АО «АЛМАТИНСКИЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Факультет инженерных и информационных технологий

Кафедра компьютерной инженерии

ПРОЕКТИРОВАНИЕ СИСТЕМ ИСКУССТВЕННОГО ИНТЕЛЛЕКТА

ЛЕКЦИЯ № 2

Постановка AI-задачи и требования к системе

Теоретико-практический конспект

Составитель: Алтыбай А.

Кафедра компьютерной инженерии

Алматы, 2026

Аннотация и назначение лекции

Лекция 2 раскрывает тему «Постановка AI-задачи и требования к системе» в контексте разработки промышленной AI-системы. Материал опирается на тематический план дисциплины и соединяет определения, принципы проектирования, математические оценки, проектные артефакты, практический код и аргументированный анализ рисков. Вопросы построены вокруг решений, которые предстоит принимать инженеру, а не только вокруг названий алгоритмов. [1, 2]

План занятия

1. Формализация бизнес-проблемы
2. Единица предсказания
3. Целевая переменная и метки
4. Функциональные требования
5. Нефункциональные требования
6. Базовый сценарий без ML
7. Матрица стоимости ошибок
8. Критерии приемки
9. Практические сценарии, контроль качества и проектные задания.

Ожидаемые результаты обучения

По завершении студент способен:

1. объяснить и аргументированно применить принцип «Формализация бизнес-проблемы» к инженерному решению;
2. объяснить и аргументированно применить принцип «Единица предсказания» к инженерному решению;

3. объяснить и аргументированно применить принцип «Целевая переменная и метки» к инженерному решению;
4. объяснить и аргументированно применить принцип «Функциональные требования» к инженерному решению;
5. объяснить и аргументированно применить принцип «Нефункциональные требования» к инженерному решению;
6. объяснить и аргументированно применить принцип «Базовый сценарий без ML» к инженерному решению;
7. реализовать упрощённый проверяемый прототип и сформулировать критерии его приемки.
8. указать угрозы достоверности результатов и предложить проверки до эксплуатации.

Сквозной кейс. В течение лекции рассматриваем университетскую службу поддержки. Существенные условия: пропуски критических обращений, избыточные эскалации. Основная проверяемая характеристика: среднее время реакции. Числа в расчётных примерах учебные и не являются результатами реальных измерений.

1. Базовые теоретические положения из конспекта курса

Ниже представлены исходные положения учебно-методического конспекта, связанные с данной темой. Следующие разделы раскрывают их с инженерными деталями и проверяемыми примерами. [1, 2] Цель лекции. научить переводить прикладную проблему в формализованную AI/ML-задачу, определять функциональные и нефункциональные требования, метрики успеха, ограничения и критерии приемки.

Ключевые понятия. problem framing, business objective, ML objective, KPI, functional requirements, non-functional requirements, SLA/SLO, acceptance criteria, baseline.

1.1 От бизнес-проблемы к ML-задаче

Проектирование начинается не с выбора нейронной сети, а с определения решения, которое должно быть принято на основе прогноза. Формулировка «предсказать отток» недостаточна. Следует уточнить: кого считать уходящим клиентом, за какой горизонт, какое действие предпринимается после прогноза, какая стоимость ложного срабатывания и пропуска.

Полезно отделять бизнес-цель от ML-цели. Бизнес-цель может заключаться в снижении потерь на 5%, а ML-цель — в построении ранжирования клиентов с высокой precision среди верхних 10% риска.

1.2 Функциональные требования

Функциональные требования описывают, что система должна делать: принимать определенный формат запроса, выдавать прогноз и объяснение, поддерживать пакетную обработку, сохранять результат, предоставлять API, выполнять ручное подтверждение и т.д. Эти требования определяют границы продукта и интерфейсы между компонентами.

Для AI-компонента необходимо также фиксировать правила поведения при отсутствии данных, выходе значения за допустимый диапазон или недоступности модели.

1.3 Нефункциональные требования

Нефункциональные требования определяют качество функционирования системы. К ним относятся latency, throughput, availability, масштабируемость, безопасность, воспроизводимость, приватность, интерпретируемость, стоимость, доступность GPU и время восстановления.

Например, модель с F1=0.94 может быть непригодна для real-time продукта, если отвечает 3 секунды при требовании 100 мс. Поэтому ML-метрики всегда рассматриваются вместе с системными ограничениями.

