



Слепынина Надежда
Product analyst

Гайд по А/В- тестированию

Гайд по A/B-тестированию

Полный цикл эксперимента: от гипотезы до решения

Практическое руководство по проектированию, запуску и анализу продуктовых A/B-тестов. Девять последовательных шагов, статистическая база, методы снижения дисперсии и альтернативы классическому эксперименту.

Содержание

- 1 Шаг 1. Сформулировать гипотезу
- 2 Шаг 2. Выбрать и изучить метрики
- 3 Шаг 3. Определить аудиторию и сплит
- 4 Шаг 4. Изучить данные для эксперимента
- 5 Шаг 5. Спроектировать эксперимент — MDE, небольшая выборка, статистические гипотезы и p-value, карта статистических методов
- 6 Шаг 6. Провалидировать эксперимент перед запуском — AA-тесты и валидация метода
- 7 Шаг 7. Пройти необходимые ревью
- 8 Шаг 8. Мониторить эксперимент
- 9 Шаг 9. Проанализировать результаты — проверка качества, анализ результатов, принятие решения, альтернативные методы



Шаг 1.

Сформулировать гипотезу

На этом этапе нужно определить проблему, сформулировать гипотезу и понять, имеет ли смысл проводить эксперимент.

Как сформулировать гипотезу

Эксперимент проводится для проверки гипотезы. От качества её формулировки зависят выбор метрик, дизайн эксперимента и дальнейшая интерпретация результатов.

Перед формулировкой гипотезы определите проблему или барьер в пользовательском сценарии, который вы собираетесь устранить. Хорошая гипотеза объясняет, какое изменение вносится, для кого оно предназначено и какого результата ожидается достичь.

Хорошая гипотеза отвечает на три вопроса: что меняем, для кого меняем, какую метрику хотим улучшить.

Если [изменение], то для [сегмент пользователей] это приведёт к [изменению метрики].

Пример удачной формулировки: если показывать новым пользователям блок с популярными функциями на первом экране, то это увеличит долю пользователей, совершивших первое ключевое действие.

Пример неудачной формулировки: сделаем новый экран регистрации и посмотрим на результат.

Если в рамках инициативы проверяются разные механики или разные ожидаемые эффекты, сформулируйте несколько отдельных гипотез.

Результаты каждого шага рекомендуется сразу фиксировать в шаблоне дизайна А/В-теста – это упрощает прохождение ревью и подготовку к запуску.

Что ещё нужно сделать на первом шаге

Подготовить гипотезу

- Определить проблему, которую решает изменение.
- Сформулировать продуктовую гипотезу.
- Подготовить обоснование гипотезы на основе аналитики, исследований или предыдущего опыта.

Оценить целесообразность эксперимента

- Оценить экономическую целесообразность изменения.
- Оценить возможные бизнес-риски.
- Определить, нужен ли эксперимент и какой именно: учтён ли сетевой эффект, отложенный эффект и другие особенности.

Подготовиться к принятию решения

- Определить критерий успеха эксперимента.
- Продумать дальнейшие действия при положительном, нейтральном и отрицательном результате.

Результат шага. Сформулирована продуктовая гипотеза, определён критерий успеха и принято решение о целесообразности проведения эксперимента. Этот этап полезно выполнять вместе со стейкхолдером или владельцем продукта: важно не только сформулировать гипотезу, но и убедиться, что эксперимент имеет ценность и его результаты помогут принять решение.

Выбрать и изучить метрики

После формулировки гипотезы нужно определить метрики, по которым будет оцениваться результат. Метрики помогают ответить на три вопроса: приносит ли изменение пользу, подтверждается ли гипотеза, не создаёт ли изменение негативных побочных эффектов.

Как выбрать метрики

В этом гайде используется рабочая классификация метрик на четыре типа: Goal, Driver, Debug и Guardrail. Это не единственно возможное деление и не универсальный стандарт: в разных командах и источниках границы и названия отличаются. Мы используем эту классификацию, потому что она удобно связывает каждый тип метрики со своим вопросом и этапом анализа и помогает не только оценивать результат, но и находить причины изменений в продукте.

Goal Metrics

Goal Metrics (они же North Star) отражают конечную цель продукта и показывают, насколько он успешен в целом. Такие метрики напрямую связаны с результатами бизнеса, но обычно слабо реагируют на локальные изменения: на них влияет много факторов, поэтому отдельный эксперимент редко заметно двигает Goal-метрику. Примеры: выручка, прибыль, LTV, retention.

Driver Metrics

Driver- и proxy-метрики ближе к пользовательскому поведению и помогают объяснить изменения бизнес-метрик. Они чувствительнее к изменениям продукта, поэтому чаще используются как основные метрики эксперимента. Например, рост выручки может быть связан с увеличением числа активных пользователей, конверсии в ключевое действие или количества действий на пользователя — эти метрики реагируют раньше выручки. Примеры: конверсии, количество целевых действий на пользователя, количество активных пользователей.

При выборе основной метрики отдавайте предпочтение Driver-метрикам, которые напрямую связаны с гипотезой и позволяют быстрее обнаружить эффект.

Debug Metrics

Debug Metrics помогают понять причины изменения основной метрики. Если Driver-метрика выросла или упала, Debug-метрики показывают, какой именно этап воронки или сценарий привёл к результату. Примеры: конверсия между шагами воронки, CTR нового блока, количество просмотров экрана.

Guardrail Metrics

Guardrail Metrics контролируют побочные эффекты. Они помогают убедиться, что изменение не ухудшило важные показатели продукта. Даже при росте основной метрики ухудшение guardrail-метрик может стать причиной отказаться от раскатки. Примеры: доля ошибок, количество обращений в поддержку, показатели стабильности системы, показатели безопасности.

Изучите метрику перед запуском

Перед запуском изучите поведение основной метрики: определите её тип, посмотрите историческую динамику, оцените сезонность, определите окно метрики, посчитайте основные статистики (среднее, медиану, стандартное отклонение), проверьте данные на выбросы и аномалии.

Окно метрики — период времени, необходимый для полного проявления эффекта. Например, если пользователи в среднем совершают целевое действие через семь дней после старта, для анализа результатов нужно дождаться завершения этого периода.

Результат шага. Определены Goal-метрика, основная, вспомогательные и guardrail-метрики, изучены их основные характеристики.



Шаг 3.

Определить аудиторию и сплит

На этом этапе нужно определить, какие пользователи участвуют в эксперименте и как они распределяются между группами.

Определите целевой сегмент

Целевой сегмент – группа пользователей, на которых направлена гипотеза. Опишите его через конкретные параметры, например: платформа (iOS, Android, Web), страна или tier, статус в программе лояльности, уровень активности, срок жизни в продукте, принадлежность к сегменту.

Пример: новые пользователи на Android из выбранной страны, совершившие первое ключевое действие за последние 30 дней.

