


Что дальше?

Михаил Трофимов
DMIA Production ML  весна 2021

- Мы обсудили немного то, что устроен как завернуть модель в сервис, задеплоить ее и сделать базовую петлю переобучения.
- Но в реальности есть еще много граней, с которыми нужно иметь дело.

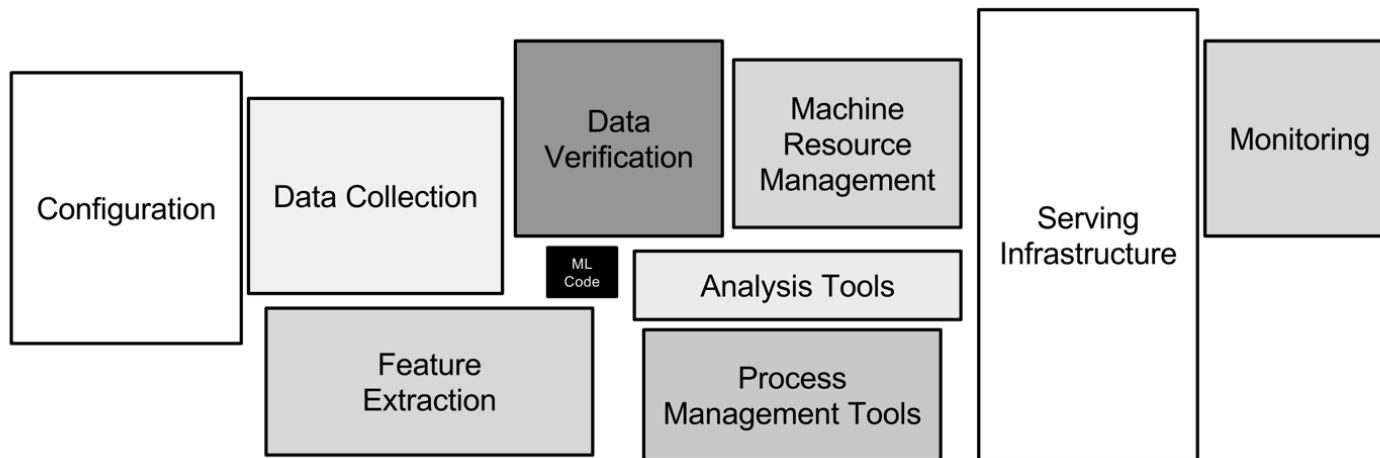


Figure 1: Only a small fraction of real-world ML systems is composed of the ML code, as shown by the small black box in the middle. The required surrounding infrastructure is vast and complex.

Картинка из статьи *"Hidden Technical Debt in Machine Learning Systems"*

- Хочется про них поговорить (очень обзорно) - чтобы в голове был концепт и несколько названий инструментов, как можно решить.
- Второй момент - прояснить роль и место некоторых известных инструментов.
 - Инструменты меняются часто, парадигмы - реже

- Полезно хотя бы немного знать про все, таковы особенности рынка (обсуждается далее, в командной работе)
- Всегда есть тонкости, всегда есть нюансы, нужно уметь залезть под капот.

О чем упомянем?

- Нагрузка на саму модель
- Переобучение модели
- Выкатка модели
- Доставка фичей
- Процессы вокруг модели
- Командная работа

Нагрузка на модель

- В зависимости от условий - может быть 1 запрос в час, а может - тысячи в секунду. Модель должна ее выдерживать.
- Тут возникает все, что относится к хайлоаду (highload).

- Полезно понимать, как работает система целиком, какие есть узкие места, как масштабируется.

- Масштабирование - вертикальное/горизонтальное

- Stateless сервисы

- Балансировщик
- Понятие sticky sessions / affinity

- Алгоритмы, умение читать код, не бояться заглядывать под капот.

- Эффективность языка программирования.
- Часто тренируют модель на python, а сервят в чем-то другом (C++, Java, Go)
- Не нужно бояться других языков

- Кэши
- Батчевание (особенно для GPU)

