empenoso Программирование на python IT ...

Машинное обучение на Московской бирже - что из этого не вышло

12 дней назад 3.5K

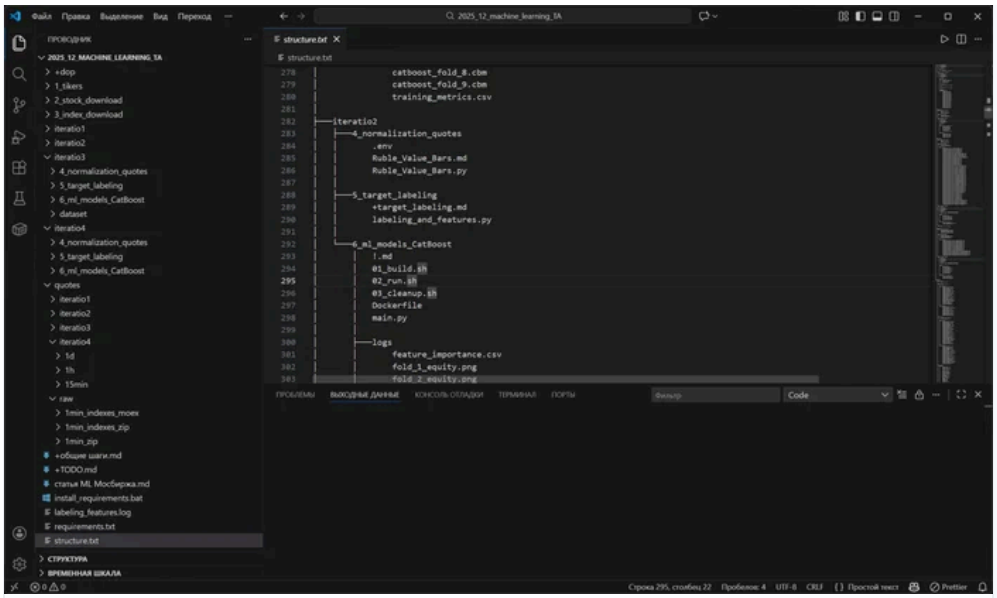
Время после нового года решил провести с пользой и окунуться в машинное обучение. Заняться Machine Learning — и посмотреть получится что-то или нет с российским рынком акций на Московской бирже.

Моей целью было построить такую систему, которая будет учиться на истории и в перспективе торговать лучше чем случайное блуждание 50/50. Но из-за комиссий и спреда подобные блуждания изначально отрицательны — чтобы выйти в плюс надо как минимум покрывать комиссии.

Если говорить о результатах очень кратко, то технически всё работает, но вот финансовый результат на грани безубыточности.

Если Вы только интересуетесь этой темой Вы можете посмотреть какие-то шаги в моей статье, а если Вы уже опытный разработчик подобных систем, то можете подсказать что-нибудь в комментариях.

Причём вся эта работа выглядит совершенно не так как показывается в фильмах про уолл-стрит: фактически это написание скриптов и монотонный запуск и всё происходит полностью локально на компьютере.



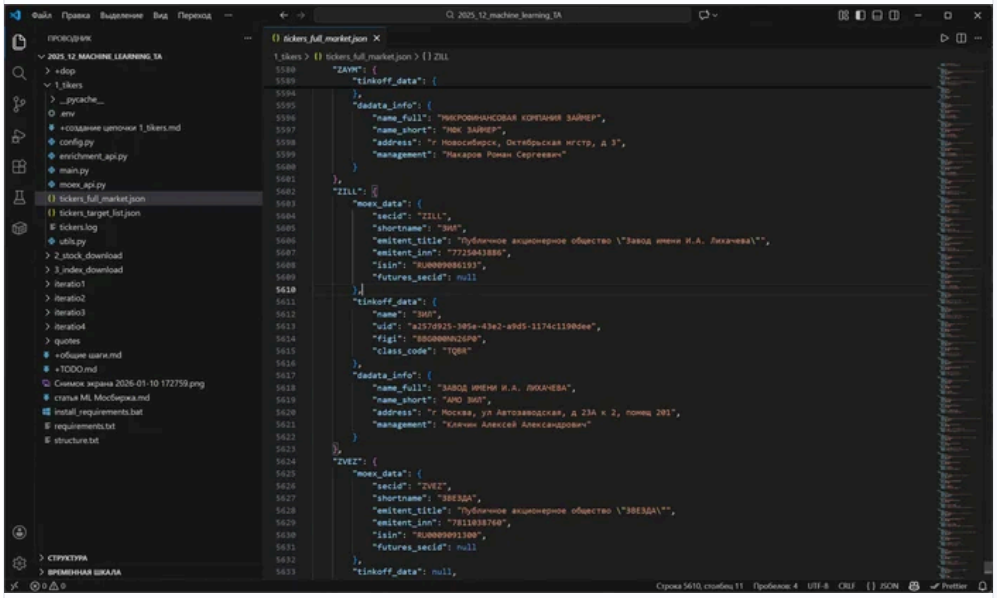
Скриншот VS Code открытым деревом проекта

Охота за данными

Вообще данные очень важны. Иначе получается что мусор на входе просто перетекает в мусор на выходе.

