Министерство образования и науки Российской Федерации
Федеральное государственное автономное образовательное учреждение
высшего образования
«Санкт-Петербургский политехнический университет Петра Великого»
Институт компьютерных наук и технологий
Высшая школа киберфизических систем и управления

Работа допущена к защите
И.о. директора ВШ КФСУ
_____В.П. Шкодырев
«___»_____2017 г.

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА БАКАЛАВРА

АВТОМАТИЗИРОВАННАЯ СИСТЕМА ОБНАРУЖЕНИЯ АНОМАЛЬНЫХ СИТУАЦИЙ НА ФИНАНСОВЫХ РЫНКАХ

направление 27.03.04 – Управление в технических системах
профиль 27.03.04_05– Интеллектуальные системы обработки информации и управления

Выполнил
студент гр. 43503/3 < > Лапко Д.В.

Научный руководитель
к.т.н., доцент < > Онуфриев В.А.

Санкт-Петербург
2017

РЕФЕРАТ

Пояснительная записка 58 стр., 19 рис., 6 табл., 9 ист., 1 прил.

Финансовые рынки, выявление событий, инсайдерская торговля, спектральный анализ, автоматизированная система обнаружения.

Рассматривается задача обнаружения событий, повлиявших на финансовые рынки, а также задача обнаружения инсайдерской торговли по данным рынка и потока заказов трейдера.

Для первой задачи рассматривается два метода решения: решение с помощью спектрального анализа данных рынка и решение с помощью использования данных об объеме рынка.

Для второй задачи рассматривается статистический метод решения и анализ позиции трейдеров, а также используются результаты решения первой задачи.

Для автоматизации обнаружения инсайдерской торговли разработано приложение с веб интерфейсом.

ПЕРЕЧЕНЬ СОКРАЩЕНИЙ, УСЛОВНЫХ ОБОЗНАЧЕНИЙ, СИМВОЛОВ, ЕДИНИЦ ИЗМЕРЕНИЯ

Финансовый рынок - структура,спомощьюкоторойвусловияхрыночнойэкономикисоздается возможностьзаимствований,купли-продажиценныхбумаг и инвестиционныхтоваров (таких как драгоценные металлы).

Биржа - организация, обеспечивающая регулярное выполнение торговых операций на финансовом рынке.

Торговая сессия – время функционирования биржи в течении рабочего дня. Например, для биржи NYSEэто 9.30 - 16.00 по Нью-Йоркскому времени.

Инструмент торговли (инструмент) – товар или ценная бумага которая торгуется на данной бирже. Например, инструментом могут быть акции компании Apple. Акции компании принято называть инструментоми прибавлять соответствующий тикер. Тикер – сокращенное название компании. Так что акции компании Applebудем называть «инструментом AAPL».На бирже одновременно торгуется несколько инструментов. В данной работе в качестве инструментов рассматриваются только акции компаний.

Трейдер – человек торгующий на бирже.

Данные рынка – поток дискретных данных для определённого инструмента, содержащий время совершения сделок по купле-продаже данного инструмента, размеры сделок и цену по которой данные сделки производились. Пример данных рынка для акций Appleприведён в таблице 1.

Time	PRICE	SIZE
2016/06/01 09:30:00.019	99.02	100
2016/06/01 09:30:00.053	99.01	100
2016/06/01 09:30:00.128	99.01	300

Таблица 1. Пример данных рынка.

Данные рынка доступны всем участникам рынка.

Поток заказов – поток дискретных данных для определённого инструментаи определённого трейдера (торговца), содержащий время совершения сделок данным трейдером, размер сделок, флаг покупки (обозначает покупал или продавал инструмент трейдер) и цену по которой данные сделки проводились. Пример потока заказов для некоторого трейдера, торгующего акциямикомпании Appleприведён в таблице 2.

Time	PRICE	BUY_FLAG	QTY
2017/03/01 09:30:00.862	137.88	1	8
2017/03/01 09:30:04.879	137.9	1	2
2017/03/01 09:30:16.734	137.94	0	1
2017/03/01 09:30:25.409	138	1	5
2017/03/01 09:30:55.058	137.92	0	5

Таблица 2. Пример потока заказов.

QTY – размер сделок, BUY_FLAG – флаг покупки. Поток заказов доступен самому трейдеру, его доверенным лицам, организаторам биржи и государственным контролирующим органам (регуляторам).

Неправомерная торговля –действия трейдера, нацеленные на получение высокой прибыли и противоречащие законодательству. Надзор за действиями трейдеров осуществляет специальный уполномоченный орган, например, в США это SEC– комиссия по ценным бумагам и биржам.

СОДЕРЖАНИЕ

Введение……...……. 6
1 Предмет задачи, общий алгоритм решения……...……. 7
1.1 Инсайдерская торговля…….. 7
1.2 Общий алгоритм решения……. 9
2 Обнаружение событий по данным рынка……..……. 11
2.1 Идентификация событий по изменению цены…..…….. 12
2.2 Идентификация событий с использованием информации об объеме рынка …….. 18
2.3 Оценка результатов …….. 23
3 Анализ потока заказов трейдера ……..…….. 24
3.1 Алгоритм …… 24
3.2 Оценка результатов……..…….. 34
Заключение…… 36
Список используемых источников……. 38
Приложение. Программный код……. 39

ВВЕДЕНИЕ

Основная задача данной работы заключается в разработке эффективного алгоритма выявления неправомерной торговли трейдера, а конкретно – инсайдерской торговли.

В первом разделе данной работы подробно рассматривается предмет задачи, описывается общий алгоритм решения.

Во втором разделерассматривается методика обнаружения событий по данным рынка, вокруг которых потенциально могла производиться инсайдерская торговля.

Втретьем разделе описываетсяалгоритм поиска интервалов времени, в которых трейдер аномально много покупал или продавал инструменты торговли, что потенциально является признаком инсайдерской торговли.

В заключении подводятся итоги работы, описывающие полученные результаты.

1 ПРЕДМЕТ ЗАДАЧИ, ОБЩИЙ АЛГОРИТМ РЕШЕНИЯ

1.1 Инсайдерская торговля

Инсайдерская торговля – это торговля с использованием инсайдерских знаний, т.е. таких знаний которые априорно ставят трейдера в более выгодное положение по отношению к другим участникам рынка. [5] Примером может служить торговля акциями компании, которая вот-вот обанкротиться, акции торгуются по низкой цене, и владельцы стараются избавится от акций, но внезапно компания гигант, договаривается с руководством банкротящейся компании о слиянии. Широкой публике об этом неизвестно, но трейдер N через свои связи узнает о готовящейся сделке и начинает скупать акции банкротящейся компании, а после объявления о слиянии, когда акции банкротящейся компании сильно вырастают в цене продаёт их. Знание о предстоящем слиянии ставят его в заведомо более выгодное положение по отношению к другим трейдерам. Такая торговля запрещена во многих странах, человек получивший инсайдерские знания не может торговать акциями данного инструмента до того момента пока эти знания не станут известны широкой публике. В противном случае он привлекается к ответственности.

Инсайдерская торговля является серьезным нарушением закона, но вместе с тем и довольно распространенным, это связано со сложностью выявления нарушителей, а доказательство использования инсайдерской информации при торговле представляется еще более сложной задачей. В России надзорная функция по выявлению данных нарушений возложена на Центральный Банк, и регулируется законом №224 ФЗ ст. 185.6 «Неправомерное использование инсайдерской информации», за нарушение которой предусмотрена в том числе и уголовная ответственность. Реальным

примером инсайдерской торговли может служить случай, выявленный Банком России в 2014 году [4]:

«Банк России установил факты неправомерного использования инсайдерами ПАО АФК «Система» (далее – Общество) Буяновым А.Н., Носовой Н.К. и Гончаруком А.Ю. инсайдерской информации при осуществлении в своих интересах в июле 2014 года операций с обыкновенными акциями Общества.Данные операции были осуществлены за несколько часов до официального раскрытия Обществом инсайдерской информации, вызвавшего снижение котировок акций Общества на Бирже более чем на 7%.Полученная в ходе проверки информация указывает на то, что перечисленными выше лицами был получен доступ к инсайдерской информации Общества. Обстоятельства подачи поручений на совершение операций и характер торгового поведения свидетельствуют об осуществлении в их интересах операций с акциями ПАО АФК «Система» с неправомерным использованием полученной инсайдерской информации.Совершение таких сделок позволило указанным лицам избежать убытков, совокупный размер которых составил десятки миллионов рублей.»

1.2 Общий алгоритм решения

Общий алгоритм решения состоит из двух частей.

В первой части по данным рынка находятся события которые могли бы спровоцировать инсайдерскую торговлю, эти события обычно сопровождаются резким изменением стоимости инструмента с последующим установлением цены на новом уровне. На рисунок 1.1 представлен график иллюстрирующий пример влияния такого события на данные рынка акций компании Apple.

Рисунок 1.1. Данные рынка, пример события

Можно заметить, что 31.01.2017 произошло событие, которое вызвало резкий скачок стоимости инструмента. Некоторый период времени до и после скачка можно считать благоприятным для инсайдерской торговли. Подробнее события рассматриваются в разделе 2.

Во второй части производится оценка потока заказов трейдера по заданному инструменту. В Этом потоке заказов выявляются периоды с аномально высоким доходом или аномально высокой убылью (периоды, когда трейдер много продавал или покупал некоторый инструмент). Данные периоды выявляются статистическим методом.

На рисунке 1.2 приведен пример потока заказов содержащий интервал с аномально высокой убылью.

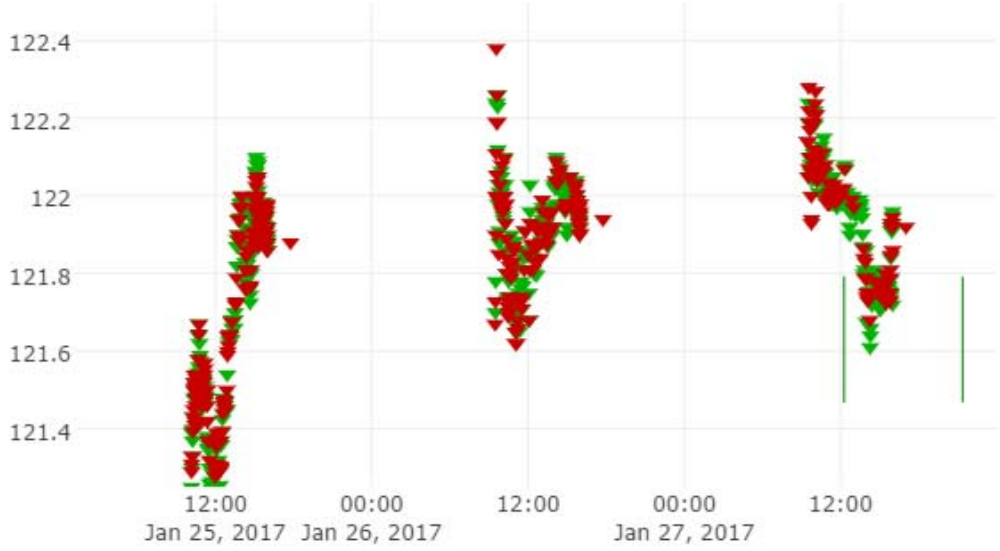


Рисунок 1.2.Поток заказов, пример аномально высоких доходов

Сделки по продаже обозначаются красными треугольниками, а по покупке зелеными. Период времени в который трейдер купил аномально большое количество акций обозначен двумя вертикальными чертами. В данный период было закуплено акций на 228 тыс. $а продано на 98 тыс.$, обычно данный трейдер покупает примерно столько же сколько и продает.Подробнее анализ потока заказов рассматриваются в разделе 3.

