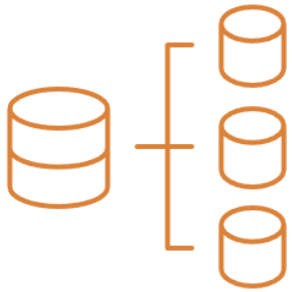
Лекция 02
Данные и признаки: сбор, очистка, валидация, DataOps

Почему качество данных важнее архитектуры модели

От системного взгляда к данным и признакам

Целый продукт

В Лекции 01 мы смотрели на ИИ-систему как на целый продукт: роли, жизненный цикл, артефакты.



Данные на каждом этапе

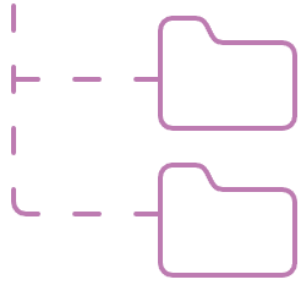
Мы увидели, что данные присутствуют почти на каждом этапе: от постановки задачи до эксплуатации.

Данные и признаки

Сегодня мы «приближаем камеру» к данным и признакам как к отдельным инженерным сущностям.

Формальное качество

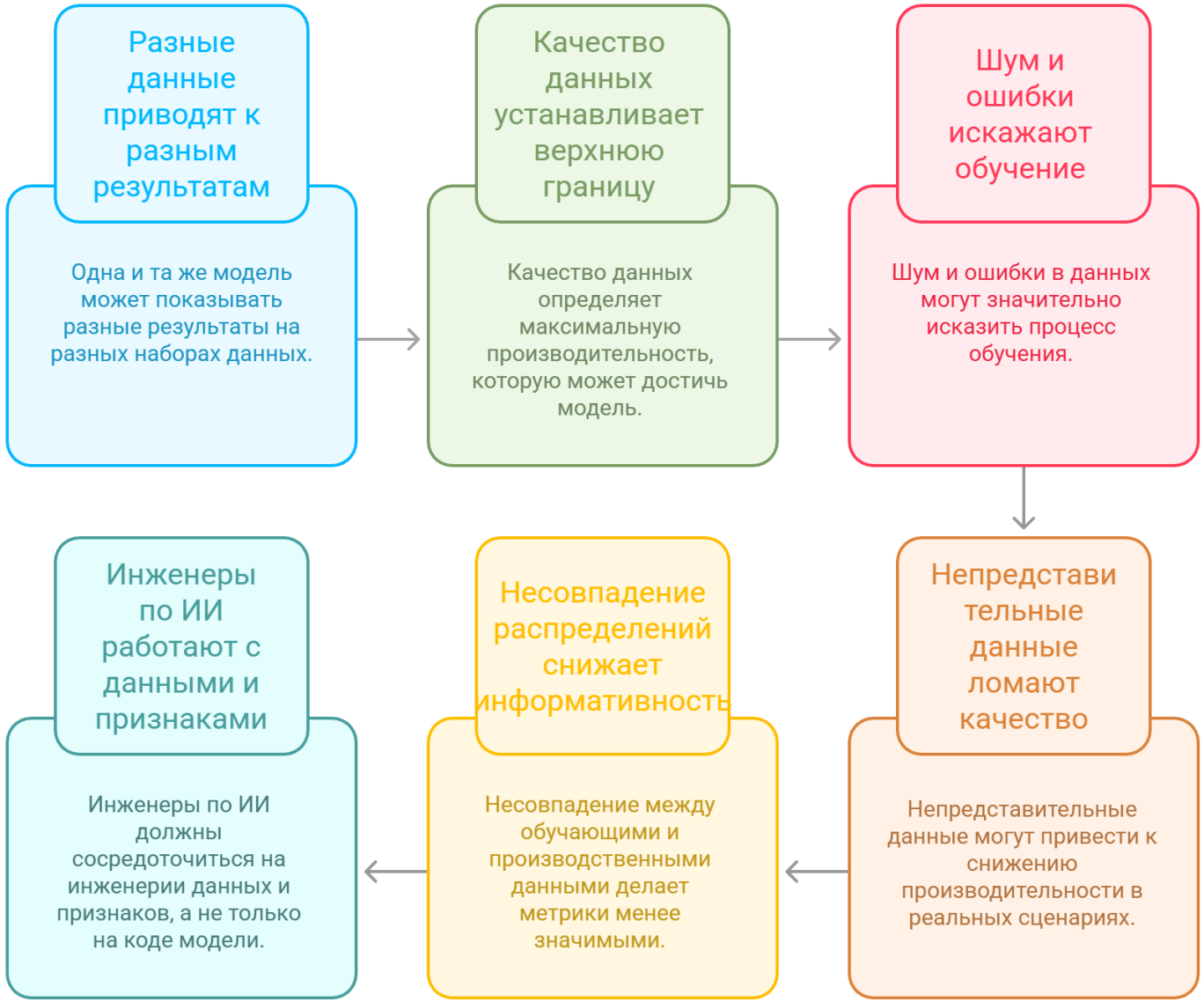
Цель: научиться говорить о качестве данных и признаков формально, а не в стиле «кажется, всё норм».



