




ВЫБИРАЙ

УЗНАТЬ БОЛЬШЕ

Рекламодатель - АНО "Центр мониторинга и защиты"



Горячее Лучшее Свежее Подписки

 **empenoso** 5 месяцев назад  Лига Инвесторов

Мой первый и неудачный опыт поиска торговой стратегии для Московской биржи

Когда закончил писать **механизм своего торгового робота** обнаружил, что самое главное всё таки не сам механизм, а стратегия, по которой этот механизм будет работать.

Первый тесты на истории показали что с доходностью и тем более с тем как доходность портфеля компенсирует принимаемый риск (коэффициент Шарпа) проблемы, но неудачный опыт тоже опыт, поэтому решил описать его в статье.

Первый и самый важный вопрос - при помощи чего проводить тесты торговой стратегии на исторических данных? В какой программе или при помощи какой библиотеки создавать стратегию и потом прогонять её на истории?

Раз мой торговый робот создан в среде исполнения JavaScript Node.js, то и тесты в идеале должны проводиться на чём-то схожем. Но забегая немного вперёд скажу что получилось по другому.

Windows? macOS? Linux?

Раз сам механизм робота кросс-платформенный, то хотелось чтобы и тесты можно было проводить при помощи кросс-платформенной утилиты. Однако когда рассматривал самые популярные программы, то обнаружилось что все программы из списка только для Windows. Кроме TradingView, который является веб-сервисом и Excel - который есть и для macOS.

Но похоже что веб-сервис и тем более Microsoft Excel - не лучший выбор. Тем не менее вот варианты, которые я рассматривал:

- **TradeStation**: комплексная торговая и аналитическая платформа; идеально подходит для построения графиков, автоматизации стратегий и бэктестинга для акций, опционов, фьючерсов и криптовалют.
- **NinjaTrader**: торговое программное обеспечение для фьючерсов и форекс; отлично подходит для расширенного построения графиков, бэктестинга и автоматизированной торговли.
- **MetaStock**: фокусируется на техническом анализе и бэктестинге с обширными инструментами для построения графиков и индикаторов, популярен среди

Используйте аккаунт Яндекса для входа на сервис

Безопасный вход без дополнительной регистрации на сайте

Войти с Яндекс ID

Войти

Логин


Пароль


Войти


Создать аккаунт


Забыли пароль?


или продолжите с


 Войти с Яндекс ID


 Войти через VK ID


 Промокоды


 Работа

 Курсы

 Реклама

 Игры

 Пополнение Steam



трейдеров акциями.

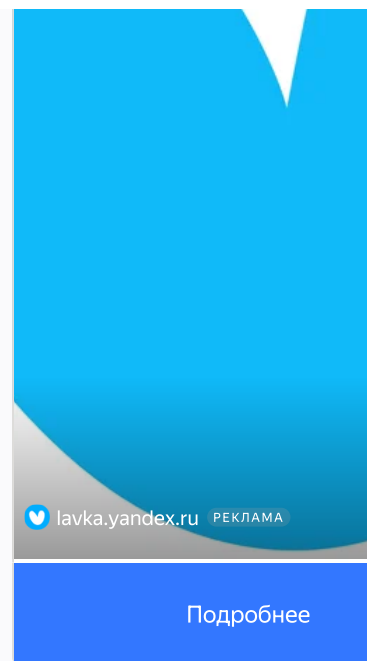
- **Wealth-Lab**: платформа, известная расширенным бэктестингом и разработкой торговых стратегий с мощной поддержкой портфелей из нескольких активов.
- **TradingView**: удобная в использовании платформа для построения графиков с социальными функциями; отлично подходит для технического анализа, обмена идеями и базового бэктестинга стратегий.
- **RealTest**: легкое программное обеспечение для бэктестинга и разработки стратегий, известное своей скоростью и простотой, ориентированное на системных трейдеров.
- **Neuroshell Trader**: специализируется на прогнозном моделировании и анализе на основе нейронных сетей; идеально подходит для трейдеров, интересующихся машинным обучением.
- **TSLab**: платформа позволяет разрабатывать, тестировать и оптимизировать торговые системы без необходимости глубокого знания программирования.
- **The Zorro Project**: бесплатная, легкая и скриптовая платформа, предназначенная для автоматизированной торговли, бэктестинга и исследований, популярная среди алгоритмических трейдеров.
- **и даже Microsoft Excel**: универсальный инструмент для работы с электронными таблицами, часто используемый для анализа портфеля, пользовательского бэктестинга и организации данных в торговле.

