Санкт-Петербургский государственный университет

МАРЦИНКОВСКАЯ Наталья Алексеевна

Выпускная квалификационная работа

Исследование A/B-тестов в разработке мобильных игр

Уровень образования: бакалавриат
Направление *02.03.01 "Математика и компьютерные науки"*
Основная образовательная программа *СВ.5152.2019 "Математика, алгоритмы и анализ данных "*

Научный руководитель:
доцент, Факультет математики
и компьютерных наук СПбГУ,
Авдюшенко Александр Юрьевич

Рецензент:
менеджер количественных исследований
«МАЙ.ГЕЙМЗ Би.Ви.»,
Северов Александр Анатольевич

Санкт-Петербург
2023

Содержание

1. Введение

1.1. Мотивация.

Для начала определим, что такое A/B тестирование и как его проводить.

Определение 1. A/B тестирование – это методология исследования пользовательского поведения, состоящая из рандомизированного эксперимента с двумя вариантами, A и B, одной и той же сущности. Задача – сравнить, какой из вариантов лучше (понятие «лучше» определяется аналитиками заранее с учетом специфики поставленной задачи). [1]

Формально процесс A/B тестирования выглядит следующим образом:

- Определяем изменение в нашем продукте(в нашем случае - в игре) и метрику, которую будем сравнивать в конце.
- Формулируем нулевую гипотезу: чаще всего, это гипотеза о том, что изменения не повлияют на метрику.
- Делим пользователей на две независимые группы с одинаковыми важными для нас показателями, одной из групп показываем вариант A нашего продукта, другой – вариант B.
- Сравниваем выбранную нами метрику на этих двух группах с помощью статистического теста (в зависимости от метрики – свой тест)
- Делаем вывод – какой вариант продукта лучше для нас.

Процесс A/B тестирования широко применяем в IT-сфере и дает достаточно достоверные результаты, но есть важные нюансы. Во-первых, проводить A/B тесты - долго и дорого, требуется много ресурсов. Во-вторых, нередко значимая разница между вариантах отсутствует, поэтому необходим большой объем выборки, а данные должны быть качественнее, чего мы не всегда можем добиться.

1.2. Задача.

Тема моей работы - "Исследование A/B тесты в разработке мобильных игр". Основной задачей является улучшение текущей системы работы с A/B тестами в команде и изучение различных методологий A/B тестирования.
Рассмотрим некоторую игру N, в которой есть несколько уровней сложности и различные активности на этих уровнях. У нас есть базы данных с информацией

о пользователях (их уникальные коды (id), дата установки игры и активность, платежи и статусы прохождения уровней).

Введем несколько определений из игровой индустрии:

- **Киты (топ донатеры)** - самые платящие игроки (в контексте сортировки игроков по средним/общим платежам, чаще всего смотрим конкретный процент от всех игроков)
- **ARPU (average revenue per user (в долларах))** – продуктовая метрика, показывающая средний доход с игрока (средние траты игрока), чаще всего за месяц

В нашей команде часто бывают небольшие эффекты, а также некачественные данные, которые еще и долго и сложно собирать, поэтому возникают вопросы: корректно ли мы используем A/B тесты и можно ли в некоторых случаях обойтись без них.
В данной работе представлены результаты по следующим задачам:

1. Проверка применимости статистических тестов.
2. A/B тест без непосредственного проведения A/B теста.
3. Влияние распределения улучшения от фичи на A/B тесты.

2. Основные результаты

2.1. Проверка применимости статистических тестов.

При проведении A/B-тестирования в команде использовались 2 статистических теста - для сравнения средних (`scipy.stats.ttest_ind`) и для сравнения пропорций (`proportions_ztest`). Задача - проанализировать применимость этих тестов для наших данных.
Формат данных: id игрока (в порядке даты установки игры) и средний платеж.

Алгоритм анализа:

1. Делим данные на две группы.

   - Используя хэш-функцию (вычисляется значение хэша для каждого id, потом берется остаток от деления на 2)
   - Рандомно

   Получаем id с принадлежностью к группе (либо A, либо B).

2. Делим все данные на подгруппы последовательных id (размеры – 100000, 10000, 1000), из них смотрим, какие id какой группе принадлежат.

3. Для каждой из подгрупп:
   Моделируем A/A тест (как A/B тест, только у нас нет изменений в продукте) и получаем p-value. [2]

   - **тест на сравнение средних платежей:**
     Можем либо выкинуть некоторый процент китов для каждой группы отдельно (около 20), либо нет. После чего применяется `ttest` для сравнения средних платежей в двух группах.
   - **тест на сравнение долей платящих:**
     Считаем число ненулевых платежей в каждой группе и общее их количество. После чего, используя `proportions_ztest` сравниваем доли.

   Получаем значение p-value для каждой из подгрупп, объединяем в массив.

