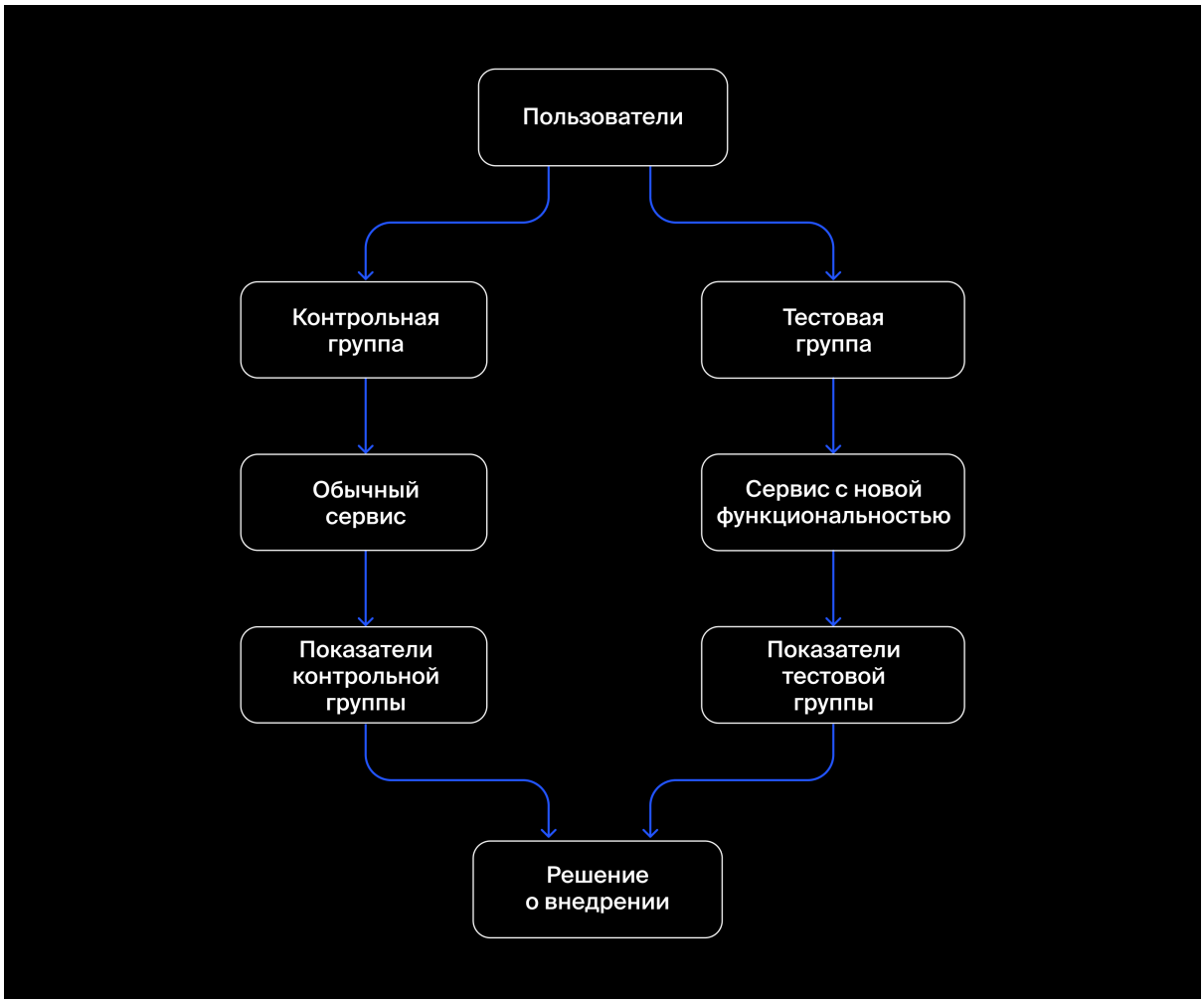
Конспект по теме «Запуск новой функциональности»

Планирование запуска

A/B-тестирование, или **сплит-тестирование**, — техника проверки гипотез. Позволяет оценить, как изменение сервиса или продукта повлияет на пользователей. Проводится так: аудиторию делят на две группы — контрольную (*A*) и тестовую (*B*). Группа *A* видит начальный сервис, без изменений. Группа *B* получает новую версию, которую и нужно протестировать. Эксперимент длится фиксированное время. В ходе тестирования собираются данные о поведении пользователей в разных группах. Если ключевая метрика в тестовой группе выросла по сравнению с контрольной, новую функциональность внедряют.



