АО «АЛМАТИНСКИЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Факультет инженерных и информационных технологий

Кафедра компьютерной инженерии

ПРОЕКТИРОВАНИЕ СИСТЕМ ИСКУССТВЕННОГО ИНТЕЛЛЕКТА

ЛЕКЦИЯ № 8

Оценка качества AI-моделей и экспериментальный дизайн

Теоретико-практический конспект

Составитель: Алтыбай А.

Кафедра компьютерной инженерии

Алматы, 2026

Аннотация и назначение лекции

Лекция 8 раскрывает тему «Оценка качества AI-моделей и экспериментальный дизайн» 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, 3]

План занятия

1. Зачем нужны метрики
2. Матрица ошибок
3. ROC и PR-анализ
4. Регрессия
5. Cross-validation
6. Интервалы неопределенности
7. Калибровка и выбор порога
8. Error analysis
9. Практические сценарии, контроль качества и проектные задания.

Ожидаемые результаты обучения

По завершении студент способен:

1. объяснить и аргументированно применить принцип «Зачем нужны метрики» к инженерному решению;
2. объяснить и аргументированно применить принцип «Матрица ошибок» к инженерному решению;

3. объяснить и аргументированно применить принцип «ROC и PR-анализ» к инженерному решению;
4. объяснить и аргументированно применить принцип «Регрессия» к инженерному решению;
5. объяснить и аргументированно применить принцип «Cross-validation» к инженерному решению;
6. объяснить и аргументированно применить принцип «Интервалы неопределенности» 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 сервис ранжирования заявок. Существенные условия: дисбаланс редких событий, доверие к средней метрике. Основная проверяемая характеристика: precision, recall и стоимость ошибок. 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, 3] Цель лекции. систематизировать offline и online оценивание, выбор метрик, error analysis, калибровку, A/B-тестирование и экспериментальный дизайн.

Ключевые понятия. offline evaluation, online evaluation, precision, recall, F1, ROC-AUC, PR-AUC, calibration, error analysis, A/B test, statistical significance.

1.1 Метрика должна соответствовать решению

Accuracy удобна, но может скрывать проблему при дисбалансе классов. Precision отвечает на вопрос, какая доля положительных прогнозов верна; recall — какая доля реальных положительных случаев найдена. F1 балансирует precision и recall, но не учитывает явную стоимость ошибок.

В ранжировании важны метрики top-K, а в генерации — комбинация автоматических и human/LLM-assisted оценок. Нет универсальной метрики качества для всех AI-систем.

1.2 Threshold и стоимость ошибок

Многие классификаторы выдают score/probability, а бинарное решение формируется порогом. Изменяя threshold, можно управлять компромиссом precision-recall. Порог следует выбирать по операционной стоимости ошибок и пропускной способности downstream-процесса.

Например, если ручная проверка может обработать только 500 объектов в день, разумно оптимизировать качество top-500, а не глобальную accuracy.

1.3 Калибровка

Калиброванная вероятность означает, что среди объектов с прогнозом около 0.8 событие происходит примерно в 80% случаев. Это особенно важно, когда score используется для risk-based decision. Сильная дискриминация не гарантирует хорошую калибровку.

Калибровку можно оценивать reliability diagrams и Brier score, а улучшать Platt scaling или isotonic regression.

1.4 Error analysis

После общей метрики необходимо изучать ошибки по сегментам: регионы, устройства, классы, длина текста, редкие категории. Такой анализ показывает, где модель систематически слаба и какие данные или признаки

следует улучшить.

Полезна taxonomy ошибок: ошибка данных, недостаток контекста, неправильная разметка, ограничение модели, shift, плохой threshold. Это делает улучшение управляемым.

1.5 Online evaluation и A/B-тестирование

Offline улучшение не всегда переносится на продукт. A/B-тест сравнивает две версии на реальных пользователях или трафике. До запуска определяют основную метрику, guardrail metrics, длительность, единицу рандомизации и правила остановки.