4. По полученному массиву p-value, строим эмпирическую функцию распределения для него, используя `sm.tools.ECDF(data)`

Для верификации результатов используем следующий факт: если статистический тест применим к нашим данным, то полученное распределение должно получаться равномерным (мы сами это выводили, но потом оказалось, что доказанный факт уже есть). (доказательство приведено в [4])

Исследования проводились для следующих групп параметров:

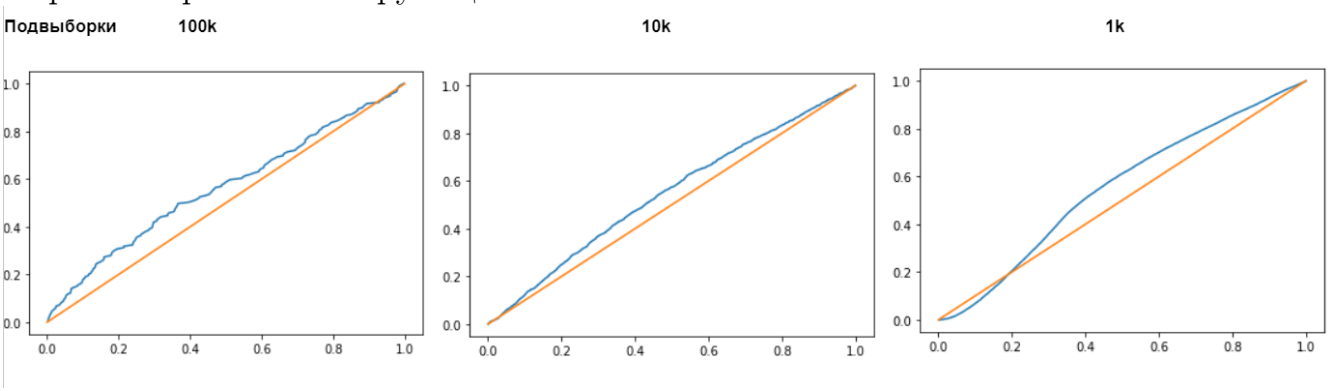
1. Тест на сравнение средних платежей: платежи за 1, 30 и 90 дни с момента установки, с 3 разными хэш-функциями и рандомно, с выкидыванием 20% китов и без.

2. Тест на сравнение долей платящих: за 30 дней, разбивка на группы с хэш-функцией и по модулю 2 или 16 (для id)