Прежде чем провести A/B-тест, часто делают проверку корректности — **A/A тест**. В нём пользователей делят на две контрольные группы, которые видят одинаковый сервис.

Ключевая метрика у групп отличаться не должна: если она разная — ищите ошибку.

Длительность A/B-теста

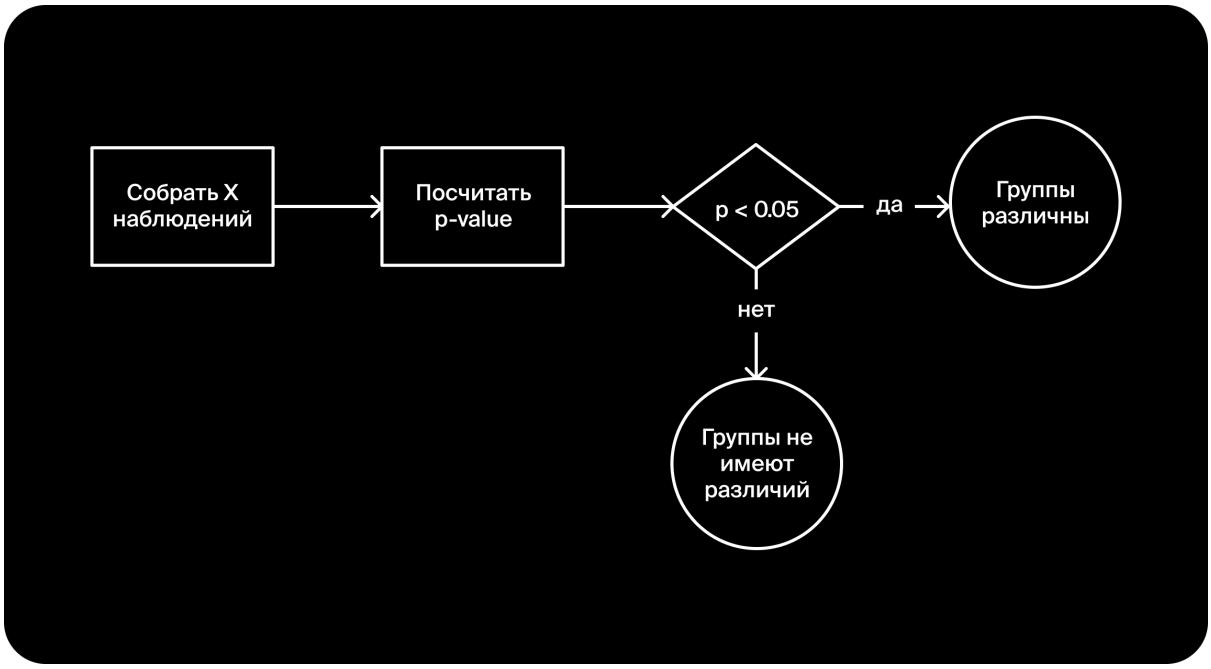
С запуском новой функциональности меняется поведение пользователей. Обычно им нужно время, чтобы привыкнуть к нововведению. Когда это произошло, успешность эксперимента можно оценить уже наверняка. Чем больше объём данных, тем меньше вероятность ошибки при проверке статистических гипотез.

У A/B-теста есть **проблема подглядывания**: общий результат искажается, если новые данные поступают в начале эксперимента. Каждый, даже небольшой фрагмент новых данных, велик относительно уже накопленных — статистическая значимость достигается за короткий срок.