- Latency — время ответа на один запрос.
- Throughput — количество запросов или объектов в единицу времени.

- Availability — доля времени, когда сервис доступен.
- SLA/SLO — договорные или внутренние целевые уровни сервиса.

1.4 Метрики успеха и критерии приемки

Метрики должны быть связаны с решением. Для классификации это не обязательно accuracy: при сильном дисбалансе классов важнее precision, recall, PR-AUC или cost-weighted metric. Для ранжирования могут использоваться NDCG и Recall@K; для генеративной системы — task success, groundedness, доля корректных ссылок, human evaluation и стоимость ответа.

Критерий приемки полезно записывать как измеримое условие: «Recall критических случаев не ниже 0.92 при precision не ниже 0.70, p95 latency не более 200 мс, стоимость 1000 запросов не выше X». Это позволяет принимать инженерные решения на основе требований, а не субъективного впечатления.

1.5 Baseline и стратегия доказательства ценности

До сложной модели следует определить baseline: бизнес-правило, простую статистическую модель или текущий процесс. Baseline дает точку сравнения и помогает понять, оправдана ли сложность ML. В некоторых задачах правило «последнее значение» или логистическая регрессия могут быть удивительно сильными.

Хорошая постановка включает план проверки гипотезы: какие данные нужны, как будет проведено offline-тестирование, что станет критерием перехода к пилоту и как будет измеряться online-эффект.

1.6 Выводы

- Постановка задачи должна связывать бизнес-решение, ML-цель и системные требования.

▷ Функциональные и нефункциональные требования одинаково важны для production AI.

▷ Метрики выбираются по стоимости ошибок и реальному способу использования прогноза.

▷ Baseline обязателен для доказательства полезности усложнения.

1.7 Контрольные вопросы

▷ Почему формулировка «построить нейронную сеть» не является корректной постановкой задачи?

▷ В чем различие бизнес-метрики и ML-метрики?

▷ Какие нефункциональные требования наиболее важны для real-time AI-сервиса?

▷ Что такое критерий приемки модели?

▷ Зачем строить baseline до сложной модели?

2. Систематическое изложение темы и инженерные решения

2.1 Формализация бизнес-проблемы

Бизнес-владелец формулирует желаемое изменение процесса, а команда AI преобразует его в прогноз или оптимизационную задачу. Сначала следует установить, какое действие зависит от ответа системы. [2]

Проверяемый пример. Для службы поддержки цель «снизить время ожидания» превращается в ранжирование срочности обращений с проверяемым влиянием на SLA.

Вопрос для обсуждения. Как проверить выполнение такого требования: Для службы поддержки цель «снизить время ожидания» превращается в ранжирование срочности обращений с проверяемым влиянием на SLA.

2.2 Единица предсказания

Единицей может быть пользователь, заказ, документ, изображение либо временное окно. Неверная гранулярность приводит к несовместимости данных и решения. [3]

Проверяемый пример. Для оттока клиентов укажите горизонт прогноза 30 дней и момент доступности признаков.

Вопрос для обсуждения. Как проверить выполнение такого требования: Для оттока клиентов укажите горизонт прогноза 30 дней и момент доступности признаков.

2.3 Целевая переменная и метки

Метка должна фиксировать наблюдаемое событие без обращения к будущим признакам. Правило разметки документируют вместе с источником и уровнем согласия экспертов. [4]

Проверяемый пример. Различите фактический уход клиента и «отсутствие визита 2 дня»; проверьте, возможны ли ложные метки.

Вопрос для обсуждения. Как проверить выполнение такого требования: Различите фактический уход клиента и «отсутствие визита 2 дня»; проверьте, возможны ли ложные метки.

2.4 Функциональные требования

Перечисляют форматы, поведение в штатном и нештатном сценариях, пределы размера запроса и семантику ответа. Контракты важнее названия архитектуры нейронной сети. [5]

Проверяемый пример. Определите JSON-схему входа, ответ со score и model_version и ошибки HTTP 422/503.

Вопрос для обсуждения. Как проверить выполнение такого требования:

Определите JSON-схему входа, ответ со score и model_version и ошибки HTTP 422/503.