Определите исключения

Отдельно зафиксируйте, кто не должен попадать в эксперимент. Примеры: пользователи из стран, где функция недоступна; пользователи старых версий приложения; сотрудники и тестовые аккаунты; пользователи, уже участвующие в конфликтующих экспериментах.

Определите сплит

Заранее опишите: сколько групп в эксперименте, какое распределение между ними, по какому объекту делим пользователей (user_id, account_id, device_id и т. д.), через какой инструмент происходит сплит, в какой момент пользователь попадает в группу.

Пример: пользователи распределяются в контрольную и тестовую группы в пропорции 50/50 по user_id в момент первого открытия целевого экрана.

Единицу анализа согласуйте с единицей рандомизации. Если рандомизация идёт по одному объекту (например, по пользователю), а метрика считается на более низком уровне (сессии, события, дни одного пользователя), наблюдения внутри объекта зависимы. В этом случае анализ на уровне событий занижает дисперсию и завышает значимость, поэтому либо агрегируйте метрику до единицы рандомизации, либо применяйте метод, учитывающий кластеризацию.

Результат шага. Определены целевой сегмент, исключения и способ распределения пользователей между группами.



Шаг 4.

Изучить данные для эксперимента

Перед расчётом дизайна нужно изучить исторические данные по выбранной аудитории и метрикам. Цель — понять особенности данных и собрать входные параметры для дизайна.

Проанализируйте исторические данные

Для целевого сегмента изучите историческое поведение основных метрик. Рекомендуется: выбрать достаточный период наблюдения, рассчитать базовые статистики, оценить размер сегмента, оценить объём трафика и количество целевых событий, визуализировать динамику метрик во времени.

Такой анализ показывает: есть ли сезонность (по дням недели, месяцам), насколько метрика волатильна, есть ли периоды аномального поведения, достаточно ли данных, насколько стабильны показатели аудитории. Например, многие продуктовые метрики имеют выраженную недельную сезонность, которую нужно учитывать при выборе длительности теста.

Важно: любая метрика имеет естественные колебания. Чем выше вариативность, тем сложнее обнаружить эффект и тем больший объём данных потребуется.

Определите окно измерения метрики

Для каждой ключевой метрики определите окно, в течение которого пользователь может совершить целевое действие после попадания в эксперимент. Примеры: регистрация — активация в течение месяца; просмотр экрана — целевое действие в течение семи дней.

Если окно измерения составляет 30 дней, финальные выводы можно делать только после того, как все участники полностью пройдут это окно.

Проверьте качество данных

Убедитесь, что данные позволяют корректно измерить эффект. Обратите внимание на выбросы, сильные различия между средним и медианой, пропуски данных, технические ограничения сбора. Для количественных метрик полезно построить распределения, гистограммы и boxplot.

Результат шага. Понятны историческое поведение метрик, окно измерения эффекта и особенности данных, которые нужно учесть при проектировании.

Спроектировать эксперимент

Нужно подготовить дизайн: определить MDE, размер выборки, длительность теста и способ анализа результатов.

1. Определите тип основной метрики

Основная метрика уже выбрана на шаге 2. Определите её тип: конверсия или доля, среднее значение, ratio-метрика. От типа зависят методы расчёта выборки, MDE и способ анализа. Подробные рекомендации по выбору критерия – в подразделе «Карта основных статистических методов» ниже.

2. Определите ограничения эксперимента

Три взаимосвязанных параметра: MDE (минимальный эффект, который эксперимент способен обнаружить), размер выборки, длительность. Изменение одного влияет на остальные: для обнаружения более слабого эффекта нужна большая выборка или более длительный тест.

Сценарий 1. Известен ожидаемый эффект. Используется, когда заранее известно, какой эффект будет значимым. Порядок: определяется MDE, рассчитывается необходимый размер выборки, определяется длительность.

Сценарий 2. Эксперимент ограничен ресурсами. Используется при ограничениях по аудитории, трафику или срокам. Порядок: фиксируются ограничения, рассчитывается достижимый MDE, принимается решение о целесообразности. Если полученный MDE существенно превышает ожидаемый эффект, стоит пересмотреть дизайн, сроки или саму гипотезу.

3. Рассчитайте параметры эксперимента

До запуска зафиксируйте статистические параметры: α (допустимая вероятность ошибки первого рода, обычно 0,05) и power (требуемая мощность, обычно 0,8). Затем определите MDE, размер выборки, длительность и статистический критерий для анализа.

4. Зафиксируйте дизайн

После определения параметров заполните шаблон дизайна и зафиксируйте принятые решения до запуска. После старта параметры дизайна не должны меняться без веских причин.

Результат шага. Подготовлен дизайн: определены параметры теста, способ анализа и критерии принятия решений.

MDE: определение, расчёт и применение

Ошибки первого и второго рода и мощность теста используются при расчёте MDE — минимального эффекта, который эксперимент способен обнаружить.

Что такое MDE

MDE (Minimum Detectable Effect) — минимальное изменение метрики, которое эксперимент способен обнаружить при заданных уровне значимости, мощности, размере выборки и вариативности метрики. MDE отвечает на вопрос: эффект какого размера эксперимент сможет обнаружить с заданной вероятностью? Например, $\text{power} = 80\%$ означает, что при истинном эффекте, равном MDE, эксперимент обнаружит статистически значимое различие примерно в 80% аналогичных запусков. Чем меньше эффект нужно обнаружить, тем больше данных потребуется.

MDE определяется до запуска и используется для расчёта размера выборки, длительности и чувствительности теста. MDE не является порогом значимости при анализе: эффект меньше MDE тоже может оказаться значимым, но вероятность его обнаружения будет ниже заданной мощности.

MDE, ожидаемый и бизнес-значимый эффект

Минимальный бизнес-значимый эффект — наименьшее изменение метрики, при котором внедрение имеет смысл с учётом выгоды, затрат и рисков.

Ожидаемый эффект — прогноз результата изменения на основе прошлых экспериментов, похожих релизов, исторических данных или экспертной оценки.

Достижимый MDE — минимальный эффект, который эксперимент сможет обнаружить при доступной выборке и заданных α и power .

Эксперимент достаточно чувствителен, если достижимый MDE не превышает минимальный бизнес-значимый эффект.

Текущая конверсия = 10%

Минимальный бизнес-значимый эффект = +1 п.п.

Достижимый MDE = 0,8 п.п.

Такой эксперимент достаточно чувствителен: он способен обнаружить эффект меньше минимального изменения, ради которого готовы внедрять решение. Если достижимый MDE равен 1,5 п.п., эксперимент может пропустить бизнес-значимое изменение в 1 п.п. Ожидаемый эффект можно использовать для проверки реалистичности гипотезы, но не следует автоматически принимать его за MDE.