Для отбора бумаг я руководствовался следующей логикой: сначала ищу общий список всех акций, торгуемых на Московской бирже и делаю выборку — в неё входят только те, у которых есть фьючерсы. Дальше оставляю только тикеры акций, которые имеют фьючерсы.

Потом беру два API — одно от брокера, а второе API, предоставляющее информацию по всем юридическим лицам России — это API DaData. У каждой акции ведь есть ИНН компании. Используя оба этих API — оба бесплатных кстати — обогащаю каждую запись дополнительными сведениями.



Фрагмент файла tickers_full_market.json с общим список тикеров

После этого скачиваю котировки акций с Московской биржи. И загружаю три индекса: IMOEX, IMOEX2, RTSI, RGBI.

Для этого скачиваю минутки — они готовые и сразу в архивах через API брокера — можно очень быстро скачать полностью все бумаги.

С индексами сложнее. Брокерский API не отдавал историю по IMOEX и RGBI, видимо, из-за ограничений лицензии биржи (привет, MOEX AlgoPack). Пришлось писать парсер для прямого API Московской биржи (ISS MOEX) — скорость гораздо медленнее, но я скачиваю тоже минутки. Скачать все доступные минутки с IMOEX, IMOEX2, RTSI, RGBI занимает около 20 минут.

Котировки индекса

Работа со временем

Я начал работу с того, что выделил три интересующих меня таймфрейма. 15 минут, 1 час и 1 день, основной 1 час.

Написал скрипт который преобразует архивы с минутками от API брокера в файлы с котировками нужных таймфреймов.

Итерация 1,3,4: временнЫе бары (обычные)

Сначала я просто агрегировал минутки в 15 минут, 1 час и 1 день через скрипт, но рынок неравномерен. Возможно для ML-модели временные свечи могут быть очень шумные, так как содержат разное количество информации.

Скрипт normalization_quotes_stock.py который читает 1-минутные архивы (ZIP) из quotes/1min_zip и агрегирует данные в 15min, 1h, 1d

Итерация 2: свечи рублевого объема

Потом я временно перешёл к событийным барам. Новая свеча формируется не по времени, а когда проходит фиксированный объем в рублях, например, 50 млн руб., но для разных акций этот порог разный, потому что рассчитывается динамически. Разделил их по условным классам:

- A: порог выше.
- B: порог ниже.
- C: отбрасываются.

Скрипт который создаёт из минутных котировок свечи рублевого объема и классифицирует бумаги по классам

Учитель для робота. Разметка

Как объяснить машине, что такое «хорошая сделка»? Рынок может продолжать рост, но перед этим выбить по стопу.

Книга Маркос Лопез де Прадо «Машинное обучение: алгоритмы для бизнеса»

Для установки цели я использовал тройной барьер по де Прадо:

- Верхний барьер (Take Profit): Цена + N×ATR
- Нижний барьер (Stop Loss): Цена - M×ATR
- Вертикальный барьер (Time Limit): Если прошло 100 баров, а цена никуда не пришла — выходим.

Скрипт расставляет метки:

Метка 1: сработал Take Profit.

Метка 0: вышло время, выход в ноль (минус комиссия).

Метка -1: сработал Stop Loss, потеря денег.

Фрагмент в Visual Studio Code

Глаза модели: инженерия признаков

Я использовал CatBoost (Categorical Boosting) это библиотека машинного обучения с открытым исходным кодом от «Яндекса», основанная на градиентном бустинге над деревьями решений. Я не подаю сырые цены (OHLCV: Open (цена открытия), High (максимальная цена), Low (минимальная цена), Close (цена закрытия) и Volume (объем торгов)), так как они не стационарны, потому что цена 100 ₽ в 2010 и 100 ₽ в 2024 — это разные сущности, а использую только относительные величины.

Я старался подавать именно Log Returns (логарифмические доходности), потому что Log returns аддитивны и симметричны: падение на 50% и рост на 100% имеют одинаковый масштаб в логарифмах.

Сами признаки, в разных итерациях по разному было:

- Качество импульса: не просто «цена выросла», а как она выросла. Использую автокорреляцию и эффективность тренда.
- Микроструктура: что происходило внутри этой свечи рублевого объема? Какая концентрация объема?
- Межрыночные связи: как актив ведет себя относительно индекса Мосбиржи и индекса гособлигаций RGBI.
- Классика: RSI (нормализованный через Z-score), расстояния до скользящих средних.