Ни один из этих вариантов мне не приглянулся из-за отсутствия кросс-платформенности или этот вариант был Экселем.




Node.js библиотеки - не смог ❌

После этого стал смотреть библиотеки для Node.js. Выбор оказался небольшой и более-менее живыми мне показались:

- **grademark**: <https://github.com/Grademark/grademark>
Библиотека Node.js для бэктестинга торговых стратегий на исторических данных.
- **Fugle Backtest**: <https://github.com/fugle-dev/fugle-backtest-node>
Библиотека Node.js для бэктестинга стратегий торговли акциями.
- **CCXT** - CryptoCurrency eXchange Trading Library: <https://github.com/ccxt/ccxt>
Библиотека Node.js для торговли криптовалютой, которая предоставляет унифицированный API для подключения и торговли на нескольких криптовалютных биржах, поддерживая как торговлю в реальном времени, так и доступ к историческим данным.





Топ прошлой недели

-  **ZaTaS**
4 поста
-  **SergVaders1999**
5 постов
-  **prapor35**
2 поста

[Посмотреть весь топ](#)


РЕКЛАМА



 **dikom.ru**

Столы для монтажных работ. От завода ДиКом.

[Узнать больше](#)

 **Лучшие посты недели** ✕

Рассылка Пикабу:
отправляем самые
рейтинговые материалы за 7
дней 🔥

Укажите [Подписаться](#)

```

// backtest_grademark_not_work.js
src > old_nodejs_not_work > backtest_grademark_not_work.js > @ runBacktest > @ strategy
32 function aggregateData(minuteData, interval) {
33   minuteData.forEach((entry, index) => {
34     aggregated.push(aggCandle);
35     temp = [];
36   });
37   return aggregated;
38 }
39 // Функция для запуска стратегии
40 async function runBacktest(startMonth, testMonth) {
41   let strategy = grademark({
42     buy: ({ fiveMinuteCandle, hourlyCandle }) => {
43       return {
44         fiveMinuteCandle.close > calculateSMA(fiveMinuteCandle, 5) &&
45         hourlyCandle.close > calculateSMA(hourlyCandle, 60)
46       };
47     },
48     sell: ({ price, maxPrice }) => {
49       return price < maxPrice * (1 - trailingStopPercent / 100);
50     }
51   });
52   // Сначала оптимизируем стратегию на данных за месяц
53   let januaryData = await readCsvData('data/ed123145-9665-43e0-8413-c0b1baa9013_20245[startMonth].csv');
54   let fiveMinuteCandles = aggregateData(januaryData, 5);
55   let hourlyCandles = aggregateData(januaryData, 60);
56   strategy.optimize({ fiveMinuteCandles, hourlyCandles });
57   // Далее, проводим тестирование на следующем месяце (например, феврале)
58   let februaryData = await readCsvData('data/ed123145-9665-43e0-8413-c0b1baa9013_20245[testMonth].csv');
59   let testFiveMinuteCandles = aggregateData(februaryData, 5);
60   let testHourlyCandles = aggregateData(februaryData, 60);
61 }

```

Ответ ChatGPT по Grademark

Для Grademark набросал через ChatGPT конкретный пример использования.

При этом криптовалюты мне не подходили, Grademark почему-то не смог установить, а Fugle Backtest не приглянулся.

Python библиотеки - заработало! ✓

В Python есть несколько популярных библиотек для бэктестинга торговых стратегий, рассчитанных на разные уровни сложности и типы активов. Вот найденные варианты:

- **Backtesting.py** <https://github.com/kernecl/backtesting.py>

Легкая, интуитивно понятная библиотека для векторизованного бэктестинга, включающая популярные индикаторы и метрики.

✗ 4 года не обновлялась.

- **Backtrader** <https://github.com/mementum/backtrader>

Одна из самых популярных и многофункциональных библиотек для бэктестинга. Поддерживает несколько активов, таймфреймов, индикаторов и оптимизацию стратегий.

- **PyAlgoTrade** <https://github.com/gbeced/pyalgotrade>

Простая библиотека бэктестинга со встроенной поддержкой технических индикаторов и создания базовой стратегии.

