

Разработка интерактивного дашборда для анализа акций при помощи фреймворка Pathway

Анишкова Анастасия Сергеевна

Приамурский государственный университет имени Шолом-Алейхема

Студент

Аннотация

Цель данного исследования — разработка интерактивного дашборда для анализа торговых акций компании Apple с использованием фреймворка Pathway. Дашборд был реализован на облачной платформе Google Colab. Полученный результат позволяет пользователю исследовать данные, изменяя параметры анализа в реальном времени. Это дает возможность гибко настраивать параметры, такие как размер окна для скользящего среднего или порог объема торгов, и мгновенно видеть, как эти изменения влияют на график и результаты анализа.

Ключевые слова: анализ акций, дашборд, Pathway, Google Colab.

Development of an interactive dashboard for stock analysis using the Pathway framework

Anishkova Anastasia Sergeevna

Sholom Aleichem Priamurskiy State University

Student

Abstract

The purpose of this research is to develop an interactive dashboard for analyzing Apple's trading stocks using the Pathway framework. The dashboard was implemented on the Google Colab cloud platform. The result allows the user to explore the data by changing the analysis parameters in real time. This makes it possible to flexibly adjust parameters such as the window size for the moving average or the trading volume threshold, and instantly see how these changes affect the chart and the analysis results.

Key words: stock analysis, Dashboard, Pathway, Google Colab.

1 Введение

1.1 Актуальность

Создание дашборда для анализа торговых акций с использованием фреймворка Pathway актуально, потому что финансовые рынки меняются очень быстро, и важно оперативно анализировать данные. Pathway позволяет работать с информацией в реальном времени, что особенно важно для тех, кто торгует акциями. Дашборд дает возможность гибко настраивать параметры, такие как скользящее среднее или объем торгов, и сразу видеть результаты.

Благодаря интеграции с Google Colab, инструмент легко доступен и не требует установки сложных программ. В итоге, такой дашборд сочетает в себе скорость, удобство и интерактивность, что делает его полезным для профессионалов на финансовых рынках.

1.2 Обзор исследований

А.И. Малик продемонстрировал построение дашборда в python с помощью библиотеки Dash [2], актуальные инструменты визуализации данных описали А.А. Юрова, Т.М. Алясева, К.Р. Зименкова [3], П.А. Пылов, А.В. Протодяконов сделали обзор на программные библиотеки языка программирования python для реализации алгоритмов визуализации данных [4], как использовать python для визуализации данных продемонстрировал С.В. Пальмов [5], Э.А. Эберле, Г.А. Гадельшина описали принципы проектирования дашбордов: исследование на примере python и flask/dash [6], инструментальную панель для задач предварительной обработки интеллектуальных данных для мобильных устройств создал Ю. Дж. Харанвала [7], Е. В. Бенко автоматизировал аналитический процесс в системе образования Челябинской области [8].

1.3 Цель исследования

Цель исследования — разработать интерактивный дашборд для анализа торговых акций компании Apple с использованием фреймворка Pathway.

2 Материалы и методы

В данном исследовании для визуализации данных используется облачная платформа Google Colab в сочетании с фреймворком Pathway. Google Colab предоставляет удобную среду для разработки и выполнения кода без необходимости установки дополнительного программного обеспечения, что делает процесс создания дашборда более доступным и масштабируемым. Pathway, в свою очередь, обеспечивает обработку потоковых данных в реальном времени, что позволяет визуализировать актуальную информацию о торговых акциях.

Для проведения анализа используется датасет ticker [1], содержащий информацию о торговых данных акций компании Apple (AAPL) за определенный временной период. Каждая строка файла представляет собой запись, отражающую торговую активность за конкретный момент времени, что позволяет детально исследовать динамику рыночных изменений (рис.1).

