# Model Monitoring & Maintenance

Page • 1 backlink • uni

Все модели – это черные ящики (почти), при этом важно отслеживать, что с ними происходит в продакшене

Бизнес хочет знать, можно ли верить модели, а разработчики должны знать, если в ней что-то "взорвется"

Между деплоем и улучшением нужно мониторить модель, а также анализировать ее качество и перф (в онлайне)

Model Monitoring спрашивают на интервью:

- зачем мониторить модели? (применительно к конкретному use case)
- какие приложения можно использовать для мониторинга?
- какие аспекты нужно мониторить?
- как бы выглядел идеальный мониторинг вокруг конкретной инфраструктуры?

У мониторинга в первую очередь человеческий запрос и требования, поэтому каждый алерт должен срабатывать по делу и в корректное время (в зависимости от критичности ситуации)

Челленджи до внедрения

- что мониторить?
- как мониторингу взаимодействовать с моделью?
- технические зависимости
- (?)

Челленджи после внедрения

- стоит ли беспокоиться при сдвиге?
- как быстро мониторинг покажет проблему при ее возникновении?
- (?)

Также системы мониторинга позволяют установить внешние источники проблем и в compliance

Естественные метки как элемент мониторинга – решили ли мы задачу пользователя? (возможность у пользователя самому определить,

устроил ли его ответ – в случае больших сервисов сигнала хватает), но есть риск получить шум и сдвиги, помимо этого для некоторых сигналов feedback loop занимает даже месяцы

Какие могут быть ошибки

- операционные (о них говорили раньше)
- машинного обучения

> По статистике Google более 60% связано именно с инфраструктурой

Но все же, какие бывают ошибки машинного обучения

- продовые данные не такие, как обучающие

> Выбили попугаев в разработке, но в продашкене метрики плохие, а также само распределение данных могло сдвинуться

- краевые случаи
- проблемы в распределении данных, связанные с feedback loop

> Особенно актуально для рекомендательных систем: показываем всем без разбора только залайканный контент; также актуально для систем скрининга резюме – такую проблему сложно отловить без прошествия нескольких итераций применения модели
> Варианты уменьшения эффекта:
> - рандомизация (однорукие бандиты)
> - позиционные факторы (на обучении вводим позиционный фактор, а на инференсе его пессимизируем), а также с помощью моделей (?)

Какие бывают сдвиги в распределении?

- $P(Y|X)$ – сдвиг концепта (тяжело отлавливать, скорее всего нужно переобучить модель)

> Могло поменяться из-за внешних факторов, сезонности

- data drift (отслеживать гораздо проще)
    - $P(Y)$ – label drift
    - $P(X)$ – feature drift / covariate shift

    > Причины в сборе данных (потенциально bias при сэмплировании, пересэмплировании представителей редикх классов и так далее)

При этом изменения не обязательно могли произойти из-за внешних изменений, мог нарушиться процесс сбора данных, как обнаружить и понять?

Observability – возможность определения неполадок при их возникновении (как правило, является часть системы мониторинга)

Метрики железа и бэкенда разбирали ранее, их всегда нужно мониторить

Какие показатели машинного обучения мониторить?

- сырые входы – сложно отслеживать, лучшее понимание
- факторы – непросто отслеживать, хорошее понимание, самый частый способ отслеживания (есть софт типа pandas-profiling, facets)
- предсказания – легко отслеживать, неплохое понимание – легко визуализировать, можно считать статистики и проводить two-sample тесты
- метрики – очень легко отслеживать, слабое понимание, можно анализировать только по прошествии feedback loop

Расстояние Кульбака-Лейблера между двумя временными окнами для определения сдвигов – не очень интерпретируемо, можно использовать тест Колмогорова-Смирнова (в целом работает и хоть что-то отлавливает только в одномерном случае)

А как насчитывать метрики?

- окном – меньше шанс срабатывания
- за всю историю (кумулятивно) – выше шанс срабатывания

Не все сдвиги одинаковы, нужно отличать

- внезапные от постепенных
- внешние от временных

При отслеживании факторов важно учитывать диапазон допустимых значений, ожидаемое распределение значений, проблемы: дороговизна – следить за многими факторами для большого количества моделей дорого и затратно по памяти, alert fatigue, ... (?)

Первый и главный инструмент – логи, второй по полезности (но более удобный) – дашборды

Решение проблем с распределением данных

- покрываем датасетом все возможные распределения
- создаем новую модель на новом распределении

Пример встраивания мониторинга в случае Amazon Alexa: есть голосовой ассистент (колонка), которая работает по этапам

- wake word recognition – свертка

- automatic speech recognition – почти whisper
- natural language understanding (скорее всего нормализация и regexp) – множество компонент по доменам, из которых алгоритмом ранжирования выбирается наилучший (легко встраивать новые сценарии)
- text-to-speech
- Alexa voice service

Оффлайн-модели для определения ошибок

В архитектуре с доменными компонентами есть проблема false routing – можем отправить дальше данные не из того домена

Такое тяжело отлавливать, как минимум, потому что каждый домен – это некоторая доля, а ошибок в этой доле еще меньше

Другая проблема: ошибки в диалоговой системе

- False Wakes
- ASR Error
- NLU Errors
- ERR – Incorrect entity recognition and resolution errors
- Results Errors

Даже когда с точки зрения моделей никто не ошибся, пользователь мог быть не устроен, потому что не соблюден интент