DOI 10.37882/2223-2966.2025.10.50

АЛГОРИТМ ОПТИМАЛЬНОГО ВЫБОРА ИНСТРУМЕНТА ДЛЯ АНАЛИЗА ИНФОРМАЦИИ НА ЯЗЫКЕ PYTHON В АНАЛИТИКЕ ДАННЫХ ИНДУСТРИИ E-COMMERCE

ALGORITHM FOR OPTIMAL TOOL SELECTION FOR DATA ANALYSIS IN PYTHON IN THE E-COMMERCE ANALYTICS INDUSTRY

I. Shkokov

Summary. Article addresses the problem of selecting the optimal data analysis tool for e-commerce web analytics. A comparative performance study of four popular Python libraries—Pandas, Polars, DuckDB, and PySpark—is conducted using typical business queries. Based on the experimental results measuring data processing time, the strengths of each library are identified according to task complexity and data volume. As a result, a decision tree algorithm is proposed to help data specialists choose the fastest tool for a specific analytical task, thereby significantly increasing workflow efficiency.

Keywords: data analysis, Python, optimization, web analytics, e-commerce, data libraries, performance.

Шкоков Игорь Олегович
Политехнический Институт Передовых Наук, г. Париж, Франция
igor.shkokov@gmail.com

Аннотация. В статье рассматривается проблема выбора оптимального инструмента для анализа данных в области веб-аналитики электронной коммерции. Проведено сравнительное исследование производительности четырех популярных библиотек языка Python: Pandas, Polars, DuckDB и PySpark на примере типичных бизнес-запросов. На основе результатов экспериментов, измеряющих время обработки данных, выявлены сильные стороны каждой библиотеки в зависимости от сложности задачи и объема данных. Предложен алгоритм в виде дерева решений, который помогает специалистам по данным выбирать наиболее быстрый инструмент для конкретной аналитической задачи, что позволяет значительно повысить эффективность работы.

Ключевые слова: анализ данных, Python, оптимизация, веб-аналитика, e-commerce, библиотеки для данных, производительность.

Введение

Согласно прогнозам, общий объем сгенерированных данных в мире продолжает стабильно расти и вырастет до 291 зеттабайта в 2027 году [1]. При таком бурном росте количество специалистов, занятых в индустрии data science и в анализе произведённых данных, будет соответственно неуклонно расти [2].

Современный анализ данных требует знания большого количества инструментов. Изобилие возможностей для процессинга наборов данных формируют «Парадокс выбора», при котором специалисты в области обработки данных используют неоптимальные инструменты и прикладывают усилия для переключение между ними.

В то же время, быстродействие скриптов обработки данных современных библиотек (написанных, в основном, для языка Python) достаточно предсказуемо и зависит от сложности запроса к базе данных, объема и формата данных и эффективности в манипуляциях с информацией самих библиотек. Перечисленные факторы позволяют представить себе возможным алгоритм для оптимального выбора необходимого инструмента. Данный алгоритм по вводным параметрам может определять, какая именно библиотека должна быть использована для конкретной задачи.

Для показательности исследования может быть рассмотрена аналитика крупного веб-сайта в индустрии e-commerce, как одна из наиболее репрезентативных областей в индустрии.

Литературный обзор

В работе [3] авторами приводится очень релевантное исследование похожего типа, которое концентрируется на большом числе параметров. Затронуты 7 популярных библиотек для анализа данных и исследованы многие показатели на нескольких наборах публичных данных. При этом, статья не имеет целью выработку алгоритма выбора оптимального инструмента для анализа данных, а также не затрагивает во всех экспериментах библиотеку DuckDB, показывающую достойные результаты в сравнительных анализах, опубликованных в профессиональной прессе.

В работе [4] авторы производят похожее исследование на синтетических датасетах. При этом, в сравнение попадают только Pandas и Polars, обходя стороной дру-

гие популярные в индустрии библиотеки для анализа данных.

Работа [5] оценивает уровень расхода энергии — кореллирующую, но не напрямую связанную с временем процессинга запроса метрику. В работе [6] автор исследования концентрируется на библиотеке Modin — достаточно быстрой для анализа, но недостаточно часто используемой в индустрии анализа данных. Целью работы [7] является сравнение разрабатываемой компанией Microsoft библиотеки PyFroid, которая призвана заменить Pandas и лучше использует ресурсы отдельно взятого процессора.