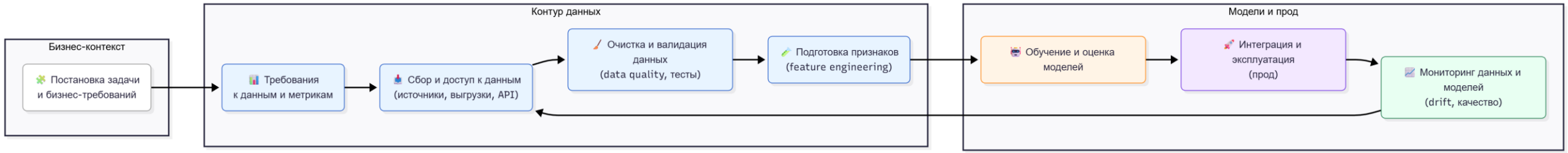
Репозиторий данных

Результат: вы будете понимать, что именно должно появиться в репозитории по данным и признакам к началу работы над моделями.

Данные задают потолок качества модели



Жизненный цикл ИИ-системы с фокусом на данных и признаках



- Данные и признаки появятся **до обучения** и остаются важны **после вывода в прод**.
- Между источниками и моделью всегда есть этапы **очистки, валидации и feature engineering**.
- Мониторинг в проде – это не только про метрики модели, но и про **качество и распределения данных**.
- В нашем курсе мы будем постепенно заполнять артефактами каждый из подсвеченных блоков.

Основные типы источников данных для ИИ-задач

Источник данных	Описание
Транзакционные системы (OLTP-БД)	Оперативные записи о действиях пользователей и бизнес-событиях
Аналитические хранилища и витрины (DWH, marts)	Агрегированные и очищенные данные «для аналитики»
Логи и потоки событий	Технические и бизнес-события: клики, запросы, метрики, телеметрия
Файловые выгрузки и обмены	CSV/Excel/Parquet, загружаемые вручную или по расписанию
Внешние API и сторонние сервисы	Геоданные, погода, курсы, контрагенты, открытые датасеты
Стриминговые платформы (Kafka, очереди и т.п.)	Непрерывные потоки данных для онлайн-обработки

Мини-кейс: какие данные есть у онлайн-сервиса такси

Заказы и поездки

Информация о заказах и поездках, включая статус и стоимость.

Пользователи и водители

Профили пользователей и водителей, рейтинги и история поездок.

Геоданные и дорожная обстановка

Координаты, маршруты, пробки и зоны повышенного спроса.

Платежи и биллинг

Суммы, комиссии, отмены, возвраты, промокоды и мошенничество.

Технические логи и телеметрия

Версии приложения, тип устройства, ошибки, задержки и качество связи.