Характеристика столбцов датасета:

- **ticker:** Символ тикера акции (всегда AAPL);
- **open:** Цена открытия акции в заданном временном интервале;
- **high:** Максимальная цена акции в заданном интервале;
- **low:** Минимальная цена акции в заданном интервале;
- **close:** Цена закрытия акции в заданном интервале;
- **volume:** Объем торгов, выраженный в количестве акций, проданных за указанный период;

- **vwap**: Средневзвешенная цена акции (Volume Weighted Average Price) за интервал;
- **t**: Временная метка (timestamp) в формате Unix-времени (миллисекунды с 1 января 1970 года);
- **transactions**: Количество сделок, совершенных за указанный интервал;
- **otc**: Поле, указывающее на возможность совершения сделки на внебиржевом рынке (ОТС). Отсутствие данных в данном поле позволяет предположить, что информация относится к биржевым сделкам.

ticker	open	high	low	close	volume	vwap	t	transactions	otc
AAPL	189.6	189.6	189.2	189.3	473.0	189.4087	1701075600000	23	
AAPL	189.2	189.2	189.2	189.2	418.0	189.1945	1701075610000	7	
AAPL	189.25	189.25	189.25	189.25	301.0	189.25	1701075764000	2	
AAPL	189.25	189.25	189.25	189.25	105.0	189.25	1701075784000	2	
AAPL	189.27	189.27	189.27	189.27	250.0	189.2758	1701076340000	3	
AAPL	189.28	189.28	189.28	189.28	300.0	189.28	1701076353000	4	
AAPL	189.28	189.28	189.28	189.28	345.0	189.28	1701076391000	2	
AAPL	189.29	189.29	189.29	189.29	764.0	189.2958	1701076396000	9	
AAPL	189.39	189.39	189.39	189.39	300.0	189.3917	1701076447000	2	
AAPL	189.41	189.41	189.4	189.4	300.0	189.405	1701076497000	2	
AAPL	189.4	189.4	189.4	189.4	259.0	189.3968	1701076515000	5	
AAPL	189.3	189.3	189.3	189.3	568.0	189.3004	1701076683000	5	
AAPL	189.28	189.28	189.28	189.28	1797.0	189.2788	1701076686000	21	

Рисунок 1. Датасет

3 Результаты

Перейдем к анализу данных с использованием фреймворка Pathway в среде Google Colab. Для начала работы необходимо установить библиотеку Pathway, которая предоставляет инструменты для эффективной обработки и анализа потоковых данных. Установка выполняется с помощью команды, которая будет представлена в коде (рис.2).

