Факультет информационных технологий и анализа больших данных

Департамент анализа данных, принятия решений и финансовых технологий

Курсовая работа на тему:

**«Проверка гипотезы о нормальном распределении логарифмической**

**доходности по критерию Шапиро-Франчиа»**

Вид исследуемых данных:

**Котировки акций компаний, входящих в индекс МосБиржи голубые фишки (MOEXBC)**

Выполнил:

студент группы ПМ19-4

Королев Д.М.

Научный руководитель:

к.ф.-м.н. Пяткина Д.А.

План:

[Введение 3](#_Toc71530770)

[Глава 1. Теоретическая часть 4](#_Toc71530771)

[1.1 Статистическая проверка гипотез 4](#_Toc71530772)

[1.2. Критерий Шапиро-Франчиа 5](#_Toc71530773)

[1.3 Критерий Колмогорова-Смирнова 6](#_Toc71530774)

[Глава 2. Практическая часть 7](#_Toc71530775)

[2.1 Проверка гипотезы на модельных данных 7](#_Toc71530776)

[Заключение 15](#_Toc71530777)

[Список используемых источников 16](#_Toc71530778)

[Литература 16](#_Toc71530779)

[Интернет-ресурсы 16](#_Toc71530780)

[Приложение 17](#_Toc71530781)

[Приложение 1. Характеристики компьютера 17](#_Toc71530782)

[Приложение 2. Коды программ 17](#_Toc71530783)

# **Введение**

В этой работе будет осуществлена проверка гипотезы о нормальном распределении логарифмических доходностей рассчитанных по ценам акций, входящих в состав индекса МосБиржи голубые фишки (MOEXBC), в период времени: с 01.01.2018 по 31.12.2020. Данные об акциях компаний, которые входят в индекс взяты с финансового портала Финам (https://www.finam.ru). Проверка гипотезы в данной работе будет осуществляться по критерию Шапиро-Франчиа.

Текущая курсовая работа включает в себя две части: теоретическую и практическую. Первая часть (теоретическая) состоит из описания статистических методов проверок гипотез, а также рассмотрения основных статистических критериев.

Практическая же часть текущей курсовой работы состоит из проверки гипотезы на модельных и реальных данных. Сама проверка будет осуществлена через сравнение выбранного критерия с альтернативным на близость распределения к нормальному.

Также нужно заметить важность и актуальность данной задачи, ведь если реальные данные как-то походят на одно из известных теоретических распределений, то появится возможность делать более точные прогнозы будущих событий и на более долгий срок.

Новизной в моей работе является то, что я взял индекс MOEXBC который до этого никто не брал, и код критерий Шапиро-Франчиа отличается от работ других. Так как данный критерий не является частью библиотеки scipy в Python, то данный критерий был запрограммирован мною полностью, ведь все расчеты реализованы на языке Python и также сопоставлены с расчетами на языке R, где этот критерий уже написан.

# **Глава 1. Теоретическая часть**

## **1.1 Статистическая проверка гипотез**

Для текущего исследования необходимо рассмотреть основные понятия. Первое из них – статистическая гипотеза. Под ней подразумевают всякое высказывание о виде неизвестного распределения генеральной совокупности, опираясь на результаты наблюдений в случайной выборке.

Никакие экспериментально полученные данные никогда не подтвердят какую-либо гипотезу. Это наше фундаментальное ограничение. Данные могут лишь не противоречить ей или, наоборот, показывать крайне маловероятные результаты (при условии, что гипотеза верна). Но и в том, и в другом случае нет оснований утверждать, что выдвинутая гипотеза доказана.

Допустим, данные гипотезе не противоречат, тогда мы её не отвергаем. Если же мы приходим к выводу, что получить такие данные в рамках этой гипотезы вряд ли возможно, у нас появляется основание отбросить эту гипотезу.

Статистическая гипотеза — на то и статистическая, чтобы утверждать что-то о параметре распределения.

Существует основная (нулевая) гипотеза, которая обозначается в которой мы предполагаем свойства эмпирических данных и проверяем их. Эта гипотеза по умолчанию считается истинной, пока не будет доказана её ложность. всегда формулируется так, чтобы использовать знак равенства. Другими словами, чтобы все значения, полученные эмпирически, не были экстремальными, а лежали там, где ожидаем их найти.

Также существует альтернативная (конкурирующая) гипотеза, которая обозначается которая принимается, если основная гипотеза оказывается неверна. По итогу, всегда верна только одна гипотеза: или , но заранее нельзя предугадать какая именно, нужно проверять.

Всегда есть риск принять неверное решение по поводу истинности гипотезы, например ввиду неудачной выборки. Это называют ошибкой и их разделяют на два вида:

1. Ошибка I рода – основная гипотеза может быть принята за ложь, при том что является истинной

2. Ошибка II рода – наоборот, нулевая гипотеза может быть принята за истину, когда на самом деле является ложной.