- Трейсинг запросов
- <https://opentelemetry.io>

Переобучение модели

Готовка данных

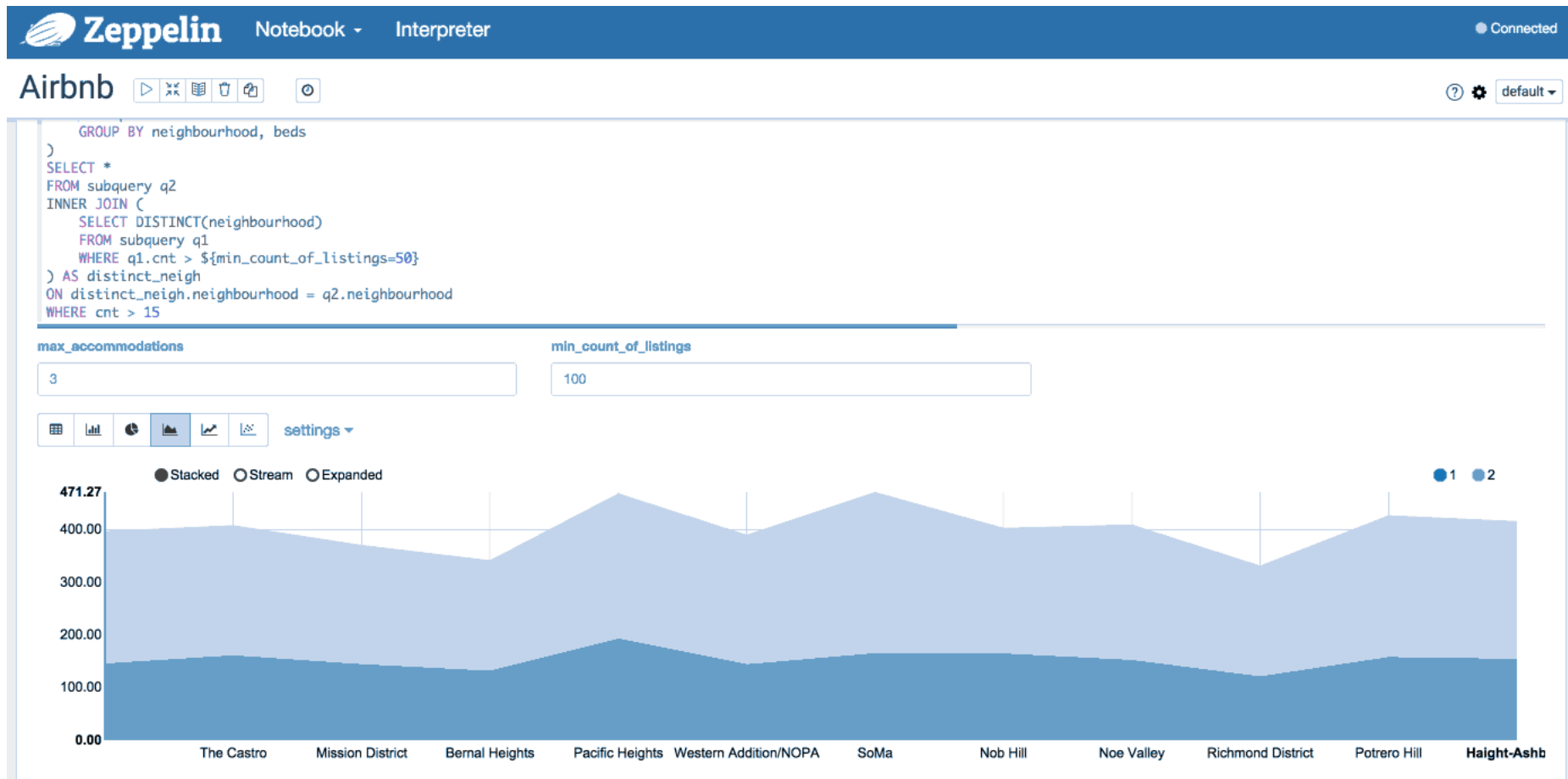
- Модель нужно готовить, данные нужно готовить.
- Хорошая новость - это оффлайн процессы, мы не так ограничены во времени и скорости.
- Как правило, на этом этапе нужно ворочать большими объемами данных.
- Возникает почти все, что относится к Big Data.

Offline

- Работа с хранилищем, вычислительными кластерами.

- Apache Spark как инструмент работы с данными.
- Оперирует объектами DataFrame, похоже на Pandas
- Поддерживает SQL и возможность написать кастомный код обработки

- Zeppelin как альтернатива Jupyter



- Код объединяется в пайплайны - их нужно запускать на кластере и менеджить.
- Кто-то должен решать задачу распределения ресурсов кластера, манипуляции и задачами и своевременный запуск задач.
- Этот класс инструментов называют "планировщики" ("шедулеры" / "schedulers")
- airflow, oozie - ровно оно

