP-OEX/components/ (дата обращения: 31.04.2021).
4. Yahoo Finance URL: https://finance.yahoo.com/ (дата обращения: 31.04.2021).
5. Кобзарь А.И. Прикладная математическая статистика. Для инженеров и научных работников. М.: ФИЗМАТЛИТ, 2006. – С. 204–209, 214–216;
6. Громова М. С. «Проверка гипотезы о нормальном распределении логарифмической доходности по критерию Дэвида-Хартли-Пирсона». Вид исследуемых данных: «Котировки акций компаний, входящих в индекс ММВБ нефти и газа» - М., 2019

9. Приложения

Приложение 1

Характеристики компьютера:

Тип процессора: Intel® Core™ i5-8250U CPU @ 1.60GHz 1.80 GHz

Память: 12 GB

Кэш-память: 6 MB Intel® Smart Cache

Время выполнения программ:

Название программы	Время работы
Код 01. Количество торговых дней.ipynb	3.55 s
Код 02. Максимальные скачки цены.ipynb	6.5 s
Код 03. Графики цен.ipynb	0.64 s
Код 04. Модельные данные.ipynb	345 s
Код 05. Мощность критерия и распределение Стьюдента.ipynb	1243 s
Код 06. Гистограммы p-value распределения Стьюдента.ipynb	376 s
Код 07. Реальные данные и равномерность.ipynb	20.5 s
Код 08. Реальные данные при разных объемах торгов и равномерность.ipynb	26.3 s

Приложение 2 – Код программ

Код 01. Количество торговых дней.ipynb (Таблица 2)

```
# подготовка библиотек
import pandas as pd

# функция, которая выводит номер начала и конца срезов таблицы соответствующие периоду одного полугодия
def half_year(df, year, half=1):
    '''
    df - датасет, в котором ищутся номера строк
    year - год, по которому нужны данные
    если half = 1 ищутся индексы строк первой половины нужного года,
    если half = 2 ищутся индексы строк второй половины нужного года
    функция выводит номер начала и конца нужного среза таблицы
    (из-за особенностей pandas фактический конец среза на единицу меньше конца, который выводится тут)
    '''
    start, end, count = 0, 0, 0
    for i in df['Date']:
        if half == 1:
            if int(i[:4]) == year and (int(i[5:7]) < 7) and start == 0:
                start = count
            if int(i[:4]) == year and (int(i[5:7]) > 6) and end == 0:
                end = count
                break # для сокращения времени работы функции
            count += 1
        elif half == 2:
            if int(i[:4]) == year and (int(i[5:7]) > 6) and start == 0:
                start = count
            if int(i[:4]) == year+1 and end == 0:
                end = count
                break
            count += 1
    if start != 0 and end == 0:
        end = count
    return start, end

# функция, которая выводит номер начала и конца срезов таблицы соответствующие периоду одного квартала
def quartal(df, year, quart=1):
    '''
    df - датасет, в котором ищутся номера строк
    year - год, по которому нужны данные
    если quart = 1 ищутся индексы строк первого квартала нужного года,
    если quart = 2 ищутся индексы строк второго квартала нужного года,
    если quart = 3 ищутся индексы строк третьего квартала нужного года,
    если quart = 4 ищутся индексы строк четвертого квартала нужного года,
    функция выводит номер начала и конца нужного среза таблицы
    (из-за особенностей pandas фактический конец среза на единицу меньше конца, который выводится тут)
    '''
    start, end, count = 0, 0, 0
    for i in df['Date']:
        if quart == 1:
            if int(i[:4]) == year and (int(i[5:7]) < 4) and start == 0:
                start = count
            if int(i[:4]) == year and (int(i[5:7]) > 3) and end == 0:
```

```
            end = count
            break # для сокращения времени работы функции
        count += 1
    elif quart == 2:
        if int(i[:4]) == year and (7 > int(i[5:7]) > 3) and start == 0:
            start = count
        if int(i[:4]) == year and (10 > int(i[5:7]) > 6) and end == 0:
            end = count
            break
        count += 1
    elif quart == 3:
        if int(i[:4]) == year and (10 > int(i[5:7]) > 6) and start == 0:
            start = count
        if int(i[:4]) == year and (int(i[5:7]) > 9) and end == 0:
            end = count
            break
        count += 1
    elif quart == 4:
        if int(i[:4]) == year and (int(i[5:7]) > 9) and start == 0:
            start = count
        if int(i[:4]) == year+1 and end == 0:
            end = count
            break
        count += 1
if start != 0 and end == 0:
    end = count

return start, end

# функция, которая выводит номер начала и конца срезов таблицы при помощи предыдущих функций
def Rows(file, year=False, half=False, quart=False):
    '''
    file - файл, из которого берется датасет
    year - год, по которому нужны данные
    half - полугодие в году, по которому нужны данные
    quart - квартал в году, по которому нужны данные
    функция выводит номер начала и конца нужного среза таблицы
    (из-за особенностей pandas фактический конец среза на единицу меньше конца, который выводится тут)
    '''

    # считывание данных из csv файла в таблицу
    df = pd.read_csv(file)

    '''
    start - начало среза
    end - конец среза +1
    count - счетчик строк

    '''

    start, end, count = 0, 0, 0
    #если не нужно извлекать полугодие, квартал или пятилетку, то находятся индексы для периода 1 год
    if not half and not quart:
        for i in df['Date']:
            if int(i[:4]) == year and start == 0:
                start = count
            if int(i[:4]) == year+1 and end == 0:
```

```
            end = count
            break # для сокращения времени работы функции
        count += 1
    # если все строки были проверены, но end не стал каким-то значение, то end - последняя строка таблицы
    if start != 0 and end == 0:
        end = count
  # если нужно извлечь полугодие, то запускается функция half_year
  elif half:
    start, end = half_year(df, year, half)
  # если нужно извлечь квартал, то запускается функция quartal
  elif quart:
    start, end = quartal(df, year, quart)
  # частный случай, который учитывается для дальнейшей работы в других функциях
  if start == 1:
    start = 0
  return start, end

# Функция, которая считает количество торговых дней в определенном году
def Num_of_days(file, year, half_year=False, quart=False):
  # считывание данных из csv файла в таблицу
  df = pd.read_csv(file)
  # при помощи функции Rows находим строку начала и конца определенного году в таблицу
  start, end = Rows(file, year, half_year, quart)
  if start == end:
    return 0
  else:
    # так как из-за особенности pandas end = конец периода в таблице + 1, то просто вычитаем из end start
    return end-start


years = range(2010, 2021)
# создаем пустую таблицу с заголовками годами
df = pd.DataFrame(columns=years)
# список тикеров
TICK = ['AAPL', 'AMZN', 'CSCO', 'INTC', 'MCD', 'NKE', 'FDX', 'PG', 'V',
    'DIS', 'FB', 'GOOG', 'NFLX', 'MSFT', 'IBM', 'KO', 'TSLA', 'PFE']
for j in TICK:
  # создание путя к файлу
  file = f'.\\ТИКЕРЫ\\{j}.csv'
  l = [] # список количества торговых дней во все года для определенного тикера
  for i in years:
    l.append(Num_of_days(file, i))
  # запись этого списка в таблицу с индексом рассматриваемого тикера
  df.loc[j] = l

# сохранение таблицы в csv файл
df.to_csv('.\\ResultWork\\Num_Of_Days.csv', sep=';')
```