2.5 Нефункциональные требования

Latency, throughput, availability, privacy, reliability и maintainability измеряются отдельно от accuracy. Приемка должна быть многокритериальной. [6]

Проверяемый пример. Для 200 запросов/с оцените p95 задержки и допустимую долю ошибок по наблюдениям нагрузочного теста.

Вопрос для обсуждения. Как проверить выполнение такого требования: Для 200 запросов/с оцените p95 задержки и допустимую долю ошибок по наблюдениям нагрузочного теста.

2.6 Базовый сценарий без ML

Эвристика, константный прогноз и правила являются обязательным baseline. Если AI не улучшает полезность относительно простого решения, внедрение может быть неоправданным. [1]

Проверяемый пример. Сравните ранжирование FIFO и ML-приоритизацию на одних и тех же исторических обращениях.

Вопрос для обсуждения. Как проверить выполнение такого требования: Сравните ранжирование FIFO и ML-приоритизацию на одних и тех же исторических обращениях.

2.7 Матрица стоимости ошибок

False positive и false negative имеют различные последствия. Фиксированный порог 0.5 не обязательно минимизирует бизнес-потери. [2]

Проверяемый пример. Рассчитайте ожидаемую стоимость FN=100 условных единиц и FP=5; обсудите выбор порога.

Вопрос для обсуждения. Как проверить выполнение такого требования: Рассчитайте ожидаемую стоимость FN=100 условных единиц и FP=5; обсудите выбор порога.

2.8 Критерии приемки

Критерии содержат измеритель, порог, тестовое множество, нагрузку и ограничения. Формулировки должны допускать однозначное «принято/не принято». [3]

Проверяемый пример. Впишите в таблицу приемки macro-F1, p95 latency, rate ошибок, правила хранения персональных данных.

Вопрос для обсуждения. Как проверить выполнение такого требования: Впишите в таблицу приемки macro-F1, p95 latency, rate ошибок, правила хранения персональных данных.

2.9 Ограничения проекта

Правовые, ресурсные и временные ограничения ограничивают область допустимых решений. Ограничение по разметке может оказаться существеннее ограничений GPU. [4]

Проверяемый пример. Для пилота на одной неделе выберите минимальный измеримый объем и явно перечислите исключенные функции.

Вопрос для обсуждения. Как проверить выполнение такого требования: Для пилота на одной неделе выберите минимальный измеримый объем и явно перечислите исключенные функции.

2.10 Стейкхолдеры и ответственность

Операторы, аналитики, клиенты, инженеры и специалисты по безопасности по-разному понимают успех. Конфликты ожиданий проясняют до программирования. [5]

Проверяемый пример. Составьте user stories и обозначьте ответственного за ложное автоматическое решение.

Вопрос для обсуждения. Как проверить выполнение такого требования: Составьте user stories и обозначьте ответственного за ложное автоматическое решение.

2.11 Декомпозиция KPI

Бизнес-метрики связаны с ML-метриками лишь опосредованно, поэтому понадобится контроль процесса. Изменение F1 не гарантирует улучшения пользовательского опыта. [6]

Проверяемый пример. Нарисуйте причинную схему: качество обнаружения срочности -> порядок обработки -> среднее время решения.

Вопрос для обсуждения. Как проверить выполнение такого требования: Нарисуйте причинную схему: качество обнаружения срочности -> порядок обработки -> среднее время решения.

2.12 План экспериментальной проверки

План должен разделять offline-проверку и пилот; указывать, на каких данных и при каких условиях результат считается убедительным. [1]

Проверяемый пример. Определите тестовую выборку по времени и A/B-метрику для оценки влияния нового маршрутизатора.

Вопрос для обсуждения. Как проверить выполнение такого требования:

Определите тестовую выборку по времени и A/B-метрику для оценки влияния нового маршрутизатора.

3. Математические инструменты и количественные критерии

Для перехода от неформального решения к проверяемой постановке введём следующие величины.

$$R(a \mid x) = \sum_{y} C(a, y)\, P(Y = y \mid X = x) \tag{1}$$

$$a^*(x) = \arg\min_{a \in \mathcal{A}} R(a \mid x) \tag{2}$$

Операционное действие a выбирают по условному ожидаемому ущербу, а не обязательно по максимальной вероятности класса.