Статистический критерий – математическое правило или формула, на основе которого с идет проверка принимается решение о принятии и отвержении статистической гипотезы. К данному решению приходят по специальной методике, которая в свою очередь состоит из следующих этапов (пусть задана случайная выборка ) – выборка из n объектов из генеральной выборки X):

1. Вначале формируется основная гипотеза о виде распределения выборки. Часто вместе с ней формируется альтернативная гипотеза , которая противоположна основной. Таких конкурирующих гипотез может быть несколько.

2. Далее выбирается статистический критерий, на основе которого будет проверяться истинность гипотезы .

3. Выбирается значение уровня значимость, которое обозначается , что является допустимой вероятностью ошибки I рода. , но на практике часто берут = 0,05 или 0,01.

4. Находится критическая область данного критерия, то есть та область, вероятность попадания в которую (в случае, когда верна) точно равна выбранному уровню значимости .

5. Делается заключение о том, что гипотеза подтверждается или опровергается.

Вероятность попадания критерия в критическую область называют мощностью критерия и обозначают M. Под ней понимают вероятность не совершить ошибку II рода. Как правило последствия ошибки 1 рода масштабнее, чем второго, а мощность рассчитывают по формуле , где – вероятность ошибки II рода.

P-значение (P-value) – вероятность получить наблюдаемый или ещё более далёкий от предполагаемого результат при условии, что нулевая гипотеза верна. Если это значение больше 10%, то нулевую гипотезу точно не стоит отвергать. Меньше — возможно, есть основания отвергнуть нулевую гипотезу. Общепринятые пороговые значения — 5% и 1%. Если вы признаёте 5%-ю вероятность слишком малой и на этом основании отвергаете нулевую гипотезу, то в среднем в одном исследовании из 20 значимый эффект будет обнаружен не потому, что она неверна, а за счёт случайной ошибки. Окончательное решение, какой порог считать достаточным, всегда остаётся за аналитиком.

То есть если , то принимается истинной, а если , то отвергается, для любого уровня значимости .

Иногда, для надежности проверяют несколько критериев.

## **1.2. Критерий Шапиро-Франчиа**

Критерий Шапиро-Франчиа довольно популярен для оценки нормальности распределения выборки. Его предложили Р.С. Франчиа и С.С. Шапиро еще в 1972 году. Он является улучшенной версией критерий Шапиро-Уилка, который в свою очередь неудобен в работе с большими объемами выборки n (n > 100). Это и делает данный критерий хорошим решением проблемы, возникающей из-за больших данных. Далее показаны математические «внутренности» критерия.

Статистика имеет вид:

, где и – математическое ожидание i-той порядковой статистики из стандартного нормального распределения.

Аппроксимация не искажает существенно критерий .

Используя аппроксимацию для квантили стандартного нормального распределения можно записать:

и тогда для имеем:

Важно подчеркнуть, что статистика показывает нам, насколько близко выборочное распределение к нормальному виду. Таким образом, если статистика будет принимать довольно большие значения, то текущее распределение можно считать нормальным.

Приближенную вероятность получения эмпирического значения статистики при нулевой гипотезе можно вычислить по формуле:

где – коэффициенты, определяемые по таблице, но по данному критерию для достаточно большого значения n данные коэффициенты берутся из таблицы как для n = 50.

Различные сравнительные исследования показывают, что статистические корреляционные тесты порядка, такие как Шапиро-Уилка и Шапиро-Франчиа, являются одними из самых достоверных и мощных критериев проверки распределения на нормальность.

## **1.3 Критерий Колмогорова-Смирнова**

Критерий Колмогорова-Смирнова – один из непараметрических критериев равенства одномерных непрерывных вероятностных распределений, используемый в математической статистике для сравнение теоретического выборочного распределения с эмпирическим или для сравнения двух выборок.

Критерию дали названия в честь двух великих советских математиков: Николая Смирнова и Андрея Колмогорова.

Часто критерий Колмогорова-Смирнова берут как два разных критерия:

1. Критерий согласия Колмогорова – служит для проверки истинности гипотезы о принадлежности выборки к определенному теоретическому типу распределения.

2. Критерий однородности Смирнова – берется для проверки гипотезы о принадлежности двух различных и независимых выборок к одному и тому же теоретическому типу распределения.

В основе метода находится статистика Колмогорова-Смирнова, а их тест с двумя выборками является одним из наиболее эффективных методов сравнения двух выборок.

Критерий Колмогорова-Смирнова часто применяется в практике для реальных задач. В текущей курсовой работе критерий Колмогорова-Смирнова будет использоваться в проверке равномерности распределений P-значений основного критерия при помощи встроенного метода kstest в библиотеке scipy.stats на языке Python.

**Глава 2. Практическая часть**

## **2.1 Проверка гипотезы на модельных данных**

## 

Как модельные данные будет использоваться случайная выборка из 250 значений, ведь примерно столько торговых дней в каждом году. Выборка будет изначально распределена согласно нормальному закону. Так мы сможем проверить, правильно ли мы записали статистику основного критерия. Стоит принять во внимание, что закон распределения статистики Шапиро-Франчиа нам изначально неизвестен.

Для проверки гипотезы о нормальном распределении пойдем по следующему алгоритму:

1. В начале с помощью программы «Таблицы квантилей.ipynb» построим таблицу 999 квантилей распределения статистики основного критерия при верной нулевой гипотезе (статистика вычислялась 10000 раз).