Код 02. Максимальные скачки цены.ipynb (Таблица 3, 4)

```
# подготовка библиотек
import pandas as pd

# функция, которая высчитывает максимальные скачки цены вверх (в процентах)
def Cost_year(file, year):
  # считывание данных из csv файла в таблицу
```

```
    df = pd.read_csv(file)
    #добавление столбца, в который записываются скачки цен в процентах по всей таблице
    df['dif'] = (df['Close']-df['Open'])/df['Open']*100
    # получение позиций среза с помощью функции Rows (см. выше)
    start, end = Rows(file, year)
    # выводит округленное до двух знаков после запятой максимальное/минимальное значение за
определенный год при помощи среза таблицы
    return round(max(df['dif'][start:end]), 2), round(min(df['dif'][start:end]), 2)


years = range(2010, 2021)
# создаем пустую таблицу с заголовками годами
df = pd.DataFrame(columns=years)
# список тикеров
TICK = ['AAPL', 'AMZN', 'CSCO', 'INTC', 'MCD', 'NKE', 'FDX',
        'PG', 'V', 'DIS', 'GOOG', 'NFLX', 'MSFT', 'IBM', 'KO', 'PFE']
for j in TICK:
    # создание путя к файлу
    file = f'.\\ТИКЕРЫ\\{j}.csv'
    l = [] # список максимальных скачков цены вверх (в процентах) во все года для определенного тикера
    for i in years:
        l.append(Cost_year(file, i)[0])
    # запись этого списка в таблицу с индексом рассматриваемого тикера
    df.loc[j] = l

# добавления столбца максимумов значений по строкам таблицы
df['Max'] = df[range(2010, 2021)].max(axis=1)

# сохранение таблицы в csv файл
df.to_csv('.\\ResultWork\\Max_Up.csv')

#аналогичные дейтсвия для таблицы максимальных скачков вниз
df = pd.DataFrame(columns=years)

for j in TICK:
    # создание путя к файлу
    file = f'.\\ТИКЕРЫ\\{j}.csv'
    l = [] # список максимальных скачков цены вниз (в процентах) во все года для определенного тикера
    for i in years:
        l.append(Cost_year(file, i)[1])
    # запись этого списка в таблицу с индексом рассматриваемого тикера
    df.loc[j] = l

# добавления столбца максимумов значений по строкам таблицы
df['Min'] = df[range(2010, 2021)].min(axis=1)

# сохранение таблицы в csv файл
df.to_csv('.\\ResultWork\\Max_Down.csv')
```