Сравнение источников данных по их свойствам

Источник	Структура	Частота/скорость	Типичный объём	Плюсы для ML	Минусы/риски для ML
OLTP-БД	Нормализованная	Высокая, мелкие транзакции	Средний-высокий	Детальные события, бизнес-семантика	Сложные джойны, нагрузка на прод
DWH/витрины	Денормализованная	Пакетно (дни/часы)	Высокий	Очищенные данные, готовые агрегаты	Потеря деталей, задержки обновления
Логи/события	Полуструктурированная	Очень высокая, потоковая	Очень высокий	Богатая динамика, временные паттерны	Шум, нестабильный формат, много мусора
Файлы (CSV/Parquet)	Табличная/полуструкт.	Пакетно (ручные/плановые)	От малых до больших	Быстрый старт, простая интеграция	Хрупкость форматов, отсутствие контроля
Внешние API/сервисы	Зависит от провайдера	По запросу, rate limits	Ограниченный	Доп. признаки (погода, карты, справочники)	SLA, лимиты, лицензии, нестабильная доступн.
Стриминговые платформы	Событийная	Непрерывная, низкая задержка	Очень высокий	Онлайн-ML, почти реальное время	Сложность инфраструктуры и отладки

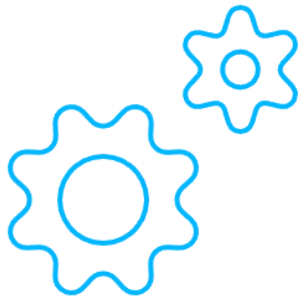
- Разные источники дают **разные компромиссы** между детализацией, стабильностью и скоростью.
- Для ИИ-проекта важно заранее понимать, **что берём «как есть», а что требует тяжёлой подготовки**.

Что такое качество данных в инженерном смысле

Качество данных – это набор измеримых свойств, а не субъективное «норм / не норм».

Полнота

Нужные поля заполнены, важные события не «теряются по дороге».



Целостность

Данные не противоречат сами себе и связаны корректными ключами.

Уникальность

Нет ненужных дублей записей и идентификаторов.

Актуальность

Данные достаточно свежие для задачи, задержки контролируются.

Валидность

Значения лежат в допустимых диапазонах и соответствуют реальному миру.

Измерения качества данных на простых примерах

Измерение	Короткое определение	Пример нарушения
Полнота	Насколько мало важных пропусков и «дырок»	У 20% заказов отсутствует способ оплаты
Целостность	Нет ли логических противоречий и «битых» связей	В платежах есть user_id, которого нет в таблице users
Уникальность	Отсутствие дублей там, где ждём уникальность	Один и тот же заказ записан в таблице дважды
Актуальность	Насколько данные своевременны для задачи	Витрина обновляется раз в неделю, а прогноз нужен по часам
Валидность	Значения попадают в допустимые диапазоны и справочники	Возраст = −5, валюта = «RUBLES» вместо кода «RUB»
Точность	Насколько данные совпадают с реальностью	В системе указан адрес офиса, который уже переехал