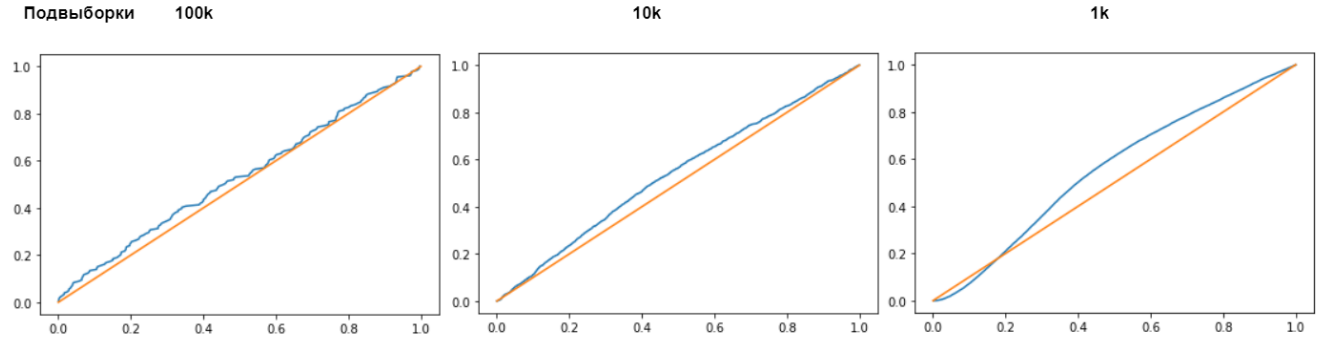
Результаты

Платежи за 1 день с момента установки, с выкидыванием 20% китов

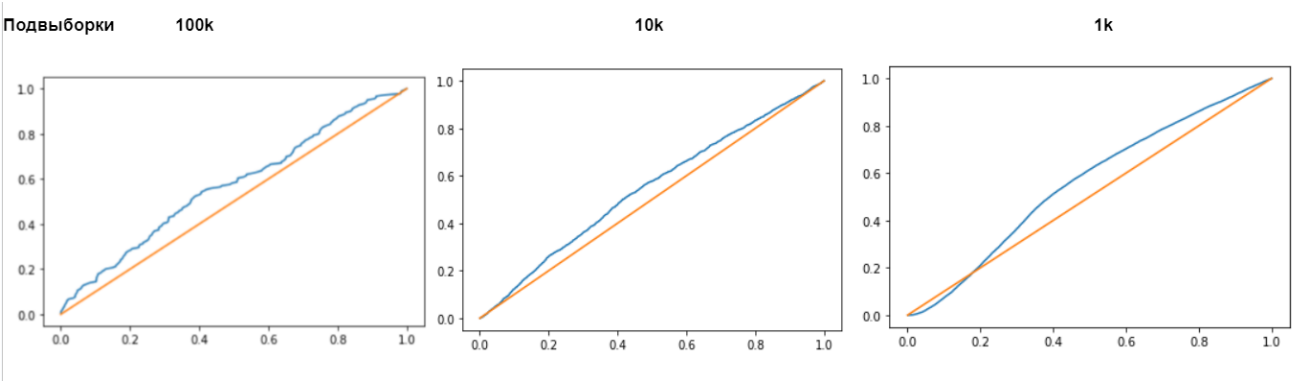
Первый вариант хэш-функции



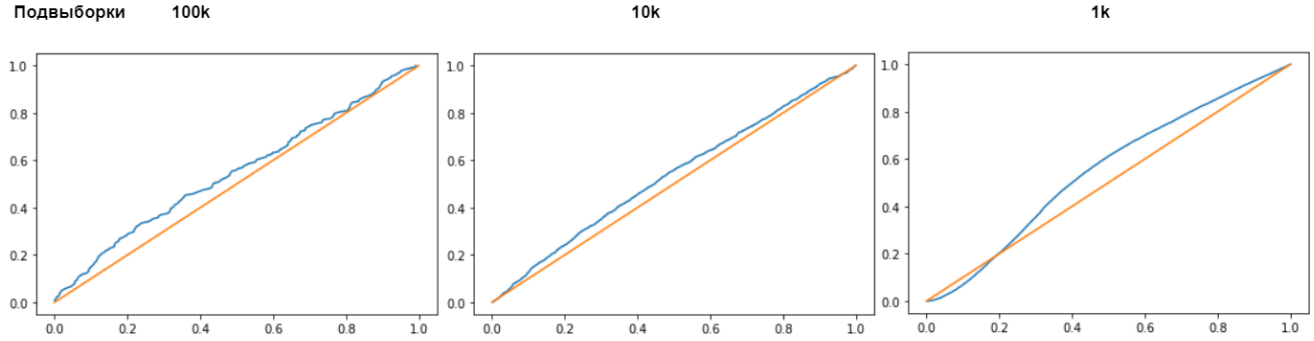
Второй вариант хэш-функции



Третий вариант хэш-функции

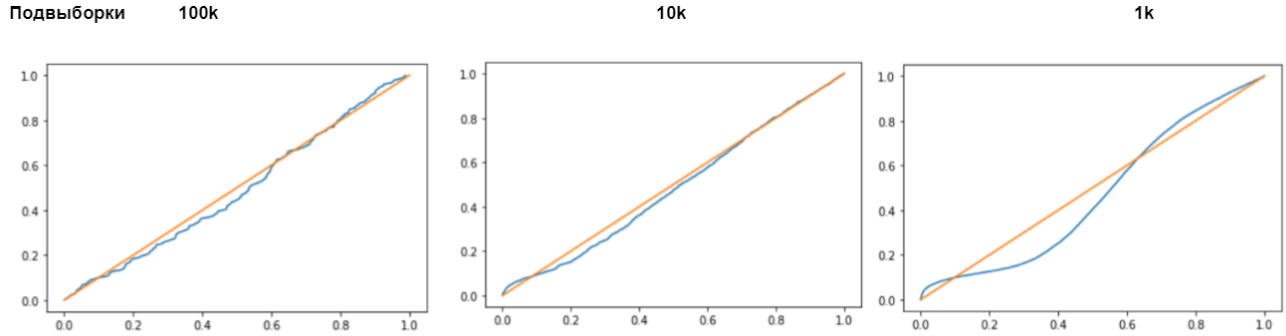


Разбивка на группы по модулю 2



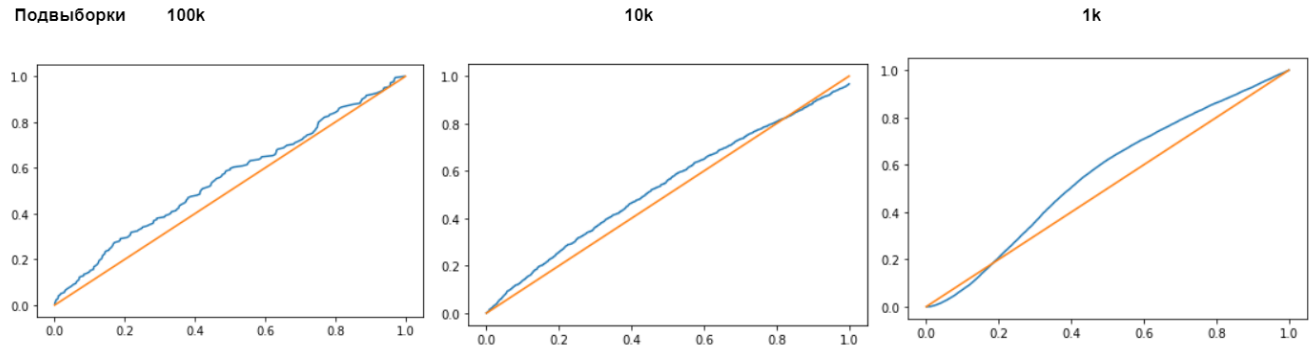
Получается, что от способа разбивки на группы результаты эксперимента не меняются (для других параметров были проведены аналогичные эксперименты, результаты для краткости приводить не буду). Далее будем рассматривать разбивку на группы только с помощью 1 хэш-функции.