Абсолютный и относительный MDE

MDE можно задавать в абсолютном или относительном выражении — в дизайне нужно явно указать вариант.

Текущая конверсия = 10%

Абсолютный MDE = 1 п.п.

Целевое значение = 11%

Относительное изменение: 1 п.п. / 10% = 10%

То есть абсолютный MDE 1 п.п. соответствует относительному MDE 10%. Важно не смешивать проценты и процентные пункты при расчёте и описании.

Чем считать MDE и размер выборки

Расчёты ниже удобно выполнять в любом калькуляторе размера выборки: онлайн-калькуляторе A/B-теста, готовой функции из библиотеки (например, statsmodels в Python) или в собственной таблице. Дальше под словом «калькулятор» имеется в виду любой такой инструмент. Главное — понимать, какие параметры он принимает на вход (baseline, MDE, alpha, power, пропорция сплита) и что возвращает: размер одной группы или суммарную выборку.

Подходы к расчёту MDE

Подход 1. От минимального бизнес-значимого эффекта. Используется, когда можно определить минимальное изменение, ради которого готовы внедрять решение. Порядок: определить минимальный бизнес-значимый эффект, задать MDE на этом уровне или ниже, рассчитать необходимый размер выборки, оценить время набора. Если требуемую выборку невозможно набрать за разумный срок, нужно пересмотреть аудиторию, длительность, метрику или применить методы снижения дисперсии.

Подход 2. От доступной выборки. Используется при ограничении по аудитории или срокам. Порядок: оценить доступную выборку, рассчитать достижимый MDE, сравнить его с минимальным бизнес-значимым эффектом. Если достижимый MDE меньше или равен минимальному бизнес-значимому эффекту, чувствительности достаточно. Если больше — эксперимент может не обнаружить нужное изменение, и следует увеличить выборку либо признать, что тест не ответит на бизнес-вопрос.

Какой сценарий расчёта выбрать

Способ зависит от типа метрики и от того, что известно на этапе проектирования: ожидаемый эффект или доступная выборка.

Тип метрики	Что известно	Что рассчитываем	Что нужно для расчёта
Конверсия	Заданный MDE	Размер выборки и длительность	Конверсия, MDE, alpha, power, пропорция сплита
Конверсия	Доступная выборка или длительность	Достижимый MDE	Конверсия, размер групп, alpha, power, пропорция сплита
Среднее значение	Заданный MDE	Размер выборки и длительность	Стандартное отклонение, MDE, alpha, power, пропорция сплита
Среднее значение	Доступная выборка или длительность	Достижимый MDE	Стандартное отклонение, размер групп, alpha, power, пропорция сплита

Если MDE для среднего задаётся в относительном выражении, дополнительно нужно текущее среднее, чтобы перевести относительный эффект в абсолютный. Для ratio-метрик расчёт должен учитывать совместную вариативность числителя и знаменателя; для них нельзя автоматически использовать расчёт как для обычного среднего – метод выбирается по «Карте статистических методов».

На что обратить внимание

- Проверьте, что именно возвращает калькулятор: размер одной группы или суммарную выборку на все группы. От этого зависит расчёт длительности.
- Укажите, задан MDE в абсолютном или относительном выражении.
- MDE используется для проектирования эксперимента до запуска.
- Чем меньше MDE, тем больше выборка потребуется.
- Достижимый MDE должен быть не больше минимального бизнес-значимого эффекта.
- Если достижимый MDE больше – увеличьте аудиторию или длительность, снизьте дисперсию либо признайте, что эксперимент не ответит на вопрос надёжно.

Как рассчитать длительность

Сначала определите размер выборки, затем оцените скорость её набора.

Ориентировочная длительность = Суммарная необходимая выборка / Количество новых eligible-пользователей в день

Если калькулятор показывает выборку на одну группу, при сплите 50/50 её нужно умножить на два. Проверьте несколько вариантов длительности и сравните достижимый MDE для каждого. Расчётная длительность показывает только время набора выборки; дополнительно учитывайте окно созревания метрики, недельные и другие бизнес-циклы, сезонность, возможное снижение трафика из-за исключений, фактическую пропорцию распределения.

Что зафиксировать в дизайне

До запуска укажите: baseline метрики, MDE и его тип, минимальный бизнес-значимый эффект, источник ожидаемого эффекта (если используется), alpha и power, размер каждой группы, пропорцию сплита, прогнозируемый eligible-трафик, расчётную длительность, окно созревания метрики, источник и период исторических данных для расчёта.

Небольшая выборка: что делать

После расчёта MDE и размера выборки может оказаться, что трафика недостаточно. Например, для нужного эффекта требуется 100 000 пользователей в группу, а за плановый период можно набрать 20 000. Варианты действий:

1. Увеличить длительность

Если позволяет контекст, собирайте данные дольше. Длительность лучше делать кратной неделе, чтобы учесть недельную сезонность.

2. Расширить аудиторию

Увеличить доступную выборку за счёт расширения сегмента.

3. Пересмотреть MDE

Если важны только крупные изменения, можно увеличить MDE: чем он больше, тем меньше выборка. Но тогда эксперимент не обнаружит слабые эффекты.

4. Пересмотреть статистические параметры

Размер выборки зависит от α и power. В отдельных случаях их пересматривают с учётом рисков: для высокорисковых изменений — более строгий уровень значимости, для исследовательских — менее строгие требования к мощности. Такие решения принимают совместно с заинтересованными сторонами. Уменьшение требований к надёжности сокращает выборку, но увеличивает риск неверного решения.

5. Использовать методы снижения дисперсии

Если выборку увеличить нельзя, повысьте чувствительность за счёт снижения дисперсии метрики. Чем ниже дисперсия, тем меньший эффект можно обнаружить при том же размере выборки.

Методы снижения дисперсии стоит применять уже на этапе дизайна, а не только при анализе. Применяя их при дизайне, вы проводите больше тестов за единицу времени и замечаете изменения, которые иначе увидеть маловероятно.

CUPED

CUPED использует историческое значение метрики как ковариату и уменьшает шум в данных. Метод применим, если для пользователей есть данные за период до эксперимента.