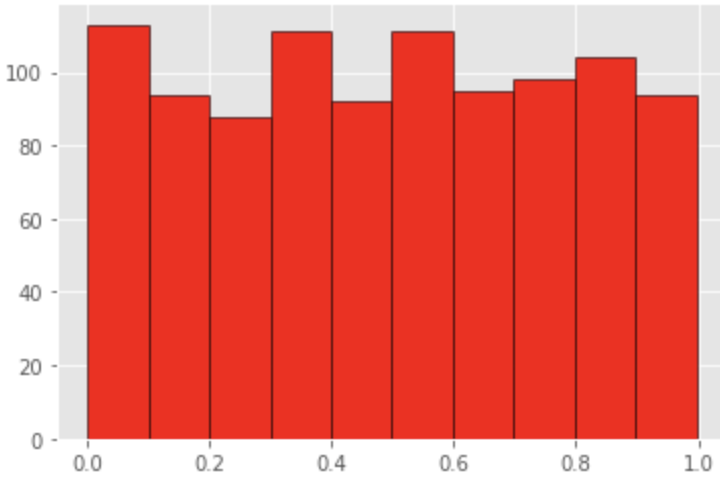
Таблица 1. Квантили распределения основной статистики

Изображение выглядит как текст

Автоматически созданное описание

2. Формируем гипотезу – выборки распределены по нормальному закону и гипотезу – выборки распределены по другому закону распределения. Далее воспользуемся программой «Гистограмма P-значений для модели» чтобы вычислить P-значения и построить их диаграмму.

Рисунок 1. Гистограмма 1000 P-значений на модельных данных



Исходя из данного графика можно сделать вывод, что множество P-значений распределено равномерно, но нужно проверить аналитически наше визуальное представление о p-значениях.

3. Далее исходя из теоремы о равномерности P-value, которая нам говорит о том, что если верна, то P-value равномерно распределено на отрезке [0;1]. Идем от обратного, если наше Р-значение распределено равномерно, то верна. На равномерность нам поможет проверить критерий Колмогорова-Смирнова.

Формируем для него новую гипотезу выборка p-значений распределена равномерно и к ней гипотезу – выборка p-значений имеет другой закон распределения. Затем в программе «Равномерность модельных P-значений по Колмогорову.ipynb» произведем проверку равномерности распределения P-значений на отрезке [0;1] на основе критерия Колмогорова-Смирнова.

Если p-value KS < 0,01 => принимается по Колмогорову-Смирнову =>  принимается в основном тесте с вероятностью 99%

0,01 <= p-value KS < 0,05 => принимается по Колмогорову-Смирнову =>  принимается в основном тесте с вероятностью 95%

0,05 <= p-value KS < 0,1 => принимается по Колмогорову-Смирнову => принимается в основном тесте с вероятностью 90%

p-value KS >= 0,1 => принимается по Колмогорову-Смирнову => принимается в основном тесте.

Программа выдает нам значение 0,14130876435218978. Это больше чем 0,1, что в свою очередь доказывает равномерность данного распределения P-значений. Тогда по Колмогорову-Смирнову принимается, благодаря чему мы можем говорить о принятии основное гипотезы по Шапиро-Франчиа о нормальности распределения модельных данных.

4. Далее выбираем 3 конкурирующие гипотезы. В данном случае это будут гипотезы о том, что выборка имеет распределение Стьюдента, Хи квадрат и Лапласа. Выбраны они в силу того, максимально приближены к нормальному в отличие от других законов распределения. Мощность критерия Шапиро-Франчиа будем приближенно оценивать проверив тысячу раз гипотезу о нормальном распределении выборки (с уровнем значимости 5%) при верных конкурирующих гипотезах при помощи программы «Проверка альтернативных гипотез для модельных данных.ipynb». В результате при уровне значимости пять сотых мощность критерия по всем гипотезам единице.

Таким образом, мощность критерия достаточно большая, что дает нам понять, что при использовании критерия Шапиро-Франчиа, вероятность ошибки второго рода стремится к нулю.

## **2.2 Предварительный анализ данных**

В качестве исходных данных мной были взяты котировки акций по 15 компаниям, входящих в индекс МосБиржи голубые фишки (MOEXBC). Как источник данных был задействован интернет ресурс finam.ru.

Сначала посчитаем количество торговых дней для каждой компании по годам с помощью программы «Торговые дни.ipynb», чтобы выявить акций компаний, которые нам не подходят.

Таблица 1. Торговые дни каждой компании

Изображение выглядит как стол

Автоматически созданное описание

По таблице видно, что нам не подходят данные по компании Mail.ru Group и X5 Retail Group, так как у них был IPO (первичный выпуск акций) только в 2020 и в 2018 году соответственно. Соответственно нам от них нужно избавиться, для более точного дальнейшего анализа.

С помощью программы «Очистка данных.ipynb» удалим все столбцы кроме '<TICKER>', '<DATE>', '<CLOSE>', '<VOL>', где '<TICKER>' -  имя тиккера, '<DATE>' – год транзакции, '<CLOSE>' – скорректированная цена закрытия и '<VOL>' – объем торгов.