Платежи за 1 день с момента установки, без выкидывания китов

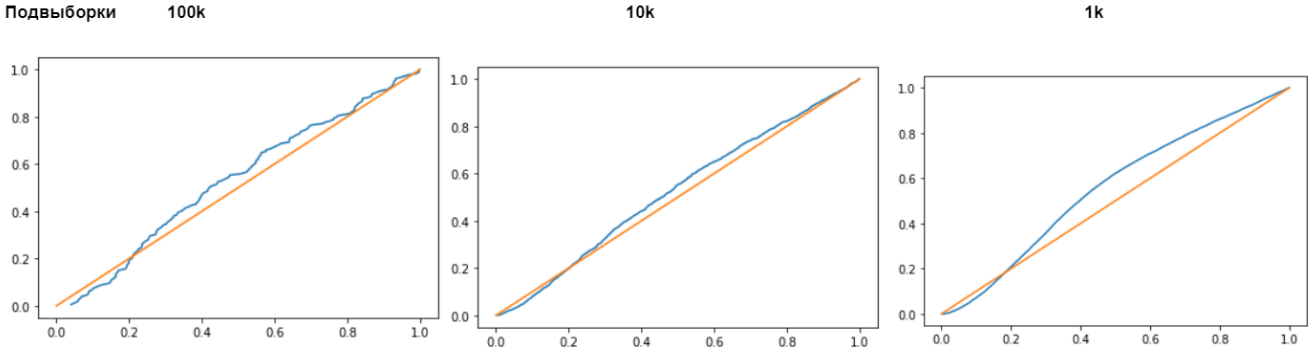


Видно, что графики противоположны предыдушим графикам (с выкидыванием китов), но в любом случае для малого значения `p-value` получается не равномерное распределение.

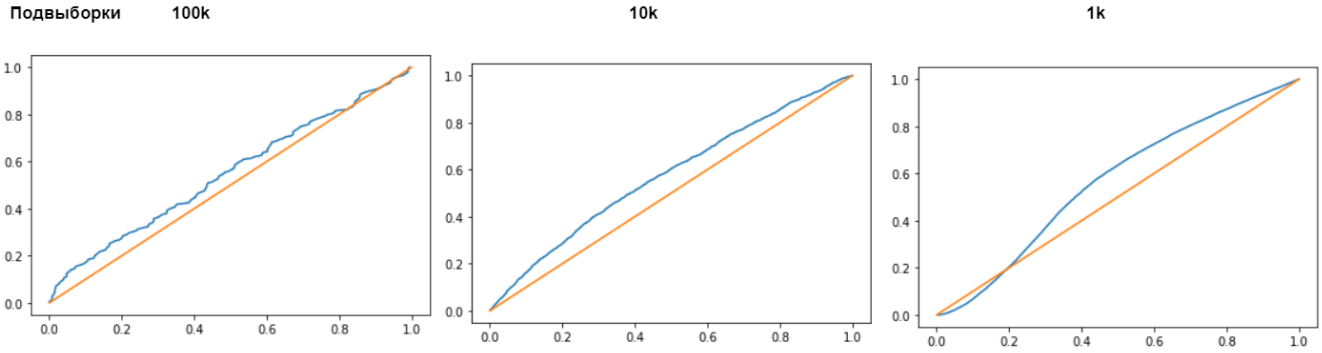
Платежи за 30 дней с момента установки, с выкидыванием 20% китов



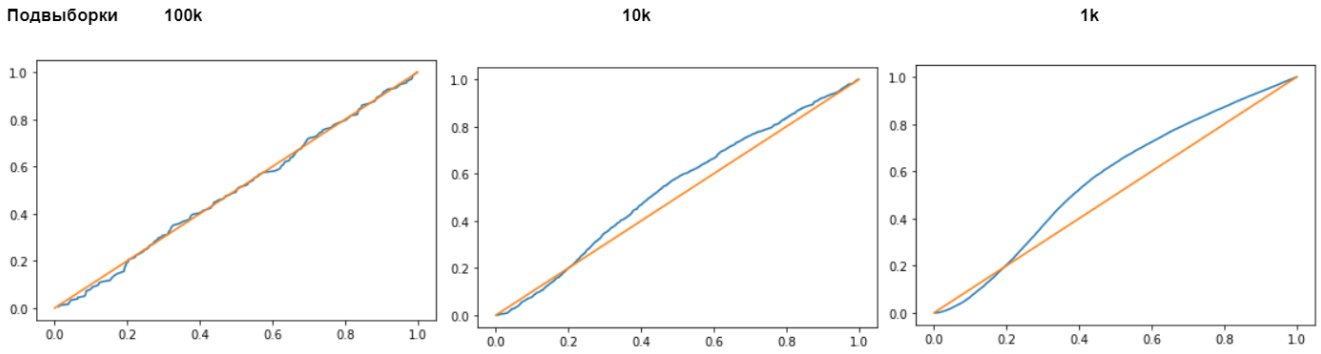
Платежи за 30 дней с момента установки, без выкидывания китов



Платежи за 90 дней с момента установки, с выкидыванием 20% китов



Платежи за 90 дней с момента установки, без выкидывания китов



Выводы: Без выкидывания китов – видно, что отвергаем гипотезу чаще, чем нужно, для платежей больших дней, для платежа первого дня – чаше принимаем.
При выкидывании китов – видно, что при более крупных разбиениях мы (при уровне значимости 0.05 обычно) гипотезу чаще принимаем, чем нужно, а при более малых – отвергаем больше, чем нужно.
Далее были проведены аналогичные исследования, но со скорректированным тестом Стьюдента:

- для неравных дисперсий тест Стьюдента с коррекцией Уэлча, устанавливаем параметр `equal_var=False`. Тогда p-value вычисляется так:

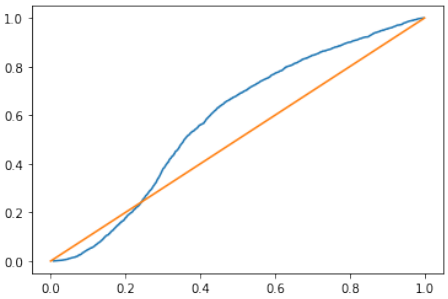
```
p = ttest_ind(group_a, group_b, equal_var=False)[1]
```