2 ОБНАРУЖЕНИЕ СОБЫТИЙ ПО ДАННЫМ РЫНКА

События, влияющие на данные рынка, можно разделить на два вида: запланированные и незапланированные.К незапланированным относятся: спонтанные или скрываемые решения руководства компаний, изменение внешней конъюнктуры, непредвидимые события природного характера и т.д. К запланированным можноотнести: финансовые отчеты компаний, презентации новой продукции, предстоящие конференции и заседания ассоциаций в которые входит компания. Отдельный вид событий — это квартальные и ежегодные отчеты о доходности компании. Торги акциями компании активизируются в преддверии выхода такого отчета, многие аналитики делают свои прогнозы. Повышается волатильность рынка, т.е. цена инструмента сильно колеблется с появлением новых прогнозов и слухов, увеличивается её̈дисперсия.[1] Также увеличивается и объем торгов в преддверии и после события. Объем торгов – совокупное количество проданных и купленных на рынке акций в определенный период.

Значимые для данного инструмента события отражаются в довольно сильном и резком изменении цены. Анонсы некоторых запланированных событий можно получить из открытых источников. Информации о многих значимых событиях нет в открытом доступе или они представлены в неформализованном виде,например, как ежедневные новостные заголовки, без указания степени значимости события.

Далее будут представлены два метода, которые позволяют выявить события, сильно повлиявшие на движение рынка, без использование данных из открытых источников.

2.1 Идентификация событий по изменению цены

В этом методе используются только данные рынка. Данные рынка каждой торговой сессии разбиваются на равные промежутки времени, и цена усредняется по ним, это делается чтобы сократить размер входного потока и уменьшить дисперсию. В данном случае я беру интервал в 15 минут, для хорошо торгуемых инструментов усреднение довольно значительное – порядка нескольких тысяч или даже десятков тысяч сделок. Исходные данные выглядят следующим образом (табл. 2.1):

Index	Symbol	Time	PRICE
1	NYSE_TAQ::AAPL	2016/06/01 09:45:00.000	99.2433040
2	NYSE_TAQ::AAPL	2016/06/01 10:00:00.000	99.0664907
3	NYSE_TAQ::AAPL	2016/06/01 10:15:00.000	99.2246921
4	NYSE_TAQ::AAPL	2016/06/01 10:30:00.000	99.2567635
5	NYSE_TAQ::AAPL	2016/06/01 10:45:00.000	99.2108794
6	NYSE_TAQ::AAPL	2016/06/01 11:00:00.000	99.2983692
7	NYSE_TAQ::AAPL	2016/06/01 11:15:00.000	99.3806437
8	NYSE_TAQ::AAPL	2016/06/01 11:30:00.000	99.4142616
9	NYSE_TAQ::AAPL	2016/06/01 11:45:00.000	99.0390728
10	NYSE_TAQ::AAPL	2016/06/01 12:00:00.000	98.9963841

Таблица 2.1 Исходные данные для инструмента AAPL.

В данном случае для каждой сессии получается 32 строки данных. Этот размер можно варьировать, в зависимости от потребностей.

Рассмотрим данные рынка для инструмента AAPLза годовой период в виде графика на рисунке 2.1. Каждая точка графика соответствует цене инструмента на момент закрытия торговой сессии.

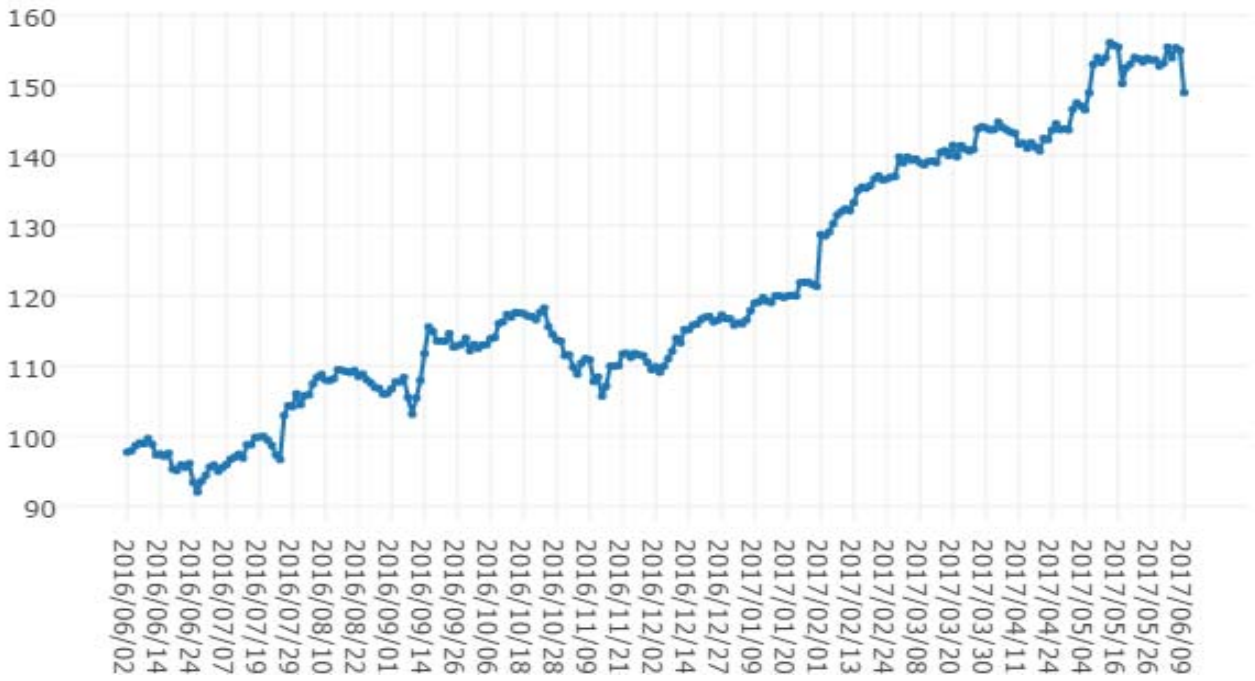


Рисунок 2.1.Годовые данные рынка инструмента AAPL

Попробуем найти события. Для этогоразобьём данные рынка на непересекающиеся сегменты. В каждом сегменте произведем дискретное преобразование Фурье (ДПФ), оценим амплитудно-частотную характеристику (спектр) сигнала в сегменте и возьмем сумму коэффициентов амплитудно-частотной характеристики как метрику данного сегмента. Проанализируем распределение метрики по сегментам и выберем сегменты с высоким значением метрики, такие сегменты возможно будут соответствовать интервалам времени, содержащим события.

ДПФ выполняется по следующей формуле:

$$x(k) = \sum_{n=0}^{N-1} x_n e^{-2\pi i \frac{kn}{N}}$$

$x_n -$ значение дискретного сигнала в $n -$ ый момент

$N -$ количество значений сигнала в сегменте

$n -$ индекс, принимает значения от 0 до $N - 1$

$x(k)$ – комплексная амплитуда соответствующей частоты

k – индекс частоты, принимает значения от 0 до $\frac{N-1}{2}$,

частота k – го сигнала равна k/T

За ширину окна возьмем два дня, т.е. 64 строки данных.

В качестве оконной функции сегмента возьмем обычное прямоугольное окно. Поскольку амплитуда основного лепестка при прямоугольном окне совпадает с истинной амплитудой данной частоты, и нам не требуется находить значения неосновных частот, а требуется найти лишь их амплитуды, а точнее – сумму всех амплитуд, то нет нужды гасить боковые лепестки жертвую при этом шириной основного лепестка и усложняя расчеты.

Исходя из того, что все случайные колебания цены мы убрали с помощью усреднения цены по интервалам в 12 минут и данные рынка не имеют свойства колебаться в короткие периоды времени с большей амплитудой, чем изменение цены в более длинные периоды времени, можно полагать что частотой с максимальной амплитудой будет низкая частота, в большинстве случаев соответствующая k=0.Это означает что неосновные частоты нас мало интересуют и не привнесут в метрику (сумму коэффициентов АЧХ) существенного вклада.

Посмотрим на первые пять коэффициентов АЧХ пяти случайно выбранных сегментов:

```
1) 11.05213   1.70177   0.4068    0.14271   0.36489
2) 4.66951  0.48777  0.22257  0.24257  0.01055
3) 1.59811  0.51304  0.01752  0.15392  0.84832
4) 0.16414  0.05447  0.34956  0.18674  0.0749
5) 6.70732  0.56421  1.9179   0.1254   0.91854
```

Можно заметить, что результаты эксперимента соответствуют гипотезе, выдвинутой выше.

Для большей наглядности построим спектрограмму (рисунок 2.2). Горизонтальная ось – ось времени, на ней отмечены номера сегментов, вертикальная ось – ось частоты. Высокие значения частот на спектрограмме отсутствуют, т.к. амплитуды соответствующих частот очень малы, и эта часть графика не информативна. Величина амплитуды частоты указывается цветом, синий и фиолетовый – маленькая амплитуда, красный и желтый – высокая.

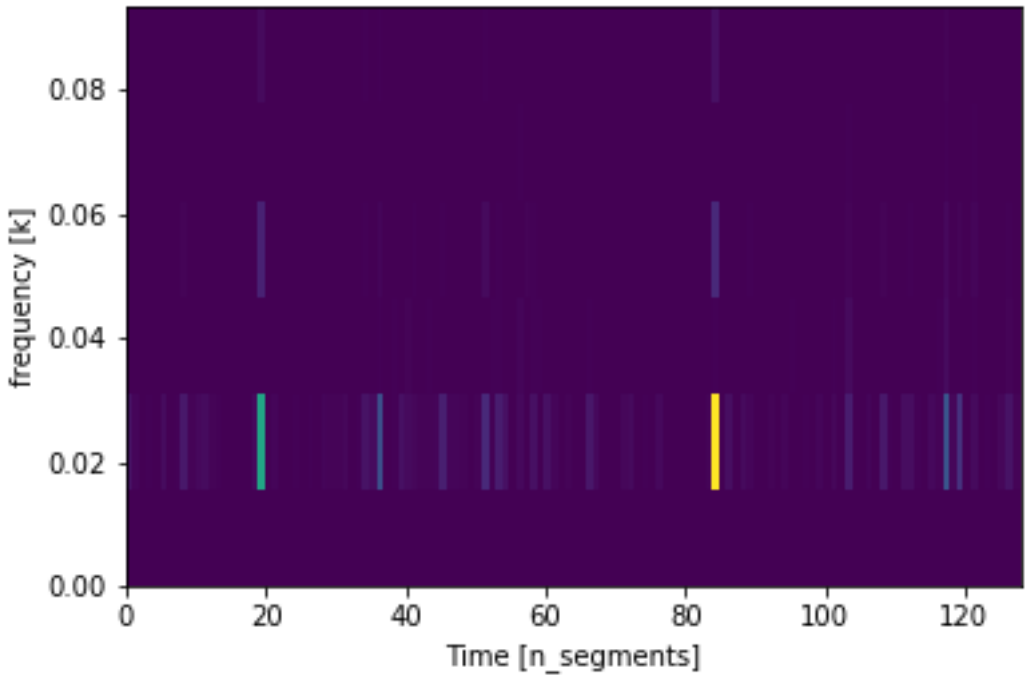


Рисунок 2.2.Спектрограмма

Найдем сегменты с высоким значением метрики, для этого возьмём верхний 5-процентный квантиль. На графике на рисунке 2.3 изображена зависимостьзначения метрики от номера сегмента. Рыжими точками указаны значения, входящие в верхний 5-процентный квантиль.