Первоисточник: Deng, Xu, Kohavi, Walker, «Improving the Sensitivity of Online Controlled Experiments by Utilizing Pre-Experiment Data» (Microsoft, 2013), exp-platform.com. Разбор с симуляциями: bytepawn.com.

```
import numpy as np
import pandas as pd

# Функция для подсчёта CUPED-метрики.
# Важно: метрика-ковариата должна быть посчитана на пред-периоде,
# до начала эксперимента.
def get_cuped_metric(df: pd.DataFrame, metric: str) -> pd.DataFrame:
    covariate = metric + '_covariate'
    cuped_metric = metric + '_cuped'
    theta = (
        np.cov(df[metric], df[covariate])[0, 1]
        / df[covariate].var(ddof=0)
    )
    df[cuped_metric] = df[metric] - theta * df[covariate]
    return df
```

Стратификация

Стратификация уменьшает дисперсию за счёт учёта структуры аудитории. Полезна, если пользователи заранее разделены на устойчивые группы: по платформе, региону или типу пользователя.

```
import pandas as pd

# Стратифицированное среднее.
# df — датафрейм с целевой метрикой и колонкой страты stratify_cluster
# weights — маппинг {название страты: вес страты в популяции}
def calc_strat_mean(df: pd.DataFrame, weights: pd.Series, metric: str) -> float:
    strat_mean = df.groupby('stratify_cluster')[metric].mean()
    return (strat_mean * weights).sum()

# Стратифицированная дисперсия.
def calc_strat_var(df: pd.DataFrame, weights: pd.Series, metric: str) -> float:
    strat_var = df.groupby('stratify_cluster')[metric].var()
    return (strat_var * weights).sum()
```

Обработка выбросов

Выбросы увеличивают дисперсию и требуемый размер выборки. Но часто это реальные наблюдения, а не ошибки: крупные покупки, большая активность отдельных пользователей могут быть важной частью продукта. Поэтому удалять или ограничивать выбросы нужно осторожно.

Допустимо: удалять технические ошибки и некорректные данные; анализировать влияние выбросов на чувствительность; заранее фиксировать правила обработки до запуска. Не рекомендуется удалять выбросы только ради уменьшения дисперсии или получения значимого результата. Если решение об обработке принято, одинаковые правила применяются ко всем группам и фиксируются в дизайне до запуска.

Когда эксперимент лучше не запускать

Если после всех изменений достижимый MDE остаётся значительно выше ожидаемого эффекта, эксперимент не сможет надёжно ответить на вопрос. В этом случае стоит пересмотреть гипотезу, выбрать другую метрику или отказаться от эксперимента.

Статистические гипотезы, p-value и ошибки

A/B-тест оценивает, отличается ли результат в тестовой группе от контрольной или наблюдаемая разница могла возникнуть случайно. Для этого используют статистические гипотезы и критерии их проверки.

Статистические гипотезы

После выбора основной метрики формулируют две гипотезы.

Нулевая гипотеза H_0 предполагает отсутствие эффекта: $H_0: \mu_{\text{test}} = \mu_{\text{control}}$. Изменение не влияет на метрику, а разница между группами объясняется случайностью.

Альтернативная гипотеза H_1 предполагает наличие эффекта: $H_1: \mu_{\text{test}} \neq \mu_{\text{control}}$. Задача теста – определить, достаточно ли данных, чтобы отвергнуть нулевую гипотезу в пользу альтернативной.

Статистическая значимость

По результатам рассчитывается p-value – вероятность при верной нулевой гипотезе получить наблюдаемое различие между группами или ещё более экстремальное. Чем меньше p-value, тем хуже наблюдаемые данные согласуются с гипотезой об отсутствии эффекта. Для решения p-value сравнивают с заранее зафиксированным уровнем значимости alpha.

$p\text{-value} < \alpha \rightarrow$ различие значимо, H_0 отвергается.
 $p\text{-value} \geq \alpha \rightarrow$ оснований отвергнуть H_0 недостаточно.

Как не читать p-value. Он не равен вероятности того, что нулевая гипотеза верна, и не показывает вероятность того, что эффект «настоящий» или «не случайный». Он говорит только одно: насколько наблюдаемые данные вероятны в мире, где эффекта нет. Значимый результат не доказывает эффект со стопроцентной уверенностью, а лишь показывает, что такие данные маловероятны при отсутствии эффекта.

Ошибки при принятии решения

Ошибка первого рода (alpha). Возникает, когда нулевая гипотеза отвергается, хотя она верна. В контексте A/B-теста – эксперимент показывает эффект, которого нет (ложноположительный результат). Обычно фиксируется на уровне $\alpha = 0,05$, то есть допускается не более 5% ложноположительных выводов.

Ошибка второго рода (beta). Возникает, когда нулевая гипотеза не отвергается, хотя эффект есть. В контексте A/B-теста – полезное изменение ошибочно признаётся неэффективным (ложноотрицательный результат).

Мощность теста ($1 - \text{beta}$). Показывает вероятность обнаружить эффект, если он действительно существует. На практике обычно используют $\text{power} = 0,8$: при наличии эффекта тест обнаружит его примерно в 80% случаев.

Почему статистика работает в A/B-тестах

Решения принимаются не по всей генеральной совокупности, а по выборке участников.

Статистические критерии позволяют оценить, насколько вероятно распространить наблюдаемый результат на всю аудиторию. Основание – центральная предельная теорема: при достаточном размере выборки распределение выборочных средних приближается к нормальному, что позволяет рассчитывать доверительные интервалы, p-value и значимость.

Карта основных статистических методов

Ниже — основные статистические методы для подсчёта наблюдаемой разницы между группами A/B-теста.

Полезные внешние материалы: бутстреп в A/B-тестировании, дельта-метод для ratio-метрик, линеаризация ratio-метрик, CUPED. Для CUPED см. первоисточник Deng et al. (2013), exp-platform.com.

1. Бинарные метрики (конверсии)

Когда использовать: для метрик «событие произошло / не произошло» (0/1), например регистрация, покупка, клик. Как работает: сравнивает доли успехов между контрольной и тестовой группами.

Важно: стандартные конверсионные метрики (например, отношение просмотров к кликам) в индустрии принято считать ratio-метрикой, так как пользователь может сделать несколько просмотров и кликов. Критерии ниже применяйте, когда нужно посчитать поюзерные пропорции 1/0 без учёта нескольких возможных событий.

Не путайте поюзерную конверсию и ratio-конверсию. Поюзерная — это доля 0/1 на пользователя: сконвертировался хотя бы раз или нет, и для неё подходят критерии этого раздела. Ratio-конверсия — это отношение суммарных событий (клики / просмотры) поверх пользователей, где на одного пользователя приходится несколько событий; она считается методами раздела «Ratio-метрики». От того, как определена метрика, зависит и корректный критерий.

1.1. Z-test для пропорций

Гипотезы: H_0 — конверсии в control и treatment равны; H_1 — различаются. Когда: большая выборка, достаточное число успехов и неудач.

```
import numpy as np
from statsmodels.stats.proportion import proportions_ztest

control_success = 1000
control_total = 10000
treatment_success = 1122
treatment_total = 10200

count = np.array([treatment_success, control_success])
nobs = np.array([treatment_total, control_total])

stat, pval = proportions_ztest(count, nobs)
print("Z-test p-value:", pval)
```

1.2. Fisher exact test

Гипотезы: H_0 — конверсии в обеих группах равны; H_1 — различаются. Когда: маленькая выборка или редкая конверсия (работает и на больших данных, ненамного дольше).

```
from scipy.stats import fisher_exact
import numpy as np

control_total = 500
control_success = 34
treatment_total = 500
treatment_success = 65

table = np.array([
    [treatment_success, treatment_total - treatment_success],
    [control_success, control_total - control_success],
])
oddsratio, pval_fisher = fisher_exact(table)
print("Fisher exact p-value:", pval_fisher)
```

2. Средние значения

Когда использовать: user-level метрики типа «выручка на пользователя», «действий на пользователя». Как работает: сравнивает средние значения метрики между группами.