Нельзя постоянно «подсматривать» на p-value без корректировки: это увеличивает вероятность ложного положительного вывода. Для критических систем внедрение часто начинается с shadow или canary режима.

1.6 Выводы

- Метрики выбираются по характеру решения и стоимости ошибок.
- Threshold является частью продукта, а не фиксированным свойством модели.
- Error analysis по сегментам важнее одной агрегированной метрики.
- Online evaluation проверяет реальную продуктовую ценность и должен быть статистически корректным.

1.7 Контрольные вопросы

- Когда PR-AUC предпочтительнее ROC-AUC?
- Как выбрать threshold для бинарного классификатора?
- Что означает калиброванная вероятность?
- Зачем анализировать метрики по сегментам?
- Чем shadow deployment отличается от A/B-теста?

2. Систематическое изложение темы и инженерные решения

2.1 Зачем нужны метрики

Метрика служит измерителем конкретного свойства, но ни одна величина не описывает всю систему. Сначала нужно определить тип ошибки и цену решения. [2]

Проверяемый пример. Сравните accuracy, macro-F1 и recall для редких критических обращений.

Вопрос для обсуждения. Как проверить выполнение такого требования: Сравните accuracy, macro-F1 и recall для редких критических обращений.

2.2 Матрица ошибок

TP, FP, TN и FN позволяют вычислять метрики бинарной классификации. Изменение порога переносит баланс между полнотой и точностью. [3]

Проверяемый пример. Вычислите precision и recall при TP=40, FP=10, FN=20.

Вопрос для обсуждения. Как проверить выполнение такого требования: Вычислите precision и recall при TP=40, FP=10, FN=20.

2.3 ROC и PR-анализ

ROC-AUC описывает ранжирование по всем порогам; PR-AUC информативна для редкого положительного класса и зависит от распространенности. [4]

Проверяемый пример. Объясните, почему PR-кривая меняется при смене base rate при одинаковой способности ранжирования.

Вопрос для обсуждения. Как проверить выполнение такого требования: Объясните, почему PR-кривая меняется при смене base rate при одинаковой способности ранжирования.

2.4 Регрессия

MAE устойчивее к отдельным большим остаткам, чем MSE, но оптимизирует другую функцию потерь. MAPE некорректна для нулевых целевых значений. [5]

Проверяемый пример. Оцените MAE и RMSE по четырем прогнозам доставки и поясните разницу.

Вопрос для обсуждения. Как проверить выполнение такого требования: Оцените MAE и RMSE по четырем прогнозам доставки и поясните разницу.

2.5 Cross-validation

Stratified, Group и TimeSeriesSplit реализуют разные допущения о независимости. Любой preprocessing должен обучаться внутри каждого training fold. [6]

Проверяемый пример. Приведите пример leakage через предварительно выполненный StandardScaler до K-fold.

Вопрос для обсуждения. Как проверить выполнение такого требования: Приведите пример leakage через предварительно выполненный StandardScaler до K-fold.

2.6 Интервалы неопределенности

Одна точка оценки не показывает устойчивость сравнения. Повторные разбиения, bootstrap и интервальные оценки дают информацию о вариабельности. [1]

Проверяемый пример. Покажите mean±std F1 по пяти seed и обсудите ограничения такого отчета.

Вопрос для обсуждения. Как проверить выполнение такого требования: Покажите mean±std F1 по пяти seed и обсудите ограничения такого отчета.

2.7 Калибровка и выбор порога

Калибровка прогнозов и threshold selection являются отдельными процедурами. Не выбирают порог по окончательному тестовому набору. [2]

Проверяемый пример. Постройте reliability diagram и выберите порог по стоимости ошибок на validation.

Вопрос для обсуждения. Как проверить выполнение такого требования: Постройте reliability diagram и выберите порог по стоимости ошибок на validation.

2.8 Error analysis

Ошибки группируют по источнику, языку, длине текста и времени, отделяя баги preprocessing от недостатков модели. [3]

Проверяемый пример. Обнаружьте, что F1 на длинных запросах значительно ниже средней метрики.