Ссылка на ноутбук
<https://colab.research.google.com/drive/1MTI38sngAevSIQ6Z1dlxp3RXustoi0EL?usp=sharing>

```

✓ 22 08:00 | pip install pathway pandas bokeh panel
28%

Collecting pathway
  Downloading pathway-0.16.3-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_x86_64.whl.metadata (40 kB)
    Requirement already satisfied: pandas in /usr/local/lib/python3.10/dist-packages (2.2.2)
    Requirement already satisfied: bokeh in /usr/local/lib/python3.10/dist-packages (3.6.2)
    Requirement already satisfied: panel in /usr/local/lib/python3.10/dist-packages (1.5.5)
    Requirement already satisfied: aiohttp>=3.8.4 in /usr/local/lib/python3.10/dist-packages (from pathway) (3.11.11)
    Requirement already satisfied: click>=8.1 in /usr/local/lib/python3.10/dist-packages (from pathway) (8.1.8)
    Requirement already satisfied: geopy>=2.4.0 in /usr/local/lib/python3.10/dist-packages (from pathway) (2.4.1)
    Collecting h3>=4 (from pathway)
      Downloading h3-4.1.2-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_x86_64.whl.metadata (18 kB)
    Requirement already satisfied: numpy>=1.21 in /usr/local/lib/python3.10/dist-packages (from pathway) (1.26.4)
    Requirement already satisfied: scikit-learn>=1.0 in /usr/local/lib/python3.10/dist-packages (from pathway) (1.6.0)
    Requirement already satisfied: shapely>=2.0.1 in /usr/local/lib/python3.10/dist-packages (from pathway) (2.0.6)
    Collecting sqlalchemy>=10.6.1 (from pathway)
      Downloading sqlalchemy-10.6.1-py3-none-any.whl.metadata (14 kB)
    Requirement already satisfied: pyarrow>=10.0.0 in /usr/local/lib/python3.10/dist-packages (from pathway) (17.0.0)
    Requirement already satisfied: requests>=2.31.0 in /usr/local/lib/python3.10/dist-packages (from pathway) (2.32.3)
    Collecting python-sat>=0.1.8.dev0 (from pathway)
      Downloading python_sat-1.8.dev14-cp310-cp310-manylinux_2_12_x86_64.manylinux2010_x86_64.manylinux_2_28_x86_64.whl.metadata (1.5 kB)
    Collecting beartype<0.16.0, >=0.14.0 (from pathway)
      Downloading beartype-0.15.0-py3-none-any.whl.metadata (28 kB)
    Requirement already satisfied: rich>=12.6.0 in /usr/local/lib/python3.10/dist-packages (from pathway) (13.9.4)
    Collecting diskcache>=5.2.1 (from pathway)
      Downloading diskcache-5.6.3-py3-none-any.whl.metadata (20 kB)
    Requirement already satisfied: exceptiongroup>=1.1.3 in /usr/local/lib/python3.10/dist-packages (from pathway) (1.2.2)
    Collecting boto3>=1.26.76 (from pathway)
      Downloading boto3-1.35.95-py3-none-any.whl.metadata (6.7 kB)
    Requirement already satisfied: google-api-python-client>=2.108.0 in /usr/local/lib/python3.10/dist-packages (from pathway) (2.155.0)
    Requirement already satisfied: typing-extensions>=4.8.0 in /usr/local/lib/python3.10/dist-packages (from pathway) (4.12.2)
    Collecting jupyter-bokeh>=3.0.7 (from pathway)
      Downloading jupyter_bokeh-4.0.5-py3-none-any.whl.metadata (7.1 kB)
    Collecting jmespath>=1.0.1 (from pathway)
      Downloading jmespath-1.0.1-py3-none-any.whl.metadata (7.6 kB)
    Collecting aiohttp-cors>=0.7.0 (from pathway)
      Downloading aiohttp_cors-0.7.0-py3-none-any.whl.metadata (20 kB)
    Requirement already satisfied: opentelemetry-api>=1.22.0 in /usr/local/lib/python3.10/dist-packages (from pathway) (1.29.0)
    Requirement already satisfied: opentelemetry-sdk>=1.22.0 in /usr/local/lib/python3.10/dist-packages (from pathway) (1.29.0)
    Collecting opentelemetry-exporter-otlp-proto-grpc>=1.22.0 (from pathway)
      Downloading opentelemetry_exporter_otlp_proto_grpc-1.29.0-py3-none-any.whl.metadata (2.2 kB)
    Collecting fs>=2.4.16 (from pathway)
      Downloading fs-2.4.16-py2.py3-none-any.whl.metadata (6.3 kB)
    Collecting async-lru>=2.0.4 (from pathway)
      Downloading async_lru-2.0.4-py3-none-any.whl.metadata (4.5 kB)
    Requirement already satisfied: networkx>=3.2.1 in /usr/local/lib/python3.10/dist-packages (from pathway) (3.4.2)
    Requirement already satisfied: google-cloud-pubsub>=2.21.1 in /usr/local/lib/python3.10/dist-packages (from pathway) (2.27.1)
    Requirement already satisfied: google-cloud-bigquery in /usr/local/lib/python3.10/dist-packages (from pathway) (3.25.0)
    Collecting pydantic>=2.9.0 (from pathway)
      Downloading pydantic-2.9.2-py3-none-any.whl.metadata (149 kB)
    Requirement already satisfied: gitpython>=3.1.43 in /usr/local/lib/python3.10/dist-packages (from pathway) (3.1.44)
    Requirement already satisfied: python-dateutil>=2.8.2 in /usr/local/lib/python3.10/dist-packages (from pandas) (2.8.2)
    Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.10/dist-packages (from pandas) (2024.2)
    Requirement already satisfied: tzdata>=2022.7 in /usr/local/lib/python3.10/dist-packages (from pandas) (2024.2)
    Requirement already satisfied: Jinja2>=2.9 in /usr/local/lib/python3.10/dist-packages (from bokeh) (3.1.5)
    Requirement already satisfied: contourpy>=1.2 in /usr/local/lib/python3.10/dist-packages (from bokeh) (1.3.1)
    Requirement already satisfied: packaging>=16.8 in /usr/local/lib/python3.10/dist-packages (from bokeh) (24.2)
  