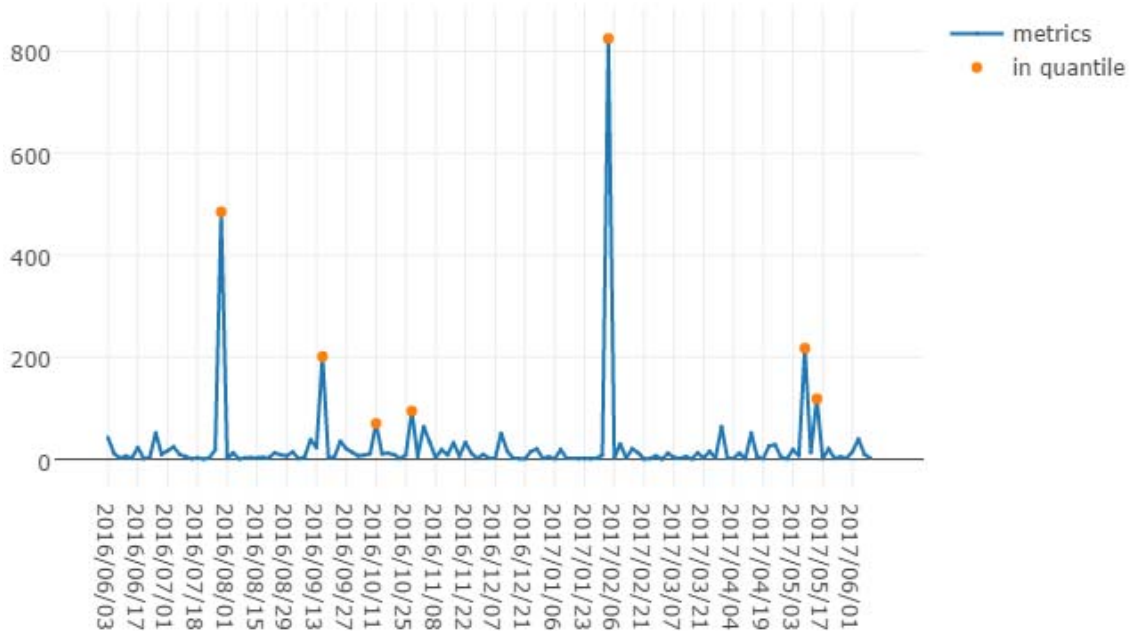


Рисунок 2.3.Метрика и значения, попадающие в квантиль

Мы получили семь дат событий, сравним их с информацией из открытых источников (табл. 2.2)

Дата события из результатов анализа	Дата события из открытых источников	Описание события
-	24/06/2016	Выход Великобритании из Евросоюза
27-28/07/2016	26/07/2016 17:00 (после завершения сессии)	Отчет о доходах за 3 квартал 2016 года[7]
-	07/09/2016	Презентация нового продукта (IPhone 7) [7]
14-15/09/2016	13/09/2016	Выпуск новой операционной системы IOS10[7]
10-11/10/2016	10/10/2016	Компания Samsung, прямой конкурент Apple, заявила о приостановке продаж смартфонов Galaxy 7 в связи с их пожароопасностью[6]
26-27/10/2016	25/10/2016	Отчет о доходах за 4 квартал 2016 года[7]

-	10/11/2016	Выборы президента США
01-02/02/2017	31/01/2017 16:00 (после завершения сессии)	Отчет о доходах за 1 квартал 2017 года[7]
-	02/05/2017 16:00 (после завершения сессии)	Отчет о доходах за 2 квартал 2017 года[7]
08-09/05/2017	08/05/2017	Аналитический отчет брокерской конторы DrexelHamilton[9]
14-15/05/2017	-	Сведения найти не удалось

Таблица2.2. Результаты анализа и события из открытых источников инструментаAAPLза годовой период

Большинство важных событий удалось обнаружить. При этом не удалось обнаружить важное событие, вызванное квартальным отчетом от 02/05/2017, по всей видимости экспертные прогнозы совпали с содержимым отчета и рынок плавно выровнялся еще в период ожидания (конец апреля – начало мая). Падение рынка 15/05/2017 по всей видимости вызвано отскоком после спекулятивного слабообоснованного сильного роста 08/05/2017.

Таким образом вероятность обнаружения истинного события составила 0,64, вероятность того что обнаруженное событие является истинным составила 0,86.

Аналогичным образом были проанализированы событияинструментаAMZN (Amazon)[8], вероятность обнаружения истинного события составила 1, вероятность того что обнаруженное событие является истинным равна 0,7.

2.2 Идентификация событий с использованием информации об объеме рынка

Объем рынка – общее количество акций, проданных в течении некоторого времени. Согласно проведенному группой исследователей Стевенского Технологического Института анализу [1] изменение цены инструмента должно быть подкреплено высоким показателем объема рынка, в противном случае с высокой вероятностью цена инструмента вернется на прежний уровень. Для хорошо торгуемых инструментов вероятность возвращения цены составляет выше 0,9. В таком случае мы можем выдвинуть гипотезу, что важные события должны происходить в периоды высокого объема рынка. Проверим данную гипотезу и попробуем найти события учитывая периоды высокого объема рынка.

На рисунке 2.4 проиллюстрированы графики изменения цены и дневного объема инструмента AAPL.

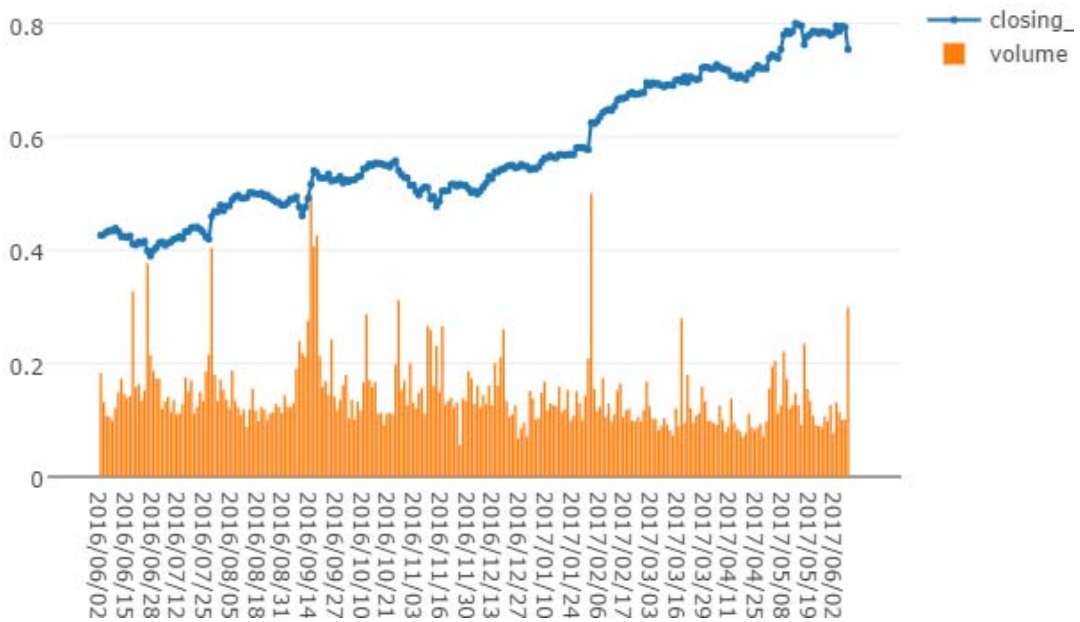


Рисунок 2.4. Цена и дневной объем AAPL

Выделим периоды, когда одновременно происходит изменение цены рынка и дневной объем высок. Для этого мы будем использовать в качестве данных рынка цену инструмента при закрытии торговой сессии.

Отразим данные рынка в более подходящем для нашей задачи виде – в виде изменения цены. Для этого произведем свертку[3] данных рынка с последовательностью[1, 1, -1, -1], т.е. последовательно пройдемся по данным рынка окном, представленным на рисунке 2.5, и определим таким образом степень похожести данных рынка на окно в каждый момент времени.

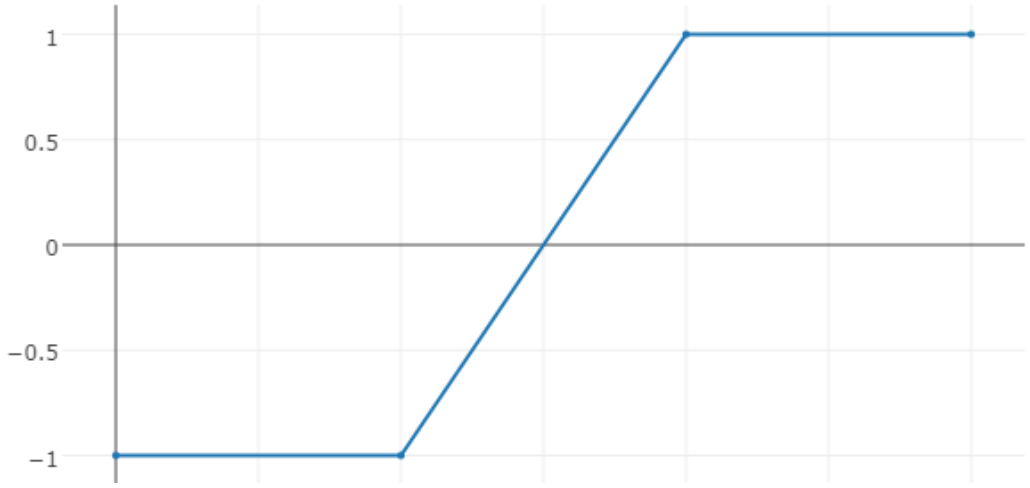


Рисунок 2.5. Окно свертки.

