# AlgoPack: Алгоритмический трейдинг | Часть 4: Бот

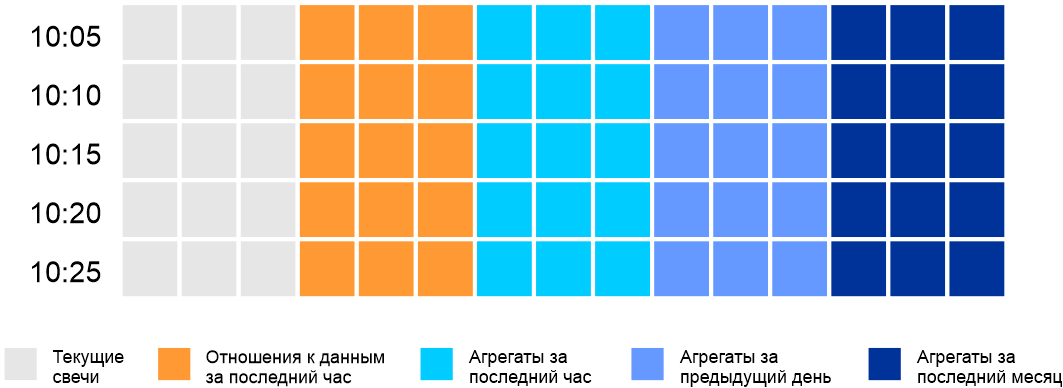
Это заключительная часть разработки автоматического торгового бота на данных AlgoPack. Здесь соберем части нашего бота в единое цело и поставим его на автоматическое выполнение.

Список статей:

* AlgoPack: Алгоритмический трейдинг | Часть 1: Введение
* AlgoPack: Алгоритмический трейдинг | Часть 2: EDA
* AlgoPack: Алгоритмический трейдинг | Часть 3: Модель
* AlgoPack: Алгоритмический трейдинг | Часть 4: Бот

## Датасет

Для выполнения предсказания нам нужно подготовить датасет. Т.к. наш бот будет выполнять предсказание на самую последнюю пятиминутку нам нужно получить одну единственную строку на эту пятиминутку. По структуре она должна точно повторять тренировочный датасет и состоять из следующих элементов:



В прошлой части мы сформировали (при ночном запуске) агрегаты за прошлый день и за прошлый месяц. А значит нам нужно получить сами пятиминутки и сформировать фичи за последний час.

Загружаем свежие данные:

today = '2023-12-08'

loader = Ticker('YNDX')

df = loader.tradestats(date=today, till\_date=today)

df['pr\_mean'] = df[['pr\_high','pr\_low']].mean(axis=1)

Формируем из них отношения и агрегаты за последний час:

for i in range(1,11+1):

df[f'ratio\_pr\_mean\_{i}'] = df['pr\_mean'] / df['pr\_mean'].shift(i)

for a in ['min','max','mean','std']:

df[f'hour\_{a}'] = df.groupby('tradedate')['pr\_mean'].transform(lambda s: s.shift(1).rolling(11).agg(a))

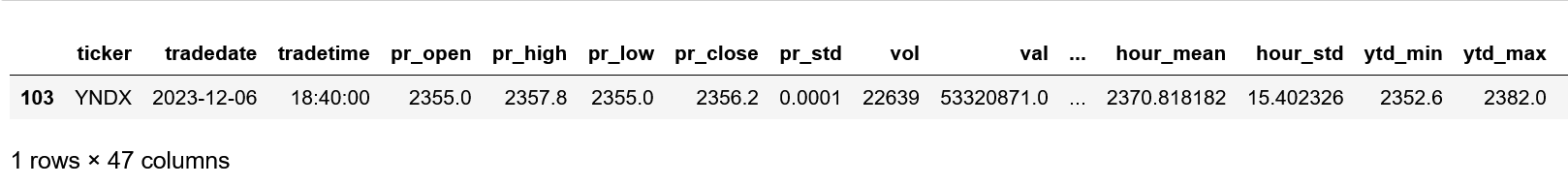
Подгружаем агрегаты за прошлый день и прошлый месяц:

ytd\_month = pd.read\_pickle(f'../data/interim/{datetime.today().strftime("%Y-%m-%d").replace("-","")}\_part.pkl')