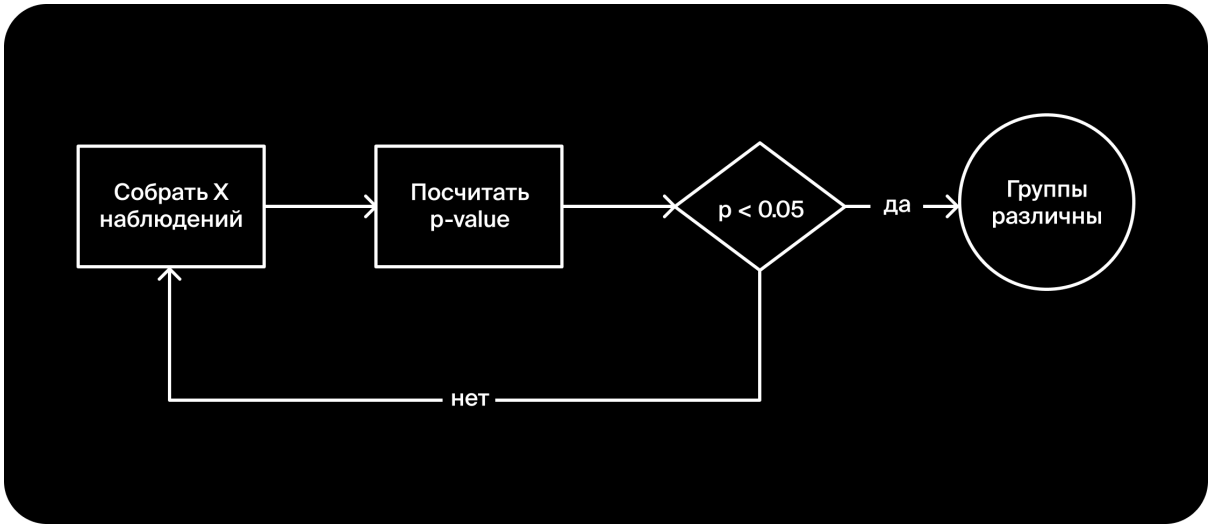
Это одно из проявлений закона больших чисел. Если наблюдений мало, их разброс больше. Если много, случайные выбросы успели друг друга компенсировать. Значит, если выборка слишком мала, различия увидеть легко. Для статистического теста это снижение *p-value* до значений, достаточно маленьких, чтобы отвергнуть нулевую гипотезу об отсутствии различий.

Чтобы избежать проблемы подглядывания, размер выборки определяют ещё до начала теста.

Вот правильная процедура A/B-тестирования:



А так делать A/B-тестирование не стоит:



Простой способ найти размер выборки — рассчитать в онлайн-калькуляторе, например: https://vwo.com/tools/ab-test-duration-calculator/

Сравнение средних значений

Научимся анализировать результаты A/B-теста: среднее значение метрики опишет поведение всех пользователей вместе.

Результаты измерения и средние значения — случайные величины. Значит, в них могут быть *случайные* погрешности. Спрогнозировать их невозможно, а вот оценить получится методами статистики.

Допустим, наша нулевая гипотеза $\mathbf{H_0}$ звучит так: «Новая функциональность не улучшает метрики». Тогда альтернативная гипотеза $\mathbf{H_1}$ такая: «Новая функциональность улучшает метрики».

На этапе проверки гипотез возможны ошибки двух типов:

1) **Ошибка первого рода** — когда нулевая гипотеза верна, но она отклоняется (ложно-*положительный* результат; новую функциональность приняли, поэтому *положительный*);

2) **Ошибка второго рода** — когда нулевая гипотеза не верна, но принимается (ложно-отрицательный результат).

Чтобы принять или отвергнуть нулевую гипотезу, вычислим знакомый вам уровень значимости — **p-value**. Он показывает вероятность ошибки первого рода. А об ошибке второго рода ничего не сообщает. Если значение *p-value* больше **порогового значения**, то нулевую гипотезу отвергать не стоит. Меньше — есть основание отказаться от нулевой гипотезы. Общепринятые пороговые значения — 5% и 1%. Но только от специалиста по Data Science зависит окончательное решение, какой порог считать достаточным.

Средние значения сравнивают методами проверки односторонних гипотез. В нашем случае одностороннюю альтернативную гипотезу принимают, если проверяемое значение намного больше принятого в нулевой гипотезе.

