Практическая работа №5. Trade-off и метрики релиза

Матрица приоритетов

Таблица 1 Матрица приоритетов

Задача	**Value**	**Effort**	**Зона в матрице**	**Почему такая оценка**
API персонализации	5	13	Высокая ценность / высокий effort	Ключевая фича релиза, влияет на бизнес, высокая сложность
ML baseline	4	8	Высокая ценность / средний effort	Нужна для работы персонализации, средняя сложность
Сбор данных	5	5	Высокая ценность / низкий effort	Без данных ML не работает
UI рекомендации	4	13	Высокая ценность / высокий effort	Улучшает UX
A/B тест	3	8	Средняя ценность / средний effort	Полезно, но не критично для запуска
Нагрузочные тесты	3	5	Средняя ценность / низкий effort	Помогают избежать деградации, низкий effort
Исправление логов	5	3	Крайне высокая ценность / низкий effort	Критично для ML, делается быстро

Расчёт TCO (Total Cost of Ownership)

TCO = Первоначальные инвестиции + (Ежегодные прямые затраты × Период владения) + (Ежегодные косвенные затраты × Период владения) + Затраты на вывод

Первоначальные инвестиции включают трудозатраты на разработку (~ 40 story points по средней стоимости 6000 рублей за один SP) и базовые работы по развертыванию инфраструктуры для персонализации и ML. Суммарно это составило около 330 000 рублей.

Далее учитываются прямые ежегодные расходы на работу ML-инференса, обслуживание серверов, хранение логов и мониторинг работоспособности. При средней стоимости облака около 50 000 рублей в месяц и дополнительных небольших расходов на хранение данных и мониторинговые сервисы годовая сумма прямых затрат получилась примерно 732 000 рублей. Косвенные ежегодные расходы включают время разработчиков, QA и аналитика на поддержку релиза, исправление ошибок, доработки модели, обновление данных и сопутствующую операционную деятельность. Если учесть, что

разработчик тратит в среднем 8 часов в месяц, QA около 4 часов, а аналитик ещё 4 часа (конкретная фитча), то совокупная годовая стоимость косвенной поддержки выходит примерно на уровне 912 000 рублей. Завершающей частью формулы является стоимость вывода фичи, то есть миграции данных, удаление сервисов и обновление API. Я оценил эти работы в 40 000 рублей.

В итоге итоговый TCO релиза за год (период владения) составил примерно 2 014 000 рублей. Это и есть фактическая стоимость владения персонализацией за первый год после её внедрения.

Расчёт CPU (Cost per Unit)

CPU = Стоимость фичи / Количество операций

Для расчета данной метрики я выделил только те расходы, которые непосредственно относятся к обработке персональных рекомендаций: реализацию API, подготовку данных, обучение начальной ML-модели и исправление логов. Общая сумма этих вложений составила примерно 174 000 рублей. При среднем объёме трафика в районе 4,2 миллиона запросов в месяц стоимость одной операции получилась около 0,041 рубля за персонализированный запрос.

ROI (Return on Investment)

ROI = (Выгоды – Затраты) / Затраты × 100 %

В качестве выгод я взял прирост дохода, который даёт персонализация. По прогнозу она должна повысить конверсию примерно на 4%, что в деньгах эквивалентно примерно 280 тысячам рублей в месяц. Затраты считаются по рассчитанному TCO — около 2 миллионов рублей за год. Если смотреть только на первый месяц, то релиз ещё не окупается: ROI около –33 %. Однако при стабильном увеличении дохода уже через несколько месяцев система выходит в плюс, и примерно ко второму месяцу вложения полностью покрываются.

Перерасход бюджета и падение производительности (Чёрный лебедь)

Что произошло

Облачный провайдер поднял тарифы на 35%, что увеличило стоимость инфраструктуры для ML.

Нагрузочные тесты выявили:

- задержка API выросла с 300 до 550 мс
- нагрузка на CPU выросла на 70%
- SLA может быть нарушен

После появления этого чёрного лебедя стало понятно, что первоначальная расстановка приоритетов внутри релиза больше не отражает реальной картины. Если до инцидента акцент делался на вывод пользовательских функций — интерфейс

персональных рекомендаций, A/B-тестирование и визуальные элементы релиза, — то теперь стало очевидно, что такое распределение усилий приведёт к дальнейшей деградации SLA и, как следствие, к росту операционных затрат. Увеличение задержек API и резкое повышение нагрузки на процессор сделали производительность ключевым фактором, определяющим экономику всей системы. Функции, которые не влияют напрямую на производительность или стоимость обработки — такие как UI-изменения и A/B-тестирование, — логично будет перенести в следующий релиз. На данном этапе они не дают значимого улучшения, а только потребляют ресурсы команды. Получается, что релиз X+1 должен переключиться с расширения функциональности на стабилизацию и оптимизацию уже существующей архитектуры. Это единственный способ сохранить SLA и снизить стоимость владения фичей.

Если говорить о релизе X+N, то после анализа инцидента становится ясно, что в следующих итерациях стоит изменить сам подход к планированию. В будущие релизы необходимо заранее включать работы, которые обычно откладываются как технические: оптимизацию ML-инференса, переработку архитектуры сервиса персонализации, внедрение кэширования для снижения нагрузки, а также настройку более агрессивного авто-масштабирования и ограничений на частоту запросов. Ещё одной ключевой составляющей становится регулярный аудит стоимости облака и проверка эффективности алгоритмов рекомендаций, чтобы избежать ситуации, когда рост инфраструктурных расходов неожиданно перекрывает экономический эффект от релиза.

Таким образом, после появления чёрного лебедя приоритеты пришлось пересчитать с учётом новых ограничений: на первое место становятся стабильность, производительность и предсказуемость затрат, а развитие функциональности переносится в будущее. Для релиза X+N основной задачей будет обеспечить, чтобы архитектура персонализации была не только эффективной, но и экономически устойчивой, а сама команда имела заранее подготовленные инструменты контроля производительности и прозрачные метрики, по которым можно оперативно выявлять и предупреждать подобные инциденты.

Примерный план корректировок

1. Оптимизировать архитектуру персонализации и сократить количество обращений к ML.
2. Ввести кэширование рекомендаций для снижения нагрузки и стоимости обработки.
3. Настроить авто-масштабирование и ограничения на частоту запросов.
4. Обязательное нагрузочное тестирование в каждом релизе.
5. Улучшить качество данных и стандартизировать логирование.
6. Проводить регулярный аудит стоимости облака.
7. Обновить SLA с фокусом на производительность и стоимость операции.
8. Перенести UI и A/B тест в следующий релиз, оставив в X+N только оптимизацию и стабилизацию.