Формула операции свертки для дискретного ряда выглядит следующим образом:

$$y(n) = \sum_{m=0}^{n} x[m] * w[n-m]$$

$y[n] -$ результат свертки

$N -$ количество значений сигнала x

$w -$ окно

Результат свертки представлен на рисунке 2.6 оранжевой кривой.

Рисунок 2.5. Результат свертки

На рисунке 2.6 представлены графики дневного объема и результата свертки.

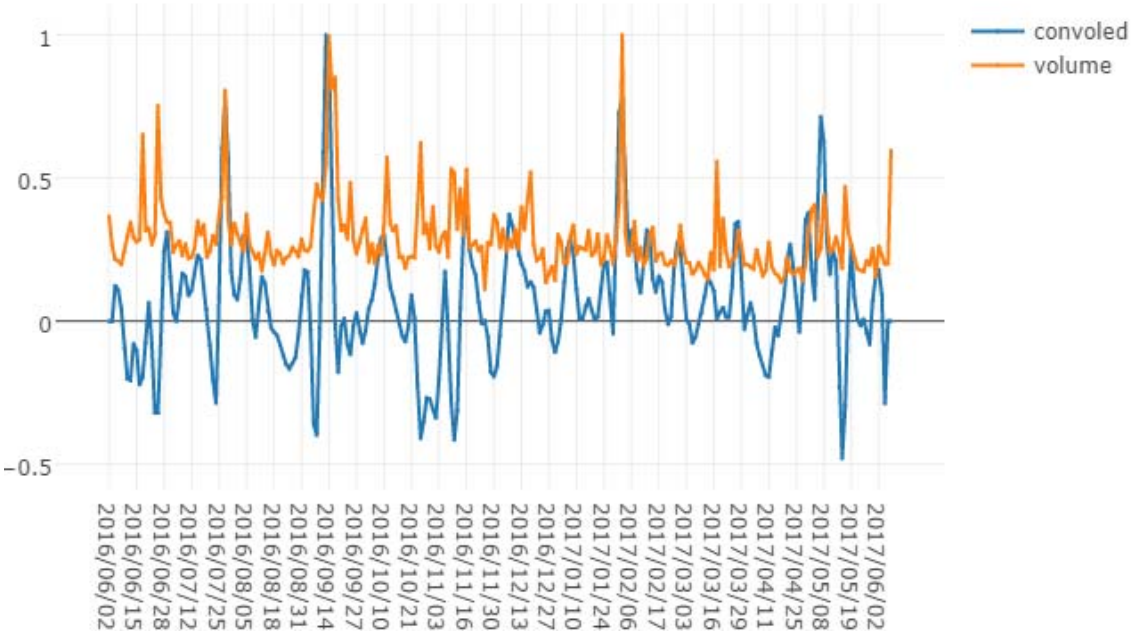


Рисунок 2.5. Результат свертки и объем рынка

Поэлементно перемножим последовательности – результат свёртки и дневной объем рынка. Таким образом мы получим последовательность, которая принимает большие значения в точках с высоким объемом рынка и сильным изменением цены. График данной последовательности представлен на рисунке 2.7

Рисунок 2.7. Результирующая последовательность

На рисунке 2.7 оранжевыми маркерами отмечены дни которые не попали в 95-процентный квантиль по признаку удаленности от среднего. Будем считать, что эти значения свидетельствуют о событиях.

Сверим результаты анализа с данными из открытых источников (табл. 2.3).

Дата события из результатов анализа	Дата события из открытых источников	Описание события
24/06/2016	24/06/2016	Выход Великобритании из Евросоюза

27/07/2016	26/07/2016 17:00 (после завершения сессии)	Отчет о доходах за 3 квартал 2016 года[7]
08/09/2016	07/09/2016	Презентация нового продукта (IPhone 7) [7]
14/09/2016	13/09/2016	Выпуск новой операционной системы IOS10[7]
-	10/10/2016	Компания Samsung, прямой конкурент Apple, заявила о приостановке продаж смартфонов Galaxy 7 в связи с их пожароопасностью[6]
26/10/2016	25/10/2016	Отчет о доходах за 4 квартал 2016 года[7]
10/11/2016	10/11/2016	Выборы президента США
01/02/2017	31/01/2017 16:00 (после завершения сессии)	Отчет о доходах за 1 квартал 2017 года[7]
-	02/05/2017 16:00 (после завершения сессии)	Отчет о доходах за 2 квартал 2017 года[7]
08/05/2017	08/05/2017	Аналитический отчет брокерской конторы DrexelHamilton[9]

Таблица2.3. Результаты анализа и события из открытых источников инструмента AAPLза годовой период

Таким образом вероятность обнаружения истинного события составила 0,8, вероятность того что обнаруженное событие является истинным равна 1.

Аналогичным образом были проанализированы событияинструмента AMZN (Amazon)[8], вероятность обнаружения истинного события составила 0,86, вероятность того что обнаруженное событие является истинным равна 0,86.

2.3 Оценка результатов

Вероятности, вычисленные выше как для первого, так и для второго метода являются лишь точечными оценками реальных вероятностей и не могут служить объективными характеристиками методов. Выборки, на которых эти значения вычислялись, малы и используют лишь два инструмента торговли. Нет уверенности в том, что события, полученные из открытых источников (в таблицах 2.2 и 2.3) имеют одинаковую степень важности, и что не было упущено еще некоторое количество важных событий.

Для более точной оценки степени достоверности методов надо проанализировать выборки за больший промежуток времени и для большего числа инструментов. Но для этого требуется знать объективную информацию о событиях для каждого анализируемого инструмента за длительный промежуток времени, такую информацию можно получить от экспертов хорошо знакомых со спецификой конкретных инструмента. Получение такой информации представляется затруднительным в рамках данной работы.

3 АНАЛИЗ ПОТОКА ЗАКАЗОВ ТРЕЙДЕРА

В данном разделе рассматривается анализ потока заказов трейдера с целью выявления периодов с аномально большим доходом или аномально большой убылью, т.е. периодов, когда трейдер аномально много покупал или аномально много продавал.

Поток заказов выглядит следующим образом (табл. 3.1):

Time	PRICE	BUY_FLAG	QTY
2017/03/01 09:30:00.862	137.88	1	8
2017/03/01 09:30:04.879	137.9	1	2
2017/03/01 09:30:16.734	137.94	0	1
2017/03/01 09:30:25.409	138	1	5
2017/03/01 09:30:55.058	137.92	0	5

Таблица 3.1. Пример потока заказов.

Где PRICE – цена по которой производилась сделка, BUY_FLAG–флаг покупки (означает покупал или продавал трейдер инструмент торговли в данной сделке), QTY – количество акций, проданных или купленных трейдером.

3.1 Алгоритм

Разобьем весь поток заказов на временные интервалы одинаковой длины, будем называть их сегментами. Посчитаем выручку трейдера в каждом сегменте по следующей формуле:

$$y[n] = \sum_{k=0}^{m} PRICE[k] * QTY[k] * (-1)^{BUY\_FLAG}$$

Где

$y[n]$ – выручка в сегменте под номером n

m – количество исполненных заказов трейдера в n – ом сегменте

На рисунке 3.1 визуально представлено распределение заказов по сегментам. Красные треугольники – сделки по продаже, зеленые – сделки по покупке, вертикальное ось – ось цены исполнения сделок.

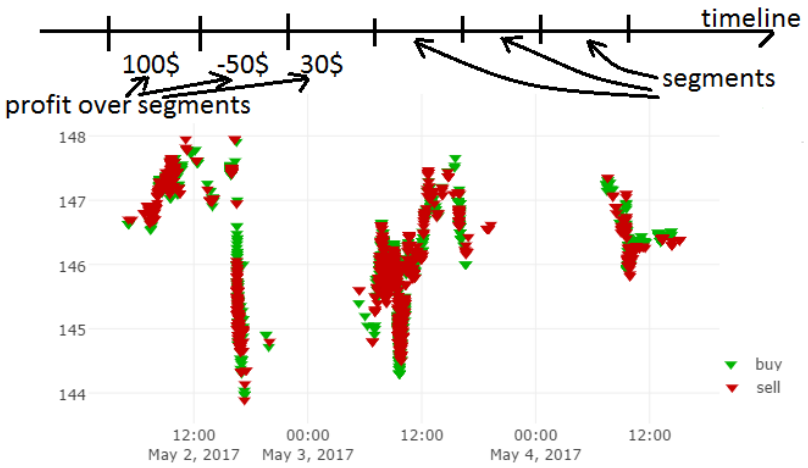


Рисунок 3.1. Распределение выручки по сегментам

Построим распределение выручки сегментов для некоторого зафиксированного разбиения исходного потока заказов на сегменты. На рисунке 3.2 представленабстрактный график (гистограмма) такого распределения: по горизонтальной оси отложим размер выручки сегмента с некоторой дискретизацией окна (для подсчета сегментов, имеющих схожий размер выручки в одном окне), по вертикальной – суммарный объем выручки всех сегментов в данном окне.

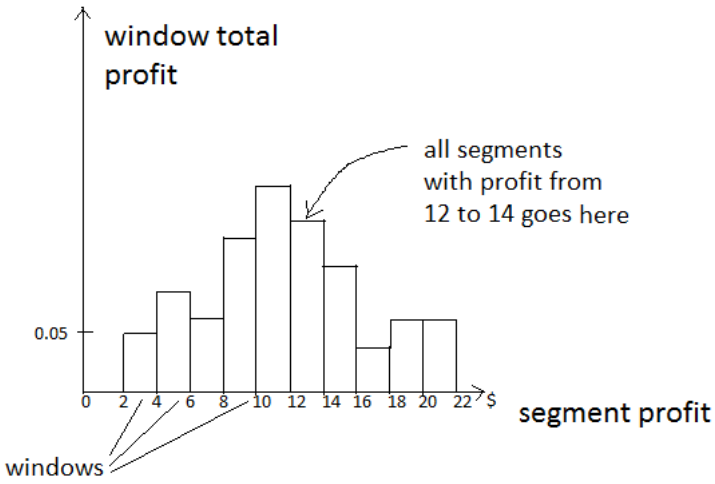


Рисунок 3.2. Гистограмма распределения сегментов

Построим такое распределение для реального потока заказов (рис. 3.3).В качестве исходных данных взят поток заказов длительностью 4 месяца одного из трейдеров инструмента AAPL.Длительность сегментов на которые мы бьём исходный поток заказов равна 2ч 20 мин. По горизонтальной оси откладывается выручка сегмента.Горизонтальная ось разбита на окна, так, что значения, попадающие в одно окно, считаются равными. Вертикальная ось – это суммарная выручка всех сегментов, попавших в одно окно. Зеленые окна– окна содержащие сегменты с отрицательной выручкой, красные – с положительной. Синяя кривая – кривая Гаусса, соответствующая нормальному распределению с такими же значениями среднего и дисперсии, как и у построенного распределения.