Объединяем датасеты и оставляем только последнюю строчку:

df = df.merge(ytd\_month, how='cross')

last\_row = df[-1:]



## Предсказание

Сначала загрузим модель. Т.к. у нас хранятся все модели, то берем последнюю:

import glob

all\_models = glob.glob('../models/\*.lgb')

all\_models.sort()

model = lgb.Booster(model\_file=all\_models[-1])

З.Ы. У нас даже есть некоторый запас прочности. Если ночной процесс обучения упадет, то у нас всегда будет модель для предсказания.

Выполним предсказание:

pred = model.predict(last\_row[model.feature\_name()]).argmax()

Теперь надо сделать с ним что-то полезное :)

## Брокер

Как уже говорилось во введении, напрямую с биржей физическим лицам торговать нельзя. Чтобы совершать сделки вам необходимо подключить своего бота к API лицензированного брокера. Их много. Из известных: [Тиньков](https://www.tinkoff.ru/invest/open-api/), [Финам](https://www.finam.ru/landings/trade-api/), и пр.

Т.к. бесплатных брокеров не бывает, то код подключения к API брокера остается на ваше усмотрение. А сейчас мы просто сделаем под него заглушку:

# Отправляем запрос брокеру

def send\_to\_broker(pred):

if pred == 0:

print(TICKER, df['tradetime'].max(), '- Ничего не делаем')

elif pred == 1:

print(TICKER, df['tradetime'].max(), '- Пик (продаем)')

elif pred == 2:

print(TICKER, df['tradetime'].max(), '- Впадина (покупаем)')

send\_to\_broker(pred)

Код предсказания: <https://github.com/slivka83/algopack_simple_bot/blob/main/src/models/predict_model.py>

## Pipeline

На текущий момент мы получили набор разрозненных py-файлов. Теперь нужно заставить их органично работать вместе: автоматически и в определенной последовательности.

### Оркестратор

Для того чтобы настроить последовательность выполнения задач нам понадобится оркестратор. Для учебных целей воспользуемся самым простым - [pypyr](https://pypyr.io/).

Установите его:

pip install pypyr

... и создайте два файла со следующим содержимым:

**train.yaml**

steps:

- name: pypyr.steps.cmd

in:

cmd: python src\data\get\_one\_day.py

- name: pypyr.steps.cmd

in:

cmd: python src\features\create\_features.py

- name: pypyr.steps.cmd

in:

cmd: python src\models\train\_model.py

Здесь мы последовательно вызываем три py-файла: загружаем файлы, формируем фичи и тренируем модель.

Чтобы запустить его необходимо выполнить такую команду:

pypyr src/pipeline/predict

**predict.yaml**

steps:

- name: pypyr.steps.cmd

in:

cmd: python src\models\predict\_model.py

Здесь мы запускаем один единственный py-файл, который выполняет всю работы по предсказанию.