- для несимметричного распределения тест Стьюдента с коррекцией Саттертуэйта, устанавливаем параметр `nan_policy='omit'`. Тогда p-value вычисляется так:

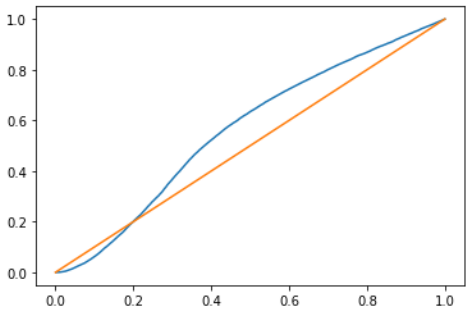
```
p = ttest_ind(group_a, group_b, nan_policy='omit')[1]
```

В целом, как и предполагалось, данные корректировки не улучшили ситуацию:

Рис. 1. Платежи за 30 дней с момента установки



Коррекция Уэлча



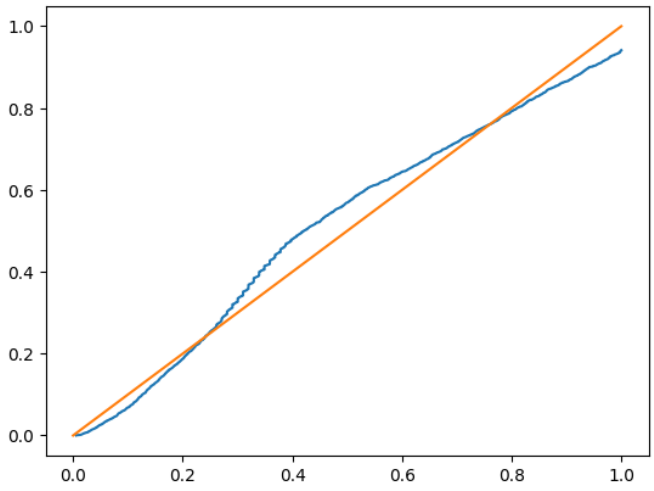
Коррекция Саттертуэйта

Далее было проведено исследование распределения наших данных, оно оказалось близко к логнормальному, поэтому было решено прологарифмировать данные, чтобы приблизить их распределение к нормальному. Большая часть данных - нули, поэтому сначала их сдвигаем на 1. Т.о. преобразование данных следующее:

```
np.log(data + 1)
```

Это не исправило ситуацию:

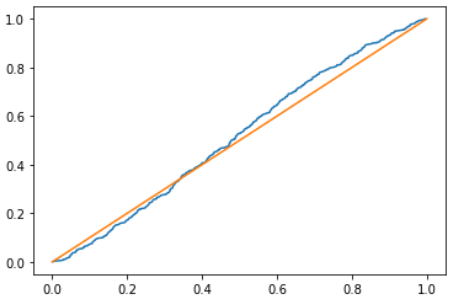
Результаты на логарифмированных данных



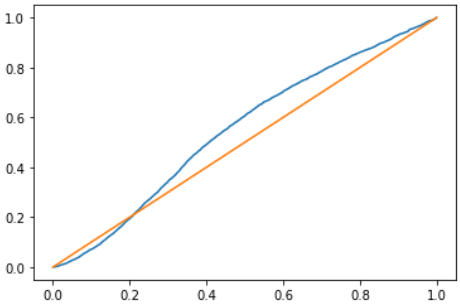
В целом, с таким подходом возникает еще много проблем, например, с тем, как верифицировать результаты тестов, ведь при логарифмировании возникают искажения в данных. Да, равенство средних соблюдается, но с отклонениями и знаками могут быть трудности.

Далее рассматриваются платежи не с момента установки игры, а с момента достижения определенного уровня: 5, 8 и 11 (он же максимальный) за 180 дней. Приведу только результаты на подвыборках размера 1000 и 100, поскольку данных мало.