Код 03. Графики цен.ipynb (Рисунок 1, 2)

```
# подготовка библиотек
import matplotlib.ticker as ticker
import matplotlib.pyplot as plt
import pandas as pd

# функция, которая возвращает массив цен и дат (таблицу) за нужный период
```

```
def Prices(file, year, half_year=False, quart=False):
    # считывание данных из csv файла в таблицу
    df = pd.read_csv(file)
    # получение позиций среза с помощью функции Rows (см. выше)
    start, end = Rows(file, year, half_year, quart)
    return df.loc[start:end, 'Date':'Close']


# нужный тикер и  год
stock = 'NFLX'
year = 2012

# создание путя к файлу
file = f'.\\ТИКЕРЫ\\{stock}.csv'

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

# создание графика цен за нужный год при помощи функции Prices
price = Prices(file, year)
# по оси икс - даты, по оси игрек - цены
ax.plot(price['Date'], price['Close'])
ax.xaxis.set_major_locator(ticker.MultipleLocator(50))
ax.xaxis.set_minor_locator(ticker.MultipleLocator(50))
ax.set_title(f'График цены закрытия {stock} за {year} год')
plt.xlabel('Дата')
plt.ylabel('Цена, дол.')
# сохранение графика в формате png
plt.savefig('.\\ResultWork\\Max_Up_Stock.png')

# нужный тикер и  год
stock = 'NFLX'
year = 2013

# создание путя к файлу
file = f'.\\ТИКЕРЫ\\{stock}.csv'

fig, ax = plt.subplots(figsize=(8, 8))
# создание графика цен за нужный год при помощи функции Prices
price = Prices(file, year)
# по оси икс - даты, по оси игрек - цены
ax.plot(price['Date'], price['Close'])
ax.xaxis.set_major_locator(ticker.MultipleLocator(50))
ax.xaxis.set_minor_locator(ticker.MultipleLocator(50))
ax.set_title(f'График цены закрытия {stock} за {year} год')
plt.xlabel('Дата')
plt.ylabel('Цена, дол.')
# сохранение графика в формате png
plt.savefig('.\\ResultWork\\Max_Down_Stock.png')
```