Таким образом, в научной литературе последних лет, в основном, имеются только частичные сравнения данных библиотек, не направленные на практические результаты. Также стоит отметить, что индустрия веб-аналитики с характерными для неё аспектами недостаточно затронута в подобных исследованиях.

Материалы и методы

Для создания оптимального алгоритма выбора инструмента было решено произвести ряд экспериментов на определение быстродействия работы нескольких библиотек обработки данных в Python. Были учтены современные реалии индустрии e-commerce и выбраны наиболее используемые метрики, представленные в таблице 1.

Таблица 1.

Метрики, используемые для эксперимента на быстродействие процессинга данных

Название запроса	Краткое описание запроса	Бизнес-применение и полезность
Количество просмотров карточек продуктов	Подсчет просмотра товаров: количество страниц, открытых пользователями.	Используется для оценки активности клиентов и как начальный этап воронки конверсии. Показывает вовлеченность пользователей.
MAU (Месячная активная аудитория)	Подсчет уникальных пользователей, проявивших активность в течение календарного месяца.	Одна из основных метрик для оценки клиентской базы. Используется для выявления сезонности, планирования развития веб-сайта и отчетности инвесторам.
Анализ воронки конверсии	Анализ пользовательского пути от просмотра товара до покупки, расчет коэффициентов конверсии на каждом этапе (просмотр → корзина → покупка).	Используется для оптимизации электронной конверсии: выявление товаров с лучшей конверсией в покупку, определение точек фрустрации пользователей.

Во время проведения экспериментов были проанализированы 4 современные библиотеки на языке Python:

- **Pandas:** самая популярная и устоявшаяся для работы с данными библиотека. Структуры данных, используемые для работы в этой библиотеке (DataFrame, Series) стали де-факто стандартом индустрии. Имея широкий набор инструментов для очистки и трансформации данных, pandas используется как для научных разработок, так и для промышленных применений. Особенное преимущество pandas в её легкой интеграции с другими современными элементами экосистемы языка Python, такими как NumPy, matplotlib и scikit-learn. При этом главное ограничение pandas в том, что она работает преимущественно в оперативной памяти на одном ядре (CPU) и плохо масштабируется для работы с большим объемом данных.
- **Polars:** относительно новая библиотека для работы с данными, написанная на языке Rust. Данная библиотека позиционируется как современная замена Pandas, предлагая низкое потребление памяти и использование колоночного формата хранения данных (Arrow), а также возможности использования привычного синтаксиса в стиле pandas.
- **DuckDB:** изначально задуманная как небольшой исследовательский проект, эта библиотека быстро набрала популярность благодаря своей легкости, понятному синтаксису и производительности. DuckDB позволяет почти полноценно использовать язык SQL в среде Python и не требует отдельного сервера. При этом она может работать с данными прямо в файлах, что представляет собой огромное преимущество по сравнению с другими библиотеками.
- **PySpark:** это библиотека и фреймворк, созданные специально для работы с большим объемом данных. По сути, PySpark представляет собой Python-интерфейс к программному продукту Apache Spark. PySpark изначально спроектирован для распределенных вычислений, что выгодно выделяет его в сравнении с другими подобными библиотеками. Он позволяет предсказуемо обрабатывать терабайты данных, используя кластеры серверов. PySpark активно применяется в корпоративной среде, где важна отказоустойчивость и масштабируемость.

В целях создания репрезентативных условий, максимально приближенных к современным условиям анализа реляционных таблиц в индустрии веб-аналитики, для создания алгоритма были использованы следующие ограничения:

- Все использованные инструменты (библиотеки, языки, среда программирования) были с открытой лицензией (open-source)

- Все манипуляции с данными производились в среде программирования Jupyter Notebook с ядром на языке Python
- Среда программирования была запущена локально, а не в облачном хранилище

Для симулирования необходимых для эксперимента данных был использован классический датасет платформы RetailRocket, опубликованный в открытом доступе на агрегаторе наборов данных Kaggle [8]. Он представляет собой анонимизированные таблицы с такими полями, как идентификатор продукта, дата открытия страницы с этим продуктом, идентификатор пользователя, время покупки, и т.д. Схема набора данных представлена на рисунке 1.