2.1. Student t-test и Welch t-test

Гипотезы: H_0 — средние равны; H_1 — различаются. Когда: почти всегда, но чувствителен к выбросам. Принцип: оценивает разницу средних с учётом дисперсии в группах.

Классический Student t-test предполагает равенство дисперсий в группах (`equal_var=True`). На практике дисперсии в контрольной и тестовой группах часто различаются, поэтому по умолчанию безопаснее использовать Welch t-test (`equal_var=False`) — он не требует равенства дисперсий и почти не теряет мощность, когда они равны.

```
from scipy.stats import ttest_ind
import numpy as np

control = np.random.normal(50, 10, 1000)
treatment = np.random.normal(52, 12, 1050)

# equal_var=False — Welch t-test (рекомендуется по умолчанию)
stat, pval = ttest_ind(treatment, control, equal_var=False)
print("Welch t-test p-value:", pval)
```

2.2. Bootstrap

Гипотезы: H_0 — среднее одинаково в обеих группах; H_1 — различаются. Когда: данные с тяжёлыми хвостами; можно применять вместо t-теста, но на маленьких выборках метод менее устойчив. Принцип: многократный ресэмплинг пользователей, строим доверительный интервал разницы средних.

```
import numpy as np
np.random.seed(42)

group_a = np.random.normal(loc=50, scale=10, size=1000) # контроль
group_b = np.random.normal(loc=52, scale=10, size=1000) # тест

n_boot = 5000
diffs = []
for _ in range(n_boot):
    sample_a = np.random.choice(group_a, size=len(group_a), replace=True)
    sample_b = np.random.choice(group_b, size=len(group_b), replace=True)
    diffs.append(np.mean(sample_b) - np.mean(sample_a))

diffs = np.array(diffs)
ci_low, ci_high = np.percentile(diffs, 2.5), np.percentile(diffs, 97.5)
print("Bootstrap 95% CI разницы средних:", (ci_low, ci_high))
print("Среднее A:", np.mean(group_a))
print("Среднее B:", np.mean(group_b))
```

3. Ratio-метрики (отношения)

Например средний чек, конверсия клик/просмотр. Особенности: числитель и знаменатель случайны и могут быть зависимы, поэтому простое сравнение средних ratio некорректно; значение ratio может сильно колебаться при маленьком знаменателе; нужен метод, учитывающий дисперсию и корреляцию числителя и знаменателя.

Когда использовать: метрики вида «числитель / знаменатель». Прежде чем считать разницу в группах, метрику приводят к юзер-лevelу, чтобы убрать зависимость в данных.

3.1. Линеаризация

Рекомендуется, так как поверх неё можно применить CUPED. Идея: превращаем ratio в линейную метрику на уровне пользователя, чтобы использовать обычные методы сравнения средних.

```
import pandas as pd
import numpy as np
from scipy.stats import ttest_ind

# Данные user-level: revenue и deposits на каждого пользователя
df_control = pd.DataFrame({'num': [100, 200, 50, 80, 60],
                           'den': [10, 20, 5, 8, 6]})
df_treatment = pd.DataFrame({'num': [120, 180, 55, 90, 65],
                              'den': [11, 18, 5, 9, 6]})
```

```

# Ratio контрольной группы
R_control = df_control['num'].sum() / df_control['den'].sum()

# Линеаризация: L = num - R_control * den
df_control['lin'] = df_control['num'] - R_control * df_control['den']
df_treatment['lin'] = df_treatment['num'] - R_control * df_treatment['den']

# Сравнение средних линеаризованной метрики
mean_diff = df_treatment['lin'].mean() - df_control['lin'].mean()
stat, pval = ttest_ind(df_treatment['lin'], df_control['lin'], equal_var=False)
print("Разница после линеаризации:", mean_diff)
print("t-test p-value:", pval)

```

3.2. Дельта-метод

Позволяет оценить стандартную ошибку ratio на агрегированных данных, учитывая дисперсию числителя и знаменателя и их ковариацию. После этого можно получить разницу средних между тестовой и контрольной группами.

```

import numpy as np
import scipy.stats as stats
np.random.seed(42)

n_users = 10000
# Группа A
views_A = np.random.randint(10, 50, n_users)
clicks_A = np.random.binomial(views_A, p=0.05)
# Группа B (чуть выше конверсия)
views_B = np.random.randint(10, 50, n_users)
clicks_B = np.random.binomial(views_B, p=0.055)

def delta_method_ratio(den, num):
    """Оценка ratio = E[num] / E[den] и её стандартной ошибки
    дельта-методом с учётом ковариации числителя и знаменателя."""
    n = len(den)
    mean_num = np.mean(num)
    mean_den = np.mean(den)
    var_num = np.var(num, ddof=1)
    var_den = np.var(den, ddof=1)
    cov = np.cov(num, den, ddof=1)[0, 1]
    ratio = mean_num / mean_den
    variance = (
        var_num
        - 2 * ratio * cov
        + ratio**2 * var_den
    ) / (mean_den**2)

```

```

std_error = np.sqrt(variance / n)
return ratio, std_error

ratio_A, se_A = delta_method_ratio(views_A, clicks_A)
ratio_B, se_B = delta_method_ratio(views_B, clicks_B)

diff = ratio_B - ratio_A
se_diff = np.sqrt(se_A**2 + se_B**2)
z_stat = diff / se_diff
p_value = 2 * (1 - stats.norm.cdf(np.abs(z_stat)))
print(f"Метрика группы A: {ratio_A:.5f}")
print(f"Метрика группы B: {ratio_B:.5f}")
print(f"Разница: {diff:.5f}")
print(f"Z-статистика: {z_stat:.4f}")
print(f"P-value: {p_value:.5f}")

```



Шаг 6.

Провалидировать эксперимент перед запуском

Перед запуском нужно убедиться, что инфраструктура работает корректно и результаты можно будет интерпретировать с доверием. Проверьте: корректность сплитования, отсутствие систематических различий между группами до старта, корректность расчёта метрик, применимость выбранных статистических методов. Для этого используют АА-тесты и проверки механизма сплитования.

Результат шага. Подтверждено, что пользователи распределяются корректно, метрики считаются без ошибок, статистические методы применимы, инфраструктура готова к запуску. Перед отправкой на ревью полезно выполнить самопроверку дизайна, чтобы выявить типовые методологические ошибки до человеческого ревью.