Код 04. Модельные данные.ipynb (Таблица 5, Рисунок 3, 4)

```
# подготовка библиотек
from math import sqrt, log, log2, isnan
import scipy.stats as sts
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
```

```
# функция, которая осуществляет оценку параметров θ1 и θ2 (мат.ожидание и дисперсию) сгенерированной рандомной выборки
def ozenka_tetta(Data):
    # упорядочивает сгенерированную выборку
    Data = sorted(Data)
    n = len(Data)
    m = 1+int(log2(n))  # по формуле Стерджесса расчитывает количество интервалов разбиения выборки
    h = (max(Data)-min(Data))/m #шаг разбиения
    eps = [min(Data)] # список который будет собирать центральны точки отрезков разбиения (первая точка в разбиении - минимум выборки)

    Razb = [] # список, который будет собирать разбитые на отрезки данные
    Razb.append([x for x in Data if x < eps[-1]+h/2])
    for i in range(2, m):
        eps.append(eps[0]+(i-1)*h)
        Razb.append([x for x in Data if eps[-1]-0.5*h < x < eps[-1]+0.5*h])

    eps.append(eps[0]+(m-1)*h)
    Razb.append([x for x in Data if x > eps[-1]-0.5*h])

    # список частот (количество элементов в интервалах)
    V = [len(x) for x in Razb]

    m = [] # список значений, которые будут суммировать для получения θ1
    for i in range(len(V)):
        m.append(V[i]*eps[i])
    m_ = 1/n*sum(m) # высчитывает θ1 по формуле

    s = [] # список значений, которые будут суммировать для получения θ2
    for i in range(len(V)):
        s.append(V[i]*(eps[i]-m_)**2)
    sigma_ = 1/n*sum(s) # высчитывает θ2 по формуле

    # выводим θ1, θ2, список частот, список центральных точек отрезков разбиений
    return m_, sigma_, V, eps

def Pirsons(per):
    n = per
    # генерация нормальной выборки
    Data = sts.norm(0, 1).rvs(n)
    m = 1+int(log2(n))  # по формуле Стерджесса расчитывает количество интервалов разбиения выборки
    h = (max(Data)-min(Data))/m # шаг разбиения
    func = ozenka_tetta(Data) # получение нужный данных при помощи функции ozenka_tetta
    m, sigma, eps, V = func[0], func[1], func[3], func[2] # считывание необходимых данных для критерия Пирсона
    Exp = sts.norm(m, sqrt(sigma)) # ожидаемое распределение (то, с которой будем сравнивать рандомную выборку)
    # формирование списка вероятностей попадания в интервалы разбиения
    P = [Exp.cdf(eps[0]+0.5*h)]
    P += [Exp.cdf(eps[x]+0.5*h)-Exp.cdf(eps[x]-0.5*h)
          for x in range(1, len(eps)-1)]
    P += [1-Exp.cdf(eps[-1]-0.5*h)]

    # высчитываем статистику критерия Пирсона
    Z = []
    for i in range(len(P)):
```

```
        Z.append(((V[i]-n*P[i])**2)/(n*P[i]))
    chi_2 = sum(Z)
    pvalue = (sts.chi2(len(V)-3).sf(chi_2))
    # возвращаем статистику критерия Пирсона и p-value
    return chi_2, pvalue


N = 10000 # количество значений статистики

# Словарь разных объемов выборки, соответствующие разным временным интервалам
periods = {'квартал': 63, 'полугодие': 126, 'год': 252}
# создание пустых таблицы для 9 и 999 квантилей
df_9, df_999 = pd.DataFrame(), pd.DataFrame()

# заполняем таблицы перебором всех возможных объемов выборки
for i, j in periods.items():
    # вычисление 10 000 значений статистики
    pirs = [Pirsons(j) for _ in range(N)]

    # списки значений статистики и p-value
    chi2, pv_pirs = [x[0] for x in pirs], [x[1] for x in pirs]

    # вычисление 9 и 999 квантилей
    chi2_q999 = np.quantile(chi2, np.arange(0.001, 1, 0.001))
    chi2_q9 = np.quantile(chi2, np.arange(0.1, 1, 0.1))

    # вычисление 999 p-value для постоения гистограммы
    pv_q999 = np.quantile(pv_pirs, np.arange(0.001, 1, 0.001))

    # запись в таблицы округленных значений
    df_9[i] = np.round(chi2_q9,7)
    df_999[i] = np.round(chi2_q999,7)

# изменение значений индексов строк в таблицах
ind = [round(i, 1) for i in list(np.arange(0.1, 1, 0.1))]
df_9.index = ind
ind = [round(i, 3) for i in list(np.arange(0.001, 1, 0.001))]
df_999.index = ind

# сохранение таблиц в csv файлы
df_9.to_csv('.\\ResultWork\\Quant_9.csv', sep=';', encoding='cp1251')
df_999.to_csv('.\\ResultWork\\Quant_999.csv', sep=';', encoding='cp1251')


n = 252

chi2 = [Pirsons(n)[0] for _ in range(N)]
chi2_q999 = np.quantile(chi2, np.arange(0.001, 1, 0.001))

# список p-value критерия Пирсона вручную вычисленных вручную
pvalue_pirs = []
for _ in range(N):
    u0 = Pirsons(n)[0]
    k = 0
    for i in range(len(chi2_q999)):
        if chi2_q999[i] > u0:
            k += 1
```

```
    pvalue_pirs.append(k/len(chi2_q999)) #p-value критерия Пирсона вручную

# список p-value критерия Колмогорова - Смирнова
pvalue_ks = []
n=250
for _ in range(N):
    Data = sts.norm(0, 1).rvs(n)
    pvalue_ks.append(sts.kstest(Data,'norm')[1]) #p-value критерия Колмогорова - Смирнова

sns.histplot(pvalue_ks,bins=10,stat='density')
plt.savefig(".\\ResultWork\\Kolmogorov-Smornov_test.png")
plt.show()
sns.histplot(pvalue_pirs,bins=10,stat='density')
plt.savefig(".\\ResultWork\\Pirsons_test.png")

# проверка, что распределения p-значений идентичны
pvalue = sts.ks_2samp(pvalue_ks,pvalue_pirs)
```

