СРАВНЕНИЕ БЫСТРОДЕЙСТВИЯ ПОПУЛЯРНЫХ БИБЛИОТЕК МАНИПУЛЯЦИИ ДАННЫХ НА ЯЗЫКЕ PYTHON

Гребенюк Я.В.
НИ ТПУ, ИШИТР, 8ПМ11, e-mail: yvg8@tpu.ru

Введение

Для интерактивной визуализации данных часто используются информационные панели, содержащие различные таблицы, графики, а также поля настройки и взаимодействия. Функционирование такого продукта включает в себя несколько частей:

1. Манипуляция и анализ данных;
2. Визуализация;
3. Настройка взаимодействия объектов;
4. Выкладка панели.

Комфортное функционирование интерактивной информационной панели предполагает высокое быстродействие всей системы, одной из самых медленных частей которой зачастую является именно первый, ведь на него приходится множество вычислений, особенно если представлен большой объём данных [1].

Наиболее распространенным инструментом манипуляции данных в Data Science можно назвать библиотеку Pandas [2] языка программирования Python. Начиная свою историю как проект с открытым кодом в 2009 году, библиотека была призвана облегчить работу с данными, предоставляя доступный синтаксис и высокую производительность. Вдохновение черпалось из мира финансов и таких инструментов как R и Excel.

Однако в 2020 году группа специалистов из области Data Science создали библиотеку под названием Polars как прямую замену Pandas с целью улучшить производительность.

Целью данной работы является сравнение производительности двух инструментов.

Описание алгоритма

Для выполнения сравнения использовались язык Python версии 3.10.2, платформа JupyterLab версии 3.3.4, библиотека Pandas версии 1.4.1, библиотека Polars версии 0.16.5.

Использовался датасет из открытого доступа, содержащий порядка 12.000.000 записей о компаниях по всему миру с числовыми и символьными полями, из датасета были взяты шесть фрагментов размером 1000, 500000, 1000000, 12000000 записей.

Последовательно выполнялись следующие операции: загрузка данных в память, выбор части датасета, фильтрация записей согласно условию на числовом поле, фильтрация записей согласно условию на строковом поле, подсчет уникальных числовых значений, подсчет уникальных строковых значений, функция агрегации по нескольким полям с подсчетом среднего, подсчет пропущенных значений, описание датасета, сохранение датасета.

Для сохранения времени выполнения использовался враппер, начинающий отсчет времени перед выполнением функции, и возвращающий время выполнения вместе с результатом выполнения.

```python
from functools import wraps
import time

def timeit(func):
    @wraps(func)
    def timeit_wrapper(*args, **kwargs):
        start_time = time.perf_counter()
        result = func(*args, **kwargs)
        return time.perf_counter() - start_time,result
    return timeit_wrapper
```

Рис. 1. Код враппера для фиксации времени

Для лучшей читаемости построим трехмерный столбчатый график с помощью библиотеки Matplotlib.

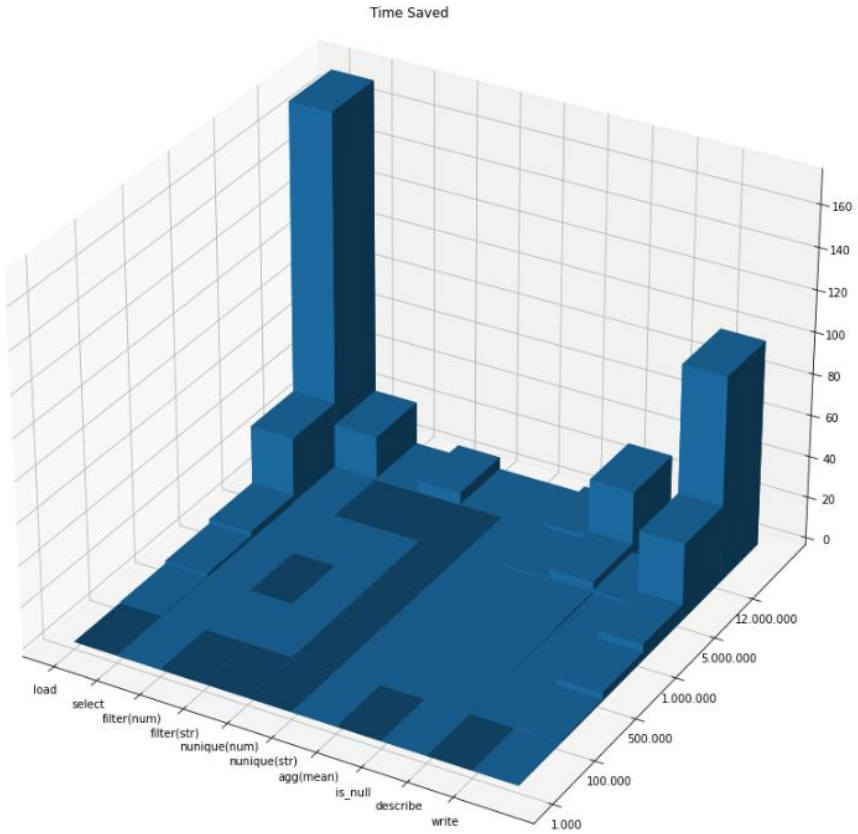

Рис. 2. График разницы во времени

Заключение

Из полученных данных можно судить о том, что Polars не сильно уступает Pandas в производительности на малых наборах данных, но чем больше набор, тем эффективнее с ним работает данная библиотека – пики быстродействия наблюдаются при загрузке и сохранении данных. В пользу Pandas же, помимо упрощенного синтаксиса, можно отнести тот факт, что многие годы, во время которых эта библиотека была популярна, расположили множество других знаковых проектов на прямую интеграцию, чего пока не наблюдается в Polars.

Список использованных источников

1. Jwo, Jung-Sing & Lin, Ching-Sheng & Lee, Cheng-Hsiung. (2021). An Interactive Dashboard Using a Virtual Assistant for Visualizing Smart Manufacturing. Mobile Information Systems. 2021. 1-9. 10.1155/2021/5578239.
2. Келлехер Джон Д. Наука о данных. Базовый курс – М.: Альпина Паблишер, 2021. – 224 с.