Вопрос для обсуждения. Как проверить выполнение такого требования: Обнаружьте, что F1 на длинных запросах значительно ниже средней метрики.

2.9 Online оценивание

A/B-тестирование рандомизирует назначение варианта и измеряет бизнес-результат при работе сервиса. Offline качество не гарантирует

причинный эффект. [4]

Проверяемый пример. Составьте эксперимент на уменьшение времени ответа оператора, сохранив guardrail на жалобы.

Вопрос для обсуждения. Как проверить выполнение такого требования: Составьте эксперимент на уменьшение времени ответа оператора, сохранив guardrail на жалобы.

2.10 Проверка гипотез

Определите эффект, единицу рандомизации, критерий остановки и множественность метрик до эксперимента. Нарушения контроля создают ложные выводы. [5]

Проверяемый пример. Различите статистическую и практическую значимость прироста 0.1%.

Вопрос для обсуждения. Как проверить выполнение такого требования: Различите статистическую и практическую значимость прироста 0.1%.

2.11 Нефункциональная оценка

Latency p50/p95, throughput и error rate измеряются при заданном профиле нагрузки. Усредненная задержка скрывает длинный хвост. [6]

Проверяемый пример. Объясните разницу между средним 50 мс и p95 250 мс.

Вопрос для обсуждения. Как проверить выполнение такого требования: Объясните разницу между средним 50 мс и p95 250 мс.

2.12 Отчет об эксперименте

Содержит данные, версию кода, протокол, доверительные ограничения и анализ ошибок. Точечная метрика без описания процедуры недоказатель-

на. [1]

Проверяемый пример. Подготовьте таблицу экспериментов с train_hash, val_hash, seed, metric, latency и примечанием.

Вопрос для обсуждения. Как проверить выполнение такого требования: Подготовьте таблицу экспериментов с train_hash, val_hash, seed, metric, latency и примечанием.

3. Математические инструменты и количественные критерии

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

$$\text{Precision} = \frac{TP}{TP + FP}, \qquad \text{Recall} = \frac{TP}{TP + FN} \tag{1}$$

$$F_1 = \frac{2TP}{2TP + FP + FN} \tag{2}$$

При $TP = 40$, $FP = 10$, $FN = 20$ получаем precision 0.8, recall $2/3$ и $F_1 = 80/110 \approx 0.727$.

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.

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

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, 3]

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

```
from sklearn.datasets import make_classification
from sklearn.model_selection import StratifiedKFold, cross_val_score
from sklearn.pipeline import make_pipeline
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import LogisticRegression

X, y = make_classification(n_samples=300, n_features=6,
                           weights=[0.85, 0.15], random_state=42)
model = make_pipeline(StandardScaler(), LogisticRegression())
cv = StratifiedKFold(n_splits=5, shuffle=True, random_state=17)
score = cross_val_score(model, X, y, cv=cv, scoring="f1")
print("F1 mean:", score.mean(), "F1 sd:", score.std())
```

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

```
from sklearn.metrics import precision_recall_fscore_support
real, pred = [0,0,1,1,1], [0,1,1,0,1]
print(precision_recall_fscore_support(real,pred,average="binary",zero_division=0))
```

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: Зачем нужны метрики

Заказчик использует сервис ранжирования заявок. Новое требование заставляет рассмотреть принцип «Зачем нужны метрики». Основная неопределённость состоит в том, как одновременно обеспечить precision, recall и стоимость ошибок, не допустить доверие к средней метрике и сохранить возможность изменения модели.

Постановка. Метрика служит измерителем конкретного свойства, но ни одна величина не описывает всю систему. Сначала нужно определить тип ошибки и цену решения.

Действие инженера. Сравните accuracy, macro-F1 и recall для редких критических обращений.

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: Интервалы неопределенности

Заказчик использует сервис ранжирования заявок. Новое требование заставляет рассмотреть принцип «Интервалы неопределенности». Основная неопределённость состоит в том, как одновременно обеспечить precision, recall и стоимость ошибок, не допустить доверие к средней метрике и сохранить возможность изменения модели.