Код 05. Мощность критерия и распределение Стьюдента.ipynb (Таблица 6, 7, 8, 9, 10)

```
# подготовка библиотек
from math import sqrt, log, log2, isnan
import scipy.stats as sts
import pandas as pd


N = 10000 # количество значений статистики

# Словарь разных объемов выборки, соответствующие разным временным интервалам
periods = {'квартал': 63, 'полугодие': 126, 'год': 252}

st = [1,2,3,5,15] # количество степеней свободы

for stepen in st:
    # создание пустой таблицы для записи значений мощности критерия
    pv = pd.DataFrame(columns=list(periods.keys()))

    for l, n in periods.items():
        pvalue = [] #список p-value критерия Пирсона
        for _ in range(N):
            # генерируем выборку нужного объема распределения Стьюдента
            Data = sts.t(stepen).rvs(n)

            # данный кусок кода поясняется выше
            m = 1+int(log2(n))
            h = (max(Data)-min(Data))/m
            func = ozenka_tetta(Data)
            m, sigma, eps, V = func[0], func[1], func[3], func[2]
            Exp = sts.norm(m, sqrt(sigma))
            P = [Exp.cdf(eps[0]+0.5*h)]
            P += [Exp.cdf(eps[x]+0.5*h)-Exp.cdf(eps[x]-0.5*h) for x in range(1, len(eps)-1)]
            P += [1-Exp.cdf(eps[-1]-0.5*h)]

            Z = []
            for i in range(len(P)):
                Z.append(((V[i]-n*P[i])**2)/(n*P[i]))
            chi_2 = sum(Z)
```

```
            pvalue.append(sts.chi2(len(V)-3).sf(chi_2))
        # отбираем p-value меньше 0.05
        PV = [v for v in pvalue if v < 0.05 or isnan(v)]
        # запись мощности критерия в таблицу
        pv[l] = [len(PV)/N]

    #сохранение таблицы в файл
    pv.to_csv(f'.\\ResultWork\\Мощность_крит_t({stepen}).csv', sep=';', encoding='cp1251',index=False)
```

Код 06. Гистограммы p-value распределения Стьюдента.ipynb (Рисунок 5,6)

```
# подготовка библиотек
from math import sqrt, log, log2, isnan
import scipy.stats as sts
import matplotlib.pyplot as plt

fig, ax = plt.subplots(1, 3, figsize=(70, 20))
n = 250
for l, k in enumerate([3, 9, 50]):
    pvalue = []
    n = 250
    for _ in range(10000):
        # генерируем выборку нужного объема распределения Стьюдента
        Data = sts.t(k).rvs(n)
        # данный кусок кода поясняется выше
        m = 1+int(log2(n))  # формула Стерджесса
        h = (max(Data)-min(Data))/m
        func = ozenka_tetta(Data)
        m, sigma, eps, V = func[0], func[1], func[3], func[2]
        Exp = sts.norm(m, sqrt(sigma))
        P = [Exp.cdf(eps[0]+0.5*h)]
        P += [Exp.cdf(eps[x]+0.5*h)-Exp.cdf(eps[x]-0.5*h)
              for x in range(1, len(eps)-1)]
        P += [1-Exp.cdf(eps[-1]-0.5*h)]

        Z = []
        for i in range(len(P)):
            Z.append(((V[i]-n*P[i])**2)/(n*P[i]))
        chi_2 = sum(Z)

        pvalue.append(sts.chi2(len(V)-3).sf(chi_2))

    ax[l].hist(pvalue, alpha=0.5)
    ax[l].set_title(f'Распределение Стьюдента с {k} степенями свободы')
plt.savefig(".\\ResultWork\\Student's_raspr_1.png")

#аналогичный код с другими значениями
fig, ax = plt.subplots(1, 2, figsize=(70, 20))
n = 250
for l, k in enumerate([ 90, 500]):
    pvalue = []
    n = 250
    for _ in range(10000):

        Data = sts.t(k).rvs(n)
        m = 1+int(log2(n))  # формула Стерджесса
```

```
        h = (max(Data)-min(Data))/m
        func = ozenka_tetta(Data)
        m, sigma, eps, V = func[0], func[1], func[3], func[2]
        Exp = sts.norm(m, sqrt(sigma))
        P = [Exp.cdf(eps[0]+0.5*h)]
        P += [Exp.cdf(eps[x]+0.5*h)-Exp.cdf(eps[x]-0.5*h)
              for x in range(1, len(eps)-1)]
        P += [1-Exp.cdf(eps[-1]-0.5*h)]

        Z = []
        for i in range(len(P)):
           Z.append(((V[i]-n*P[i])**2)/(n*P[i]))
        chi_2 = sum(Z)

        pvalue.append(sts.chi2(len(V)-3).sf(chi_2))

    ax[l].hist(pvalue, alpha=0.5)
    ax[l].set_title(f'Распределение Стьюдента с {k} степенями свободы')
plt.savefig(".\\ResultWork\\Student's_raspr_2.png")
```