Платежи за 180 дней с момента достижения 5 уровня

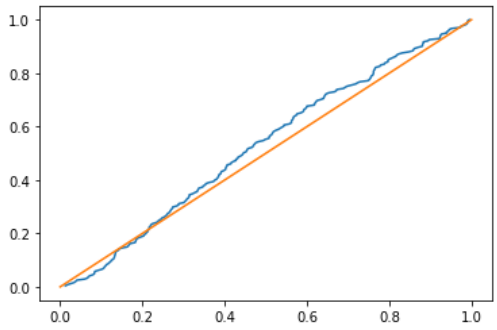


Подвыборки размера 1000

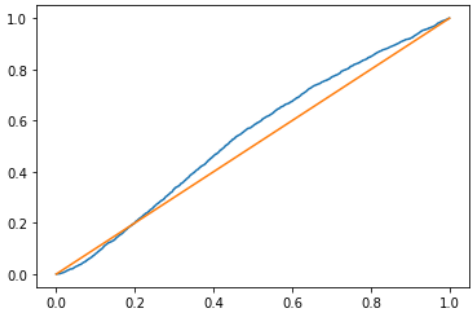


Подвыборки размера 100

Платежи за 180 дней с момента достижения 8 уровня

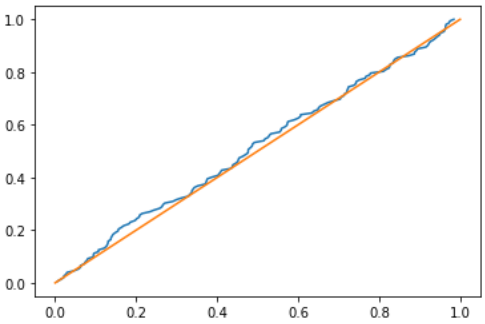


Подвыборки размера 1000

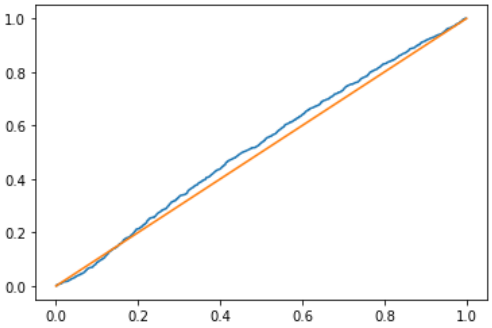


Подвыборки размера 100

Платежи за 180 дней с момента достижения 11 уровня



Подвыборки размера 1000

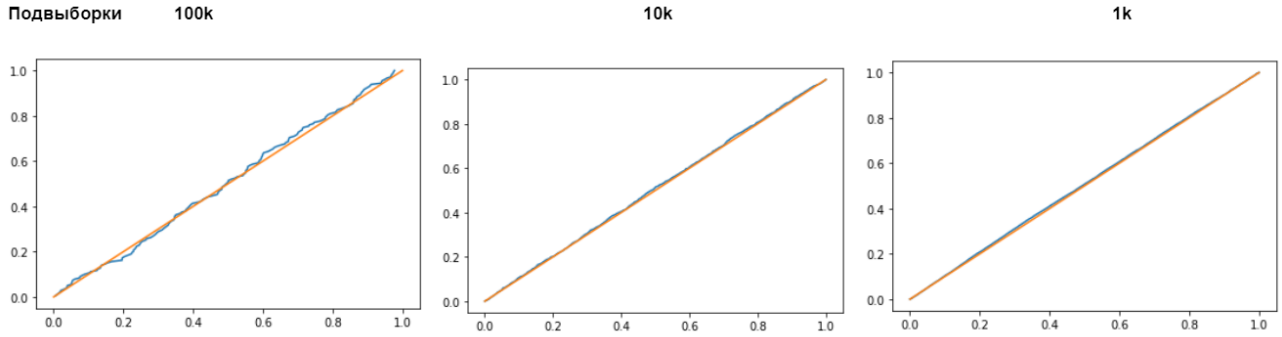


Подвыборки размера 100

Видно, что на малых уровнях ситуация аналогична тому, что мы видели ранее. Но на более высоких уровнях, распределение p-value все больше похоже на равномерное (особенно в интересующей нас части, где значение меньше 0.05). Это связано с тем, что на более высоких уровнях игроки в целом больше платят, уже мало тех, кто не заплатил ничего и распределение платежей становится лучше с точки зрения скорости сходимости сумм к нормальному.

Далее проведем эксперименты с долей платящих. С ней все получилось намного лучше:

Тест на сравнение доли платящих (за 30 дней)



Выводы: На всех других экспериментах результаты получились такими же.

Здесь уже на размере подгрупп 10к распределение p-value практически равномерно. Значит, тест на сравнение долей мы можем использовать спокойно.

2.2. A/B тест без непосредственного проведения A/B теста.

Задача: выяснить, хорошо или плохо для компании то, что игроки быстро проходят уровни. В нашем случае хорошо - приносят много денег. Вопрос – какие игроки больше платят в среднем, те, кто быстро проходят уровни или те, кто медленно.
Мотивация задачи в том, что может быть имеет смысл механически удлинить время прохождения уровней или наоборот уменьшить, чтобы проект принес больше денег.

Почему A/B тест без A/B теста? У нас уже есть информация по всем игрокам, мы хотим по ней проанализировать поведение игроков и доход с них.

Подробнее про анализ задачи:

Пускай мы анализируем игроков на 8 уровне игры. Тогда:

1. Рассмотрим игроков с примерно одинаковыми начальными характеристиками, такими как дата установки (в одном промежутке, чаще всего – неделя), среднее время в игре, платящий игрок или нет, среднее время игровой сессии и т.д.

2. Из этой группы игроков выберем тех, кто дошел до 8 уровня быстро (т.е. быстрее, чем медианное время достижения 8 уровня в группе). Так мы выбрали игроков, максимально похожих друг на друга до 8 уровня.

При проведении A/B теста после того, как мы нашли максимально похожих пользователей, мы их рандомно делим на две группы и показываем разные варианты какой-то части игры. Поскольку мы не проводим A/B тест, а считаем, что он как бы уже произошел, нам нужно разделить полученную ранее группу игроков на две так, чтобы выполнялись следующие условия:

1. В одной группе игроки прошли 8 уровень быстрее медианного времени прохождения этого уровня (именно прохождения, мы анализируем уровень), а в другой – медленнее.

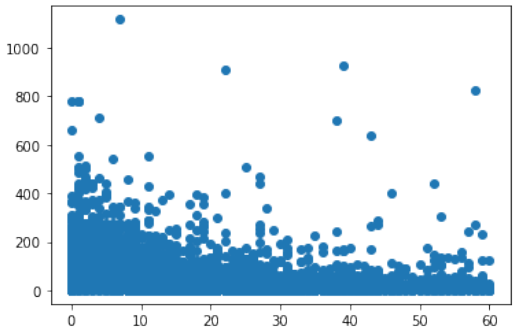
2. Средние показатели до 8 уровня у двух получившихся групп должны совпадать .