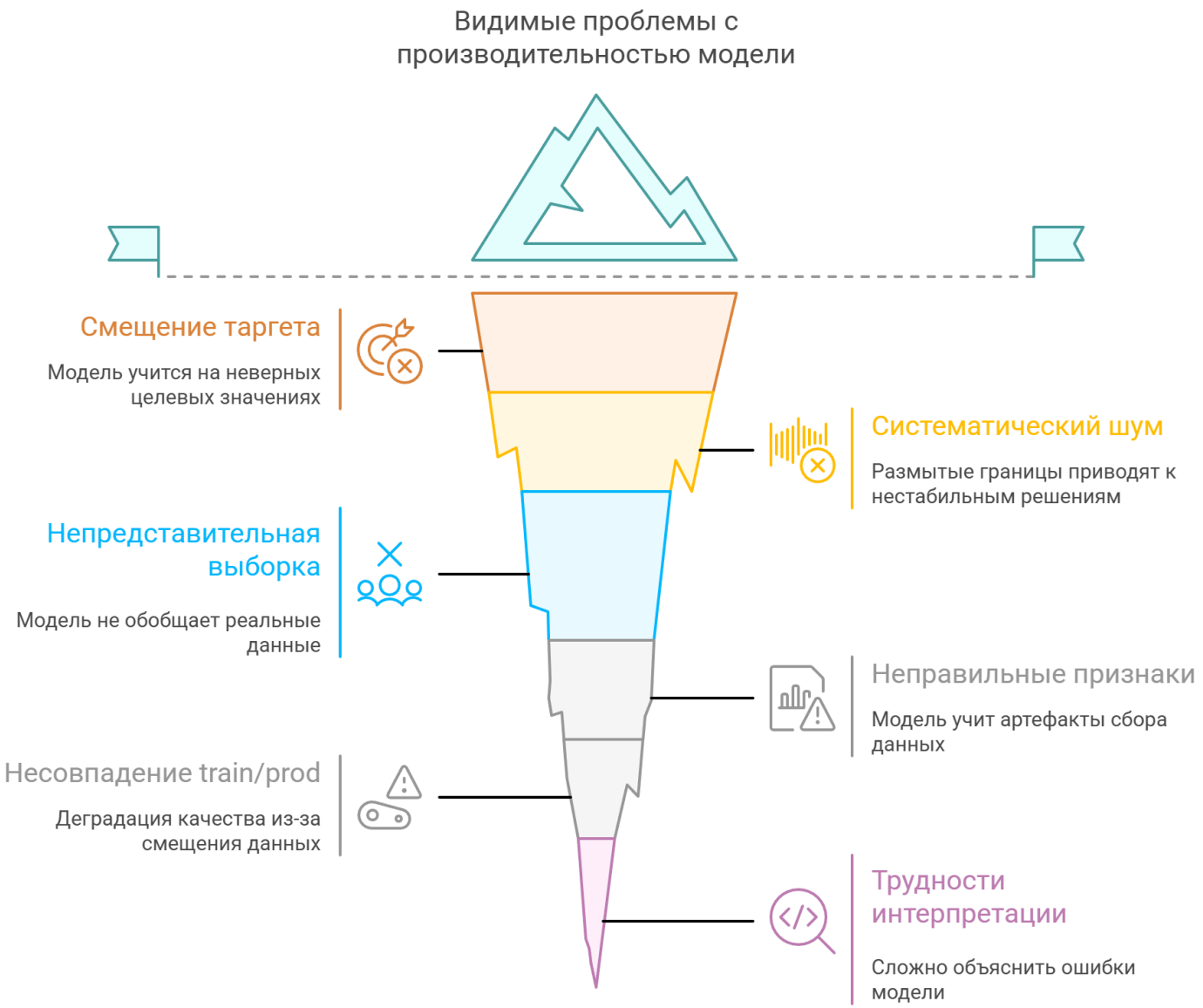
Эти измерения можно превращать в **конкретные правила и тесты** для датасета!

Пример «грязного» датафрейма и его проблемы

order_id	user_id	age	amount	status	created_at
101	u_001	25	1200.00	paid	2025-09-01 10:05:00
101	u_001	25	1200.00	paid	2025-09-01 10:05:00
102	*NULL*	40	*NULL*	paid	2025-09-01 10:10:00
103	u_003	-5	500000.0	paid	2027-01-01 00:00:00
104	u_004	30	N/A	???	2025-13-40 99:99:00

- **Дублирующая строка по order_id=101** (нарушение уникальности).
- **user_id и amount = NULL у заказа 102** (проблема полноты).
- **age = -5 и слишком большой amount = 500000.0** (проблемы валидности и, возможно, точности).
- **Статус ??? и сумма N/A у заказа 104** (невалидные категориальные и числовые значения).
- **Дата 2025-13-40 99:99:00 и будущее 2027-01-01** (нарушения валидности дат и временной логики).