Код 07. Реальные данные и равномерность.ipynb (Таблица 11, 12, Рисунок 7, 8)

```
# подготовка библиотек
from math import sqrt, log, log2, isnan
import scipy.stats as sts
import matplotlib.pyplot as plt
import pandas as pd

def Pirsons(file, year=False, half=False, quart=False):
    # высчитываем объем выборки при помощи функции Num_of_days (см.выше)
    n = Num_of_days(file, year, half, quart)
    df = pd.read_csv(file)
    start, end = Rows(file, year, half, quart)
    # формируем список лог доходностей определенного периода и акции
    Data = np.log(df['Close']/df['Open'])[start:end]
    # данный кусок кода поясняется выше
    m = 1+int(log2(n))  # формула Стерджесса
    h = (max(Data)-min(Data))/m
    func = ozenka_tetta(Data)
    m, sigma, eps, V = func[0], func[1], func[3], func[2]
    Exp = sts.norm(m, sqrt(sigma))
    P = [Exp.cdf(eps[0]+0.5*h)]
    P += [Exp.cdf(eps[x]+0.5*h)-Exp.cdf(eps[x]-0.5*h) for x in range(1, len(eps)-1)]
    P += [1-Exp.cdf(eps[-1]-0.5*h)]

    Z = []
    for i in range(len(P)):
        Z.append(((V[i]-n*P[i])**2)/(n*P[i]))
    chi_2 = sum(Z)
    pvalue = (sts.chi2(len(V)-3).sf(chi_2))

    return chi_2, round(pvalue, 3)

years = range(2010, 2021)
df = pd.DataFrame(columns=years)
TICK = ['AAPL', 'AMZN', 'CSCO', 'INTC', 'MCD', 'NKE', 'FDX',
        'PG', 'V', 'DIS', 'GOOG', 'NFLX', 'MSFT', 'IBM', 'KO', 'PFE']
```

```
pvalue_pirs=[] # список p-value критерия Пирсона на реальных данных
for j in TICK:
    file = f'.\\ТИКЕРЫ\\{j}.csv'
    l = []
    for i in years:
        l.append(Pirsons(file, year = i)[1])
    df.loc[j] = l # заполняем таблицу
    pvalue_pirs += l
# сохраняем таблицу и гистограмму в файлы
df.to_csv(f'.\\ResultWork\\Real_data_year.csv', sep=';')
sns.histplot(pvalue_pirs,bins=10)
plt.savefig(".\\ResultWork\\Real_data.png")

""" аналогично для других периодов """
pvalue_half = []
pvalue_quart = []
year = range(2010, 2021)
TICK = ['AAPL', 'AMZN', 'CSCO', 'INTC', 'MCD', 'NKE', 'FDX',
        'PG', 'V', 'DIS', 'GOOG', 'NFLX', 'MSFT', 'IBM', 'KO', 'PFE']
for j in TICK:
    file = f'.\\ТИКЕРЫ\\{j}.csv'
    l, k = [], []
    for i in year:
        l.append(Pirsons(file, i, half=2)[1])
        k.append(Pirsons(file, i, quart=1)[1])
    pvalue_half += l
    pvalue_quart += k


fig, ax = plt.subplots(1, 2, figsize = (6,3))
ax[0].hist(pvalue_half, bins=10)
ax[0].set_title('Полгода')
ax[1].hist(pvalue_quart, bins=10)
ax[1].set_title('Квартал')

plt.savefig(f".\\ResultWork\\Real_data_for_different_periods.png")

#высчитывание доли проверок для которых гипотеза принималась
pv001, pv005, pv010 = 0, 0, 0
pv = pd.DataFrame(columns=['1%','5%','10%'])
for j in range(len(TICK)):
    for i in range(len(years)):
        if df.iloc[j,i]>0.01:
            pv001 = pv001+1
        if df.iloc[j,i]>0.05:
            pv005 = pv005+1
        if df.iloc[j,i]>0.1:
            pv010 = pv010+1
percent =
[round(pv001/len(TICK)/len(years),3),round(pv005/len(TICK)/len(years),3),round(pv010/len(TICK)/len(years),3)]
pv.loc[0] = percent
pv.to_csv(f'.\\ResultWork\\Real_data_percent.csv', sep=';',index=False)
```