АА-тесты и валидация статистического метода

Перед запуском нужно убедиться, что сплитование, сбор данных, расчёт метрик и статистический метод работают корректно. Используются три разные проверки, которые не заменяют друг друга: реальный АА-тест, АА-симуляции на исторических данных, синтетические А/В-симуляции с искусственным эффектом.

Реальный АА-тест

Пользователи распределяются через продовую сплитовалку, но обе группы получают одинаковый опыт. Проверяет: корректность распределения, соответствие фактической пропорции запланированному сплиту, отсутствие пересечения групп, корректность логирования и сбора данных, одинаковый расчёт метрик, отсутствие систематических различий до воздействия.

Один реальный АА-тест не проверяет, что значимые различия возникают ровно в $\alpha\%$ случаев: он проверяет конкретный запуск инфраструктуры, а не долгосрочную частоту ложных срабатываний. Особенно важен при новой сплитовалке, изменении логирования или запуске нового процесса. Перед каждым экспериментом в уже проверенной инфраструктуре отдельный АА необязателен.

АА-симуляции на исторических данных

Проверяют, корректно ли метод контролирует ошибку первого рода на данных, похожих на будущий эксперимент. На каждой итерации историческая выборка случайно делится на контроль и тест, реального эффекта нет. При $\alpha = 0,05$ значимый результат должен появляться примерно в 5% симуляций. Одна симуляция ничего не говорит о фактической ошибке — нужна серия, например 1000 разбиений.

Синтетические А/В-симуляции

Проверяют мощность. В тестовую группу искусственно добавляется эффект заданного размера, затем применяется запланированный метод. Если добавлен эффект, равный MDE, а целевая мощность 80%, значимый результат должен появляться примерно в 80% симуляций.

Когда нужна историческая валидация

Отдельная серия симуляций особенно полезна, если: используется нестандартная метрика или метод; метрика с длинным хвостом или множеством выбросов; анализируется ratio-метрика; выборка небольшая; единица рандомизации отличается от пользователя; используется кластерная рандомизация; применяются CUPED, стратификация или другая корректировка; есть сомнения в расчётной мощности. Для стандартной метрики и проверенного подхода серия симуляций перед каждым экспериментом может быть не нужна.

Как провести историческую валидацию

Используйте исторические данные, максимально похожие на будущий эксперимент по аудитории, размеру выборки, длительности наблюдения, окну метрики, сезонности, правилам фильтрации. Данные агрегируйте до единицы рандомизации: если рандомизация по пользователю, в симуляции — одна строка на пользователя.

Шаг 1. Проверить ошибку первого рода. На каждой итерации: случайно сформируйте контроль и тест, используйте запланированные размеры групп, рассчитайте метрику тем же способом, что и в будущем эксперименте, примените выбранный критерий, сохраните p-value.

```
import numpy as np
from scipy.stats import ttest_ind, binomtest

rng = np.random.default_rng(42)
alpha = 0.05
n_simulations = 1000
n_control = 5000
n_treatment = 5000

# Одна строка на единицу рандомизации.
# Пропуски и нулевые значения обрабатываются так же,
# как в будущем эксперименте.
metric_values = df_user["metric"].to_numpy()

required_size = n_control + n_treatment
if len(metric_values) < required_size:
    raise ValueError(
        "Исторической выборки недостаточно "
        "для формирования групп заданного размера"
    )

pvalues = np.empty(n_simulations)
for i in range(n_simulations):
    indices = rng.choice(len(metric_values), size=required_size, replace=False)
    control = metric_values[indices[:n_control]]
    treatment = metric_values[indices[n_control:]]
    result = ttest_ind(treatment, control, equal_var=False, alternative="two-sided")
```

```

pvalues[i] = result.pvalue

n_false_positives = np.sum(pvalues < alpha)
false_positive_rate = n_false_positives / n_simulations
calibration_check = binomtest(
    k=n_false_positives, n=n_simulations, p=alpha, alternative="two-sided"
)
confidence_interval = calibration_check.proportion_ci(confidence_level=0.95)
print("False positive rate:", false_positive_rate)
print("95% CI:", confidence_interval)
print("Calibration p-value:", calibration_check.pvalue)

```

Если для реального эксперимента выбран другой метод (Z-test, Fisher exact test, линеаризация), в симуляции используйте именно его.

Как интерпретировать. Фактическая доля ложных срабатываний не обязана быть ровно 5% — она сама зависит от случайности симуляций. Сравнивайте результат с доверительным интервалом, а не с фиксированными порогами вроде «4,8% хорошо, а 6% плохо». Если целевое α попадает в доверительный интервал фактической ошибки, результат согласуется с уровнем значимости. Если ложные срабатывания возникают заметно чаще, возможны проблемы с единицей анализа, расчётом дисперсии, зависимостью наблюдений, методом, формированием групп, обработкой пропусков или выбросов. Если заметно реже, тест может быть излишне консервативным и терять мощность.

Шаг 2. Проверить распределение p-value. Для непрерывных тестов при выполнении их предположений p-value в AA-симуляциях должны быть приблизительно равномерно распределены от 0 до 1.

```
import matplotlib.pyplot as plt

plt.hist(pvalues, bins=20, range=(0, 1))
plt.xlabel("p-value")
plt.ylabel("Количество симуляций")
plt.title("Распределение p-value в AA-симуляциях")
plt.show()
```

Избыточная концентрация p-value около нуля может указывать на нарушение предположений или ошибку в симуляции. Для дискретных exact-тестов (например, Fisher) p-value не обязаны быть равномерными, поэтому распределение – вспомогательный диагностический инструмент, а основная проверка остаётся за фактической ошибкой первого рода.

Шаг 3. Проверить мощность. В тестовую группу добавляется эффект, равный MDE.

```
target_power = 0.80
mde_absolute = 5.0

significant_results = np.empty(n_simulations, dtype=bool)
for i in range(n_simulations):
    indices = rng.choice(len(metric_values), size=required_size, replace=False)
    control = metric_values[indices[:n_control]]
    treatment = metric_values[indices[n_control:]].copy()
    # Добавляем абсолютный эффект
    treatment = treatment + mde_absolute
    result = ttest_ind(treatment, control, equal_var=False, alternative="two-sided")
    significant_results[i] = result.pvalue < alpha

n_detected = np.sum(significant_results)
empirical_power = n_detected / n_simulations
power_check = binomtest(
    k=n_detected, n=n_simulations, p=target_power, alternative="two-sided"
)
power_interval = power_check.proportion_ci(confidence_level=0.95)
print("Empirical power:", empirical_power)
print("95% CI:", power_interval)
print("Power calibration p-value:", power_check.pvalue)
```

Если целевая мощность попадает в доверительный интервал фактической, результат согласуется с расчётом. Если фактическая мощность существенно ниже, эксперимент может не обнаружить эффект, равный MDE.