Затем необходимо составить таблицы максимальных относительных скачков как вверх, так и вниз, чтобы отсечь выбросы. Это было сделано при помощи программ «Скачки вверх.ipynb» и «Скачки вниз.ipynb».

Таблица 2. Максимальные относительные скачки цен вверх

Изображение выглядит как стол

Автоматически созданное описание

Таблица 3. Максимальные относительные скачки цен вниз

Изображение выглядит как стол

Автоматически созданное описание

Как видно из таблиц, то выбросов нет. Также построим графики максимальных скачков, с помощью программы «Графики скачков.ipynb». Для скачков вверх это ПАО «Газпром» (GAZP), а для скачков вниз это компания ПАО «Мобильные ТелеСистемы» (MTSS).

Рисунок 2. График цен акций GAZP.

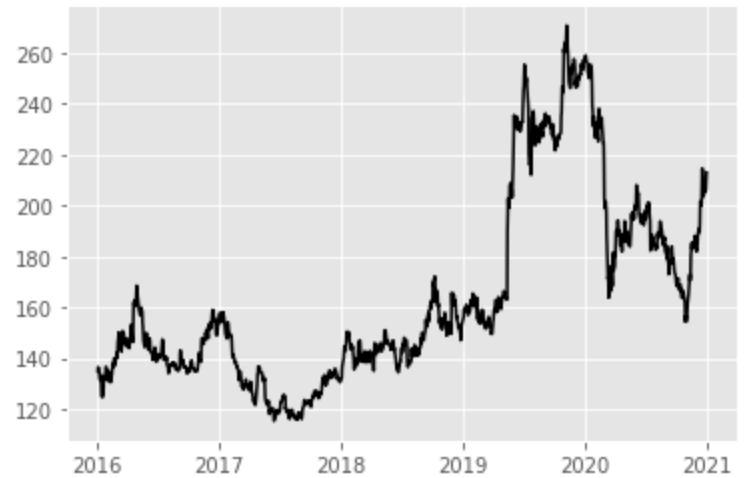


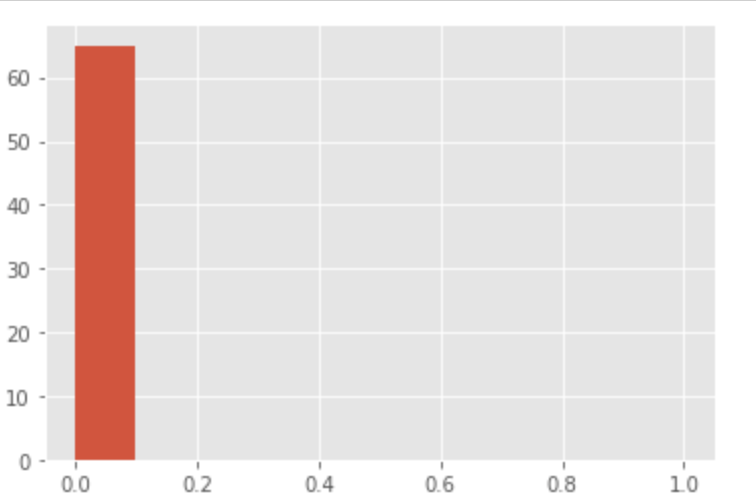
Рисунок 3. График цен акций MTSS.



**2.3 Проверка гипотезы для реальных данных.**

Вычислим P-значения для реальных данных. Для этого воспользуемся программой «P-значения для реальных данных.ipynb». логарифмическая доходность акций компаний по годам имеет нормальное распределение, логарифмическая доходность имеет другой тип распределения.

Рисунок 4. P-значения реальных данных

**

Для всех данных мы получили нулевое P-значение (это же отражено на диаграмме), что говорит нам о неверности основной гипотезы и принятии конкурирующей гипотезы .

**Заключение**

Была проверена гипотеза о нормальном распределении логарифмических доходностей компании, и во всех случаях было выявлено то, что данные распределения не нормального вида. Также была посчитана мощность критерия Шапиро-Франчиа. С моей точки зрения, данный критерий лучше рассматривать как меры приближения данного распределения к нормальному, когда гипотеза о нормальном распределении уже принята.

В ходе написания курсовой работы были выполнены все поставленные задачи, главной из которой была реализация критерия Шапиро-Франчиа на языке программирования Python.

Гипотеза о нормальном распределении дневных логарифмических доходностей не принимается, то есть реальные данные доходностей, взятые с котировок входящих в данный индекс, нельзя назвать нормально распределенными. Тем самым лишний раз убедившись, что в реальности практически ничего не подчиняется строгим теоретическим законам.

Из различных в расчетах в литературе можно сделать вывод, что критерий Шапиро-Франчиа лучше подходит для опровержения гипотез о нормальном распределении выборки, когда она действительно не нормально распределена.

**Список используемых источников**

## **Литература**

1. Браилов А.В. Лекции по математической статистике. М.: Финакадемия, 2007.