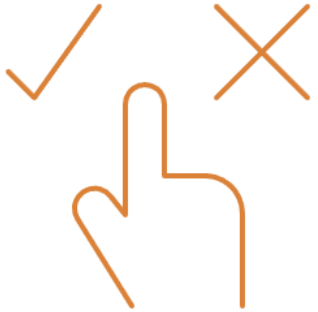
Как дефекты данных ломают обучение и инференс



Уровни проверки данных: от «на глаз» до автоматизации

Уровень 1

Быстрый визуальный осмотр данных, разовые графики и выборки. Выводы нигде формально не фиксируются.



Уровень 2

Есть список «что посмотреть» (пропуски, дубли, диапазоны), но проверки выполняются вручную перед проектом.

Уровень 3

Набор скриптов/ функций или правил, которые запускаются автоматически при обновлении датасета или перед обучением.

Уровень 4

Проверки встроены в конвейер данных, есть пороги, алерты, отчёты; отслеживается качество и в обучении, и в проде.

Простейшие проверки качества в Pandas-подобном коде


```
import pandas as pd

df = pd.read_csv("rides.csv")

problems = {}
problems["dup_orders"]          = df["order_id"].duplicated().sum()
problems["null_user_or_amount"] = df["user_id"].isna().sum() + df["amount"].isna().sum()
problems["invalid_age"]         = ((df["age"] < 0) | (df["age"] > 110)).sum()
problems["invalid_amount"]      = ((df["amount"] <= 0) | (df["amount"] > 100000)).sum()

allowed_status = {"created", "paid", "cancelled"}
problems["invalid_status"]      = (~df["status"].isin(allowed_status)).sum()

report = pd.Series(problems, name="n_bad_rows")
print(report)
```

- Читаем данные в df и заводим словарь problems для счётчиков нарушений.
- Считаем дубли по order_id и пропуски в ключевых полях user_id и amount.
- Проверяем валидность числовых полей age и amount по простым диапазонам.
- Проверяем, что status принадлежит допустимому множеству; печатаем отчёт для анализа.

«Validation as code»: проверки как полноценный артефакт

Проверки качества данных

Проверки качества данных становятся кодом, а не разовыми действиями в голове или в блокноте.

Хранение кода проверок

Код проверок хранится в репозитории рядом с пайплайнами данных и моделями, проходит ревью.

Результаты валидации как артефакт

Результаты валидации – это артефакт: отчёты, логи, метрики, по которым можно судить о здоровье данных.

Явные правила валидации

Правила валидации описываются явно: диапазоны, уникальность, обязательные поля, допустимые категории.