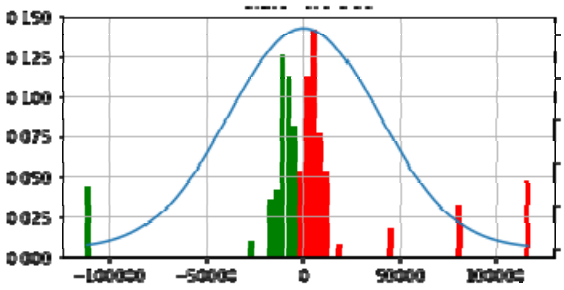


Рисунок 3.3. Реальная гистограмма распределения сегментов

Откинем несколько крайних значений выборки (рис. 3.4) и заметим, что наша выборка стала более похожей на кривую Гаусса.

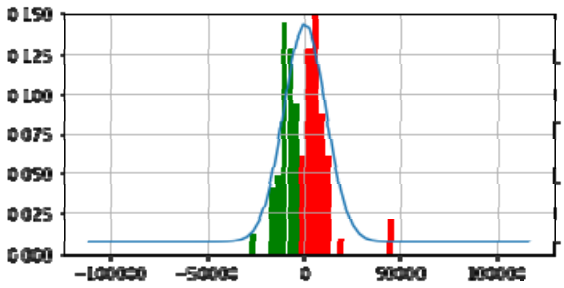


Рисунок 3.4. Реальная гистограмма с откинутыми девиантными значениями

Выдвинем гипотезу, что поток заказов трейдера, постоянно торгующего данным инструментом, разбитый на сегменты распределён нормально. Сегменты, сильно отстоящие от основного распределения, будем считать аномальными и потенциально возможными для инсайдерской торговли.

Реализуем следующий алгоритм по выявлению таких сегментов:

1) Строим распределение на основе рассуждений, приведенных выше, приводим его к вероятностям, деля каждое значение на суммарный объем рынка по всем сегментам.

2) Считаем метрики описательной статистики распределения: среднее, среднеквадратичное отклонение, дивергенцию Ку́льбака—Ле́йблера[2](формула ниже).

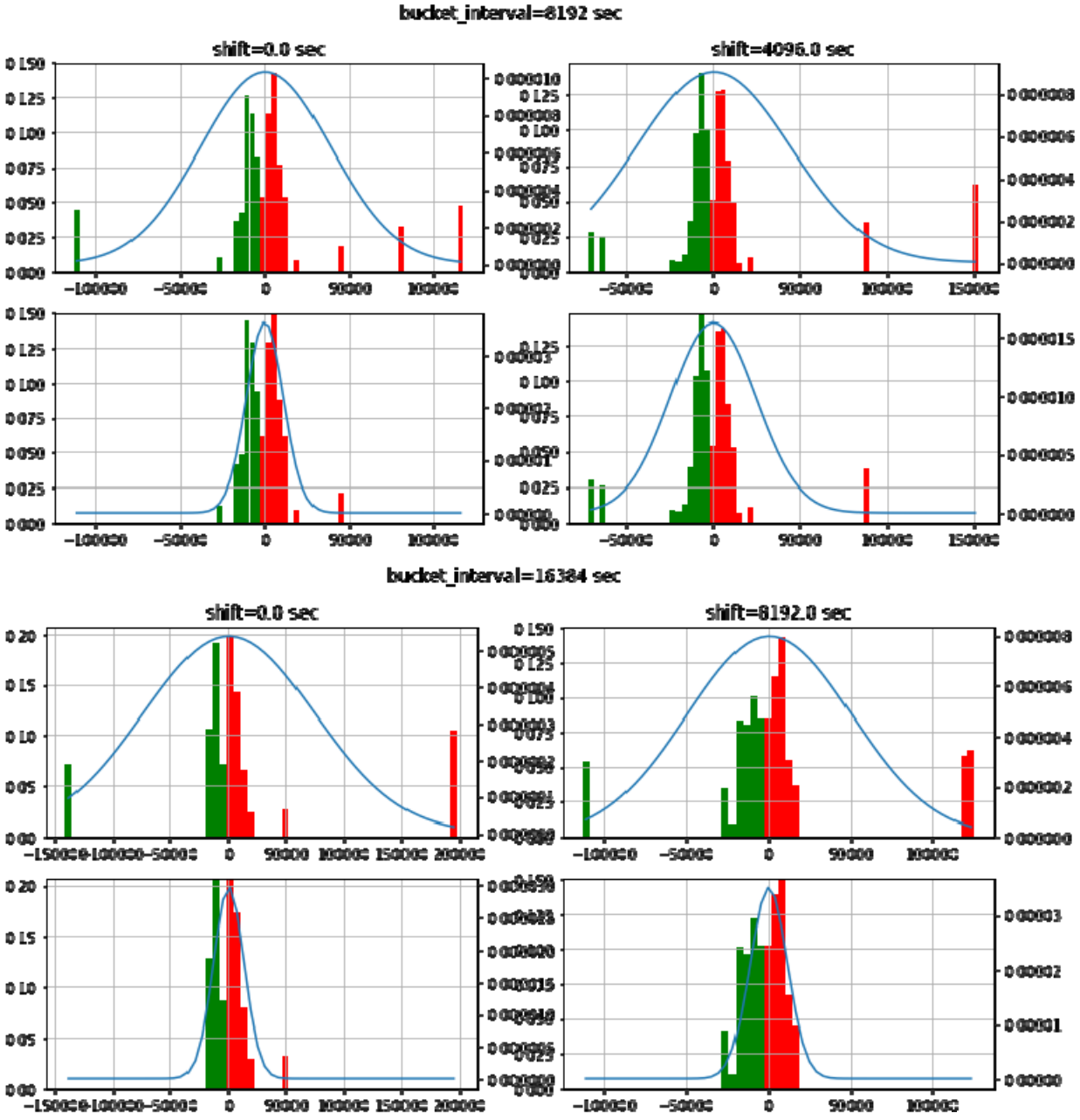
$$D = \sum_{i=1}^{n} \left| p_i * log \frac{p_i}{q_i} \right|$$

Где p_i – вероятности реального распределения, а q_i-вероятности идеального распределения с такими же средним и дисперсией. Дивергенция показывает насколько реальная выборка отличется от идеального распределения.

3) Откидываем от оригинального распределения те крайние сегменты, отсутствие которых приводит наибольшему уменьшению дисперсии. Опять считаем метрики описательной статистики. Оцениваем дивергенцию. Если дивергенция меньше некоторого порога как для нового, так и для исходного расперделения, считаем, что девиантные сегменты незначительно влияют на распределение.

4) Проделываем п.1-4 для нескольких различных длин сегментов и для разных фаз (время, когда сегмент начинается), и выбираем из них тот, в котором разница дисперсий исходного распределения и распределения с откинутыми крайними значениями больше.

На рисунке 3.5 приведена серия таких распределений для потока заказов некоторого трейдера для инструмента AAPL.Bucket_interval – размер сегментов, shift – сдвиг фазы от начала данных. Верхние два распределения в каждом квадрате – это исходные распределения, нижние – распределения с откинутыми девиантными значениями.


bucket_interval=8192 sec
shift=0.0 sec
shift=4096.0 sec
bucket_interval=16384 sec
shift=0.0 sec
shift=8192.0 sec

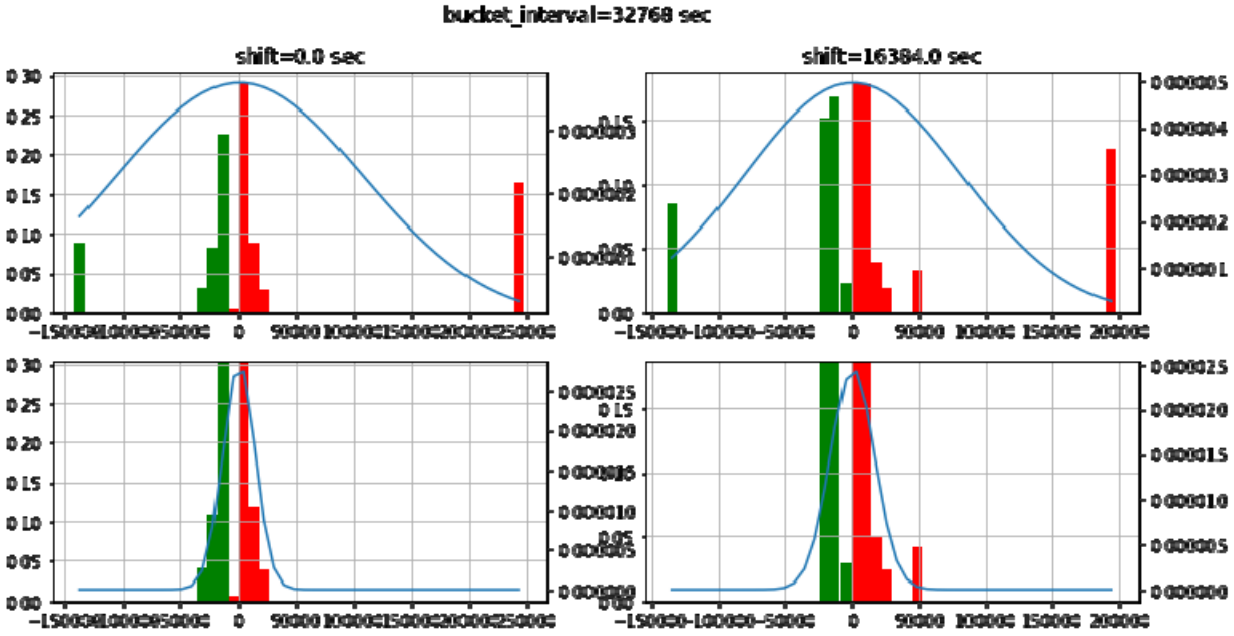


Рисунок 3.5. Серия распределений

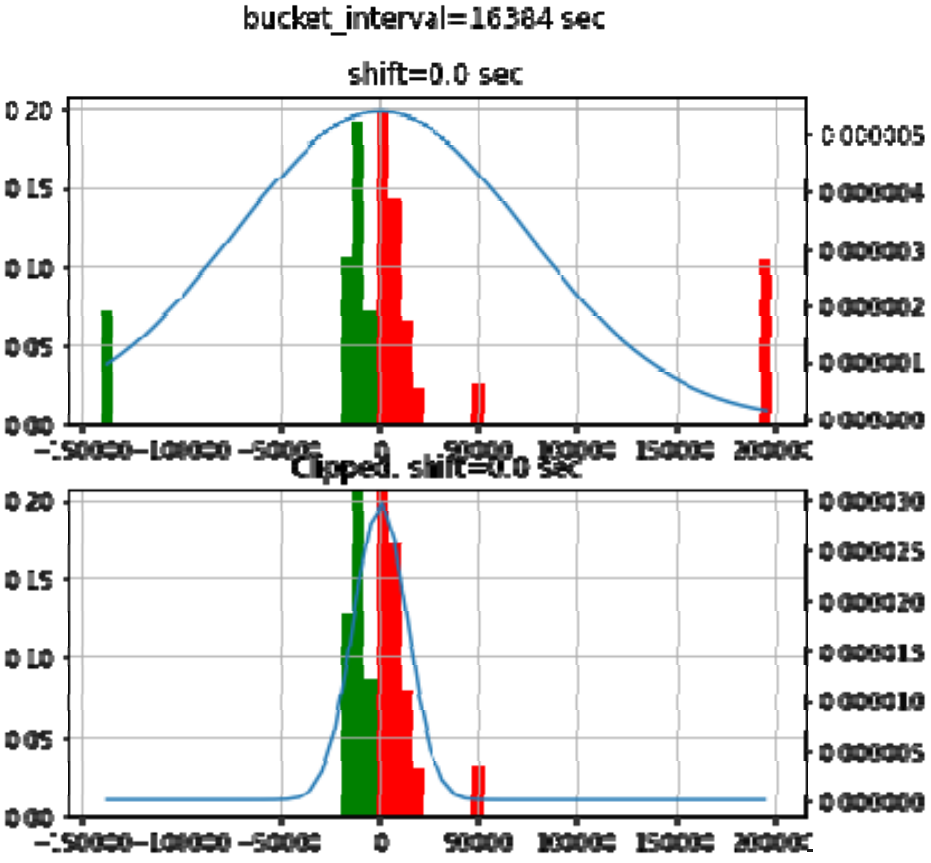


Рисунок 3.6. Распределения с наименьшей итоговой дисперсией

На рисунке 3.6 нижнее распределение имеет наименьшую дисперсию по отношению к верхнему. Дивергенция верхнего распределения сильно выше дивергенции нижнего.