При выборе числового критерия нужно отличать обучающую функцию потерь, продуктовую ценность и эксплуатационные ограничения. Первую можно оптимизировать дифференцированием, вторая определяется процессом принятия решения, а третьи проверяются нагрузочными испытаниями, политиками безопасности и мониторингом.

3.1 Пошаговый расчёт на учебных данных

Рассмотрим две конфигурации одной системы. Вариант A имеет условную оценку качества $q_A = 0.86$ и задержку $t_A = 90$ мс; вариант B — $q_B = 0.89$ и $t_B = 280$ мс. Предположим, что требование заказчика задаёт $q \geq 0.85$ и $p95 \leq 150$ мс. Тогда A удовлетворяет обоим условиям, тогда как B нарушает второе. Следовательно, выбор B только потому, что его offline-метрика выше, не является обоснованным. Числа иллюстративны; реальный вывод требует измерений при одинаковой нагрузке и одинаковом наборе данных.

$$\begin{aligned} \text{accept}(A) &= \mathbf{1}\{0.86 \geq 0.85 \wedge 90 \leq 150\} = 1, \\ \text{accept}(B) &= \mathbf{1}\{0.89 \geq 0.85 \wedge 280 \leq 150\} = 0. \end{aligned}$$

3.2 Как интерпретировать число, а не только вычислить его

Сначала следует проверить единицы измерения, состав выборки и смысл агрегирования. Например, p95 на спокойном стенде нельзя переносить на рабочую нагрузку с иным числом одновременных запросов. Затем исследуют чувствительность к изменениям входных данных и неопределённость оценок. Наконец, фиксируют последствия порога: какую ошибку он предотвращает, а какую может увеличить. Этот подход делает количественный критерий частью инженерной аргументации, а не украшением отчёта. [1, 2]

4. Практическое программирование на Python

Пример предназначен для демонстрации ключевого инженерного принципа. Он не претендует на обработку реального production-трафика. Запускать код следует в изолированном окружении и фиксировать версии библиотек, если они используются. Вывод зависит от указанных входных данных; для стохастических алгоритмов он может меняться.

4.1 Основной воспроизводимый пример

```python
from dataclasses import dataclass
@dataclass
class Requirements:
    min_f1: float = 0.82
    max_p95_ms: float = 200.0
    max_error_rate: float = 0.01

def accepted(metrics, req=Requirements()):
    return (metrics["f1"] >= req.min_f1 and
            metrics["p95_ms"] <= req.max_p95_ms and
            metrics["error_rate"] <= req.max_error_rate)

example = {"f1": 0.85, "p95_ms": 180.0, "error_rate": 0.004}
print("release accepted:", accepted(example))
```

Проверка результатов. Сначала убедитесь, что код завершился без исключений; проверьте формы массивов, схему результата и отсутствие недопустимых значений. Если используются внешние данные, включите в отчёт способ их получения и идентификатор версии. Измените один входной параметр и объясните, почему наблюдаемый результат изменился или остался прежним.

4.2 Компактная проверка контракта или метрики

```python
def expected_cost(fp, fn, fp_price=5, fn_price=100):
    return fp * fp_price + fn * fn_price
```

```
print("loss:", expected_cost(fp=5, fn=2))
```

При самостоятельном повторении предложите минимум один негативный тест: некорректное значение, пустой вход, несовместимую версию схемы либо нарушение SLO. Переход от успешного демонстрационного запуска к надежной системе требует именно таких проверок.

5. Разбор трёх инженерных ситуаций

5.1 Ситуация 1: Формализация бизнес-проблемы

Заказчик использует университетскую службу поддержки. Новое требование заставляет рассмотреть принцип «Формализация бизнес-проблемы». Основная неопределённость состоит в том, как одновременно обеспечить среднее время реакции, не допустить избыточные эскалации и сохранить возможность изменения модели.

Постановка. Бизнес-владелец формулирует желаемое изменение процесса, а команда AI преобразует его в прогноз или оптимизационную задачу. Сначала следует установить, какое действие зависит от ответа системы.

Действие инженера. Для службы поддержки цель «снизить время ожидания» превращается в ранжирование срочности обращений с проверяемым влиянием на SLA.

Альтернативы. Сравните три подхода: (A) простое правило без ML, (B) модульное ML-решение с отдельной проверкой входных данных и (C) интегрированный сервис со сложной моделью. Для каждого укажите потребность в данных, ожидаемые инженерные риски, возможность тестирования и минимальную инфраструктуру. Таблица ниже задаёт форму отчёта, а не готовый рейтинг.