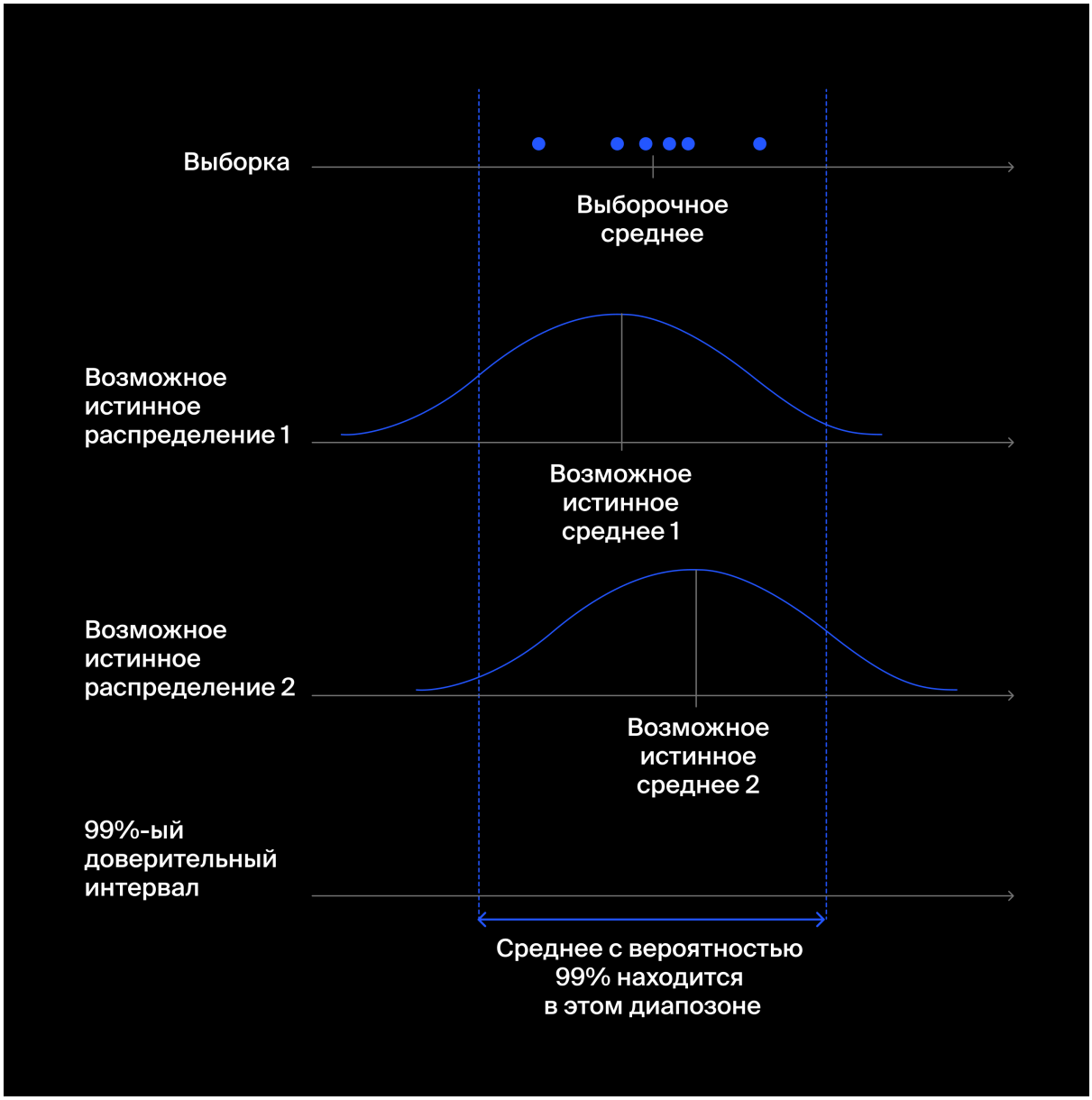
Если распределение данных близко к нормальному (в данных нет существенных выбросов), для сравнения средних значения используют стандартный *t*-тест. Этот метод предполагает нормальное распределение средних из всех выборок и определяет, достаточно ли велика разница между сравниваемыми значениями, чтобы отклонить нулевую гипотезу об их равенстве.