Таким образом мы получили два девиантных сегмента (указаны время начала, время конца и размер выручки данных сегментах):

1) '27.01.2017 12:22:44', '27.01.2017 16:55:48', -137708.0
2) '03.02.2017 12:46:12', '03.02.2017 17:19:16', 201327.0

Будем считать данные сегменты потенциально возможными для инсайдерской торговли.

Проверим их. Для этого рассмотрим их положение на оси времени вместе с данными рынка. На рисунке 3.7 первый сегмент расположен между двумя вертикальными зелеными растрами, второй между двумя красными растрами.

Рисунок 3.7.Девиантные сегменты и данные рынка

Данные сегменты расположены вблизи от важного события – выхода квартального отчета о доходах компании.

Удостоверимся, что на сделки, совершенные в данные интервалы времени стоит обратить внимание. Для этого посчитаем позицию трейдера за двухнедельный период – неделю до и неделю после события, т.е. рассмотрим все сделки, совершенные трейдером за этот период и оценим его итоговую прибыль или убыль.

Пусть трейдер в начальный момент времени имеет нулевую сумму денег и нулевое количество акций. Рассчитаем итоговую суммуденег на руках трейдера и итоговое количество акций в конце периодаследующим образом:

$$PROFIT = \sum_{k=0}^{m} PRICE[k] * QTY[k] * (-1)^{BUY\_FLAG}$$

$$SHARES\_NUMBER = \sum_{k=0}^{m} QTY[k] * (-1)^{BUY\_FLAG}$$

Тогда итоговая позиция трейдера будет:

$$POSITION = PROFIT - SHARES\_NUMBER * PRICE$$

Где $PRICE$ – цена акций в конце периода.

Если итоговая позиция значительно больше нуля, на сделки в девиантных сегментах действительно стоит обратить внимание. График на рисунке 3.8 иллюстрирует изменение количества акций на руках трейдера (красная кривая) на фоне изменения цены (синяя кривая).

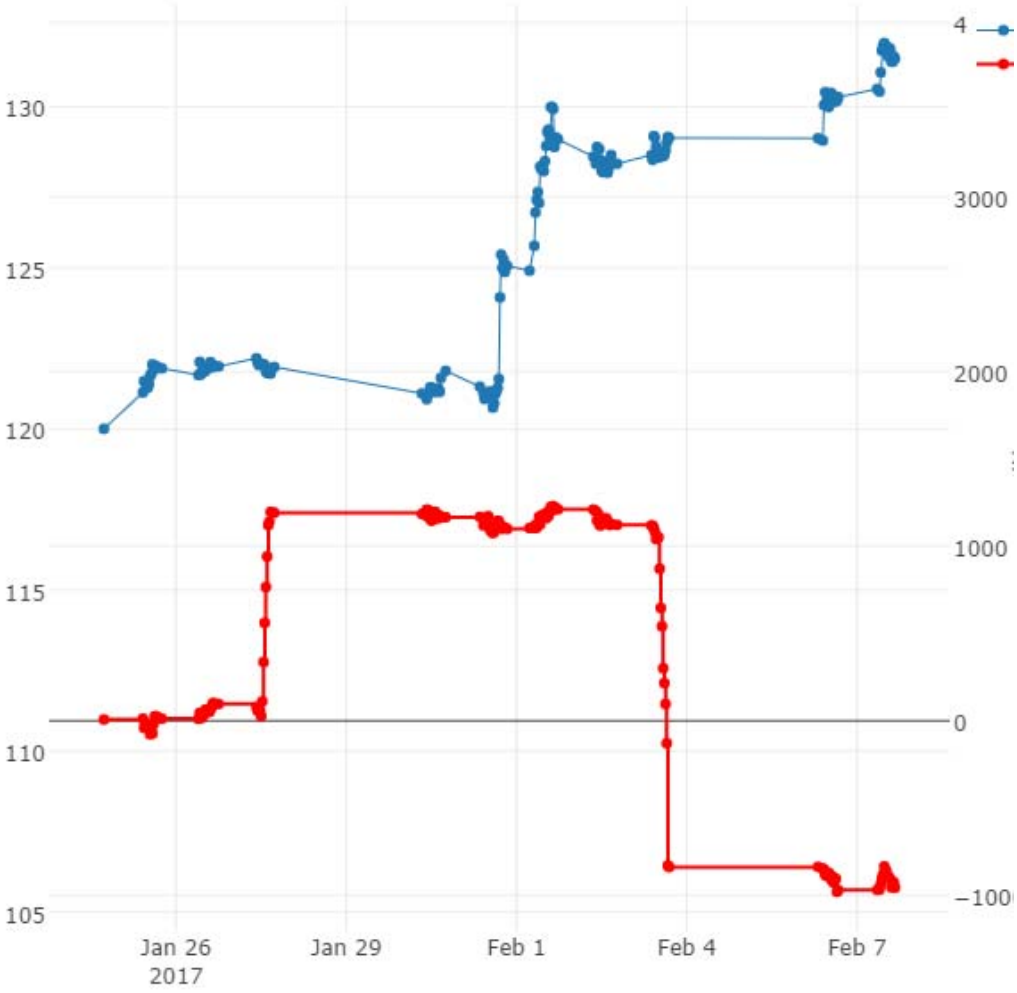


Рисунок 3.7. Изменение количества акций

3.2Оценка результатов

В данном разделе был рассмотрен не реальный поток заказов, а специально сгенерированный. Алгоритм описанный ваше может быть работоспособным только в случае, когда имеется полный поток заказов трейдера за достаточно длительный промежуток времени. В противном случае распределение сегментов, описанное выше, будет иметь всего несколько значений и говорить о каких-либо отклонениях не имеет смысла.

Для автоматизированного выполнения данной части задачи было разработано приложение с веб интерфейсом, позволяющее задавать основные параметры алгоритмов и визуализировать результаты. Интерфейсы приложения изображены на рисунке 3.8. Серверная часть приложения написана на языке pythoni использует библиотекиpandas и scipyдля обработки и анализа данных. Клиентская часть приложения написана на языке javascriptи использует библиотеку plotly для отображения графиков.

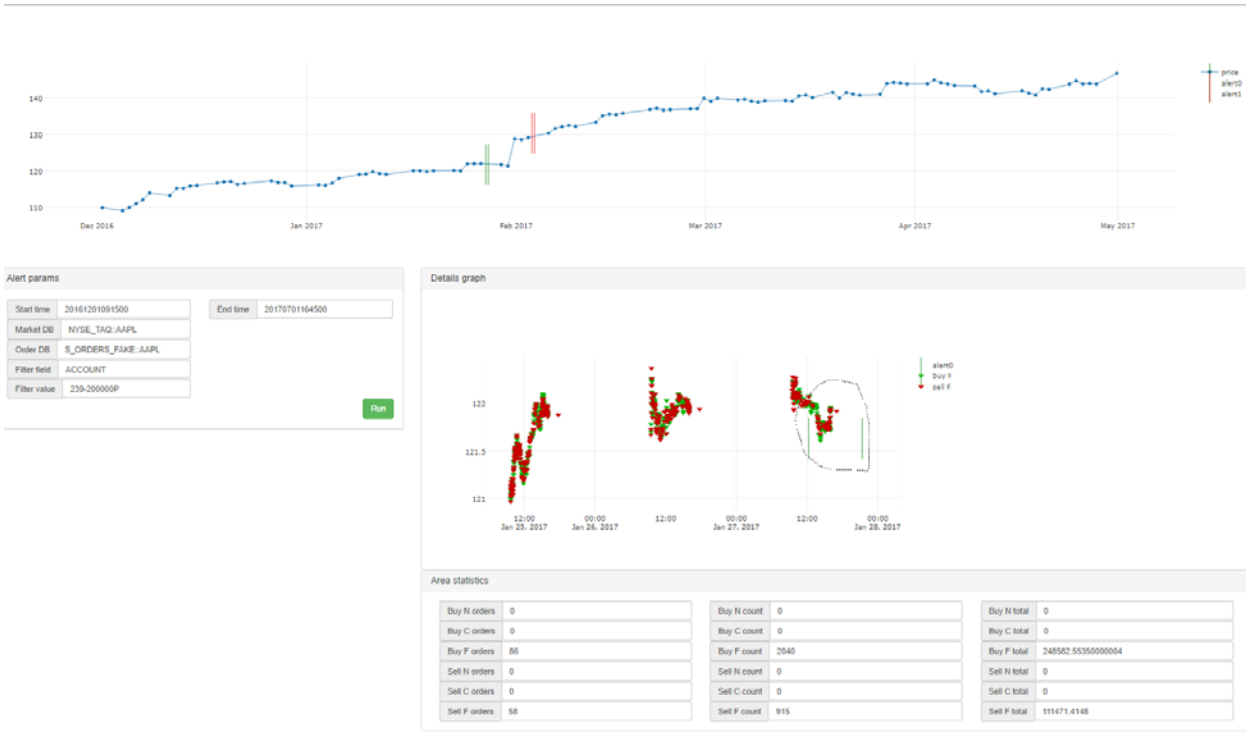


Рисунок 3.8. Интерфейс программы

ЗАКЛЮЧЕНИЕ

В данной работе была рассмотрена задача обнаружения событий, значительно влияющих на движение цены определенного инструмента торговли, а также задача обнаружения инсайдерской торговли по данным рынка и данным потока заказов трейдера.

Для задачи обнаружения событий было предложено два метода её решения: решение с помощью спектрального анализа и решение с помощью анализа с использованием данных об объеме рынка. На имеющихся тестовых данных второй метод показал себя лучше первого.