Вариант	Данные и зависимости	Проверка	Основной риск
Правила	Фиксированные условия	Набор тест-кейсов	Ограниченная гибкость
ML-модуль	Версионированные признаки	Offline и API tests	Drift и skew
Сложный сервис	Несколько компонентов	End-to-end и нагрузка	Каскадный отказ

Проверка решения. Подготовьте две позитивные и две негативные пробы; задайте метрику успеха, базовый сценарий и критерий остановки внедрения. Обсудите, какие допущения останутся непроверенными даже после удачного теста.

5.2 Ситуация 2: Базовый сценарий без ML

Заказчик использует университетскую службу поддержки. Новое требование заставляет рассмотреть принцип «Базовый сценарий без ML». Основная неопределённость состоит в том, как одновременно обеспечить среднее время реакции, не допустить избыточные эскалации и сохранить возможность изменения модели.

Постановка. Эвристика, константный прогноз и правила являются обязательным baseline. Если AI не улучшает полезность относительно простого решения, внедрение может быть неоправданным.

Действие инженера. Сравните ранжирование FIFO и ML-приоритизацию на одних и тех же исторических обращениях.

Альтернативы. Сравните три подхода: (A) простое правило без ML, (B) модульное ML-решение с отдельной проверкой входных данных и (C) интегрированный сервис со сложной моделью. Для каждого укажите потребность в данных, ожидаемые инженерные риски, возможность тестирования и минимальную инфраструктуру. Таблица ниже задаёт форму отчёта, а не готовый рейтинг.

Вариант	Данные и зависимости	Проверка	Основной риск
Правила	Фиксированные условия	Набор тест-кейсов	Ограниченная гибкость
ML-модуль	Версионированные признаки	Offline и API tests	Drift и skew
Сложный сервис	Несколько компонентов	End-to-end и нагрузка	Каскадный отказ

Проверка решения. Подготовьте две позитивные и две негативные пробы; задайте метрику успеха, базовый сценарий и критерий остановки внедрения. Обсудите, какие допущения останутся непроверенными даже после удачного теста.

5.3 Ситуация 3: Декомпозиция KPI

Заказчик использует университетскую службу поддержки. Новое требование заставляет рассмотреть принцип «Декомпозиция KPI». Основная неопределённость состоит в том, как одновременно обеспечить среднее время реакции, не допустить избыточные эскалации и сохранить возможность изменения модели.

Постановка. Бизнес-метрики связаны с ML-метриками лишь опосредованно, поэтому понадобится контроль процесса. Изменение F1 не гарантирует улучшения пользовательского опыта.

Действие инженера. Нарисуйте причинную схему: качество обнаружения срочности -> порядок обработки -> среднее время решения.

Альтернативы. Сравните три подхода: (A) простое правило без ML, (B) модульное ML-решение с отдельной проверкой входных данных и (C) интегрированный сервис со сложной моделью. Для каждого укажите потребность в данных, ожидаемые инженерные риски, возможность тестирования и минимальную инфраструктуру. Таблица ниже задаёт форму отчёта, а не готовый рейтинг.

Вариант	Данные и зависимости	Проверка	Основной риск
Правила	Фиксированные условия	Набор тест-кейсов	Ограниченная гибкость
ML-модуль	Версионированные признаки	Offline и API tests	Drift и skew
Сложный сервис	Несколько компонентов	End-to-end и нагрузка	Каскадный отказ

Проверка решения. Подготовьте две позитивные и две негативные пробы; задайте метрику успеха, базовый сценарий и критерий остановки внедрения. Обсудите, какие допущения останутся непроверенными даже после удачного теста.

6. Самостоятельная работа: индивидуальные проектные задания

Для каждого задания студент создаёт отдельный небольшой артефакт: схему, набор тестов, код либо таблицу сравнения, прикладывает наблюдаемый результат и объясняет ограничения.

1. **Формализация бизнес-проблемы.** Для службы поддержки цель «снизить время ожидания» превращается в ранжирование срочности обращений с проверяемым влиянием на SLA. Сформулируйте проверяемое требование, продемонстрируйте альтернативу и приложите протокол хотя бы одного эксперимента.