Доверительный интервал

Доверительный интервал (*confidence interval*) — отрезок числовой оси, в который с заданной вероятностью попадает нужный нам параметр генеральной совокупности. Параметр неизвестен, но его можно оценить по выборке. Если величина с вероятностью 99% попадает в интервал от 300 до 500, то 99%-й доверительный интервал для неё — это **(300, 500)**.

При вычислении доверительного интервала с каждой из его сторон обычно выбрасывают одинаковую долю экстремальных значений.

Доверительный интервал — это не просто диапазон значений случайной величины. Величина, которую мы оцениваем, изначально неслучайная. Вероятность возникает потому, что число неизвестно и оценивается по выборке. Случайность выборки вносит случайность и в оценку. Доверительный интервал оценивает уверенность в этой оценке.



Расчёт доверительного интервала

Исходя из выборки можно построить доверительный интервал для среднего с помощью центральной предельной теоремы.

Представим, что наша выборка взята из распределения с такими параметрами:

$$\mu - \text{среднее}$$

$$\sigma^2 - \text{дисперсия}$$

Обозначим оценку среднего:

$$\bar{X} - \text{оценка среднего}$$

Центральная предельная теорема говорит, что все средние всех возможных выборок размера *n* распределены нормально вокруг истинного среднего генеральной совокупности. «Вокруг» означает, что среднее этого распределения всех выборочных средних будет равно истинному среднему значению генеральной совокупности. Дисперсия будет равна дисперсии генеральной совокупности, делённой на n — размер выборки.

$$\bar{X} \sim \mathbf{N}\left(\mu, \frac{\sigma^2}{n}\right)$$

Стандартное отклонение этого распределения называется **стандартной ошибкой** (англ. *standart error of mean*):

$$\mathrm{SEM}(\bar{X}) = \frac{\sigma}{\sqrt{n}}$$

Чем больше размер выборки, тем меньше стандартная ошибка, то есть тем ближе к истинному среднему «прижимаются» все выборочные средние. Чем больше наша выборка, тем точнее оценка.

Стандартизируем это нормальное распределение:

$$\frac{\bar{X} - \mu}{\mathrm{SEM}(\bar{X})} \sim \mathbf{N}(0, 1^2)$$

У стандартного нормального распределения возьмём 5%-квантиль *F*(0.05) и 95%-квантиль *F*(0.95) для 90%-го доверительного интервала:

$$P\left(F(0.05) < \frac{\bar{X} - \mu}{\mathrm{SEM}(\bar{X})} < F(0.95)\right) = 90\%$$

Преобразуем:

$$P\left(\bar{X} - F(0.05) \cdot \mathrm{SEM}(\bar{X}) < \mu < \bar{X} + F(0.95) \cdot \mathrm{SEM}(\bar{X})\right) = 90\%$$

Это и есть 90%-й доверительный интервал для истинного среднего!

Осталась только одна проблема: при вычислении стандартной ошибки берут дисперсию генеральной совокупности, оценивать её нужно по выборке. Это влияет и на распределение выборочных средних: если дисперсия неизвестна, его нужно описывать уже не нормальным распределением, а распределением Стьюдента. Подставив в формулу 5%-квантиль *t*(0.05) и 95%-квантиль *t*(0.95), получаем:

$$P\left(\bar{X} - t(0.05) \cdot \mathrm{SEM}(\bar{X}) < \mu < \bar{X} + t(0.95) \cdot \mathrm{SEM}(\bar{X})\right) = 90\%$$

Упростить вычисления поможет распределение Стьюдента *scipy.stats.t*. В нём есть функция для доверительного интервала *interval()*, которая принимает на вход:

- *alpha* — уровень значимости;
- *df* (от англ. *degrees of freedom)* — количество степеней свободы, равное n - 1;

- loc (от англ. *location*) — среднее распределение, равное оценке среднего. Для выборки *sample* вычисляется так: `sample.mean()`;
- *scale* (англ. «масштаб») — стандартное отклонение распределения, равное оценке стандартной ошибки. Вычисляется так: `sample.sem()`.