2. Кобзарь А.И. Прикладная математика и статистика. Для инженеров и научных работников. – М.: ФИЗМАТЛИТ, 2006. – 816 с. – ISBN 5-9221-0707-0.

3. Alfred K. Mbah & Arnut Paothong (2015) Shapiro–Francia test compared to other normality test using expected p-value, Journal of Statistical Computation and Simulation, 85:15, 3002-3016, DOI: 10.1080/00949655.2014.947986

4. Математическая статистика с элементами теории вероятностей в задачах с решениями: Учебное пособие для бакалавров / Л.И. Ниворожкина, З.А. Морозова, И.Э. Гурьянова; под ред. проф. Л.И. Ниворожкиной. – 2-е изд., перераб. и доп. – М.: Издательско-торговая корпорация «Дашков и К», 2015. – 480 с. – ISBN 978-5-394-02517-4

5. Сибирцева М.С. Курсовая работа «Проверка гипотезы о нормальном распределении логарифмической доходности по критерию Шапиро-Франчиа», 2020

## **Интернет-ресурсы**

1. Сайт Финам.ru: <https://www.finam.ru>

**Приложение**

## **Приложение 1. Характеристики компьютера**

Тип процессора - 8‑ядерный процессор Intel Core i9

Тактовая частота - 2,3 GHz

Частота системной шины – 1066 МГц

Объем кэш-памяти второго уровня (L2) - 256 КБ

## **Приложение 2. Коды программ**

1. Таблица квантилей.ipynb

import time

import numpy as np

import scipy.stats as st

from math import log

from scipy.stats import kstest

import pandas as pd

start\_time = time.time() # время начала

#запрограммируем критерий Шапиро-Франчиа

def m\_i(i, n):#математическое ожидание i-той порядковой статистики из стандартного нормального распределения

return 4.91\*((((i-0.375)/(n+0.25))\*\*(0.14)) - (((n-i+5/8)/(n+0.25))\*\*(0.14)))

def up(data):#Посчитаем отдельно числитель в формуле статситики критерия

result = 0

n = len(data)

for i in range(1, n+1):

m\_1 = m\_i(i, n)

m\_up = m\_1 \* data[i-1]

result += m\_up

return (result)\*\*2

def down(data):#Посчитаем отдельно знаменатель в формуле статистики критерия

down\_sum11 = 0

m\_down = 0

n = len(data)

mean = np.mean(data)

for i in range(1, n+1):

m\_1 = m\_i(i, n)

m\_down += m\_1\*\*2

for i in range(n):

down\_sum\_1 = (data[i] - mean)\*\*2

down\_sum11 += down\_sum\_1

result = down\_sum11\*m\_down

return result

def SF\_value(data):#Делим и округляем посчитав саму статистику

result = up(data)/down(data)

return round(result, 5)

def p\_value(data):

SF\_number = SF\_value(data)

n = len(data)

mu = -1.2725 + 1.0521\*(log(log(n)) - log(n))

sigma = -0.26758\*(log(log(n)) + 2/log(n)) + 1.0308

z = (log(1 - SF\_number) - mu)/sigma

p = 1 - st.norm.cdf(z)

return round(p, 5)

#Формирование выборки методом Монте-Карло

n = 250

k = 10000

SF = []

P\_value = []

for i in range(k):

x\_n = np.sort(np.random.normal(0, 1, size=n))

SF.append(SF\_value((x\_n)))

#Формирование таблицы из 999 квантилей

q999 = pd.DataFrame(data=SF).quantile(np.arange(0.001,1,0.001))

#Формирование таблицы из 9 квантилей

q9 = pd.DataFrame(data=SF).quantile(np.arange(0.1,1,0.1))

print("--- %s seconds ---" % (time.time() - start\_time)) #время выполннения программы в секундах

q9

2. Гистограмма P-значений для модели.ipynb

import time

import numpy as np

import scipy.stats as st

from math import log

from scipy.stats import kstest

from matplotlib import pyplot as plt

import matplotlib

start\_time = time.time() # время начала

#запрограммируем критерий Шапиро-Франчиа

def m\_i(i, n):#математическое ожидание i-той порядковой статистики из стандартного нормального распределения

return 4.91\*((((i-0.375)/(n+0.25))\*\*(0.14)) - (((n-i+5/8)/(n+0.25))\*\*(0.14)))

def up(data):#Посчитаем отдельно числитель в формуле статситики критерия

result = 0

n = len(data)

for i in range(1, n+1):

m\_1 = m\_i(i, n)

m\_up = m\_1 \* data[i-1]

result += m\_up

return (result)\*\*2

def down(data):#Посчитаем отдельно знаменатель в формуле статистики критерия

down\_sum11 = 0

m\_down = 0

n = len(data)

mean = np.mean(data)

for i in range(1, n+1):

m\_1 = m\_i(i, n)

m\_down += m\_1\*\*2

for i in range(n):

down\_sum\_1 = (data[i] - mean)\*\*2

down\_sum11 += down\_sum\_1

result = down\_sum11\*m\_down

return result

def SF\_value(data):#Делим и округляем посчитав саму статистику

result = up(data)/down(data)

return round(result, 5)

def p\_value(data):