Как правильно добавить эффект. Способ добавления должен соответствовать типу метрики. Для непрерывной метрики – абсолютное изменение $\text{treatment} = \text{treatment} + \text{mde_absolute}$. Для относительного MDE сначала переведите его в абсолютное изменение на основе baseline. Для бинарной метрики нельзя использовать $\text{treatment} * = 1.05$: значения 0/1 перестанут быть бинарными. Нужно моделировать изменение вероятности события и применять метод для сравнения конверсий. Для ratio-метрики эффект добавляют в числитель или знаменатель в соответствии с механизмом изменения, затем применяют запланированный метод: линеаризацию, дельта-метод или другой выбранный подход.

Что делать, если валидация не проходит. Сначала определите, какая проверка не прошла. Если проблема в реальном AA-тесте – проверьте сплитование и бакетизацию, пересечение групп, SRM, логирование попадания в эксперимент, сбор и расчёт метрик. Если не проходит историческая проверка α – проверьте соответствие единицы анализа единице рандомизации, независимость наблюдений, предположения метода, правила агрегации, обработку пропусков и выбросов, соответствие исторических данных будущей аудитории. Если фактическая мощность ниже ожидаемой – проверьте размер групп, способ добавления эффекта, вариативность метрики, корректность расчёта MDE, возможность снизить дисперсию или увеличить выборку.

Результат валидации. Подтверждено, что инфраструктура работает корректно, группы формируются без систематических ошибок, фактическая ошибка первого рода согласуется с α , фактическая мощность – с power, и используется тот же метод, что будет применён при анализе. Если обязательная проверка не пройдена, запускать эксперимент до выяснения причины не следует.



Шаг 7.

Пройти необходимые ревью

Перед запуском стоит получить необходимые согласования. Цель ревью — убедиться, что эксперимент решает реальную задачу, корректно спроектирован и не создаёт неоправданных рисков. В зависимости от типа эксперимента может потребоваться несколько видов ревью.

Ревью со стейкхолдером или владельцем продукта

Начинается ещё на этапе формирования гипотезы. Оцениваются ценность инициативы, ожидаемый эффект, возможные риски и критерии успеха.

Ревью дизайна с независимым аналитиком

Дизайн эксперимента полезно вынести на независимое ревью — второго аналитика или аналитическое сообщество. Цель — независимо оценить логику гипотезы, выбор метрик и корректность решений. Разумно установить срок ревью, например не более двух рабочих дней с момента публикации дизайна.

Ревью финансовых рисков

Для экспериментов, которые влияют на выручку, ценообразование или другие денежные показатели, а также для заведомо ухудшающих экспериментов, стоит отдельно оценить потенциальные потери. Это согласование проводят с финансовой функцией или ответственными за юнит-экономику.

Ревью с профильными командами

Если эксперимент затрагивает специфические для продукта области (например, экономику ключевого процесса), согласуйте его с профильными командами, которые отвечают за эти области.

Результат шага. Получены все необходимые согласования, эксперимент готов к запуску.



Шаг 8.

Мониторить эксперимент

После запуска контролируйте корректность проведения и качество данных. На этом этапе проверяется состояние эксперимента, а не его результат.

1. Контролируйте набор выборки. Пользователи набираются по плану, эксперимент достигнет нужного размера в срок.
2. Проверяйте распределение по группам. Размеры групп соответствуют дизайну.
3. Контролируйте качество данных. Следите за пропусками, сбоями логирования и другими техническими проблемами.
4. Отслеживайте guardrail-метрики. При существенном ухудшении эксперимент можно остановить досрочно.
5. Не принимайте решение раньше времени. Не завершайте эксперимент раньше срока и не меняйте параметры анализа после запуска.

Результат шага. Подтверждено, что эксперимент проводится корректно, данные собираются без ошибок и пригодны для анализа.



Проанализировать результаты

После завершения нужно убедиться, что результатам можно доверять, и оценить влияние изменения. Перед решением: проверьте качество эксперимента, проанализируйте основные, вспомогательные и guardrail-метрики, оцените статистическую и практическую значимость, определите дальнейшие действия.

Результат шага. Проверено качество эксперимента, проанализированы результаты, принято решение о дальнейших действиях; итог зафиксирован.

Проверка качества эксперимента

Перед анализом эффекта убедитесь, что эксперимент проведён корректно. Иначе значения метрик и p-value могут привести к неверным выводам.

- 1. Проверьте Sample Ratio Mismatch (SRM).** SRM – ситуация, когда фактическое распределение по группам отличается от запланированного. Например, по дизайну 50/50, а после завершения 60/40. Это может указывать на ошибку в сплитовании, фильтрации или логировании. Сравните фактическое количество пользователей в каждой группе с плановым, проверьте отклонение от ожидаемых долей, при наличии SRM разберитесь в причине до анализа. Если SRM подтверждается, результаты нельзя интерпретировать без дополнительной проверки.
- 2. Проверьте сопоставимость групп до эксперимента.** Даже при похожем размере группы могут отличаться по составу. Сравните характеристики за период до эксперимента: активность, конверсии, платформу, страну или регион, тип пользователя, другие важные сегменты. Эти метрики не должны систематически отличаться до старта. Если различия есть, эффект может быть смещён.
- 3. Проверьте полноту и корректность данных.** Убедитесь, что данные собраны корректно и одинаково доступны для всех групп: не потерялись ли события, нет ли неожиданного роста пустых значений, одинаково ли считается метрика для control и treatment, нет ли резких аномалий в логировании после старта.
- 4. Проверьте соответствие дизайну.** Убедитесь, что эксперимент шёл запланированное количество дней, длительность была кратна недельному циклу (если это предусмотрено), набрана достаточная выборка, группы формировались по заданным правилам, тест не останавливали и не меняли без зафиксированной причины.

Если обнаружены критичные проблемы (SRM, существенные различия до эксперимента, ошибки в данных), не переходите сразу к интерпретации. Сначала разберитесь в причине; возможно, эксперимент придётся исключить из принятия решений и запланировать повторный запуск.

Анализ результатов A/B-теста

После проверки качества анализируйте метрики в порядке: сначала основную, затем guardrail, затем debug.

1. Основная метрика. Сравните значения в контроле и тесте, оцените абсолютный и относительный эффект, примените выбранный на этапе проектирования критерий, проанализируйте p-value и доверительный интервал. Оценивайте два показателя: p-

value (вероятность получить наблюдаемое различие или большее, если эффекта нет) и доверительный интервал (диапазон возможных значений истинного эффекта). Если p-value меньше уровня значимости (обычно 0,05), различия статистически значимы. Но значимость не означает ценности: после p-value оцените практическую значимость — сравните эффект с ожидаемым, зафиксированным при проектировании.