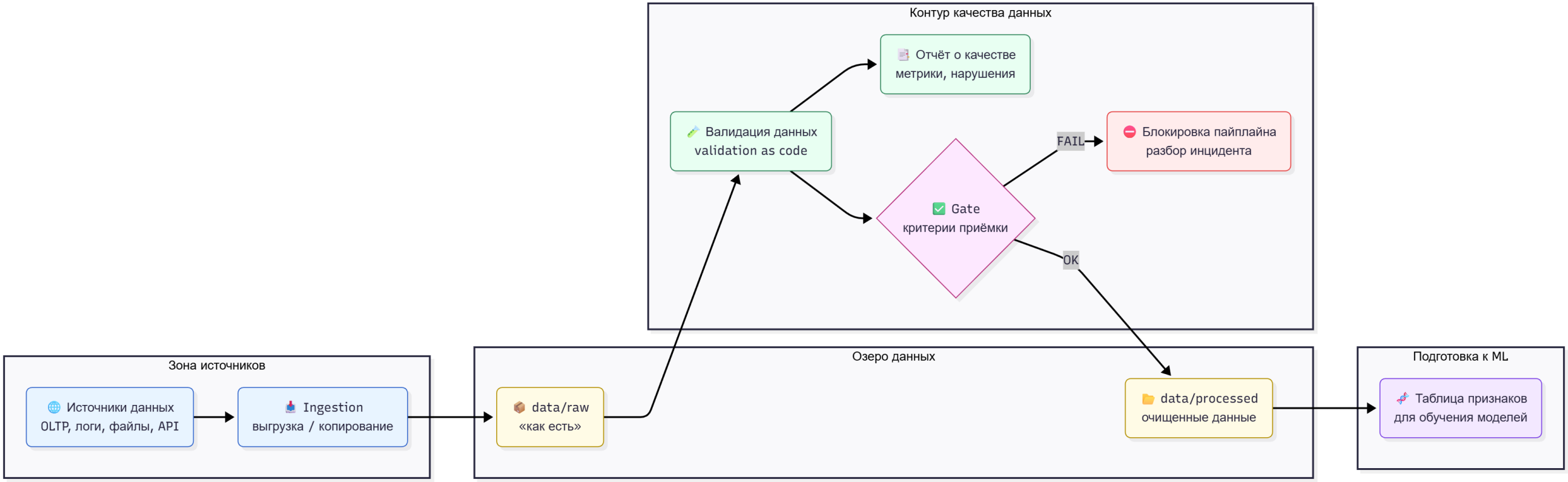
Автоматический запуск проверок

Проверки запускаются автоматически: при загрузке данных, перед обучением, по расписанию.

Фундамент DataOps / MLOps

Подход «validation as code» – фундамент DataOps / MLOps: без него доверять моделям нельзя.

Пайплайн проверки и приёмки данных перед обучением



- Сырые данные сначала **сохраняются** в data/raw, чтобы всегда знать, «что нам реально пришло».
- Поверх raw запускаются **автоматические проверки**, результаты фиксируются в отчётах.
- Решение «пропускать/не пропускать» формализовано в виде **gate с критериями приёмки**.
- Только прошедшие приёмку данные попадают в data/processed и далее в **таблицу признаков для моделей**.

DataOps: инженерный подход к работе с данными

Определение

Набор практик и культурных привычек для надёжной и предсказуемой доставки данных «как продукта».

Основные принципы

Пайплайны как код, версионирование, тесты, мониторинг, прозрачные SLA на данные.

Фундамент для ИИ

Без стабильных пайплайнов и контроля качества данных MLOps не взлетает.

Аналогия с DevOps

Те же идеи – автоматизация, непрерывность, совместная работа Dev/OPS, только фокус на данных.

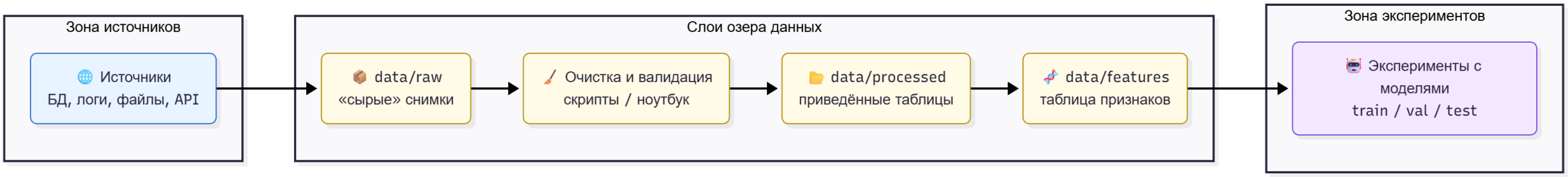
Связанные роли

DataOps связывает владельцев источников, дата-инженеров, аналитиков, ML-инженеров в один цикл.

Уровень курса

Мы не строим платформу, но думаем о своих датасетах как о маленьком продукте DataOps-уровня.

Мини-пайплайн данных до таблицы признаков



- data/raw – **немодифицируемые снимки** исходных данных: что пришло «снаружи» в этот день/запуск.
- Блок очистки/валидации – **код** (скрипт/ноутбук) с проверками и явными шагами трансформаций.
- data/processed – **очищенные и согласованные** таблицы, удобные для агрегаций и джойнов.
- data/features – **стабильная таблица признаков** с понятной схемой для обучения моделей.
- Эксперименты с моделями опираются **только** на data/features, а не напрямую на «сырой хаос».