SF\_number = SF\_value(data)

n = len(data)

mu = -1.2725 + 1.0521\*(log(log(n)) - log(n))

sigma = -0.26758\*(log(log(n)) + 2/log(n)) + 1.0308

z = (log(1 - SF\_number) - mu)/sigma

p = 1 - st.norm.cdf(z)

return round(p, 5)

n = 250

k = 10000

P\_value = []

#Формирование выборки методом Монте-Карло

for i in range(k):

x\_n = np.sort(np.random.normal(0, 1, size=n))

P\_value.append(p\_value(x\_n))

#Построение гистограммы на основе этих 1000 P-значений

matplotlib.style.use('ggplot')

plt.hist(P\_value, edgecolor='black', bins=10, color='red')

plt.show()

print("--- %s seconds ---" % (time.time() - start\_time)) #время выполннения программы в секундах

3. Равномерность модельных P-значений по Колмогорову.ipynb

import time

import numpy as np

import scipy.stats as st

from math import log

from scipy.stats import kstest

start\_time = time.time() # время начала

#запрограммируем критерий Шапиро-Франчиа

def m\_i(i, n):#математическое ожидание i-той порядковой статистики из стандартного нормального распределения

return 4.91\*((((i-0.375)/(n+0.25))\*\*(0.14)) - (((n-i+5/8)/(n+0.25))\*\*(0.14)))

def up(data):#Посчитаем отдельно числитель в формуле статситики критерия

result = 0

n = len(data)

for i in range(1, n+1):

m\_1 = m\_i(i, n)

m\_up = m\_1 \* data[i-1]

result += m\_up

return (result)\*\*2

def down(data):#Посчитаем отдельно знаменатель в формуле статистики критерия

down\_sum11 = 0

m\_down = 0

n = len(data)

mean = np.mean(data)

for i in range(1, n+1):

m\_1 = m\_i(i, n)

m\_down += m\_1\*\*2

for i in range(n):

down\_sum\_1 = (data[i] - mean)\*\*2

down\_sum11 += down\_sum\_1

result = down\_sum11\*m\_down

return result

def SF\_value(data):#Делим и округляем посчитав саму статистику

result = up(data)/down(data)

return round(result, 5)

def p\_value(data):

SF\_number = SF\_value(data)

n = len(data)

mu = -1.2725 + 1.0521\*(log(log(n)) - log(n))

sigma = -0.26758\*(log(log(n)) + 2/log(n)) + 1.0308

z = (log(1 - SF\_number) - mu)/sigma

p = 1 - st.norm.cdf(z)

return round(p, 5)

n = 250

k = 10000

P\_value = []

#Формирование выборки методом Монте-Карло

for i in range(k):

x\_n = np.sort(np.random.normal(0, 1, size=n))

P\_value.append(p\_value(x\_n))

p = kstest(P\_value, 'uniform')[1]

if p < 0.01:

print('H\_1 принимается по Колмогорову-Смирнову => H\_1 принимается в основном тесте с вер. 99%')

print('P-значение', p)

elif 0.01 <= p < 0.05:

print('H\_1 принимается по Колмогорову-Смирнову => H\_1 принимается в основном тесте с вер 95%')

print('P-значение', p)

elif 0.05 <= p < 0.1:

print('H\_1 принимается по Колмогорову-Смирнову => H\_1 принимается в основном тесте с вер 90%')

print('P-значение', p)

elif p >= 0.1:

print('H\_0 принимается по Колмогорову-Смирнову => H\_0 принимается в основном тесте.')

print('P-значение', p)

print("--- %s seconds ---" % (time.time() - start\_time)) #время выполннения программы в секундах

4. Проверка альтернативных гипотез для модельных данных.ipynb

#мощность критерия с уровнем значимости 0.05

import time

import numpy as np

import scipy.stats as st

from math import log

from scipy.stats import kstest

start\_time = time.time() # время начала

#запрограммируем критерий Шапиро-Франчиа

def m\_i(i, n):#математическое ожидание i-той порядковой статистики из стандартного нормального распределения

return 4.91\*((((i-0.375)/(n+0.25))\*\*(0.14)) - (((n-i+5/8)/(n+0.25))\*\*(0.14)))

def up(data):#Посчитаем отдельно числитель в формуле статситики критерия

result = 0

n = len(data)

for i in range(1, n+1):

m\_1 = m\_i(i, n)

m\_up = m\_1 \* data[i-1]

result += m\_up

return (result)\*\*2

def down(data):#Посчитаем отдельно знаменатель в формуле статистики критерия

down\_sum11 = 0

m\_down = 0

n = len(data)

mean = np.mean(data)

for i in range(1, n+1):

m\_1 = m\_i(i, n)

m\_down += m\_1\*\*2

for i in range(n):

down\_sum\_1 = (data[i] - mean)\*\*2

down\_sum11 += down\_sum\_1

result = down\_sum11\*m\_down

return result

def SF\_value(data):#Делим и округляем посчитав саму статистику

result = up(data)/down(data)

return round(result, 5)

def p\_value(data):