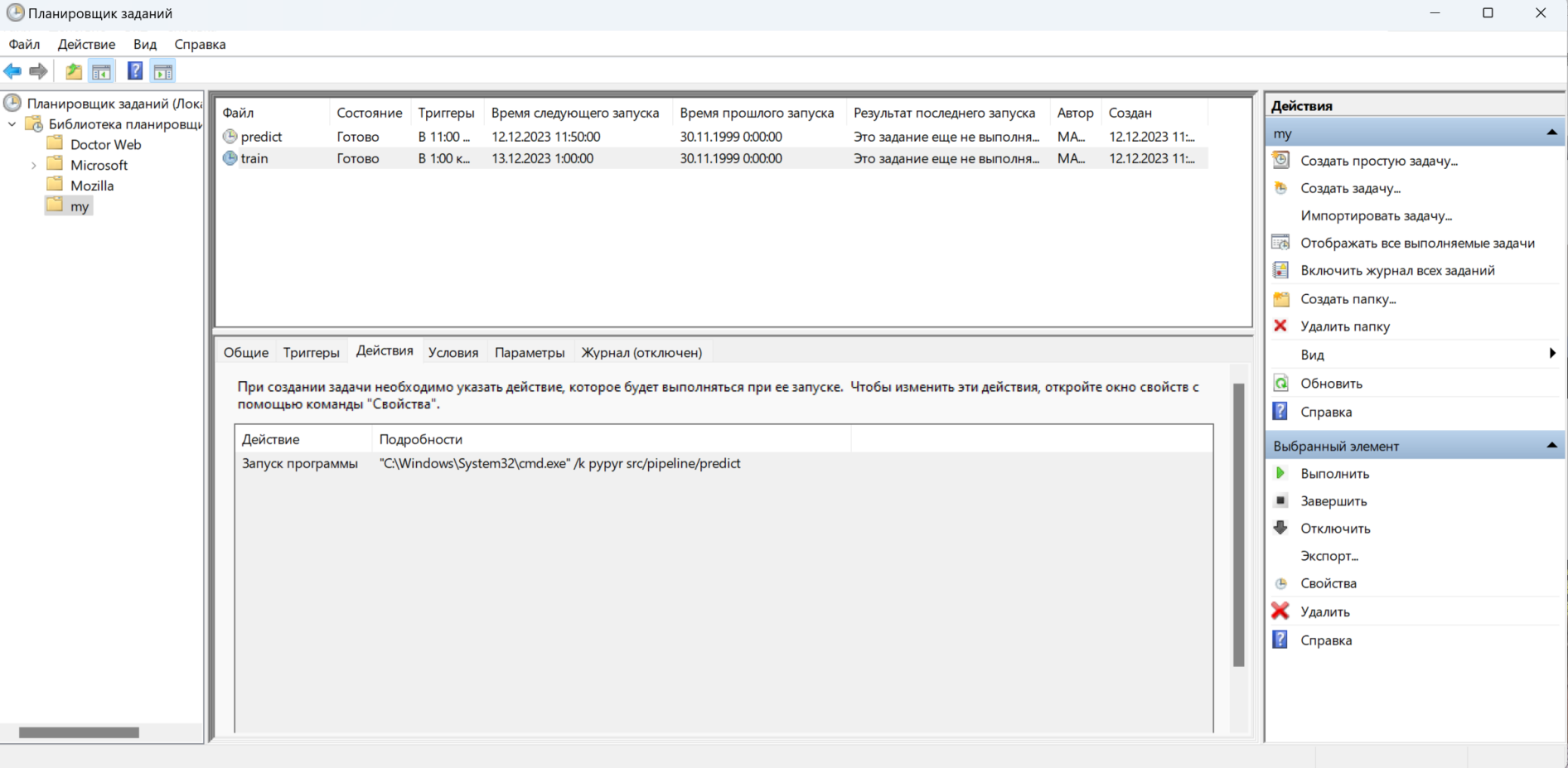
Запустит его можно так:

pypyr src/pipeline/predict

### Планировщик

Теперь надо автоматизировать нашего робота.

Для Windows можно воспользоваться стандартным Планировщиком.



Чтобы выполнить консольную команду нужно задать такую строку: "C:\Windows\System32\cmd.exe" /k pypyr src/pipeline/predict

Настройте запуск:

* train.yaml на 1 час ночи каждый день.
* predict.yaml каждые 5 минут с 11 до 19.

А если у вас Linux можете воспользоваться программной Cron.

<https://1cloud.ru/help/linux/kak-nastroit-planirovshchik-cron-na-crontab-linux>

**To-Do**

1. Использовать более продвинутые оркестраторы: Airflow, Dagster.
2. Перенести код в облако. Например, Яндекс.Облако.
3. Хранение предсказаний в БД.

## Заключения

На этом разработка нашего учебного бота закончена. Можете модифицировать его как вам угодно. Очевидные улучшения я выписал под каждым модулем. Но вы можете добавить и свои.