Таким образом мы получим две группы, которые были одинаковыми по характеристикам до 8 уровня, а 8 уровень проходили с разной скоростью. Моя задача

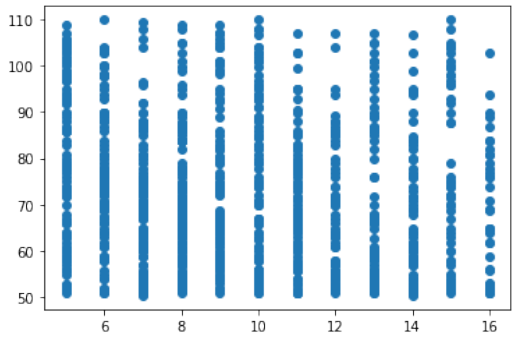
была в том, чтобы выбрать начальную группу игроков так, чтобы после разделения на группы по времени прохождения уровня выполнялось условие 2.

Первый вариант того, как это можно сделать - использовать кластеризацию. Делим игроков на кластеры, в каждом кластере делим игроков на две группы по медианному времени и смотрим показатели до 8 уровня. Я попробовала несколько вариантов кластеризации (`k-means`, `kNN`, иерархическая). Но все варианты не привели к успеху – после деления на две группы получалось так, что в них было отличие изначально (до 8 уровня).

После этого я решила попробовать выбрать группу вручную – для этого понадобился график (`scatter plot`) платежей/времени прохождения до 8 уровня (Рис.2 а). Для удобства сразу отбросим выбросы - тех, кто проходил до 8 уровня более 60 дней (медианное время достижения 8 уровня - 14 дней).



а) без ограничений



б) с ограничениями на платежи и время

Рис. 2. График платеж/время прохождения до 8 уровня

Идея выбора начальной группы следующая: выбираем наиболее плотный участок графика (т.е. где точки практически равномерно распределены по вертикали и горизонтали), после чего выбираются границы для платежей и времени прохождения до 8 уровня. (Рис.2 б)

Такой вариант показал лучшие результаты, хоть и количество людей в группах не очень большое.

	Size	ARPU	Session_time	Av. Time reaching 8th level (days)	8-th level time (days)	ARPU (on 8-th lvl)	ARPU (after 8-th lvl)
Quick	4702	74.7	1.2	9.2	8.2	97.6	498.3
Slow	4695	73.6	1.4	9.5	22.9	78.5	256.4

Рис. 3. Результаты

Первые 4 столбца – данные «ДО» 8 уровня (размер групп, средние платежи, среднее время сессии и время достижения 8 уровня), остальные – «ПОСЛЕ».

Видно, что значения «ДО» практически совпадают, а время прохождения 8 уровня отличается более, чем в два раза. Также показатели «ПОСЛЕ» сильно отличаются. Из этого можно сделать вывод, что быстрые игроки в среднем приносят больше денег проекту. Однако может быть и такое, что есть и другая причинно-следственная связь: те игроки, которые больше платят, проходят уровни быстрее. Это пока еще открытый вопрос для изучения.

2.3. Влияние распределение улучшения от фичи на A/B тесты.

Поскольку проверка гипотез с помощью A/B тестов - процесс дорогостоящий и трудозатратный, хочется понимать, в каких случаях проводить его не обязательно. Бывают ситуации, в которых мы вложим больше ресурса в проведение эксперимента, чем получим прибавку от фичи. В таких кейсах проводить A/B-тестирование не разумно. [3]

Будем считать, что вся кривая LTV примерно пропорциональна с самого начала. Пусть начальное значение arpu равно 1(нам важно только то, во сколько раз оно увеличится, поскольку у нас мультипликативная форма). Рассмотрим группу фичей, чье улучшение имеет симметричное треугольное распределение.

Далее моделируем проведение A/Б теста:
Первый шаг: сравниваем нормальное распределение с матожиданием 1 и нормальное распределение с матожиданием 1*k, где k – коэффициент (как раз улучшение от фичи), он генерируется нами случайно; коэффициент берется из треугольного распределения (максимальное значение коэффициента - параметр);

