Вводная лекция: отличия от MLOps, состав команды

Page • 1 backlink • uni

Формат: две пары

1. лекция по ML System Design
2. семинар по машинному обучению

Из проектов, которые нам прислали, обсудим, что можно брать, на семинаре

MLOps vs ML System Design – первое близко к DevOps (набор практик, как не развалить сервис), второе про архитектуру сервисов и продуктов (можно считать, что первое включается во второе)

ML Software в отличие от обычного превращает входы и выходы в паттерны (в отличие от входы + паттерны = выходы)

В ML версионировать нужно не только код, а еще и данные, модели на выходе, связи между ними, внедрение определяется A/B тестированием (и оффлайн: t-test и wilcoxon)

Стоит различать между ML-продуктами и ML-системами – Алиса и беспилотник – это продукты, использующие ML-системы, модель Llama – это компонента ML-системы

Дополнительная информация по источникам:

- medium etc
- конференции
- 10 years of practice
- Stanford CS329 – курс по ML System Design
- cherry pick лекций и материала от коллег

AI сервис состоит из сервиса и ML-системы, к которой обращается этот сервис (в каком-то смысле сервис инкапсулирует ML-систему)

ML-система является непрерывно развивающейся сущностью – функцией, необходимой для реализации клиента

При создании ML-системы возникает множество вопросов: откуда брать данные? как оценить качество? как гарантировать работу? как исправлять ошибки?

В отличие от ресерча, в котором важен перфоманс модели, в проде у разных стейкхолдеров могут быть разные цели; также в проде быстрый ответ важнее, чем быстрое обучение (влияет на количество пользований сервисом); данные в реальности постоянно меняются в отличие от статических бенчмарков (и не всеми данными можно пользоваться!); в бизнесе важна интерпретируемость модели; качество важно не в среднем, а по некоторой высокой квантили (цена ошибки сильно выше)

Стейкхолдеры и их цели:

- ML (train something)
- Sales (sell something)
- Product (make users happy)
- Dev (Mongo vs Postgre)
- SRE (99.999%)
- Business (EBITDA)

По итогу ML System Design развивается по времени, требует хорошего менеджмента большого количества разноплановых специалистов и стейкхолдеров, опирается как на качественный математический аппарат и инженерию

При этом пользователи, взаимодействующие с ML-системой, сами создают подспорье для е улучшения: генерируют сигналы качества, а порой сами приходят с ошибками нашей модели – первое уходит в data development, а второе напрямую связано с quality analysis

В cross-functional team специалисты узконаправленные (оттого и проще найм), по нужно много коммуникации, а в full stack team быстрее разработка, проще допускать нетривиальные архитектурные решения – в ходе курса будем идти от последнего к первому