Базовые понятия: объект, признаки и таргет

Термин	Определение	Пример
Объект	Единица предсказания	Заказ такси
Признаки	Набор измеримых свойств	Маршрут, время, рейтинг
Таргет	Целевая переменная	Время подачи или цена

В табличных данных: **строка = объект**, **колонка = признак**, отдельная колонка/метка = таргет.

Признаки ≠ необработанные поля: они могут быть рассчитаны, агрегированы, преобразованы.

Типы признаков и их представление в данных

Характеристика	Значения	Примеры
Числовые	Целые и вещественные	Сумма, возраст, расстояние
Бинарные	Да/нет, 0/1	Есть промокод, пользователь активен
Категориальные	Значения из конечного набора	Город, тип тарифа, способ оплаты
Временные	Даты/времена и функции	Час суток, день недели
Текстовые	Короткие тексты/ теги/описания	Отзывы, категории, заголовки
Счётчики и агрегаты	Количество событий, средние/ максимумы	По пользователю, по объекту, по району

Примеры признаков для кейса онлайн-такси

Признаки маршрута

Прямое расстояние, длина маршрута по дороге, время в пути по историческим данным.

Признаки клиента

Стаж в сервисе, число поездок за N дней, доля отмен, средний чек, средний рейтинг.

Признаки окружения

Плотность заказов в районе, наличие пробок, погода, спец-события (концерты, праздники).

Временные признаки

Час суток, день недели, выходной/будний, сезон, «сколько времени до/после час-пика».