✗ Этот репозиторий был заархивирован владельцем 13 ноября 2023 г.

- **Zipline** <https://github.com/quantopian/zipline>

Разработанная Quantopian (теперь поддерживаемая сообществом), Zipline — это надежная библиотека бэктестинга, ориентированная на событийно-управляемое бэктестирование, используемая профессионалами.

✗ 4 года не обновлялась.

- **QuantConnect/Lean** <https://github.com/QuantConnect/Lean>

Движок с открытым исходным кодом, лежащий в основе QuantConnect; поддерживает бэктестинг и торговлю в реальном времени для нескольких классов активов.

- **VectorBT** <https://github.com/polakowo/vectorbt>

Разработан для быстрого векторизованного бэктестинга и анализа стратегий непосредственно на Pandas DataFrames.

Нажимая кнопку «Подписаться на рассылку», я соглашаюсь с **Правилами Пикабу** и даю согласие на обработку персональных данных.



Портал для инженеров-проектировщиков стального проката



Готовый дом с участком в Култаево



Новости Пикабу
Помощь Награды
Кодекс Пикабу Контакты
Реклама О проекте
О компании Зал славы

Промокоды Купоны Мегамаркет
Скидки Купоны AliExpress
Работа Купоны М.Видео
Курсы Купоны YandexTravel
Блоги Купоны Lamoda

Мобильное приложение



АМА

- **Fastquant** <https://github.com/enzoampil/fastquant>

Удобная библиотека бэктестинга, разработанная для быстрого тестирования с минимальной настройкой, вдохновленная Prophet от Facebook.

✗ 3 года не обновлялась.

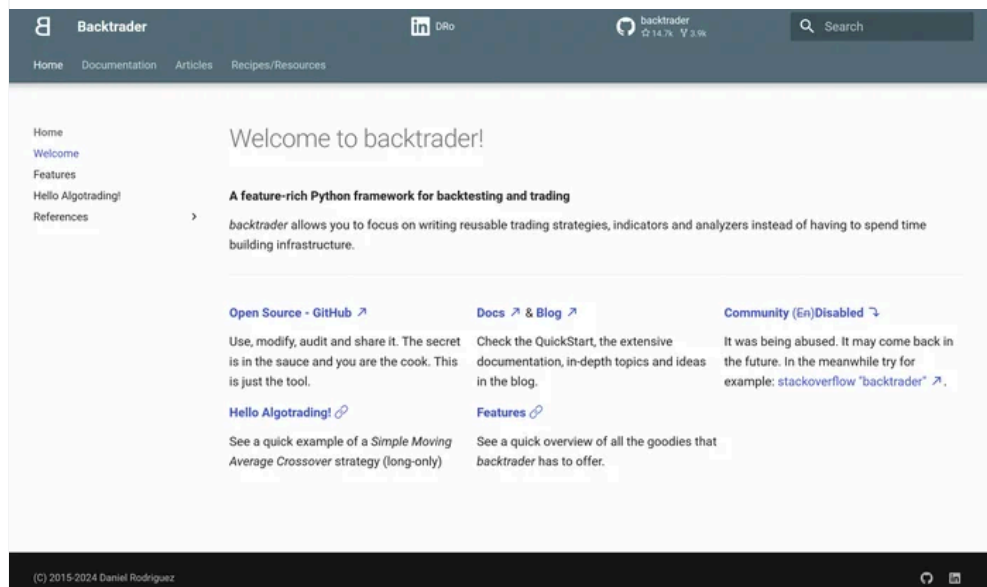
- **MibianLib** <https://github.com/yassinemaaroufi/MibianLib>

Фокусируется на ценообразовании и волатильности опционов, а не на полном бэктестинге, но полезен для стратегий, связанных с опционами.

✗ 11 лет не обновлялась.

Сначала выбрал использовать **Backtesting.py**, потому что она упоминалась на многих сайтах, но уже на первоначальном этапе использования стали вылезать проблемы. Ошибка возникла из-за несоответствия в том, как новые версии pandas обрабатывают метод `get_loc()`. Аргумент `method='nearest'` больше не поддерживается в последних версиях pandas. Эта проблема связана с тем, как библиотека **Backtesting.py** взаимодействует с новыми версиями pandas, в частности, при повторной выборке данных для построения графиков. А новой версии **Backtesting.py**, которая решает эту проблему и поддерживает последние изменения API pandas просто нет.