2. **Единица предсказания.** Для оттока клиентов укажите горизонт прогноза 30 дней и момент доступности признаков. Сформулируйте проверяемое требование, продемонстрируйте альтернативу и приложите протокол хотя бы одного эксперимента.

3. **Целевая переменная и метки.** Различите фактический уход клиента и «отсутствие визита 2 дня»; проверьте, возможны ли ложные метки. Сформулируйте проверяемое требование, продемонстрируйте альтернативу и приложите протокол хотя бы одного эксперимента.

4. **Функциональные требования.** Определите JSON-схему входа, ответ со score и model_version и ошибки HTTP 422/503. Сформулируйте проверяемое требование, продемонстрируйте альтернативу и приложите протокол хотя бы одного эксперимента.

5. **Нефункциональные требования.** Для 200 запросов/с оцените p95 задержки и допустимую долю ошибок по наблюдениям нагрузочного теста. Сформулируйте проверяемое требование, продемонстрируйте альтернативу и приложите протокол хотя бы одного эксперимента.

6. **Базовый сценарий без ML.** Сравните ранжирование FIFO и ML-приоритизацию на одних и тех же исторических обращениях. Сформулируйте проверяемое требование, продемонстрируйте альтернативу и приложите протокол хотя бы одного эксперимента.

7. **Матрица стоимости ошибок.** Рассчитайте ожидаемую стоимость

FN=100 условных единиц и FP=5; обсудите выбор порога. Сформулируйте проверяемое требование, продемонстрируйте альтернативу и приложите протокол хотя бы одного эксперимента.

8. **Критерии приемки.** Впишите в таблицу приемки macro-F1, p95 latency, rate ошибок, правила хранения персональных данных. Сформулируйте проверяемое требование, продемонстрируйте альтернативу и приложите протокол хотя бы одного эксперимента.

9. **Ограничения проекта.** Для пилота на одной неделе выберите минимальный измеримый объем и явно перечислите исключенные функции. Сформулируйте проверяемое требование, продемонстрируйте альтернативу и приложите протокол хотя бы одного эксперимента.

10. **Стейкхолдеры и ответственность.** Составьте user stories и обозначьте ответственного за ложное автоматическое решение. Сформулируйте проверяемое требование, продемонстрируйте альтернативу и приложите протокол хотя бы одного эксперимента.

11. **Декомпозиция KPI.** Нарисуйте причинную схему: качество обнаружения срочности -> порядок обработки -> среднее время решения. Сформулируйте проверяемое требование, продемонстрируйте альтернативу и приложите протокол хотя бы одного эксперимента.

12. **План экспериментальной проверки.** Определите тестовую выборку по времени и A/B-метрику для оценки влияния нового маршрутизатора. Сформулируйте проверяемое требование, продемонстрируйте альтернативу и приложите протокол хотя бы одного эксперимента.

7. Вопросы для текущего и итогового контроля

1. В чём назначение и ограничения принципа «Формализация бизнес-проблемы»?

2. Как проверить на практике утверждение: «Для службы поддержки цель «снизить время ожидания» превращается в ранжирование срочности обращений с проверяемым влиянием на SLA.»?

3. В чём назначение и ограничения принципа «Единица предсказания»?
4. Как проверить на практике утверждение: «Для оттока клиентов укажите горизонт прогноза 30 дней и момент доступности признаков.»?
5. В чём назначение и ограничения принципа «Целевая переменная и метки»?
6. Как проверить на практике утверждение: «Различите фактический уход клиента и «отсутствие визита 2 дня»; проверьте, возможны ли ложные метки.»?
7. В чём назначение и ограничения принципа «Функциональные требования»?
8. Как проверить на практике утверждение: «Определите JSON-схему входа, ответ со score и model_version и ошибки HTTP 422/503.»?
9. В чём назначение и ограничения принципа «Нефункциональные требования»?
10. Как проверить на практике утверждение: «Для 200 запросов/с оцените p95 задержки и допустимую долю ошибок по наблюдениям нагрузочного теста.»?
11. В чём назначение и ограничения принципа «Базовый сценарий без ML»?
12. Как проверить на практике утверждение: «Сравните ранжирование FIFO и ML-приоритизацию на одних и тех же исторических обращениях.»?
13. В чём назначение и ограничения принципа «Матрица стоимости ошибок»?
14. Как проверить на практике утверждение: «Рассчитайте ожидаемую стоимость FN=100 условных единиц и FP=5; обсудите выбор порога.»?
15. В чём назначение и ограничения принципа «Критерии приемки»?
16. Как проверить на практике утверждение: «Впишите в таблицу приемки macro-F1, p95 latency, rate ошибок, правила хранения персональных данных.»?