```

Рисунок 2. Установка библиотеки Pathway

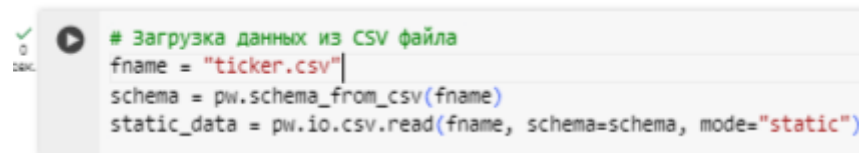
Для выполнения анализа данных установим необходимые библиотеки (рис.3): Pathway, Pandas, NumPy, Bokeh, Panel и Datetime. Установка и настройка этих библиотек обеспечивают полноценный анализ данных, начиная с их обработки и заканчивая визуализацией и интерактивным представлением результатов.

```

import datetime
import bokeh.plotting
import panel as pn
import pathway as pw
import pandas as pd
import numpy as np
  
```

Рисунок 3. Импорт библиотек

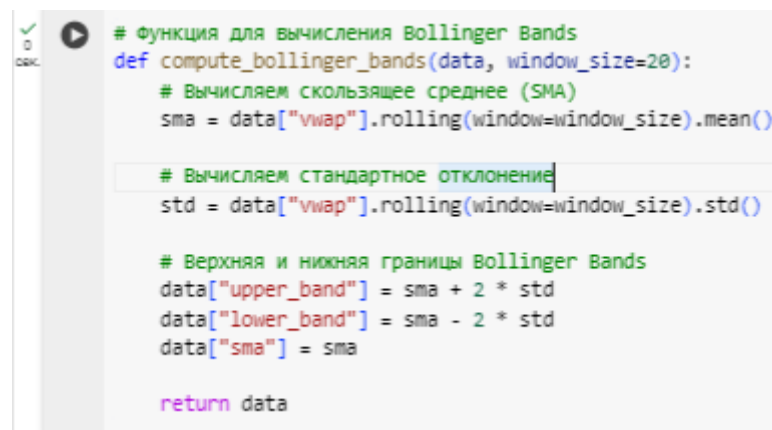
Загрузим данные из CSV-файла в таблицу с использованием фреймворка Pathway (рис.4).



```
# Загрузка данных из CSV файла
fname = "ticker.csv"
schema = pw.schema_from_csv(fname)
static_data = pw.io.csv.read(fname, schema=schema, mode="static")
```

Рисунок 4. Загрузка данных из CSV файла

Используем функцию для вычисления полос Боллинджера (Bollinger Bands) (рис.5) — индикатора, состоящего из трех линий: скользящего среднего (SMA) и двух границ (верхней и нижней), которые рассчитываются как SMA плюс/минус удвоенное стандартное отклонение цен. Этот инструмент позволяет трейдерам оценивать, насколько текущая цена отклоняется от среднего значения, и прогнозировать возможные развороты тренда.



```
# Функция для вычисления Bollinger Bands
def compute_bollinger_bands(data, window_size=20):
    # Вычисляем скользящее среднее (SMA)
    sma = data["vwap"].rolling(window=window_size).mean()

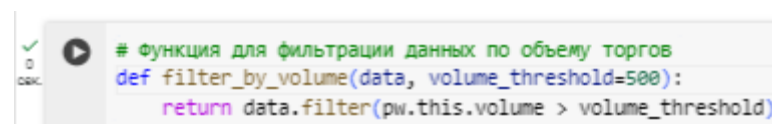
    # Вычисляем стандартное отклонение
    std = data["vwap"].rolling(window=window_size).std()

    # Верхняя и нижняя границы Bollinger Bands
    data["upper_band"] = sma + 2 * std
    data["lower_band"] = sma - 2 * std
    data["sma"] = sma

    return data
```

Рисунок 5. Функция Bollinger Bands

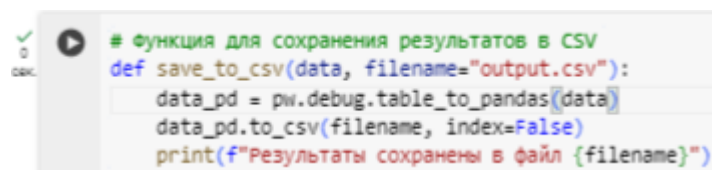
Необходимо вызывать функцию с параметром `volume_threshold=500`, чтобы в результате остались только строки, где объем торгов превышает 500. Это поможет оставить более значимые рыночные движения (рис.6)



```
# Функция для фильтрации данных по объему торгов
def filter_by_volume(data, volume_threshold=500):
    return data.filter(pw.this.volume > volume_threshold)
```