Следующий в списке был **Backtrader** - с ним и продолжил работать.



Backtrader от Дэниел Родригес (Daniel Rodriguez)

Идея моей торговой стратегии 💡

Хотя считается что торговая стратегия необязательно должна быть "человекочитаемой" - это вполне может быть результат обучения алгоритма, основанного на интеллектуальных технологиях (нейросети, машинное обучение и т.п.), но я решил начать с простого.

Мои условия:

1. Торговать только в лонг (длинная позиция) - покупать акции с целью их последующей продажи по более высокой цене.
2. Торговать только 15 лучших акций по объему на Московской бирже.
3. Использовать два разных таймфрейма для тестов - это временные интервалы на которых отображается движение цен на графике финансового инструмента.

2 CLOUD

Защищенное облако под 1С

Разместите 1С в безопасной среде с инфраструктурой под ваши задачи

Планирую использовать 5 минут и час. Это из-за того что **моё АПИ медленное**.

Моя торговая стратегия основана на пересечении скользящих средних двух разных таймфреймов со скользящим стоп-лоссом для продажи.

Условие покупки представляет собой комбинацию двух пересечений скользящих средних:

1. Краткосрочное подтверждение: цена закрытия на пятиминутном интервале выше пятиминутной скользящей средней.
2. Долгосрочное подтверждение: цена закрытия на часовом интервале выше часовой скользящей средней.

ПАО "Сбербанк России" (SBER:MOEX): 5 минут и час

Требуя выполнения обоих этих условий, гарантирую что акция будет иметь бычий импульс как на коротких, так и на длинных таймфреймах перед входом в позицию. Такое выравнивание двух таймфреймов помогает избегать покупок во время временного шума или незначительных колебаний на более коротком таймфрейме, отфильтровывая менее стабильные движения.

Условие продажи: трейлинг стоп, который предназначен для защиты прибыли и ограничения риска падения. Как работает лучше всего показано на картинке:

Бэктестинг моей торговой стратегии с помощью библиотеки backtrader на Python

Моя, описанная выше стратегия для двух таймфреймов на нескольких бумагах, выглядит в библиотеке backtrader на Python следующим образом:

strategy0_ma_5min_hourly.py

[Код на GitHub](#)

Сделал переключатель одиночный тест или оптимизация: singleTest / optimization для основного файла запуска: SingleTestOrOptimization = "optimization"

Основной файл запуска main.py

[Код на GitHub](#)

В данные загрузил котировки за октябрь 2024:

1. AFLT_1hour.csv
2. AFLT_5min.csv
3. EUTR_1hour.csv
4. EUTR_5min.csv
5. GAZP_1hour.csv
6. GAZP_5min.csv
7. MTLR_1hour.csv
8. MTLR_5min.csv
9. RNFT_1hour.csv
10. RNFT_5min.csv

11. ROSN_1hour.csv
12. ROSN_5min.csv
13. RUAL_1hour.csv
14. RUAL_5min.csv
15. SBER_1hour.csv
16. SBER_5min.csv
17. SGZH_1hour.csv
18. SGZH_5min.csv
19. SNGSP_1hour.csv
20. SNGSP_5min.csv
21. UWGN_1hour.csv
22. UWGN_5min.csv
23. VKCO_1hour.csv
24. VKCO_5min.csv
25. VTBR_1hour.csv
26. VTBR_5min.csv

Время выполнения оптимизации для таких параметров составило 74 минуты:

```
# Оптимизация стратегии start_date = 2024-
10_MovingAveragesOnDifferentTimeIntervalsStrategy
cerebro.optstrategy(MovingAveragesOnDifferentTimeIntervalsStra
tegy,
ma_period_5min=range(10, 61, 5), # Диапазон для 5-минутной
скользящей средней
ma_period_hourly=range(15, 61, 2), # Диапазон для часовой
скользящей средней
trailing_stop=[0.03]) # Разные проценты для трейлинг-стопа
0.03, 0.05, 0.07
```

Для того чтобы визуально представить результаты оптимизации написал модуль, который строит трехмерный график.

Модуль `3dchart.py`

[Код на GitHub](#)

Результат оптимизации в виде графика:

Выводы из этой оптимизации

Цифры по шкале Z показывают лишь степень убытков в рублях. Они со знаком минус.