Для задачи обнаружения инсайдерской торговли был предложен следующий алгоритм решения:

1) Разбиение потока заказов на сегменты и выявление подозрительных сегментов с аномально высокой прибылью или с аномально высокой убылью

2) Если сегменты из п.1 были найдены, то оценивается их временное положение – близость к событиям выявленным в первой части работы.

3) В случае если данные сегменты находятся в близи от событий, то оценивается позиция трейдера в некотором интервале времени возле события.

4) Если позиция на выходе из интервала большая (больше некоторого порога), генерируется сигнал примерно следующего содержания:

   «Возможно имела место инсайдерская торговля между 2017-01-24 и 2017-02-07. Трейдер с id = 42 покупал AAPL в большом

объемемежду 2017-01-27 12:22:44 и 2017-01-27 16:55:48 и продавал между 2017-02-03 12:46:12 и 2017-02-03 17:19:16. Его общая прибыль составила 6984 доллара. Торговля осуществлялась в непосредственной близости от событиявызвавшего изменение цены 2017-01-31.»

Для автоматизация и визуализация результатов анализа было разработано приложение с веб-интерфейсом.

Некоторые части алгоритма не были проверены на большом количестве реальных данных ввиду их отсутствия. Алгоритм был проверен на специально сгенерированных тестовых данных. Возможно потребуется корректировка отдельных частей алгоритма при тестировании на реальных данных.

Методы обнаружения событий могут быть полезны трейдерам и другим участникам рынка.

Алгоритм обнаружения инсайдерской торговли может быть полезен брокерским конторам и инвестиционным фондам для отслеживание действий своих сотрудников, биржам и надзорным органам для обнаружения злоумышленников на финансовом рынке.

СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ

1. «Rare Events Analysis for High-Frequency Equity Data» -DragosBozdog, Ionuţ Florescu1, KhaldounKhashanah, Jim Wang; Stevens Institute of Technology, Hoboken, 2011
2. «Kullback S. Information Theory and Statistics.» — John Wiley & Sons, 1959.
3. «Элементы теории функций и функционального анализа», — Колмогоров А. Н., Фомин С. В., М.: Наука, 2004 (7-е изд.)
4. «Об установлении фактов инсайдерской торговли акциями ПАО АФК Система». Банк России. [электронныйресурс] https://www.cbr.ru/press/pr.aspx?file=16032017_163009sbrfr2017-03-16t16_13_52.html
5. «Insider Trading» Chapter 29; Larry Harris, Trading & Exchanges, Oxford Press, Oxford, 2003.
6. «Samsung Expands Recall of Galaxy Note7 Devices to Include Original and Replacement Devices» [электронныйресурс] https://news.samsung.com/us/samsung-expands-recall-of-galaxy-note7-devices-to-include-original-and-replacement-devices-company-offers-refund-and-exchange-program/
7. «Applereports» [электронный ресурс]http://investor.apple.com/
8. «Amazon Recent Press Releases»[электронныйресурс]http://phx.corporate-ir.net/phoenix.zhtml?c=176060&p=irol-mediahome
9. «Drexel Hamilton Highlights Buying Opportunity for Apple Inc.»[электронный ресурс] http://drexelhamilton.com/

ПРИЛОЖЕНИЕ. ПРОГРАММНЫЙ КОД

Веб приложение.

```
import NumPy_OneTickQueryas OTQ
import json
import datetime

import pandas as pd
import plotly
import plotly.graph_objsas go
from flask import Flask, render_template
from flask_socketioimport SocketIO, emit
from plotly.graph_objsimport Scatter, Figure

from query_argsimport QueryArgs, OtqParams
from utilsimport get_by_name, plotly_div

CONTEXT = "PHANTOM_47030"
# CONTEXT = "DEFAULT"
MARKET_DB = 'NYSE_TAQ::AAPL'
ORDER_DB = 'S_ORDERS_FAKE::AAPL'
EVENTS_DB = 'EVENTS_FAKE::AAPL'
FILTER_FIELD = 'ACCOUNT'
FILTER_VALUE = '239-200000P'
START_DATE = '20161201093000'
END_DATE = '20170701160000'
SCREENING_INTERVAL = 7
THRESHOLD = 0
```

```
app = Flask(__name__)
app.debug = True
app.secret_key = 'any random string'
socketio = SocketIO(app)
session = dict()


defget_alerts():
query_args = QueryArgs()
query_args.otq_file = './Position.otq::events'
query_args.s = START_DATE
query_args.e = END_DATE
query_args.timezone = "America/New_York"
query_args.context = CONTEXT
query_args.treat_byte_arrays_as_strings = ''
query_args.otq_params = OtqParams()
query_args.otq_params.EventsDB = EVENTS_DB

   print('\tQUERYget_events:', str(query_args))
   _, data, _, _ = OTQ.run_query([str(query_args)])[0]

if len(data) == 0:
return tuple()

df = pd.DataFrame(
       {'time': get_by_name('Time',
data).dt.tz_localize('UTC').dt.tz_convert('America/New_York').dt.tz_localize(None),
        'id': get_by_name('ID', data)})
```

```
for i, event in df.iterrows():

start_time = event['time'] - datetime.timedelta(days=SCREENING_INTERVAL)
end_time = event['time'] + datetime.timedelta(days=SCREENING_INTERVAL)
        print('\t screening time (start,event,end)')
        print('\t', start_time)
        print('\t', event['time'])
        print('\t', end_time)

query_args = QueryArgs()
query_args.otq_file = './Position.otq::position'
query_args.s = start_time.strftime('%Y%m%d%H%M%S')
query_args.e = end_time.strftime('%Y%m%d%H%M%S')
query_args.timezone = "America/New_York"
query_args.context = CONTEXT
query_args.treat_byte_arrays_as_strings = ''
query_args.otq_params = OtqParams()
query_args.otq_params.OrderDB = ORDER_DB
query_args.otq_params.filter_field = 'ACCOUNT'
query_args.otq_params.filter_value = '239-200000P'
query_args.otq_params.threshold = THRESHOLD

        print('\tQUERYget_position:', str(query_args))
        _, data, _, _ = OTQ.run_query([str(query_args)])[0]

if len(data) > 0:
            # print('event_id=', event['id'])
position_df = pd.DataFrame(
```

```
            {'time': get_by_name('Time', data).dt.tz_localize('UTC').dt.tz_convert(
               'America/New_York').dt.tz_localize(None),
             'price': get_by_name('PRICE', data),
             'bank_size': get_by_name('BANK_SIZE', data),
             'bank_total': get_by_name('BANK_TOTAL', data)})
yield (event, position_df)
        # print(position_df)


@app.route('/position')
def dash():
   # initialize_session()
   print("NEW /dash session")
   figure = []

for event, position_dfin get_alerts():
price_min = min(position_df['price'])
price_max = max(position_df['price'])
price_delta = price_max - price_min
position_min = min(position_df['bank_size'])
position_max = max(position_df['bank_size'])
position_delta = position_max - position_min

      trace1 = Scatter(x=position_df['time'], y=position_df['price'],
mode='markers+lines', name='market',
                  line=dict(width=1))

      trace2 = Scatter(x=position_df['time'], y=position_df['bank_size'],
mode='markers+lines', name='number of shares',
```

```
yaxis='y2',  line=dict(shape='hv', color='red'))
        # fill="tozeroy",
        # trace2 = go.Bar(x=position_df['time'], y=position_df['bank_size'],
name='position',
        #                 yaxis='y2', marker=dict(
        #         color=['red' if row['bank_size'] > 0 else 'green' for i, row in
position_df.iterrows()]))
        scatters = [trace1, trace2]

        layout = go.Layout(
yaxis=dict(title='market', range=[price_min - 1.3 * price_delta, price_max + 0.1 *
price_delta]),
            yaxis2=dict(title='shares number', overlaying='y', side='right',
                        range=[position_min - 0.1 * position_delta, position_max + 1.3 *
position_delta]),
            height=700
        )
        figure = Figure(data=scatters, layout=layout, )

    print("END /dash")
return render_template('layouts/position.html', div_placeholder=plotly_div(figure,
'graphDiv'))


@socketio.on('run_alert')
defrun_alert(message):
    print("NEW socketio: run_alert session")
    print('\tmessage=', message)
```

```
global MARKET_DB, ORDER_DB, FILTER_FIELD, FILTER_VALUE,
START_DATE, END_DATE, EVENTS_DB, SCREENING_INTERVAL,
THRESHOLD

   MARKET_DB = message['MARKET_DB']
   ORDER_DB = message['ORDER_DB']
   EVENTS_DB = message['EVENTS_DB']
   FILTER_FIELD = message['FILTER_FIELD']
   FILTER_VALUE = message['FILTER_VALUE']
   START_DATE = message['START_DATE']
   END_DATE = message['END_DATE']
   SCREENING_INTERVAL = int(message['SCREENING_INTERVAL'])
   THRESHOLD = float(message['THRESHOLD'])

   figure = []
for event, position_dfin get_alerts():
price_min = min(position_df['price'])
price_max = max(position_df['price'])
price_delta = price_max - price_min
position_min = min(position_df['bank_size'])
position_max = max(position_df['bank_size'])
position_delta = position_max - position_min

      trace1 = Scatter(x=position_df['time'], y=position_df['price'],
mode='markers+lines', name='market',
                  line=dict(width=1))
      # trace1 = Scatter(x=position_df['time'], y=position_df['price'], fill='tozeroy')
      trace2 = Scatter(x=position_df['time'], y=position_df['bank_size'],
mode='markers+lines', name='number of shares',
```

```
yaxis='y2',  line=dict(shape='hv', color='red'))
        scatters = [trace1, trace2]

        layout = go.Layout(
yaxis=dict(title='market', range=[price_min - 1.3 * price_delta, price_max + 0.1 *
price_delta]),
            yaxis2=dict(title='shares namber', overlaying='y', side='right',
                        range=[position_min - 0.1 * position_delta, position_max + 1.3 *
position_delta]),
            height=700
        )
        figure = Figure(data=scatters, layout=layout)

    print("END socketio: run_alert")
    emit('render_alerts', json.dumps(figure, cls=plotly.utils.PlotlyJSONEncoder))


if __name__ == '__main__':
socketio.run(app, ping_timeout=300, host='0.0.0.0', port=40003)
```