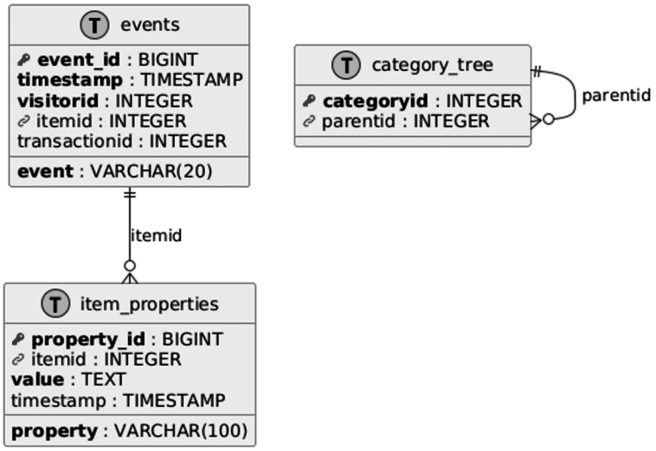

Рис. 1. Схема набора данных RetailRocket, использованного для экспериментов

Данные вычислительной техники, на которой проводился анализ, были специально подобраны для того, чтобы симулировать стандартную локальную среду рабочего ноутбука на Windows 11:

- Оперативная память: 8Gb RAM LPDDR3
- Процессор: Intel Core i5 8265U 1.6 GHz
- Накопитель: 256GB SSD

С помощью среды программирования Jupyter Notebook на языке Python был разработан необходимый код и произведены замеры быстродействия библиотек для каждого представленного запроса. Примеры запроса и ответа приведены в листинге 1 и в листинге 2.

```
import duckdb
import time
con = duckdb.connect()
start_time = time.time()
mau_duckdb = con.execute(f»»»
    SELECT
strftime(TO_TIMESTAMP(CAST(timestamp/1000 AS BIGINT)), '%Y-%m') AS month,
COUNT(DISTINCT visitorid) AS mau
FROM read_csv_auto('{csv_path}')
GROUP BY month
ORDER BY month
«»»).df()
end_time = time.time()
duckdb_runtime = float(f»{end_time — start_time:.3f}»)
print(«DuckDB runtime:», duckdb_runtime, «s»)
print(mau_duckdb.head())
```

Листинг 1. Пример использования библиотеки duckdb для анализа быстродействия запроса месячной активной аудитории

```
DuckDB runtime: 1.064 s
     month   mau
0 2015-05 306454
1 2015-06 313649
2 2015-07 377238
3 2015-08 311167 4 2015-09 174956
```

Листинг 2. Пример ответа библиотеки duckdb после выполнения запроса

Для уменьшения влияния случайных факторов каждый замер был сделан несколько раз. Таким образом, по результатам замеров было выявлено среднее время обработки запроса каждой библиотекой. После проведения экспериментов была получена сравнительная таблица 2, где по каждому запросу к базе данных приведено среднее время обработки запроса библиотекой в секундах.

Результаты

Таблица 2.

Среднее время обработки запроса к данным различными библиотеками

Тип запроса	Pandas, секунд	PySpark, секунд	DuckDB, секунд	Polars, секунд
Количество просмотров карточек продуктов	4,619	2,242	1,195	0,558
Месячная активная аудитория	3,814	2,347	1,365	0,694
Анализ воронки конверсии	10,778	14,103	2,309	7,142

Анализируя полученные данные, были выявлены следующие закономерности:

- Библиотека Polars проявляет наибольшее быстродействие с простыми запросами, но при тяжелых аналитических задачах, таких как анализ конверсии, наиболее быстра DuckDB.
- Библиотека PySpark работает гораздо медленнее для такого типа задач. При этом, как это было опи-

сано выше, только данная библиотека может стабильно работать при объеме данных, превышающем доступную оперативную память.
- Библиотека Pandas, являясь по сути стандартом индустрии, показывает гораздо более низкое быстродействие в сравнении с DuckDB и Polars.

Таким образом, благодаря выявленным при эксперименте закономерностям автором предлагается алгоритм оптимального выбора инструмента для анализа информации на языке Python при аналитике веб-данных, представленный на рисунке 2.