Постановка. Одна точка оценки не показывает устойчивость сравнения. Повторные разбиения, bootstrap и интервальные оценки дают информацию о вариабельности.

Действие инженера. Покажите mean±std F1 по пяти seed и обсудите ограничения такого отчета.

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: Нефункциональная оценка

Заказчик использует сервис ранжирования заявок. Новое требование заставляет рассмотреть принцип «Нефункциональная оценка». Основная неопределённость состоит в том, как одновременно обеспечить precision, recall и стоимость ошибок, не допустить доверие к средней метрике и сохранить возможность изменения модели.

Постановка. Latency p50/p95, throughput и error rate измеряются при заданном профиле нагрузки. Усредненная задержка скрывает длинный хвост.

Действие инженера. Объясните разницу между средним 50 мс и p95 250 мс.

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. **Зачем нужны метрики.** Сравните accuracy, macro-F1 и recall для редких критических обращений. Сформулируйте проверяемое требование, продемонстрируйте альтернативу и приложите протокол хотя бы одного эксперимента.

2. **Матрица ошибок.** Вычислите precision и recall при TP=40, FP=10, FN=20. Сформулируйте проверяемое требование, продемонстрируйте альтернативу и приложите протокол хотя бы одного эксперимента.

3. **ROC и PR-анализ.** Объясните, почему PR-кривая меняется при смене base rate при одинаковой способности ранжирования. Сформулируйте проверяемое требование, продемонстрируйте альтернативу и приложите протокол хотя бы одного эксперимента.

4. **Регрессия.** Оцените MAE и RMSE по четырем прогнозам доставки и поясните разницу. Сформулируйте проверяемое требование, продемонстрируйте альтернативу и приложите протокол хотя бы одного эксперимента.

5. **Cross-validation.** Приведите пример leakage через предварительно выполненный StandardScaler до K-fold. Сформулируйте проверяемое требование, продемонстрируйте альтернативу и приложите протокол хотя бы одного эксперимента.

6. **Интервалы неопределенности.** Покажите mean±std F1 по пяти seed и обсудите ограничения такого отчета. Сформулируйте проверяемое требование, продемонстрируйте альтернативу и приложите протокол хотя бы одного эксперимента.

7. **Калибровка и выбор порога.** Постройте reliability diagram и выберите порог по стоимости ошибок на validation. Сформулируйте проверяемое требование, продемонстрируйте альтернативу и приложите прото-

кол хотя бы одного эксперимента.

8. **Error analysis.** Обнаружьте, что F1 на длинных запросах значительно ниже средней метрики. Сформулируйте проверяемое требование, продемонстрируйте альтернативу и приложите протокол хотя бы одного эксперимента.

9. **Online оценивание.** Составьте эксперимент на уменьшение времени ответа оператора, сохранив guardrail на жалобы. Сформулируйте проверяемое требование, продемонстрируйте альтернативу и приложите протокол хотя бы одного эксперимента.

10. **Проверка гипотез.** Различите статистическую и практическую значимость прироста 0.1%. Сформулируйте проверяемое требование, продемонстрируйте альтернативу и приложите протокол хотя бы одного эксперимента.

11. **Нефункциональная оценка.** Объясните разницу между средним 50 мс и p95 250 мс. Сформулируйте проверяемое требование, продемонстрируйте альтернативу и приложите протокол хотя бы одного эксперимента.

12. **Отчет об эксперименте.** Подготовьте таблицу экспериментов с train_hash, val_hash, seed, metric, latency и примечанием. 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 «Зачем нужны метрики»?
2. Как проверить на практике утверждение: «Сравните accuracy, macro-F1 и recall для редких критических обращений.»?
3. В чём назначение и ограничения принципа «Матрица ошибок»?
4. Как проверить на практике утверждение: «Вычислите precision и recall при TP=40, FP=10, FN=20.»?