17. В чём назначение и ограничения принципа «Ограничения проекта»?
18. Как проверить на практике утверждение: «Для пилота на одной неделе выберите минимальный измеримый объем и явно перечислите исключенные функции.»?
19. В чём назначение и ограничения принципа «Стейкхолдеры и ответственность»?
20. Как проверить на практике утверждение: «Составьте user stories и обозначьте ответственного за ложное автоматическое решение.»?
21. В чём назначение и ограничения принципа «Декомпозиция KPI»?
22. Как проверить на практике утверждение: «Нарисуйте причинную схему: качество обнаружения срочности -> порядок обработки -> среднее время решения.»?
23. В чём назначение и ограничения принципа «План экспериментальной проверки»?
24. Как проверить на практике утверждение: «Определите тестовую выборку по времени и A/B-метрику для оценки влияния нового маршрутизатора.»?
25. Почему высокое offline-качество не гарантирует надежность промышленной AI-системы?
26. Какие артефакты нужны для воспроизводимости и анализа инцидента?

8. Глоссарий ключевых понятий

Формализация бизнес-проблемы
Бизнес-владелец формулирует желаемое изменение процесса, а команда AI преобразует его в прогноз или оптимизационную задачу.

Единица предсказания
Единицей может быть пользователь, заказ, документ, изображение либо временное окно.

Целевая переменная и метки
Метка должна фиксировать наблюдаемое событие без обращения к будущим признакам.

Функциональные требования
Перечисляют форматы, поведение в штатном и нештатном сценариях, пределы размера запроса и семантику ответа.

Нефункциональные требования
Latency, throughput, availability, privacy, reliability и maintainability измеряются отдельно от accuracy.

Базовый сценарий без ML
Эвристика, константный прогноз и правила являются обязательным baseline.

Матрица стоимости ошибок
False positive и false negative имеют различные последствия.

Критерии приемки
Критерии содержат измеритель, порог, тестовое множество, нагрузку и ограничения.

Ограничения проекта
Правовые, ресурсные и временные ограничения ограничивают область допустимых решений.

Стейкхолдеры и ответственность
Операторы, аналитики, клиенты, инженеры и специалисты по безопасности по-разному понимают успех.

Декомпозиция KPI

Бизнес-метрики связаны с ML-метриками лишь опосредованно, поэтому понадобится контроль процесса.

План экспериментальной проверки

План должен разделять offline-проверку и пилот; указывать, на каких данных и при каких условиях результат считается убедительным.

9. Итоги и связь со следующей темой

Представленные методы имеют смысл как части целостного управляемого жизненного цикла. Ключевой результат инженерного проектирования состоит не в максимальном числе компонентов или сложности модели, а в убедительной связи требований, проектных решений, измерений и механизмов сопровождения. Дальнейшее изучение курса развивает эти связи на следующем этапе жизненного цикла.

Список литературы

[1] Huyen C. Designing Machine Learning Systems. O'Reilly Media, 2022. https://www.oreilly.com/library/view/designing-machine-learning/9781098107956/.

[2] Google Developers. Rules of Machine Learning: Best Practices for ML Engineering. https://developers.google.com/machine-learning/guides/rules-of-ml.

[3] Géron A. Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow. 3rd ed., O'Reilly Media, 2022. https://www.oreilly.com/library/view/hands-on-machine-learning/9781098125967/.

[4] Murphy K. P. Probabilistic Machine Learning: An Introduction. MIT Press, 2022. https://probml.github.io/pml-book/.

[5] NIST. AI Risk Management Framework (AI RMF 1.0). NIST AI 100-1, 2023. https://doi.org/10.6028/NIST.AI.100-1.

[6] Sculley D. et al. Hidden Technical Debt in Machine Learning Systems. NeurIPS, 2015. https://research.google/pubs/hidden-technical-debt-in-machine-learning-systems/.