Используя данный алгоритм, специалисты в области работы с данными могут заметно ускорить сроки анализа информации. Применение этого алгоритма может позволить аналитику данных при повседневных задачах сократить сроки анализа данных на 20–30 %, что эквивалентно освобождению 1–1.5 часов рабочего дня и заметному повышению производительности труда.

Заключение

Эксперименты, проведенные с наиболее популярными библиотеками для анализа данных на языке программирования Python, позволили оценить быстродействие типичных запросов к базе данных, полученной от e-commerce сервиса. На основе данных экспериментов был разработан алгоритм, который позволяет быстро осуществить выбор библиотеки с помощью простого дерева решений. Разработанный алгоритм позволяет сократить время процессинга типичных запросов и оптимизировать ресурсы для анализа данных.

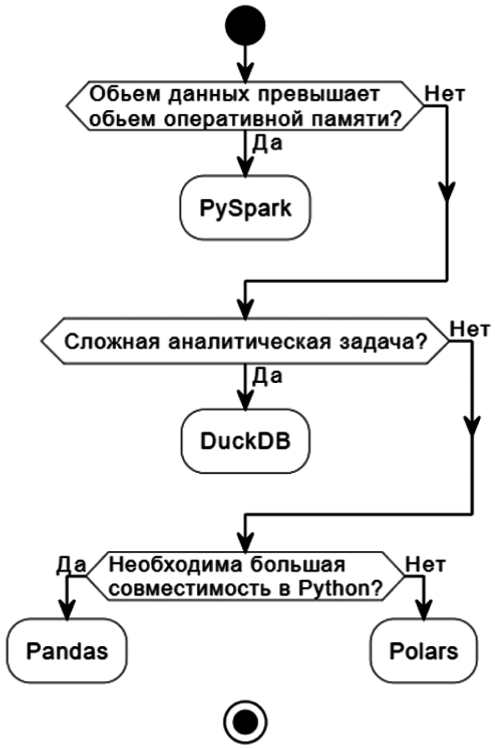

Рис. 2. Алгоритм оптимального выбора инструмента для анализа информации на языке Python при аналитике веб-данных

ЛИТЕРАТУРА

1. B. Alshemaimri, A. Badshah, A. Daud, A. Bukhari, R. Alsini, and O. Alghushairy, «Regional computing approach for educational big data», Scientific Reports, vol. 15, no. 1, p. 7619, 2025. doi: 10.1038/s41598-025-92120-7.
2. Data Strategy Pros. «Data management job projections». (2024), [Online]. Available: https://www. datastrategypros.com/resources/data-management-job-projections (visited on 08/08/2025).
3. A. Mozzillo, L. Zecchini, L. Gagliardelli, A. Aslam, S. Bergamaschi, and G. Simonini, «Evaluation of dataframe libraries for data preparation on a single machine», in Proceedings of the 28th International Conference on Extending Database Technology (EDBT), Barcelona, Spain, Mar. 2025, pp. 337–349.
4. F. Nahrstedt, M. Karmouche, K. Bargie, P. Banijamali, A.P.K. Nalini, and I. Malavolta, «An empirical study on the energy usage and performance of pandas and polars data analysis python libraries», in Proceedings of the 28th International Conference on Evaluation and Assessment in Software Engineering (EASE 2024), Salerno, Italy, Jun. 2024, p. 10.
5. S. Shanbhag and S. Chimalakonda, «On the energy consumption of different dataframe processing libraries — an exploratory study», arXiv:2209.05258, 2022.
6. D. Petersohn et al., «Flexible rule-based decomposition and metadata independence in modin: A parallel dataframe system», Proceedings of the VLDB Endowment, vol. 15, no. 3, pp. 739–751, 2021.
7. V. Emani, A. Floratou, and C. Curino, «Pyfroid: Scaling data analysis on a commodity workstation», in Proceedings of the 2024 Conference on Extending Database Technology (EDBT), 2024.
8. R. Zykov. «Retailrocket recommender system dataset». (2022), [Online]. Available: https://www.kaggle. com/datasets/retailrocket/ecommerce-dataset (visited on 08/08/2025).

© Шкоков Игорь Олегович (igor.shkokov@gmail.com)
Журнал «Современная наука: актуальные проблемы теории и практики»