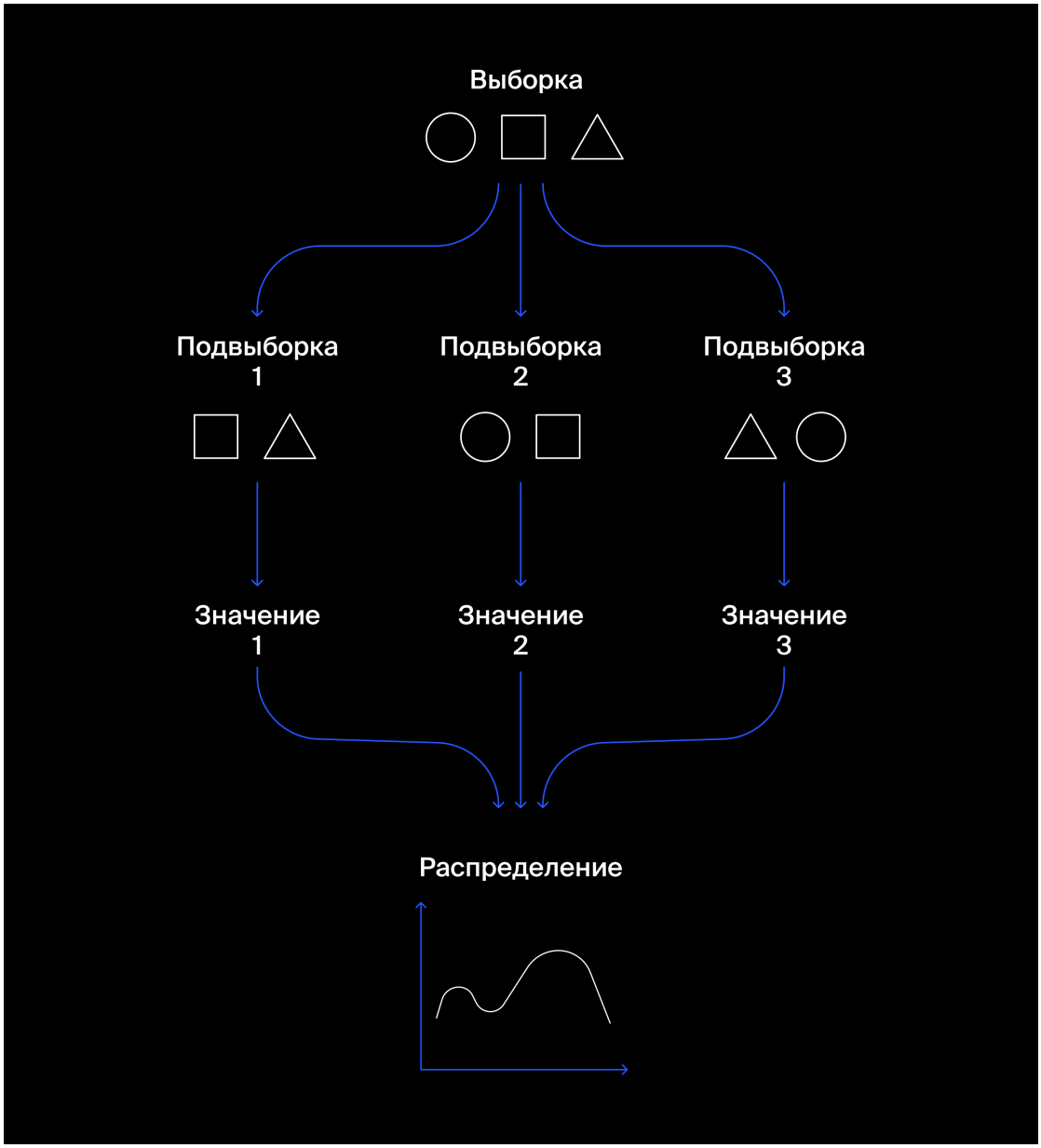
```
import pandas as pd
from scipy import stats as st

confidence_interval = st.t.interval(alpha, len(sample)-1,
                                    loc=sample.mean(), scale=sample.sem())
```

Bootstrap

Вычислить сложные величины, не прибегая к формулам, поможет техника **Bootstrap**.