Вы можете сами полностью повторить мой опыт потому что код загружен на GitHub:

https://github.com/empenoso/SilverFir-TradingBot_backtesting

Тем не менее:

1. Некоторые стратегии эффективны только в определенных рыночных условиях. Например, стратегии следования за трендом, как правило, хорошо работают на трендовых рынках, но не работают на боковых рынках.
2. Курвефитинг, подгонка под историю. Не хочу вводить много параметров, чтобы этого избежать. Переобучение прошлыми данными: если стратегия хорошо работает на исторических данных, но плохо на будущих данных в режиме скользящего окна, она может быть слишком адаптирована к историческим моделям, которые не будут повторяться.
3. Транзакционные затраты: хорошо, если тестирование учитывает реалистичное проскальзывание, комиссии и спреда.

Будущие шаги - где искать прибыльные торговые стратегии 📝

Я хочу использовать подход скользящего окна - когда данные разбиваются на более мелкие последовательные периоды например по месяцам, за которым следует период тестирования вне этой выборки. Например, оптимизация идёт на месячных данных, а тестировать уже на следующем месяце. То есть происходит сдвиг вперед: после каждого периода тестирования окно «скользит» вперед на указанный интервал, и процесс повторяется. Таким образом, каждый сегмент данных используется как для обучения, так и для тестирования с течением времени, но никогда одновременно. Это помогает проверить, что стратегия работает стабильно в меняющихся рыночных условиях.

Также планирую использовать Technical Analysis of STOCKS & COMMODITIES для поиска новых идей. Их советы трейдерам [доступны в открытом доступе](#).

А ещё планирую использовать ChatGPT, отправляя запросы вроде:

Действуй как опытный издатель. Отобрази 10 ведущих авторов в области алгоритмической торговли на рынке Америки. Для каждого автора перечисли три самые популярные книги, включая сведения о книге (дату публикации, издателя и ISBN), и предоставь русские переводы для каждого названия книги.

Ответ ChatGPT

и дальше после ответа:

Действуй как опытный пользователь библиотеки backtrader на Python. Хочу использовать торговую стратегию из книги Yves Hilpisch "Python for Finance: Mastering Data-Driven Finance" для тестов. Добавляй все комментарии на русском языке, продолжай со мной общение на английском.

И дальше подобные промты.

Итоги

Несмотря на то, что первоначальный выбор стратегии на двух разных таймфреймах и сразу для 15 активов был не самый удачный - впереди ещё очень большое поле исследований и тестов.

Автор: [Михаил Шардин](#)

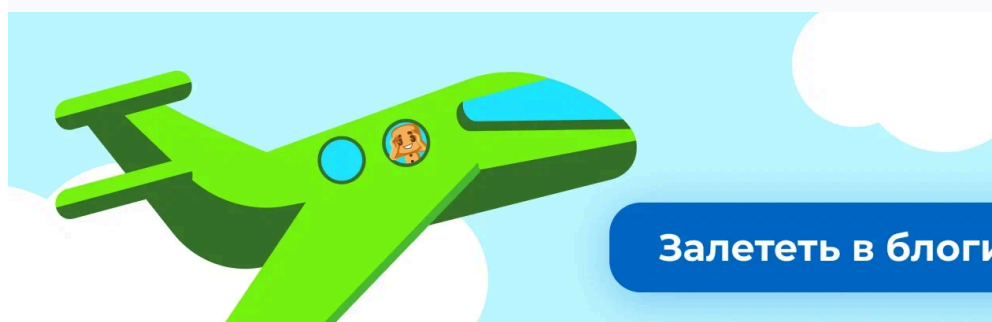
18 ноября 2024 г.



1 3.7K

₽ Поддержать

Эмоции



Лига Инвесторов

9.7K постов • 7.7K подписчиков

Добавить пост

Подписаться



Правила сообщества

1. Необходимо соблюдать правила Пикабу
2. Запрещены посты, не относящиеся к тематике сообщества
3. Запрещается откровенная реклама

4. Нельзя оскорблять участников сообщества.

Все комментарии [Автора](#)

Раскрыть 2 комментария

Чтобы оставить комментарий, необходимо [зарегистрироваться](#) или [войти](#)

● — ■ —

—
—
—
—
—
—
—

● — ■ —

—
—
—
—
—
—
—

● — ■ —

—
—
—
—
—
—
—