2. Guardrail-метрики. Убедитесь, что эксперимент не ухудшил другие важные показатели. Оцените величину эффекта, p-value и доверительный интервал. Даже при улучшении основной метрики значимое ухудшение guardrail-метрик может стать причиной отказаться от раскатки или провести дополнительное исследование.

3. Debug-метрики. Помогают объяснить результат и понять, почему изменилась основная метрика. Используются для поиска причин и формирования следующих гипотез, но обычно не как критерий принятия решения.

4. Итоговый вывод. Зафиксируйте: подтвердилась ли гипотеза, получен ли статистически значимый эффект, значим ли он практически, не ухудшились ли guardrail-метрики, помогают ли debug-метрики объяснить результат. На основании этого принимается итоговое решение.

Принятие решения по результатам

Решение принимается на основании критериев успеха и правил, зафиксированных в дизайне до запуска. Это избавляет от субъективной интерпретации после теста. Учитывайте: результаты основной метрики, состояние guardrail-метрик, статистическую и практическую значимость, качество эксперимента.

Возможные решения

Раскатить изменение. Если эксперимент достиг заранее определённых критериев успеха и качество не вызывает сомнений. После раскатки продолжайте мониторинг ключевых метрик: некоторые эффекты проявляются позже или ослабевают по мере привыкания пользователей.

Доработать гипотезу и повторить. Если результаты не позволяют принять однозначное решение или гипотеза подтвердилась частично, эксперимент не обязательно неудачен. Результаты можно использовать для уточнения гипотезы, изменения дизайна или выбора другой аудитории и провести повторную проверку.

Отказаться от изменения. Если эксперимент не достиг критериев успеха, выявил существенное ухудшение ключевых метрик или неприемлемые риски. Отрицательный результат тоже полезен: он позволяет отказаться от неэффективной идеи на основании данных.

Если результатам нельзя доверять. Иногда решение принять нельзя: обнаружены SRM, ошибки в данных, нарушение дизайна или другие критичные отклонения. Сначала разберитесь с причиной, потом решайте о повторном запуске.

Зафиксируйте результаты. Обновите дизайн и зафиксируйте: итоговые значения ключевых метрик, принятое решение, основные выводы и инсайты, дальнейшие действия. Даже если гипотеза не подтвердилась, результаты ценны и пригодятся при формировании следующих гипотез.

Альтернативные методы проверки гипотез

Что делать, если классический A/B невозможен: изменение уже внедрено для всех и нельзя выделить контроль; нет технической возможности разделить пользователей случайно; слишком мало трафика или редкие события для достаточной мощности.

В таких случаях применяют альтернативные методы оценки эффекта:

- **Difference-in-Differences (DiD)** – оценивает эффект по данным до и после внедрения для treatment и контрольной группы.
- **CausallImpact** – моделирует контрфактический сценарий (что было бы без изменения) на основе временных рядов и контрольных метрик, когда контрольной группы нет.

Преимущества: работают с наблюдательными данными без рандомизации, учитывают временные тренды и различия между группами, применимы к редким событиям и агрегированным метрикам.

Ограничения: требуют проверки предпосылок (например, параллельные тренды для DiD), чувствительны к шуму, сезонности и автокорреляции, могут давать смещённые оценки при плохом выборе контрольных рядов, требуют аккуратной интерпретации. Поэтому классический A/B всегда предпочтительнее, если он возможен.

CausallImpact

Метод каузального вывода для временных рядов, когда нет явной контрольной группы или вмешательство применено на всех данных. Моделирует, как метрика выглядела бы без вмешательства, и сравнивает с наблюдаемыми данными после.

Как работает: используется Bayesian structural time-series (BSTS) модель; на пред-периоде она обучается прогнозировать метрику; в пост-период даёт контрфактический прогноз, как развивалась бы метрика без вмешательства; разница между наблюдаемым и контрфактическим – оценка эффекта.

```

import numpy as np
import pandas as pd
from causalimpact import CausalImpact
import matplotlib.pyplot as plt
np.random.seed(42)

n_days = 105
# До интервенции: стабильные значения с малым шумом
y = np.random.normal(50, 1, n_days)
x1 = np.random.normal(30, 1, n_days)
x2 = np.random.normal(200, 2, n_days)

# После интервенции (дни 89-104) добавляем эффект
effect_start = 89
y[effect_start:] += 40
x1[effect_start:] += 3
x2[effect_start:] += 10

data = pd.DataFrame({'y': y, 'x1': x1, 'x2': x2})
pre_period = [0, 88]
post_period = [89, 104]

ci = CausalImpact(data, pre_period, post_period)
ci.plot(figsize=(15, 14))
plt.show()
print(ci.summary())

```

Смотрим на график эффекта и на доверительный интервал.

Difference-in-Differences

Внешний материал: глава про Difference-in-Differences в открытой книге Matheus Facure «Causal Inference for the Brave and True», matheusfacure.github.io.

Метод сравнивает изменение метрики во времени между группой, подвергшейся изменению (treatment), и контрольной группой. Здесь уже нужна контрольная группа. Как работает: считаем метрику для treatment и control до и после вмешательства, вычисляем разницу изменений, получаем оценку эффекта с учётом общих временных трендов. Предпосылка — параллельные тренды между группами. Метод чувствителен к внезапным внешним шокам и сезонности.

```

import pandas as pd
import numpy as np
import statsmodels.formula.api as smf
np.random.seed(42)

# Две группы: A (treatment) и B (control)
n_days = 30
a_metric = np.random.poisson(lam=50, size=n_days)
b_metric = np.random.poisson(lam=45, size=n_days)

# Эффект интервенции для группы A после середины периода
effect_start = n_days // 2
a_metric[effect_start:] += 20

```

```

data = pd.DataFrame({
    'value': np.concatenate([a_metric, b_metric]),
    'treat': [1]*n_days + [0]*n_days, # 1 = treatment, 0 = control
    'post': ([0]*(n_days//2) + [1]*(n_days - n_days//2)) * 2, # период после
})

a_before = data.query("treat==1 & post==0")["value"].mean()
a_after = data.query("treat==1 & post==1")["value"].mean()
b_before = data.query("treat==0 & post==0")["value"].mean()
b_after = data.query("treat==0 & post==1")["value"].mean()

naive1 = a_after - a_before
print(f"Наивная оценка (before-after, без контроля): {naive1:.2f}")

naive2 = a_after - b_after
print(f"Наивная оценка (A vs B, только после): {naive2:.2f}")

did_estimate = (a_after - a_before) - (b_after - b_before)
print(f"DiD estimate: {did_estimate:.2f}")

# DiD через регрессию (с доверительными интервалами)
model = smf.ols("value ~ treat * post", data=data).fit()
print(model.summary().tables[1])
# Коэффициент treat:post — это и есть DiD-оценка

```