Код 08. Реальные данные при разных объемах торгов и равномерность.ipynb (Рисунок 9, 10, 11, 12)

```
# подготовка библиотек
```

```
from math import sqrt, log, log2, isnan
import scipy.stats as sts
import matplotlib.pyplot as plt
import pandas as pd

def Volumes(file, year = False, half = False, quart = False):
    # считывание данных из csv файла в таблицу
    df = pd.read_csv(file)
    # получение позиций среза с помощью функции Rows (см. выше)
    start, end = Rows(file, year = year, half = half, quart = quart)
    # формируем столбец новой цены и новой объема продаж, высчитываем лог доходность также в новых
столбец
    df['new_price'] = (df['Adj Close'] + df['Open']) / 2
    df['new_vol'] = df['new_price'] * df['Volume']
    df['log'] = np.log(df['Close'] / df['new_price'])
    return df.loc[start:end]


def Pirsons(file, size,  year = False, half = False, quart = False):
    # получаем таблицу при помощи функции Volumes
    df = Volumes(file, year = year,  half = half, quart = quart)
    df.index = range(df.shape[0])
    # находим квантили 1/3 и 2/3 уровней в столбце нового объема продаж
    quant = np.quantile(df['new_vol'], np.arange(0, 1, 1/3)[1:])
    Data = []
    # разбиваю данные в соответсвии с нужным параметром (какой объем нужен)
    if size == 1:
        for i in range(df.shape[0]):
            if df.at[i,"new_vol"] <= quant[size - 1]:
                Data.append(df.at[i,'log'])
    elif size == 2:
        for i in range(df.shape[0]):
            if quant[size - 2] < df.at[i,"new_vol"] <= quant[size - 1]:
                Data.append(df.at[i,'log'])
    else:
        for i in range(df.shape[0]):
            if df.at[i,"new_vol"] > quant[size - 2]:
                Data.append(df.at[i,'log'])

    # данный кусок кода поясняется выше
    n = len(Data)
    m = 1+int(log2(n))  # формула Стерджесса
    h = (max(Data)-min(Data))/m
    func = ozenka_tetta(Data)
    m, sigma, eps, V = func[0], func[1], func[3], func[2]
    Exp = sts.norm(m, sqrt(sigma))
    P = [Exp.cdf(eps[0]+0.5*h)]
    P += [Exp.cdf(eps[x]+0.5*h)-Exp.cdf(eps[x]-0.5*h) for x in range(1, len(eps)-1)]
    P += [1-Exp.cdf(eps[-1]-0.5*h)]

    Z = []
    for i in range(len(P)):
        Z.append(((V[i]-n*P[i])**2)/(n*P[i]))
    chi_2 = sum(Z)
    pvalue = (sts.chi2(len(V)-3).sf(chi_2))
    return chi_2, round(pvalue, 3)
```

```
pvalue = [] # список p-value
sizes = {'small': 1,'medium': 2,'large': 3} # соответсвие объема продаж параметру для функции
years = range(2010, 2021)
year = 2016
TICK = ['AAPL', 'AMZN', 'CSCO', 'INTC', 'MCD', 'NKE', 'FDX',
        'PG', 'V', 'DIS', 'GOOG', 'NFLX', 'MSFT', 'IBM', 'KO', 'PFE']
l1, l2, l3 = [], [], [] # списки p-value для разных объемов продаж
for j in TICK:
    file = f'.\\ТИКЕРЫ\\{j}.csv'

    for k in years:

        l1.append(Pirsons(file, 1, year = k, half = False, quart = False)[1])
        l2.append(Pirsons(file, 2, year = k, half = False, quart = False)[1])
        l3.append(Pirsons(file, 3, year = k, half = False, quart = False)[1])

    pvalue += l1+l2+l3

# создание гистограмм
fig,ax = plt.subplots(1,3,figsize=(10,3))
size=list(sizes.keys())
ax[0].hist(l1,density=True)
ax[0].set_xlim(0,1)
ax[0].set_title(size[0])

ax[1].hist(l2,density=True)
ax[1].set_xlim(0,1)
ax[1].set_title(size[1])

ax[2].hist(l3,density=True)
ax[2].set_xlim(0,1)
ax[2].set_title(size[2])

plt.savefig(f".\\ResultWork\\Volume_for_years.png")

# аналогичный код для других гистограмм
pvalue = []
sizes = {'small': 1,'medium': 2,'large': 3}
years = range(2010, 2021)
year = 2016
df = pd.DataFrame(columns = sizes.keys())
TICK = ['AAPL', 'AMZN', 'CSCO', 'INTC', 'MCD', 'NKE', 'FDX',
        'PG', 'V', 'DIS', 'GOOG', 'NFLX', 'MSFT', 'IBM', 'KO', 'PFE']
for j in TICK:
    file = f'.\\ТИКЕРЫ\\{j}.csv'
    l = []
    for i in sizes.values():
        l.append(Pirsons(file, i, year = year, half = False, quart = False)[1])

    df.loc[j] = l
    pvalue += l


fig,ax = plt.subplots(1,3,figsize=(10,3))
size=list(sizes.keys())
for i in range(3):
    ax[i].hist(df[size[i]],bins=10)
```

```
    ax[i].set_xlim(0,1)
    ax[i].set_title(size[i])
plt.savefig(f".\\ResultWork\\Volume_for_{year}.png")

pvalue = []
sizes = {'small': 1,'medium': 2,'large': 3}
years = range(2010, 2021)
year = 2016
df = pd.DataFrame(columns = sizes.keys())
TICK = ['AAPL', 'AMZN', 'CSCO', 'INTC', 'MCD', 'NKE', 'FDX',
        'PG', 'V', 'DIS', 'GOOG', 'NFLX', 'MSFT', 'IBM', 'KO', 'PFE']
for j in TICK:
    file = f'.\\ТИКЕРЫ\\{j}.csv'
    l = []
    for i in sizes.values():
        l.append(Pirsons(file, i, year = year, half = 2, quart = False)[1])

    df.loc[j] = l
    pvalue += l

fig,ax = plt.subplots(1,3,figsize=(10,3))
size=list(sizes.keys())
for i in range(3):
    ax[i].hist(df[size[i]],bins=10)
    ax[i].set_xlim(0,1)
    ax[i].set_title(size[i])
plt.savefig(f".\\ResultWork\\Volume_for_second_half_of_{year}.png")

pvalue = []
sizes = {'small': 1,'medium': 2,'large': 3}
years = range(2010, 2021)
year = 2016
df = pd.DataFrame(columns = sizes.keys())
TICK = ['AAPL', 'AMZN', 'CSCO', 'INTC', 'MCD', 'NKE', 'FDX',
        'PG', 'V', 'DIS', 'GOOG', 'NFLX', 'MSFT', 'IBM', 'KO', 'PFE']
for j in TICK:
    file = f'.\\ТИКЕРЫ\\{j}.csv'
    l = []
    for i in sizes.values():
        l.append(Pirsons(file, i, year = year, half = False, quart = 4)[1])

    df.loc[j] = l
    pvalue += l

fig,ax = plt.subplots(1,3,figsize=(10,3))
size=list(sizes.keys())
for i in range(3):
    ax[i].hist(df[size[i]])
    ax[i].set_xlim(0,1)
    ax[i].set_title(size[i])

plt.savefig(f".\\ResultWork\\Volume_for_fourth_quartal_of_{year}.png")
```