SF\_number = SF\_value(data)

n = len(data)

mu = -1.2725 + 1.0521\*(log(log(n)) - log(n))

sigma = -0.26758\*(log(log(n)) + 2/log(n)) + 1.0308

z = (log(1 - SF\_number) - mu)/sigma

p = 1 - st.norm.cdf(z)

return round(p, 5)

n = 250

k = 10000

P\_value = []

#Формирование выборки методом Монте-Карло

for i in range(k):

x\_n = np.sort(np.random.normal(0, 1, size=n))

P\_value.append(p\_value(x\_n))

#мощность критерия с уровнем значимости 0.05

n = 250

m = 1000

laplace = []

## Формирование выборок из распределения Лапласа размера n

for i in range(m):

x\_n = st.laplace.rvs(loc = 0, scale = 1,size = n)

laplace.append(p\_value(x\_n))

## Вычисление мощности критерия

power\_laplace = 0

for i in laplace:

if i < 0.05:

power\_laplace += 1

print(power\_laplace/m)

#мощность критерия с уровнем значимости 0.05

n = 250

m = 1000

chi2 = []

## Формирование выборок из распределения Хи квадрат размера n

for i in range(m):

x\_n = st.chi2.rvs(df = 2, loc = 0, scale = 1,size = n)

chi2.append(p\_value(x\_n))

## Вычисление мощности критерия

power\_chi2 = 0

for i in chi2:

if i < 0.05:

power\_chi2 += 1

print(power\_chi2/m)

#мощность критерия с уровнем значимости 0.05

n = 250

m = 1000

t = []

## Формирование выборок из распределения Стьюдента размера n

for i in range(m):

x\_n = st.t.rvs(df = 2, loc = 0, scale = 1,size = n)

t.append(p\_value(x\_n))

## Вычисление мощности критерия

power\_t = 0

for i in t:

if i < 0.05:

power\_t += 1

print(power\_t/m)

print("--- %s seconds ---" % (time.time() - start\_time)) #время выполннения программы в секундах

5. Торговые дни.ipynb

import time

start\_time = time.time() # время начала

import os

os.chdir('/Users/korol787/Desktop/Курсовая Королев/Данные')

import pandas as pd

tickers = ['FIVE', 'GAZP', 'GMKN', 'LKOH', 'MAIL', 'MGNT', 'MTSS', 'NVTK', 'PLZL', 'POLY', 'ROSN', 'SBER', 'SNGS', 'TATN', 'YNDX']

shares\_dataframes = []

for i in range(len(tickers)):

df\_i = pd.read\_csv(tickers[i] + '.csv')

df\_i['<DATE>'] = pd.to\_datetime(df\_i['<DATE>'], format='%Y%m%d').dt.year.astype('int')

shares\_dataframes.append(df\_i)

shares = pd.concat(shares\_dataframes)

shares\_day\_per\_year = shares.pivot\_table(index='<TICKER>', columns='<DATE>', values='<CLOSE>', aggfunc='count').fillna(0).astype('int')

print("--- %s seconds ---" % (time.time() - start\_time)) #время выполннения программы в секундах

shares\_day\_per\_year

6. Скачки вверх.ipynb

import time

start\_time = time.time() # время начала

import os

os.chdir('/Users/korol787/Desktop/Курсовая Королев/Данные')

import pandas as pd

def tables(ticker):# формирует таблицы

company = pd.read\_csv(ticker+'.csv')

company['<DATE>'] = pd.to\_datetime(company['<DATE>'], format='%Y%m%d').dt.year.astype('int')

return company

new\_tickers = ['GAZP', 'GMKN', 'LKOH', 'MGNT', 'MTSS', 'NVTK', 'PLZL', 'POLY', 'ROSN', 'SBER', 'SNGS', 'TATN', 'YNDX']

leap\_up = []

for ticker in new\_tickers: # формирует список с максимальными отклонениями вверх

company = tables(ticker)

company['Leap'] = company['<CLOSE>'].pct\_change()

up = company.groupby(company['<DATE>'])['Leap'].max()

leap\_up.append(up)

max\_up = dict.fromkeys(list(company['<DATE>'].unique()))

for year in max\_up.keys():# формирует словарь и из него таблицу

max\_up[year] = [leap\_up[i][year] for i in range(len(leap\_up))]

up\_table = pd.DataFrame(max\_up, index = new\_tickers).round(3)

print("--- %s seconds ---" % (time.time() - start\_time)) #время выполннения программы в секундах

up\_table

7. Скачки вниз.ipynb

import time

start\_time = time.time() # время начала

import os

os.chdir('/Users/korol787/Desktop/Курсовая Королев/Данные')

import pandas as pd

def tables(ticker):# формирует таблицы

company = pd.read\_csv(ticker+'.csv')

company['<DATE>'] = pd.to\_datetime(company['<DATE>'], format='%Y%m%d').dt.year.astype('int')

return company

new\_tickers = ['GAZP', 'GMKN', 'LKOH', 'MGNT', 'MTSS', 'NVTK', 'PLZL', 'POLY', 'ROSN', 'SBER', 'SNGS', 'TATN', 'YNDX']

leap\_down = []

for ticker in new\_tickers: # формирует список с максимальными отклонениями вверх