Рисунок 6. Фильтрация данных

Реализуем функцию для сохранения результатов анализа в CSV-файл уже отфильтрованных данных (рис.7).



```
# Функция для сохранения результатов в CSV
def save_to_csv(data, filename="output.csv"):
    data_pd = pw.debug.table_to_pandas(data)
    data_pd.to_csv(filename, index=False)
    print(f"Результаты сохранены в файл {filename}")
```

Рисунок 7. Функция для сохранения результатов в CSV

Создадим интерактивные элементы управления с использованием библиотеки Panel (pn), которые позволяют пользователю динамически изменять параметры анализа данных (рис.8).

```
✓ D  
DBK. # Создаем интерактивные элементы управления  
window_size = pn.widgets.IntSlider(name="Размер окна для SMA", value=20, start=5, end=50)  
volume_threshold = pn.widgets.IntSlider(name="Порог объема торгов", value=500, start=100, end=2000)
```

Рисунок 8. Создание интерактивных элементов управления

Далее реализуем интерактивную визуализацию, которая дает возможность пользователю изучать данные, динамически изменяя параметры анализа в реальном времени (рис.9).

```
✓ D  
DBK. # Функция для обновления графика  
@pn.depends(window_size.param.value, volume_threshold.param.value)  
def update_plot(window_size, volume_threshold):  
    # Фильтрация данных  
    filtered_data = filter_by_volume(static_data, volume_threshold)  
  
    # Преобразуем таблицу Pathway в Pandas DataFrame  
    filtered_data_pd = pw.debug.table_to_pandas(filtered_data)  
  
    # Проверяем, что данные не пустые  
    if filtered_data_pd.empty:  
        print("Нет данных для отображения.")  
        return bokeh.plotting.figure(title="Нет данных", height=400, width=600)  
  
    # Выводим первые несколько строк для проверки  
    print("Отфильтрованные данные:")  
    print(filtered_data_pd.head())  
  
    # Вычисление Bollinger Bands  
    data_with_bands = compute_bollinger_bands(filtered_data_pd, window_size)  
  
    # Визуализация  
    fig = bokeh.plotting.figure(  
        height=400,  
        width=600,  
        title="Bollinger Bands и скользящее среднее",  
        x_axis_type="datetime",  
    )  
    fig.line(data_with_bands["t"], data_with_bands["vwap"], legend_label="VWAP", line_color="blue")  
    fig.line(data_with_bands["t"], data_with_bands["sma"], legend_label="SMA", line_color="green")  
    fig.line(data_with_bands["t"], data_with_bands["upper_band"], legend_label="Upper Band", line_color="red")  
    fig.line(data_with_bands["t"], data_with_bands["lower_band"], legend_label="Lower Band", line_color="red")  
  
    return fig
```

Рисунок 9. Интерактивная визуализация

Приступим к созданию дашборда (рис.10).

```

# Создаем панель с элементами управления и графиком
dashboard = pn.Column(
    pn.Row(window_size, volume_threshold),
    update_plot
)

```

WARNING:pathway_engine.connectors.monitoring:PosixLikeReader-0: Closing the data source

Отфильтрованные данные:

	ticker	open	high	low	close \
^M3Q4CMQPGQQPPDS6Y6WY68300	AAPL	190.320	190.3200	190.310	190.3100
^6DCBAJADTPWCS0J7ZJT1FFR500	AAPL	189.490	189.5000	189.490	189.4901
^8EXCKQY3PAH4V2KJDHFWG58800	AAPL	190.260	190.2700	190.255	190.2602
^6ZEA759HXF265Z9JVQY0HYRA00	AAPL	189.935	189.9350	189.920	189.9300
^6CSB43PDC81MTA8SHQA1BS8800	AAPL	190.120	190.1201	190.120	190.1201

	volume	vwap	t	transactions	etc
^M3Q4CMQPGQQPPDS6Y6WY68300	3420.0	190.3149	1701106640000		42
^6DCBAJADTPWCS0J7ZJT1FFR500	3147.0	189.4908	1701095905000		28
^8EXCKQY3PAH4V2KJDHFWG58800	4472.0	190.2611	1701104144000		42
^6ZEA759HXF265Z9JVQY0HYRA00	6507.0	189.9305	1701117944000		49
^6CSB43PDC81MTA8SHQA1BS8800	552.0	190.1202	1701111511000		13

Рисунок 10. Создание дашборда

Далее необходимо написать код, который отобразит дашборд (рис.11).