Приложение 3 - Список файлов

Имя файла	Имя программы
Рисунок 1. График цен NFLX за 2012 год	Код 03. Графики цен.ipynb
Рисунок 2. График цен NFLX за 2013 год	Код 03. Графики цен.ipynb
Рисунок 3. Гистограмма p-value критерия Пирсона, посчитанных вручную	Код 04. Модельные данные.ipynb
Рисунок 4. Гистограмма p-value критерия Колмогорова – Смирнова	Код 04. Модельные данные.ipynb
Рисунок 5–6. Гистограммы p-value критерия Пирсона для распределения Стьюдента с разными степенями свободы	Код 06. Гистограммы p-value распределения Стьюдента.ipynb
Рисунок 7. Гистограмма p-value критерия Пирсона для реальных данных (период – год)	Код 07. Реальные данные и равномерность.ipynb
Рисунок 8. Гистограмма p-value критерия Пирсона для реальных данных для других периодов	Код 07. Реальные данные и равномерность.ipynb
Рисунок 9. Гистограммы p-value критерия Пирсона для реальных данных за 2016 год	Код 08. Реальные данные при разных объемах торгов и равномерность.ipynb
Рисунок 10. Гистограммы p-value критерия Пирсона для реальных данных за вторую половину 2016 года	Код 08. Реальные данные при разных объемах торгов и равномерность.ipynb
Рисунок 11. Гистограммы p-value критерия Пирсона для реальных данных за четвертый квартал 2016 года	Код 08. Реальные данные при разных объемах торгов и равномерность.ipynb
Рисунок 12. Гистограммы p-value критерия Пирсона для реальных данных за все года (2010–2020)	Код 08. Реальные данные при разных объемах торгов и равномерность.ipynb
Таблица 1. Список компаний	-
Таблица 2. Количество торговых дней	Код 01. Количество торговых дней.ipynb
Таблица 3. Максимальные скачки цены вверх (в %)	Код 02. Максимальные скачки цены.ipynb
Таблица 4. Максимальные скачки цены вниз (в %)	Код 02. Максимальные скачки цены.ipynb
Таблица 5. Квантили основной статистики	Код 04. Модельные данные.ipynb
Таблица 6. Мощность критерия для стандартного распределения Коши	Код 05. Мощность критерия и распределение Стьюдента.ipynb
Таблица 7. Мощность критерия для распределения Стьюдента t(2)	Код 05. Мощность критерия и распределение Стьюдента.ipynb
Таблица 8. Мощность критерия для распределения Стьюдента t(3)	Код 05. Мощность критерия и распределение Стьюдента.ipynb
Таблица 9. Мощность критерия для распределения Стьюдента t(5)	Код 05. Мощность критерия и распределение Стьюдента.ipynb
Таблица 10. Мощность критерия для распределения Стьюдента t(15)	Код 05. Мощность критерия и распределение Стьюдента.ipynb
Таблица 11. P-value критерия Пирсона для реальных данных (период – год)	Код 07. Реальные данные и равномерность.ipynb
Таблица 12. Доля проверок, для которых гипотеза принималась	Код 07. Реальные данные и равномерность.ipynb