company = tables(ticker)

company['Leap'] = company['<CLOSE>'].pct\_change()

down = company.groupby(company['<DATE>'])['Leap'].min()

leap\_down.append(down)

max\_down = dict.fromkeys(list(company['<DATE>'].unique()))

for year in max\_down.keys():# формирует словарь и из него таблицу

max\_down[year] = [leap\_down[i][year] for i in range(len(leap\_down))]

down\_table = pd.DataFrame(max\_down, index = new\_tickers).round(3)

print("--- %s seconds ---" % (time.time() - start\_time)) #время выполннения программы в секундах

down\_table

8. Графики скачков.ipynb

import time

start\_time = time.time() # время начала

import pandas as pd

import matplotlib.pyplot as plt

import matplotlib

import os

os.chdir('/Users/korol787/Desktop/Курсовая Королев/Данные')

def table(ticker):# формирует таблицы

company = pd.read\_csv(ticker+'.csv')

company['<DATE>'] = pd.to\_datetime(company['<DATE>'], format='%Y%m%d')

return company

GAZP = table('GAZP')#строим графики

MTSS = table('MTSS')

matplotlib.style.use('ggplot')

plt.plot(GAZP['<DATE>'], GAZP['<CLOSE>'], color='black')

plt.show()

plt.plot(MTSS['<DATE>'], MTSS['<CLOSE>'], color='black')

plt.show()

print("--- %s seconds ---" % (time.time() - start\_time)) #время выполннения программы в секундах

9. P-значения для реальных данных.ipynb

import time

start\_time = time.time() # время начала

import numpy as np

import scipy.stats as st

from math import log

import pandas as pd

from matplotlib import pyplot as plt

import matplotlib

from scipy.stats import kstest

import os

os.chdir('/Users/korol787/Desktop/Курсовая Королев/Данные')

new\_tickers = ['GAZP', 'GMKN', 'LKOH', 'MGNT', 'MTSS', 'NVTK', 'PLZL', 'POLY', 'ROSN', 'SBER', 'SNGS', 'TATN', 'YNDX']

def tables(ticker):#формируем таблицу

company = pd.read\_csv(ticker+'.csv')

company['<DATE>'] = pd.to\_datetime(company['<DATE>'], format='%Y%m%d').dt.year.astype('int')

return company

def m\_i(i, n):#программируем Критерий Шапиро-Франчиа

return 4.91\*((((i-0.375)/(n+0.25))\*\*(0.14)) - (((n-i+5/8)/(n+0.25))\*\*(0.14)))

def up(data):

result = 0

n = len(data)

for i in range(1, n+1):

m\_1 = m\_i(i, n)

m\_up = m\_1 \* data[i-1]

result += m\_up

return (result)\*\*2

def down(data):

down\_sum11 = 0

m\_down = 0

n = len(data)

mean = np.mean(data)

for i in range(1, n+1):

m\_1 = m\_i(i, n)

m\_down += m\_1\*\*2

for i in range(n):

down\_sum\_1 = (data[i] - mean)\*\*2

down\_sum11 += down\_sum\_1

result = down\_sum11\*m\_down

return result

def SF\_value(data):

result = up(data)/down(data)

return result

def p\_value(data):#ищем P-value

SF\_number = SF\_value(data)

n = len(data)

mu = -1.2725 + 1.0521\*(log(log(n)) - log(n))

sigma = -0.26758\*(log(log(n)) + 2/log(n)) + 1.0308

z = (log(1 - SF\_number) - mu)/sigma

p = 1 - st.norm.cdf(z)

return p

P\_value = []

for ticker in new\_tickers:

company = tables(ticker)

company["logret"] = company.groupby('<TICKER>')['<CLOSE>'].apply(lambda x: np.log(x) - np.log(x.shift()))

company = company.rename({'<DATE>': 'date'}, axis=1)

for year in company['date'].unique():

company\_year = company.query('date == @year')

P\_value.append(p\_value(list(company\_year['logret'][1:])))

matplotlib.style.use('ggplot')

print(kstest(P\_value,'uniform'))

plt.hist(P\_value, range=(0, 1))

plt.show()

print("--- %s seconds ---" % (time.time() - start\_time)) #время выполннения программы в секундах

## **Приложение 3. Список файлов**

|  |  |  |
| --- | --- | --- |
| Название программы | Полученные данные | t |
| 1. Таблица квантилей.ipynb | Таблица 1 | 3.9c |
| 2. Гистограмма P-значений для модели.ipynb | Рисунок 1 | 5.09 |
| 3. Равномерность модельных P-значений по Колмогорову.ipynb | - | 4.8c |
| 4. Проверка альтернативных гипотез для модельных данных.ipynb | - | 6.95c |
| 5. Торговые дни.ipynb | Таблица 2 | 0.66c |
| 6. Скачки вверх.ipynb | Таблица 3 | 0.63c |
| 7. Скачки вниз.ipynb | Таблица 4 | 0.64c |
| 8. Графики скачков.ipynb | Рисунок 2, Рисунок 3 | 1.09c |
| 9. P-значения для реальных данных.ipynb | Рисунок 4 | 0.39c |