5. В чём назначение и ограничения принципа «ROC и PR-анализ»?
6. Как проверить на практике утверждение: «Объясните, почему PR-кривая меняется при смене base rate при одинаковой способности ранжирования.»?
7. В чём назначение и ограничения принципа «Регрессия»?
8. Как проверить на практике утверждение: «Оцените MAE и RMSE по четырем прогнозам доставки и поясните разницу.»?
9. В чём назначение и ограничения принципа «Cross-validation»?
10. Как проверить на практике утверждение: «Приведите пример leakage через предварительно выполненный StandardScaler до K-fold.»?
11. В чём назначение и ограничения принципа «Интервалы неопределенности»?
12. Как проверить на практике утверждение: «Покажите mean±std F1 по пяти seed и обсудите ограничения такого отчета.»?
13. В чём назначение и ограничения принципа «Калибровка и выбор порога»?
14. Как проверить на практике утверждение: «Постройте reliability diagram и выберите порог по стоимости ошибок на validation.»?
15. В чём назначение и ограничения принципа «Error analysis»?
16. Как проверить на практике утверждение: «Обнаружьте, что F1 на длинных запросах значительно ниже средней метрики.»?
17. В чём назначение и ограничения принципа «Online оценивание»?
18. Как проверить на практике утверждение: «Составьте эксперимент на уменьшение времени ответа оператора, сохранив guardrail на жалобы.»?
19. В чём назначение и ограничения принципа «Проверка гипотез»?
20. Как проверить на практике утверждение: «Различите статистическую и практическую значимость прироста 0.1%.»?

21. В чём назначение и ограничения принципа «Нефункциональная оценка»?
22. Как проверить на практике утверждение: «Объясните разницу между средним 50 мс и p95 250 мс.»?
23. В чём назначение и ограничения принципа «Отчет об эксперименте»?
24. Как проверить на практике утверждение: «Подготовьте таблицу экспериментов с train_hash, val_hash, seed, metric, latency и примечанием.»?
25. Почему высокое offline-качество не гарантирует надежность промышленной AI-системы?
26. Какие артефакты нужны для воспроизводимости и анализа инцидента?

8. Глоссарий ключевых понятий

Зачем нужны метрики

Метрика служит измерителем конкретного свойства, но ни одна величина не описывает всю систему.

Матрица ошибок

TP, FP, TN и FN позволяют вычислять метрики бинарной классификации.

ROC и PR-анализ

ROC-AUC описывает ранжирование по всем порогам; PR-AUC информативна для редкого положительного класса и зависит от распространенности.

Регрессия

MAE устойчивее к отдельным большим остаткам, чем MSE, но оптимизирует другую функцию потерь.

Cross-validation

Stratified, Group и TimeSeriesSplit реализуют разные допущения о независимости.

Интервалы неопределенности

Одна точка оценки не показывает устойчивость сравнения.

Калибровка и выбор порога

Калибровка прогнозов и threshold selection являются отдельными процедурами.

Error analysis

Ошибки группируют по источнику, языку, длине текста и времени, отделяя баги preprocessing от недостатков модели.

Online оценивание

A/B-тестирование рандомизирует назначение варианта и измеряет бизнес-результат при работе сервиса.

Проверка гипотез

Определите эффект, единицу рандомизации, критерий остановки и множественность метрик до эксперимента.

Нефункциональная оценка

Latency p50/p95, throughput и error rate измеряются при заданном профиле нагрузки.

Отчет об эксперименте

Содержит данные, версию кода, протокол, доверительные ограничения и анализ ошибок.

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

[2] 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/.

[3] Scikit-learn Developers. User Guide: Model Selection, Pipelines and Metrics. https://scikit-learn.org/stable/user_guide.html.

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

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

[6] Breck E. et al. The ML Test Score: A Rubric for ML Production Readiness and Technical Debt Reduction. 2017. https://research.google/pubs/the-ml-test-score-a-rubric-for-ml-production-readiness-and-technical-debt-reduction/.