Файл для генерации признаков и разметки

Моделирование

Для того чтобы заниматься ML все инструменты у меня были: компьютер 32 Гб оперативки с видеокартой GPU 16 Гб, Python как основной язык, Docker чтобы не зависеть от капризов драйверов, Numba для ускорения расчётов, Linux для администрирования Docker контейнеров.

Для моей задачи как будто даже избыточная конфигурация — потому что все расчёты протекают очень быстро.

Dockerfile

Делаю это из под Ubuntu, хотя всю разработку веду из-под Windows.

01_build.sh

Ещё использую измененную версию пошагового тестирования (Walk-Forward Optimization), которая используется в трейдинге для поиска и проверки торговых стратегий, но добавляя «очистку» (purging) данных: она избегает перекрытия обучающих и тестовых периодов, чтобы предотвратить подгонку стратегии под шум истории, делая результаты более реалистичными и устойчивыми к будущим изменениям рынка.

Результаты и боль

При просмотре результатов магия машинного обучения быстро испаряется.

Я учитываю комиссии:

```
COMMISSION_PCT = 0.04 / 100  # комиссия брокера
SLIPPAGE_PCT  = 0.02 / 100  # проскальзывание
```

Каждая сделка автоматически теряет 0,06% или 0,12% на круг для акций. Кажется мелочью, но при сотнях сделок именно эти десятые доли процента превращают модель в убыточную. Модели нужно предсказывать движения >0,3-0,5%, чтобы быть в плюсе.

По качеству прогнозов мой ML стабильно показывает AUC 0,54–0,55. Формально это лучше случайного угадывания (0.50), но до Грааля тут очень далеко. Почему? Даже небольшая ошибка в вероятностях, умноженная на комиссии и шум, быстро съедает весь перевес.

Чтобы понимать, что именно я меряю, важно разобраться в трёх ключевых метриках.

AUC (Area Under the Curve) — это мера того, насколько хорошо модель умеет отличать «хорошие» сделки от «плохих». Если AUC = 0.5, модель — это монетка. Если 0.55 —

она угадывает чуть чаще, чем случайность. В вакууме это звучит неплохо, но в трейдинге такого преимущества часто недостаточно, чтобы перекрыть издержки.

Fold — это один из прогонов в Walk-Forward Optimization. История рынка режется на последовательные отрезки: на одном модель учится, на следующем тестируется. Каждый такой отрезок — отдельный fold. Это имитация реальности: мы всегда торгуем на будущем, которого модель «не видела». Поэтому один fold может быть прибыльным, а следующий — убыточным, просто потому что режим рынка поменялся.

Тесты

Precision (точность) — это ответ на вопрос: «если модель сказала „покупай“, как часто она оказывается права?». Это критично для торговли, потому что даже модель с неплохим AUC может генерировать кучу ложных сигналов, которые будут съедать депозит комиссиями и стоп-лоссами.

И вот здесь появляется самая болезненная часть. На одном из фолдов я получаю красивую Equity Curve, где капитал растёт.

График fold_2_best_equity

На другом — та же самая модель превращает счёт в “пилу”: заработали на тренде, потом долго и мучительно всё отдали на боковике.

График fold_3_best_equity

Когда я писал этот текст мне в голову пришло, а что если модель обучать на максимизацию финансового показателя, такого как Коэффициент Шарпа?

Заключение: вопросы к залу

Я проделал некоторую работу, но результаты пока выглядят не особо приятными.

Мои гипотезы:

1. Ошибка в методологии?
2. Мало данных?
3. Предсказывать не направление, а волатильность?
4. Перейти на более высокие таймфреймы (4H, 1D), где комиссия съедает меньшую долю движения?
5. Нужно использовать данные из стакана (Order Book)? С получением истории стакана для частного лица большие проблемы. Бесплатно доступен лишь очень ограниченный набор инструментов.

6. CatBoost слишком прост, нужны трансформеры?

Я занимаюсь Machine Learning (ML), когда система учится на таблицах. Но есть ведь ещё Deep learning (глубокое обучение) когда идёт анализ больших объёмов данных и выявления сложных закономерностей автономно. Но боюсь для моей задачи слишком мало данных. Наверное только на истории стаканов OrderBook будет работать.

Приглашаю в комментарии: кто реально запускал Machine Learning на Мосбирже в плюс? Или просто опытных людей. Где я свернул не туда?

Не сдерживайте себя — напишите комментарий — буду рад любой критике в комментариях.

Автор: Михаил Шардин
Моя онлайн-визитка
Telegram «Умный Дом Инвестора»

13 января 2026 г.

Раскрыть 12 комментариев

Чтобы оставить комментарий, необходимо зарегистрироваться или войти