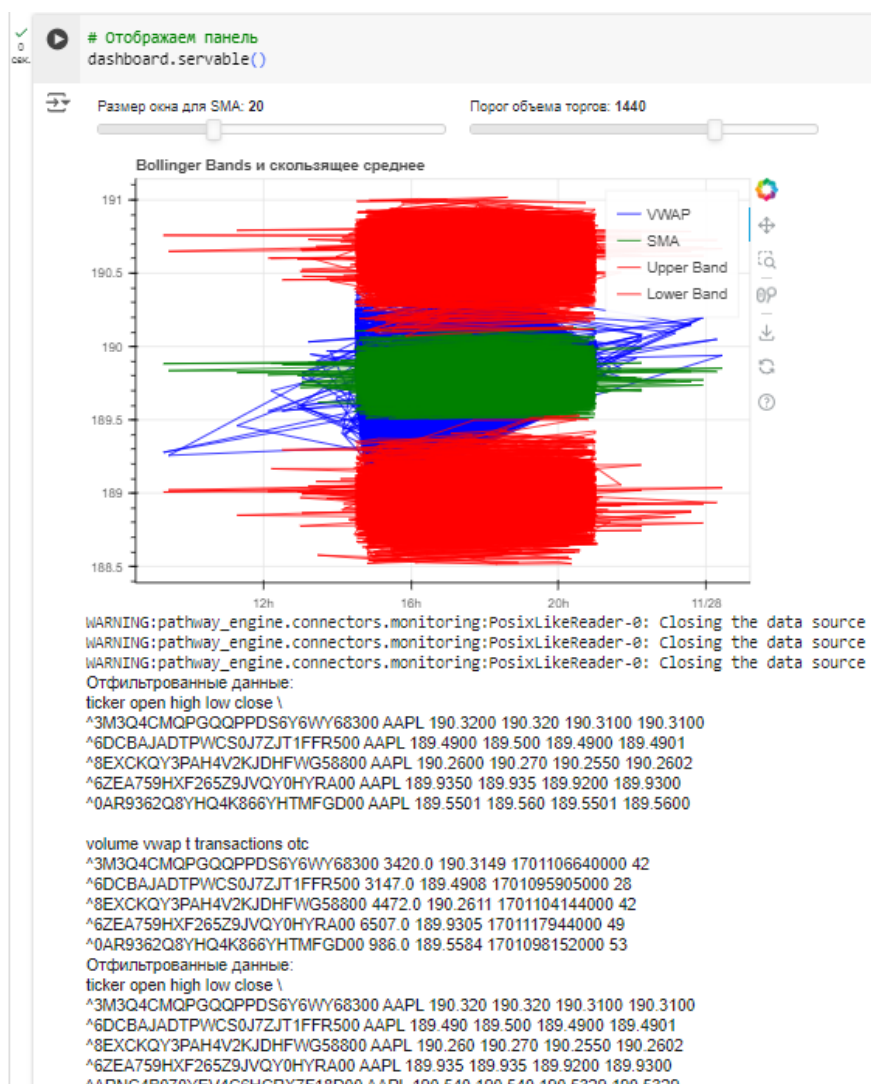


Рисунок 11. Дашборд

Если на дашборде переместить ползунки, изменяя значения размера окна для SMA и порога объема торгов, то дашборд автоматически обновит свой вид (рис.12), подстраиваясь под новые параметры. Это позволяет пользователю мгновенно видеть, как изменения влияют на анализ данных.