DAGs

| All 26 | | Active 10 | Paused 16 | Filter DAGs by tag | | Search DAGs | |
|--|---------|-----------|---------------|----------------------|--------------|--|-------|
| DAG | Owner | Runs | Schedule | Last Run | Recent Tasks | Actions | Links |
| <input checked="" type="checkbox"/> example_bash_operator example example2 | airflow | 2 | 0 0 *** | 2020-10-26, 21:08:11 | 6 | ▶ ↺ 🗑️ | ... |
| <input checked="" type="checkbox"/> example_branch_dop_operator_v3 example | airflow | | * / 1 * * * * | | | ▶ ↺ 🗑️ | ... |
| <input type="checkbox"/> example_branch_operator example example2 | airflow | 1 | @daily | 2020-10-23, 14:09:17 | 11 | ▶ ↺ 🗑️ | ... |
| <input checked="" type="checkbox"/> example_complex example example2 example3 | airflow | 1 1 | None | 2020-10-26, 21:08:04 | 37 | ▶ ↺ 🗑️ | ... |
| <input checked="" type="checkbox"/> example_external_task_marker_child | airflow | 1 | None | 2020-10-26, 21:07:33 | 2 | ▶ ↺ 🗑️ | ... |
| <input checked="" type="checkbox"/> example_external_task_marker_parent | airflow | 1 | None | 2020-10-26, 21:08:34 | 1 | ▶ ↺ 🗑️ | ... |
| <input checked="" type="checkbox"/> example_kubernetes_executor example example2 | airflow | | None | | | ▶ ↺ 🗑️ | ... |
| <input checked="" type="checkbox"/> example_kubernetes_executor_config example3 | airflow | 1 | None | 2020-10-26, 21:07:40 | 5 | ▶ ↺ 🗑️ | ... |
| <input checked="" type="checkbox"/> example_nested_branch_dag example | airflow | 1 | @daily | 2020-10-26, 21:07:37 | 9 | ▶ ↺ 🗑️ | ... |
| <input type="checkbox"/> example_passing_params_via_test_command example | airflow | | * / 1 * * * * | | | ▶ ↺ 🗑️ | ... |

Пример интерфейса airflow

Online

- Горячие факторы, быстрая обратная связь

- Концепция “брокера сообщений”
 - Apache Kafka.
- Поточная обработка данных
 - Spark Streaming
 - Apache Beam
 - Apache Flink

Разные типы базы данных

- in-memory (redis)
- nosql (mongo)
- Apache Cassandra
- ...

Воспроизводимость

- dvc
- mlflow
- CACE = "Change Anything Change Everything"
- GIGO = "Garbage In Garbage Out"

Управление артефактами

- Артефакты нужно как минимум хранить - s3, gcs, ...
 - Nexus
 - Artifactory
- Желательно иметь access control и версионирование
 - Quilt

Выкатка модели

- Модель уже в проде, ее кто-то использует. Надо обновить
- Нужно ли делать бесшовную выкатку (с точки зрения времени ответа)?
- Как правило, старую модель уже кто-то использует, возможно новую надо катить рядом, а не вместо
- Разные способы деплоя за балансером
 - Canary Deployment
 - Blue-Green Deployment

Выкатка данных

- Модель функционирует, но нужно обновить для нее данные
- Как организовать этот процесс?

Процессы вокруг модели и данных

- Как добавляется новая фича? Когда можно удалить? Какой жизненный цикл?
- Как заводить новую модель? Какой жизненный цикл у нее?
- А можно ли фичи переиспользовать между моделями?
- Кто отвечает за фичу и поддерживает ее?
- А как правильно собирать фичи в оффлайне и рантайме?

- Концепция "feature store"
 - <https://www.featurestore.org>



Feature Store Comparison

| Platform | Open-Source | Offline | Online | Metadata | Feature Engineering | Supported Platforms | TimeTravel / Point-in-Time Queries |
|---------------------------------------|-------------|-----------------|-----------------------|--------------------------------|---------------------------------------|--------------------------|------------------------------------|
| Hopsworks | AGPL-V3 | Hudi/Hive | MySQL Cluster | DB Tables, Elasticsearch | (Py)Spark, Python | AWS, GCP, On-Prem | SQL Join or Hudi Queries |
| Michelangelo | N/A | Hive | Cassandra | Content | Spark, DSL | Proprietary | SQL Join |
| Feast | Apache V2 | BigQuery | BigTable/Redis | DB Tables | Beam, Python | GCP | SQL Join |
| Conde Nast | N/A | Kafka/Cassandra | Kafka/ Cassandra | Protocol Buffers | Shared libraries | Proprietary | ? |
| Zipline | N/A | Hive | KV Store | KV Entries | Flink, Spark, DSL | Proprietary | Schema |
| Comcast | N/A | HDFS, Cassandra | Kafka / Redis | Github | Flink, Spark | Proprietary | No? |
| Netflix Metaflow | N/A | Kafka & S3 | Kafka & Microservices | Protobufs | Spark, shared libraries | Proprietary | Custom |
| Twitter | N/A | HDFS | Strato / Manhattan | Scala shared feature libraries | Scala DSL, Scalding, shared libraries | Proprietary | No |
| Facebook FBLearder | N/A | ? | Yes, no details | Yes, no details | ? | Proprietary | ? |
| Pinterest Galaxy | N/A | S3/Hive | Yes, no details | Yes, no details | DSL (Linchpin), Spark | Proprietary | ? |
| Iguazio Feature Store | N/A | Parquet | Yes, in mem database | Yes, no details | Spark, Python, Nuclio | AWS, Azure, GCP, on-prem | Yes, native time series or SQL |

Работа в команде

- Есть 2 модели устройства команды
 - узкие специалисты
 - широкие специалисты (фуллстеки)
- Я топлю за фуллстеков и считаю, что надо знать всего по немногу.

Полезно стремиться к stateless состоянию

Проблема разделения ресурсов (вычислительных)

Проритезация гипотез