Обоснование процесса разработки и введения в эксплуатацию проектов в сфере анализа данных

К.А. Григорян[1], М.В. Фишер[1], А.Р. Мангушева[2]

[1]*Казанский федеральный университет, Кремлёвская, 18, Казань, Россия, 420008*
[2]*Казанский национальный исследовательский технологический универсистет, Карла Маркса, 68, Казань, Россия, 420015*

Аннотация

В данной статье представлены результаты исследования возможности интеграции различных инструментов разработки ML-приложений и их совместного использования с Jupyter Notebook. Показано, как интеграция различных инструментов разработки моделей машинного обучения могут обеспечить масштабируемость и высокую скорость реализации проектов в сфере анализа данных. В результате, благодаря использованию различных фреймворков и систем, предложен процесс разработки от первичного анализа сырых данных до создания полноценного web-приложения, позволяющий экономить ресурсы и обеспечивающий более быстрый time to market.

Ключевые слова

Анализ данных, ML-приложения, инструменты анализа данных, модели машинного обучения

1. Введение

В настоящее время самым популярным инструментом разработки проектов в области анализа данных, особенно на этапе первичного анализа и построения визуализаций, является Jupyter Notebook. Он представляет собой среду разработки, нацеленную на быстрый анализ и разработку прототипов, но становится неудобным при масштабировании проекта. [1][2][3].

Project Jupyter — это некоммерческий проект с открытым исходным кодом, родившийся в 2014 году в рамках проекта IPython и призванный поддерживать интерактивную науку о данных и научные вычисления на всех языках программирования. [4] Популярными продуктами проекта являются Jupyter Notebook и JupyterLab.

Главное преимущество Jupyter перед другими средами разработки заключается в том, что код можно разделять на ячейки и запускать в произвольном порядке, сразу же получая результат. Это удобно при первичном анализе данных и создании прототипов, однако несет за собой побочные эффекты, такие как плохая читаемость кода, сложность масштабирования, трудность переноса кода в отдельное приложение, что замедляет процесс разработки. [3] Кроме того, строение ноутбука подразумевает, что каждый ноутбук является самостоятельным скриптом и несколько ноутбуков не могут взаимодействовать между собой без сторонних расширений.

Целью данной работы является исследование возможности интеграции инструментов и фреймворков для проектов в сфере анализа данных, построение процесса разработки приложений от первичного анализа сырых данных до создания полноценного web-приложения с использованием методов машинного обучения.

2. Традиционные методы и инструменты реализации проектов в сфере анализа данных

Как уже было сказано ранее, Jupyter Notebook и его аналоги являются самой распространенной средой разработки в сфере анализа данных и машинного обучения [1]. При выстраивании процесса разработки планируется опираться на его достоинства и

преимущества, одновременно нивелируя его неудобства и недостатки, используя прочие инструменты.

Зачастую после разработки модели машинного обучения ее ввод в эксплуатацию осуществляется с помощью создания REST API для обращения к модели. Чаще всего для этого используется web-фреймворк Flask, а точнее расширение для Flask — Flask-RESTful, в которое добавлена поддержка быстрого построения REST API [6][7].

Как правило, разработка и вывод в эксплуатацию web-серверов не входит в служебные обязанности аналитика данных, из-за чего создание даже простого сервера требует затрат дополнительных ресурсов.

Кроме этого, может возникнуть ситуация, в которой web-интерфейс для взаимодействия с моделью и отображения результатов ее работы становится необходимостью. Чаще всего это означает разработку Single Page Application с использованием таких фреймворков, как Vue, React и т.п., что требует привлечения других работников, а зачастую и целой команды.

Таким образом, традиционная методика разработки удобна лишь до создания первых прототипов. Использование Jupyter Notebook в чистом виде затрудняет процесс масштабирования проекта и добавления новых функций.

Для решения этих проблем можно выделить два основных пути:

1)Расширение возможностей Jupyter Notebook: упрощение процесса контроля версий; параметризация ноутбуков и возможность их запуска как обычных скриптов;автоматизация создания отчетов на основе ноутбуков.

2) Приоритизация использования инструментов на Python: возможность создания web-приложений аналитиком данных; использование Python для автоматизации процессов.

3. Общая схема интеграции инструментов разработки проекта по анализу данных

За основу была взята структура проекта от Cookie Cutter Data Science[8]. Весь ETL-процесс начинается с хранилища данных. В нашем проекте хранилище представлено в виде базы данных SQLite, в которую были загружены данные о заказах в формате csv, предоставленные партнерами из сервиса доставки продуктов.

Выгрузка сырых данных осуществляется с помощью отдельного ноутбука, принимающего параметры даты начала и даты окончания желаемого временного окна. Полученные данные сохраняются в директории проекта data/raw/ в формате csv и с указанием временного промежутка данных в названии файла. Запуск этого ноутбука производится с помощью регулярных DAG Airflow, которые выгружают заказы за данный период времени.

Далее в тех же DAG идет запуск ноутбуков, формирующих отчеты по данным из data/raw/ и сохраняющих их в директории reports/[period]/[format]/, где period указывает на период отчета, а format — на формат файла, например, html или pdf.

Кроме этого, после первоначальной выгрузки сырых данных отдельно запускается задача предобработки этих данных в датасет, используемый для (до)обучения модели. Предобработанные данные сохраняются в директорию data/preprocessed/ и впоследствии используются ноутбуком, отвечающим за обучение модели. Полученная модель и ее веса сохраняются в model/.

Наконец, данные из /data/processed/ и модель и веса из model/ используются web-интерфейсом Streamlit для создания предсказаний и отображения визуализаций.

В результате проведенного исследования был получен гибкий процесс разработки, опирающийся на удобство среды разработки Jupyter Notebook, одновременно избавляясь от ее недостатков.

4. Литература

[1] Perkel, J.M. Why Jupyter is data scientists' computational notebook of choice? // Nature. – 2018. – Vol. 563(7732). – P. 145-147.

[2] Kluyver, T. Jupyter Notebooks — a publishing format for reproducible computational workflows / T. Kluyver, B. Ragan-Kelley, F. Pérez, B.E. Granger, M. Bussonnier, J. Frederic, P. Ivanov // Elpub. – 2016. – P. 87-90.
[3] Chattopadhyay, S. What's Wrong with Computational Notebooks? Pain Points, Needs, and Design Opportunities / S. Chattopadhyay, I. Prasad, A.Z. Henley, A. Sarma, T. Barik // Conference on Human Factors in Computing Systems, 2020.
[4] Project Jupyter, 2020 [Электронный ресурс]. – Режим доступа: https://jupyter.org.
[5] Jupyter Documentation, 2020 [Электронный ресурс]. – Режим доступа: https://jupyter.org/documentation.
[6] Flask Documentation, 2010 [Электронный ресурс]. – Режим доступа: https://flask.palletsprojects.com/en/1.1.x/.
[7] Flask-RESTful Documentation, 2020 [Электронный ресурс]. – Режим доступа: https://flask-restful.readthedocs.io/en/latest/.
[8] CookieCutter Data Science, 2020 [Электронный ресурс]. – Режим доступа: https://drivendata.github.io/cookiecutter-data-science/.