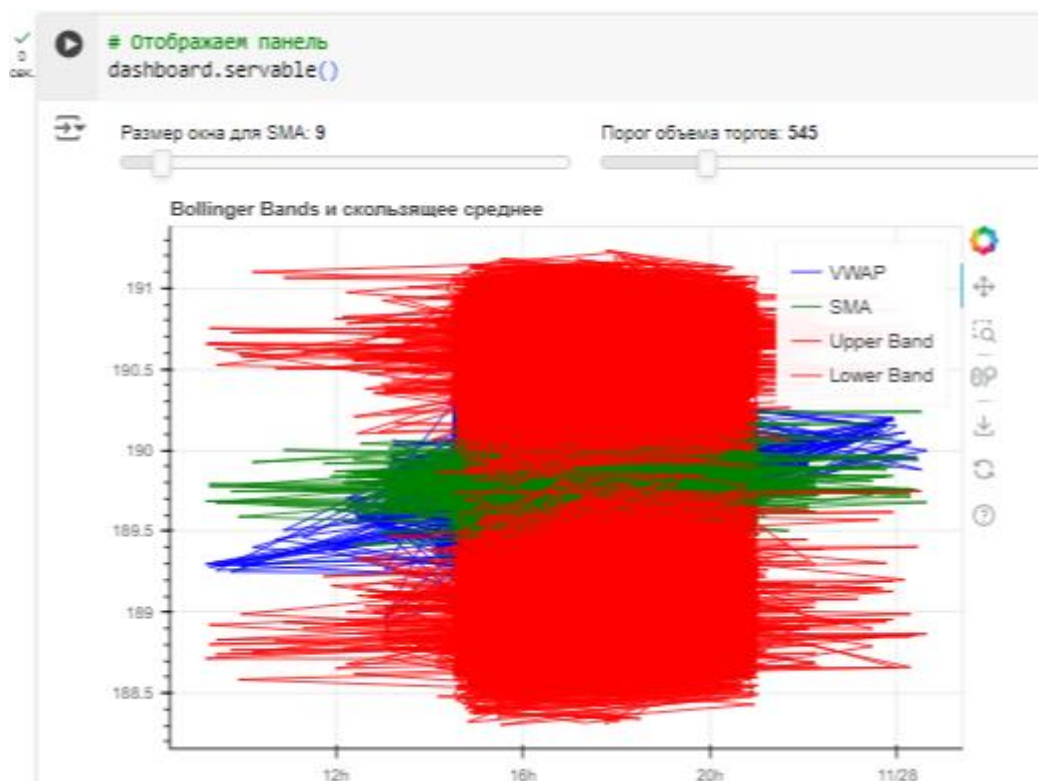


Рисунок 12. Обновление дашборда

Завершающим этапом сохраним результаты анализа в файл csv (см.13).

```
# Сохранение результатов в файл
save_to_csv(static_data, "static_results.csv")

WARNING:pathway_engine.connectors.monitoring:PosixLikeReader-0: Closing the data source
Результаты сохранены в файл static_results.csv
```

Рисунок 13. Сохранение результатов

Библиографический список

1. ticker.csv URL: <https://gist.githubusercontent.com/janchorowski/e351af72ecd8d206a34763a428826ab7/raw/ticker.csv>
2. Малик А.И. Построение Dashboard в python с помощью библиотеки Dash // Постулат. 2024. №8
3. Юрова А.А., Алясева Т.М., Жиленкова К.Р. Актуальные инструменты визуализации данных // Новая российская экономика: инвестиции, кластеры, инновации и дорожные карты. 2021. №6. С. 89-91.
4. Пылов П. А., Протоdjяконов А. В. Программные библиотеки языка программирования Python для реализации алгоритмов визуализации

- данных // Инновации. Наука. Образование, 2020. С. 233-239.
5. Пальмов С.В. Использование python для визуализации данных // Международный центр научного сотрудничества «Наука и просвещение». Пенза: МЦНС «Наука и просвещение», 2022. С. 20-23;
 6. Эберле Э. А., Гадельшина Г. А. Принципы проектирования дашбордов: исследование на примере python и flask/dash // Современные научные исследования: гуманитарные и технические науки: сборник материалов LVI-ой международной очно-заочной научно-практической конференции. М.: Издательство НИЦ «Империя», 2024. С. 23-24.
 7. Haranwala Y. J. et al. A dashboard tool for mobility data mining preprocessing tasks // 2022 23rd IEEE International Conference on Mobile Data Management (MDM). IEEE, 2022. С. 278-281.
 8. Бенко Е. В. Автоматизация аналитических процессов в системе образования Челябинской области // Научно-методическое обеспечение оценки качества образования, 2024. С. 52-57.