Чтобы получить нужную величину, например, среднее, из исходного набора данных формируют подвыборки (псевдовыборки). На каждой из них и вычисляют среднее. Теоретически формировать подвыборки и рассчитывать по ним нужную величину можно многократно. Так мы получим несколько значений интересующего показателя и оценим распределение.



Бутстреп применим для любых выборок. Это полезно, когда:

- Наблюдения не описываются нормальным законом;
- Для искомых величин нет статистических тестов.

В действительности не стоит всегда рассчитывать на нормальное распределение.

Бутстреп для доверительного интервала

Бутстреп также применяется для оценки доверительного интервала. Узнаем, как создавать подвыборки для бутстрепа. Вам уже известна функция *sample()*. В этой задаче её нужно многократно вызвать в цикле. Но здесь возникает проблема:

```
for i in range(5):
    # извлекаем из выборки 1 случайный элемент
```

```
    # указываем random_state для воспроизводимости
    print(data.sample(1, random_state=12345))
```

Из-за указания `random_state` случайный элемент всегда одинаковый. Чтобы это исправить, создадим объект `RandomState()` из модуля `numpy.random`:

```
from numpy.random import RandomState
state = RandomState(12345)
```

Этот объект можно передавать аргументу `random_state` в любой функции. Важно, что при каждом новом вызове его состояние будет меняться на случайное. Так получаем разные подвыборки:

```
for i in range(5):
    # извлекаем из выборки 1 случайный элемент
    print(data.sample(1, random_state=state))
```

Ещё одна важная деталь при создании подвыборок — они должны обеспечить выбор элементов с возвращением. То есть один и тот же элемент может попадать в подвыборку несколько раз. Для этого укажите аргумент `replace=True` в функции `sample()`. Сравните:

```
# без возвращения
print(example_data.sample(frac=1, replace=False, random_state=state))
# с возвращением
print(example_data.sample(frac=1, replace=True, random_state=state))
```

Бутстреп для анализа A/B-теста

Bootstrap применяют и в анализе результатов A/B-теста.

Пока проводили тест, мы получали данные о показателях в контрольной и экспериментальной группах. Считаем *фактическую разницу целевых показателей* в группах. Затем формулируем и проверяем гипотезы. Нулевая предполагает равенство целевых показателей в обеих группах. Альтернативная — в экспериментальной группе целевой показатель выше. Найдём *p-value*.

Вычислим разницу целевого показателя по выборкам. Найдём вероятность того, что такую разницу получили случайно (это и будет *p-value*). Объединим выборки. Бутстрепом получим распределение среднего чека.

Создадим много подвыборок, каждую из которых с номером *i* разделим пополам:

A_i — первая половина выборки

B_i — вторая половина выборки

Найдём между ними разницу целевого показателя:

Оценим долю разниц целевых показателей в бутстрепе, которые оказались не меньше, чем разницы целевых показателей между исходными выборками:

$$p\text{-value} = P(D_i \geq D)$$

```
import pandas as pd
import numpy as np

# фактическая разность средних значений в группах
AB_difference = samples_B.mean() - samples_A.mean()

alpha = 0.05

state = np.random.RandomState(12345)

bootstrap_samples = 1000
count = 0
for i in range(bootstrap_samples):
    # подсчитываем, сколько раз разница целевых показателей
    # фактическое значение при условии верности нулевой гипотезы
    united_samples = pd.concat([samples_A, samples_B])
    subsample = united_samples.sample(frac=1, replace=True, random_state=state)

    subsample_A = subsample[:len(samples_A)]
    subsample_B = subsample[len(samples_A):]
    bootstrap_difference = subsample_B.mean() - subsample_A.mean()

    if bootstrap_difference >= AB_difference:
        count += 1

pvalue = 1. * count / bootstrap_samples
print('p-value =', pvalue)

if pvalue < alpha:
    print("Отвергаем нулевую гипотезу: скорее всего, целевой показатель увеличился")
```

```
else:
    print("Не получилось отвергнуть нулевую гипотезу: скорее всего, целевой показатель не увеличился")
```

Бутстреп для моделей

Процедурой *Bootstrap* можно оценить доверительные интервалы для моделей машинного обучения.