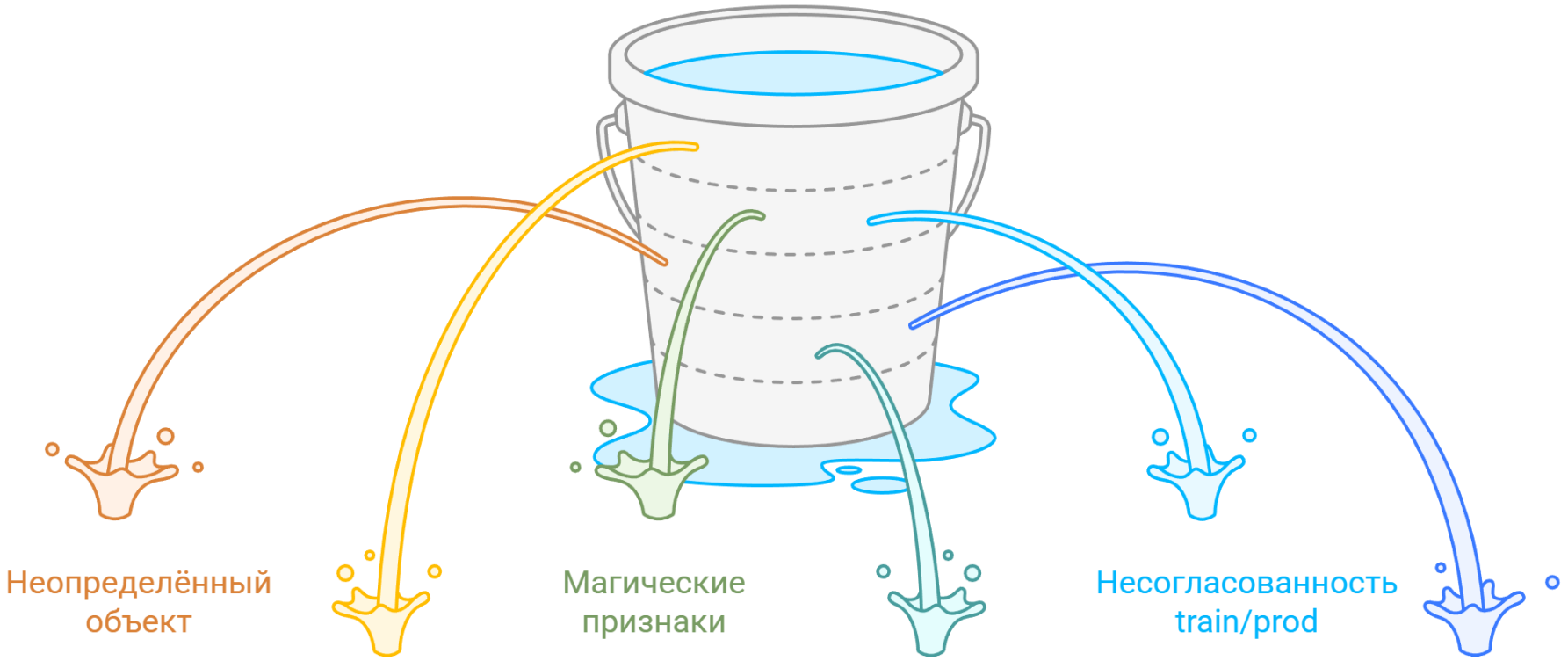
Признаки водителя/авто

Стаж в системе, класс автомобиля, средний рейтинг, доля отмен, среднее время подачи.

Источники признаков

Заказы и поездки, профили пользователей и водителей, геосервисы, биллинг, внешние API.

Типичные ошибки в работе с признаками



Неопределённый объект

Признаки и таргет на разных уровнях

Утечки данных

Информация из будущего загрязняет признаки

Магические признаки

Случайные комбинации колонок без смысла

Слишком сложные признаки

Хитрые трансформации вместо простых

Несогласованность train/prod

Разные вычисления признаков в офлайне и проде

Отсутствие документации

Неизвестно, как вычисляются признаки

Карточка признака: что обязательно зафиксировать

Имя признака

Машинопонятное имя признака без магии.

Смысл на человеческом

Описание признака на простом человеческом языке.

Временной срез

На какой момент времени признак вычисляется.

Ограничения и проверки

Ожидаемый диапазон, допустимые значения, как ловим аномалии.

Уровень объекта

Уровень объекта, с которым будет строка в data/features.

Формула / алгоритм

Как признак считается из сырых данных.

Источник данных

Из каких таблиц/ сервисов берутся сырые поля.

Как оформить данные и признаки в репозитории проекта

Папка	Описание
data/raw/	Сырые выгрузки и снимки
data/processed/	Очищенные и приведённые таблицы
data/features/	Итоговые таблицы признаков
notebooks/	EDA и прототипирование
features/	Код для расчёта признаков
docs/	Короткие описания

Итоги лекции: данные, качество, признаки, DataOps

Что читать и смотреть дальше про данные и DataOps

1. Качество данных и надёжные пайплайны
- **Книга**: Barr Moses, Lior Gavish, Molly Vorwerck – *Data Quality Fundamentals: A Practitioner's Guide to Building Trustworthy Data Pipelines*.
- **Статья/манифест**: Data Quality / Data Observability blog от Monte Carlo.

2. Feature engineering и работа с признаками
- **Книга**: Max Kuhn, Kjell Johnson – *Feature Engineering and Selection: A Practical Approach for Predictive Models*, CRC Press.
- **Обзоры/статьи**: Обзор *Feature Engineering and Selection* на Machine Learning Mastery.

3. DataOps, MLOps и долговечность ML-систем
- **Классическая статья**: D. Sculley et al. – *Hidden Technical Debt in Machine Learning Systems*.
- **Практика DataOps**: DataOps Manifesto и сопутствующие материалы.

4. Структура проектов и пайплайнов (репозитории, data/...)
- **Шаблон проекта**: *Cookiecutter Data Science* (DrivenData).
- **Руководства**: документация Cookiecutter Data Science и статьи «How to Structure Your Data Science Projects».

5. Практика Python / Pandas / EDA (для закрепления)
- **Документация**: Pandas User Guide – разделы по работе с пропусками, типами данных и индексацией (активно обновляется).
- **Книга**: Wes McKinney – *Python for Data Analysis*.

Q&A и контакты

Группа по дисциплине:

https://t.me/+8dShF1tFSDg0ZmJi