Аналитическиевставки

```
# ------------------------------------------------

window = np.array([1,1,0,-1,-1])

closing_df['CONVOLVED'] =
np.concatenate([[0,0],np.convolve(closing_df['PRICE'],window,'valid'),[0,0]])

iplot([Scatter(x=closing_df['Time'], y=closing_df['PRICE']-
closing_df['PRICE'].mean(), name='closing',

mode='markers+lines', marker=dict(size=4)),

    Scatter(x=closing_df['Time'], y=closing_df['CONVOLVED'],
name='convoled',

            mode='markers+lines', marker=dict(size=4))],

show_link=False)

# ------------------------------------------------

closing_df =
closing_df[pd.to_datetime(closing_df['Time']).dt.date.isin(pd.to_datetime(volume_
df['Time']).dt.date)]

volume_df =
volume_df[pd.to_datetime(volume_df['Time']).dt.date.isin(pd.to_datetime(closing
_df['Time']).dt.date)]

volatility_df =
volatility_df[pd.to_datetime(volatility_df['Time']).dt.date.isin(pd.to_datetime(closi
ng_df['Time']).dt.date)]
```

```
closing_df = closing_df.reset_index(drop=True)

volume_df = volume_df.reset_index(drop=True)

volatility_df = volatility_df.reset_index(drop=True)

# ------------------------------------------------

result_df = pd.DataFrame()

result_df['Time'] = closing_df['Time']

print(len(closing_df['CONVOLVED']), len(volatility_df['STDDEV']),
len(volume_df['VOLUME']))

result_df['VALUE'] = closing_df['CONVOLVED'] *volume_df['VOLUME']

result_df['VALUE'] = result_df['VALUE']/max(abs(result_df['VALUE']))


result_df['ABS_VALUE'] = abs(result_df['VALUE'] - result_df['VALUE'].mean())

result_df = result_df.sort_values(by='ABS_VALUE')

upper_bound = result_df['ABS_VALUE'].quantile(q=0.95)

result_df['EVENT_FLAG'] = result_df['ABS_VALUE']//upper_bound


result_df = result_df.sort_values(by='Time')
```

```
max_indexes, = argrelmax(np.array(result_df['VALUE']))

min_indexes,=argrelmin(np.array(result_df['VALUE']))

extrem_indexes = sorted(np.concatenate((min_indexes,max_indexes)))

result_df.loc[~result_df.index.isin(extrem_indexes),'EVENT_FLAG']=0


iplot([Scatter(x=result_df['Time'], y=result_df['VALUE'], name='result',

mode='markers+lines', marker=dict(size=2)),

    Scatter(x=result_df['Time'][result_df['EVENT_FLAG']>=1],
y=result_df['VALUE'][result_df['EVENT_FLAG']>=1], name='abnormals',

            mode='markers', marker=dict(size=6))],

    show_link=False)
```

Поискдевиантныхинтервалов

```
import NumPy_OneTickQueryas OTQ
import argparse
import re
from copy import deepcopy
from datetimeimport datetime
from datetimeimport timedelta
from time import strftime
```

```
import numpyas np
import pandas as pd
import scipy.statsas st
from pytzimport timezone
from query_argsimport QueryArgs, OtqParams

# ---- config -----#

std_coefs_range = np.arange(3.1, 0.399, -0.1)
bucket_intervals_range = range(13, 16)
n_steps_coef = 3
k_diverg_threshold = 0.65
k_stddev_threshold = 0.7

# ---- END config -----#

defget_data(formatter_string):
""" otq execution """
(_, data, _, _), = OTQ.run_query([formatter_string])
   # data = OTQ.run_query([query_formatter_string])[0][1]
   # print(data)
return data



defget_by_name(name, data):
""" transform data to pandas dataframe """
for desc, values in data:
if desc == name:
```

```
return pd.Series(values)

    print('WARN : for %s return none' % name)
return None



defbuild_df(d_names, data):
""" transform data to pandas dataframe """
d = {}
for k, v in d_names.items():
        d[k] = get_by_name(v, data)

return pd.DataFrame(d)



defget_filtered_data(data):
data_df = build_df({
        'time': 'Time',
        'profit': 'PROFIT',
    },
        data)

return data_df

# ------------ LOGIC ---------------- #

defget_best_std_reduce_coef(series, delimiters):
    mean = np.mean(series)
    probabilities = build_probabilities(series, delimiters)
```

```
std = stddev(mean, delimiters, probabilities)

prev_std_reduced = std
cur_std_reduced = std
best_std_reduce_coef = 100000
best_std_ratio = cur_std_reduced / prev_std_reduced

for std_coefin std_coefs_range:

cur_siries_reduced = build_reduced_siries(series, delimiters, probabilities,
std_reduce_coef=std_coef)
cur_probabilities_reduced = build_probabilities(cur_siries_reduced, delimiters)
cur_mean_reduced = np.mean(cur_siries_reduced)
cur_std_reduced = stddev(cur_mean_reduced, delimiters,
cur_probabilities_reduced)

cur_std_ratio = cur_std_reduced / prev_std_reduced

if cur_std_ratio<best_std_ratio:
best_std_ratio = cur_std_ratio
best_std_reduce_coef = std_coef

prev_std_reduced = cur_std_reduced

return best_std_reduce_coef, best_std_ratio



defstddev(mean, delimiters, probabilities):
std = 0
```

```
for i, x in enumerate(delimiters[:-1]):
std += abs(probabilities[i]) * (x - mean) ** 2
std **= 0.5
return std

defbuild_probabilities(series, delimiters):
   probabilities = list()
for j in range(len(delimiters) - 1):
cond = (delimiters[j] - 1 < series).T * (series <= delimiters[j + 1])
probabilities.append(sum(series[cond]))

   probabilities = np.array(probabilities) / sum(abs(np.array(probabilities)))

return probabilities

defbuild_reduced_siries(series, delimiters, probabilities, std_reduce_coef):
   mean = np.mean(series)
std = stddev(mean, delimiters, probabilities)
series_reduced = np.array(
      [iif mean - std_reduce_coef * std<= i<= mean + std_reduce_coef * stdelse 0
for iin series])
return series_reduced


defget_suspicious_intervals(data, bucket_interval, lower_lim, upper_lim):
   result = []
bucket_interval = timedelta(seconds=2 ** bucket_interval)
for i, bucket in data.iterrows():
      profit = bucket['profit']
if not lower_lim< profit <upper_lim:
```

```
start_time = bucket['time']
start_time =
start_time.replace(tzinfo=timezone('UTC')).astimezone(tz=timezone('America/New_York'))
end_time = start_time + bucket_interval

        # print(start_time.strftime('%d.%m.%Y_%H:%M:%S'),end='  ')
        # print(end_time.strftime('%d.%m.%Y_%H:%M:%S'))

start_time = int(start_time.timestamp() * 1000)  #
strftime('%d.%m.%Y_%H:%M:%S')
end_time = int(end_time.timestamp() * 1000)  #
strftime('%d.%m.%Y_%H:%M:%S')

result.append((start_time, end_time, profit))
return result
# ========== MAIN ===========
if __name__ == '__main__':
   # ----- read command line -----

   parser = argparse.ArgumentParser()
parser.add_argument("start_time", type=str)
parser.add_argument("end_time", type=str)
parser.add_argument("timezone", type=str)
parser.add_argument("context", type=str)
parser.add_argument("OrderDB", type=str)
parser.add_argument("filter_field", type=str)
parser.add_argument("filter_value", type=str)
args = parser.parse_args()
```

```
query_args = QueryArgs()
query_args.otq_file = 'C:/OMD/tasks/SURV-31/InsiderTrading.otq::EarnedMoney'
query_args.context = args.context
query_args.s = args.start_time
query_args.e = args.end_time
query_args.timezone = args.timezone
query_args.treat_byte_arrays_as_strings = ''

query_args.otq_params = OtqParams()
query_args.otq_params.OrderDB = args.OrderDB
query_args.otq_params.bucket_interval = 120
query_args.otq_params.filter_field = args.filter_field
query_args.otq_params.filter_value = args.filter_value



   # ----- end command line ------



max_diverg = 2.5
best_k_stddev = 2
best_std_reduce_ratio = 1
best_k_diverg = 0
best_bucket_interval = 0
best_shift = 0
best_siries = []
best_std_reduce_coef = 0
   # best_start_time = 0
best_filtered_data = []
```

```
best_mean = 0
best_std = 0

for bucket_interval in bucket_intervals_range:  # (5,17)
for shift in range(2):
query_args.otq_params.bucket_interval = 2 ** bucket_interval

tmp_start_time = datetime.strptime(query_args.s, '%Y%m%d%H%M%S')
tmp_start_time -= timedelta(seconds=2 ** bucket_interval / 2 * shift)
query_args.s = tmp_start_time.strftime("%Y%m%d%H%M%S")

        # print(query_args.s)
        data = get_data(str(query_args))

filtered_data = get_filtered_data(data)
siries = np.array(filtered_data['profit'])
siries = np.sort(siries)

        ######### LOGIC #########

min_elem, max_elem = siries[0], siries[-1]
n_steps = n_steps_coef * int(len(siries) ** 0.5 / 1)
        step = (max_elem - min_elem) / n_steps
        delimiters = np.linspace(min_elem, max_elem, n_steps)

std_reduce_coef, std_reduce_ratio = get_best_std_reduce_coef(siries, delimiters)

        probabilities = build_probabilities(siries, delimiters)
siries_reduced = build_reduced_siries(siries, delimiters, probabilities,
```

```
std_reduce_coef=std_reduce_coef)
probabilities_reduced = build_probabilities(siries_reduced, delimiters)

        # -- ststistics --
        mean = np.mean(siries)
std = stddev(mean, delimiters, probabilities)
        pdf = st.norm(mean, std)
pdf_vals = pdf.pdf(delimiters[:-1])
diverg = st.entropy(abs(probabilities), pdf_vals)

mean_reduced = np.mean(siries_reduced)
std_reduced = stddev(mean_reduced, delimiters, probabilities_reduced)
pdf_reduced = st.norm(mean_reduced, std_reduced)
pdf_reduced_vals = pdf_reduced.pdf(delimiters[:-1])
diverg_reduced = st.entropy(abs(probabilities_reduced), pdf_reduced_vals)

k_stddev = std_reduced / std
k_diverg = diverg_reduced / diverg

        # if k_stddev<best_k_stddev and diverg<max_diverg and
diverg_reduced<max_diverg:
if std_reduce_ratio<best_std_reduce_ratioand diverg<max_divergand
diverg_reduced<max_diverg:
best_std_reduce_ratio = deepcopy(std_reduce_ratio)
best_k_stddev = k_stddev
best_bucket_interval = deepcopy(bucket_interval)
best_shift = deepcopy(shift)
best_siries = deepcopy(siries)
            # best_start_time = deepcopy(start_time)
```

```
best_std_reduce_coef = deepcopy(std_reduce_coef)
best_k_diverg = deepcopy(k_diverg)
best_filtered_data = deepcopy(filtered_data)
best_mean = deepcopy(mean)
best_std = std

            ######### END LOGIC #########
if best_k_stddev == 2:
    print('NOT NORMAL')
elifbest_k_diverg>k_diverg_thresholdor best_k_stddev>k_stddev_threshold:
    print('NOT SUSPICIOUS INTERVALS')
else:
suspicious_intervals = get_suspicious_intervals(best_filtered_data,
best_bucket_interval,
best_mean - best_std_reduce_coef * best_std,
best_mean + best_std_reduce_coef * best_std)

    # print(suspicious_intervals)
    print('#msectime START_TIME, msectime END_TIME, double PROFIT')
for interval in suspicious_intervals:
for valin interval:
            print(val, end=' ')
        print()
```