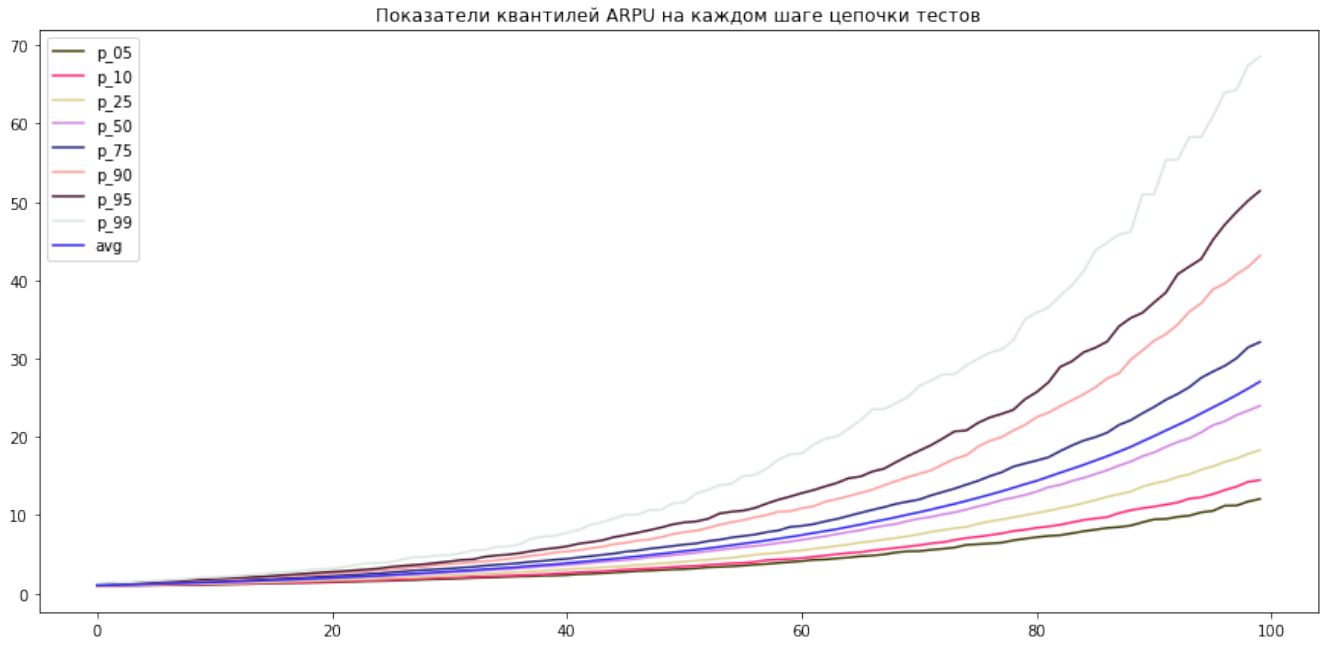
Будем рассматривать несколько вариантов теста:

1. Выбираем всегда бОльшее значение
2. Выбираем всегда тестовое значение
3. Делаем выбор на основе статзначимости, если она есть – берем тот, что больше, если ее нет – берем тестовое значение.

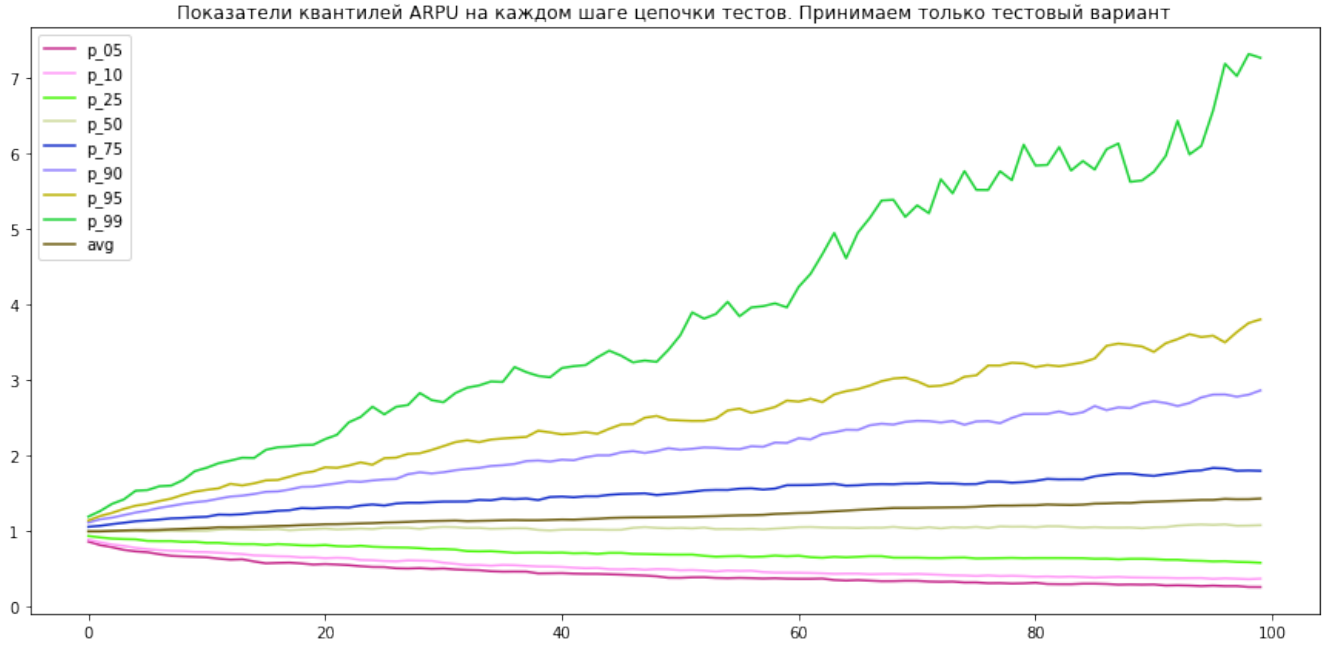
На следующем шаге мы уже сравниваем распределение с “победившими” параметрами (матожидание m) и распределение с матожиданием m*k (k – коэффициент, который снова сгенерирован случайно из распределения).

Проведем по такому алгоритму цепочку из 100 A/B тестов.
После этого, повторим алгоритм 1000 раз (получим 1000 цепочек из 100 A/B тестов).
Построим график квантилей нашего матожидания на каждом шаге из цепочки тестов:

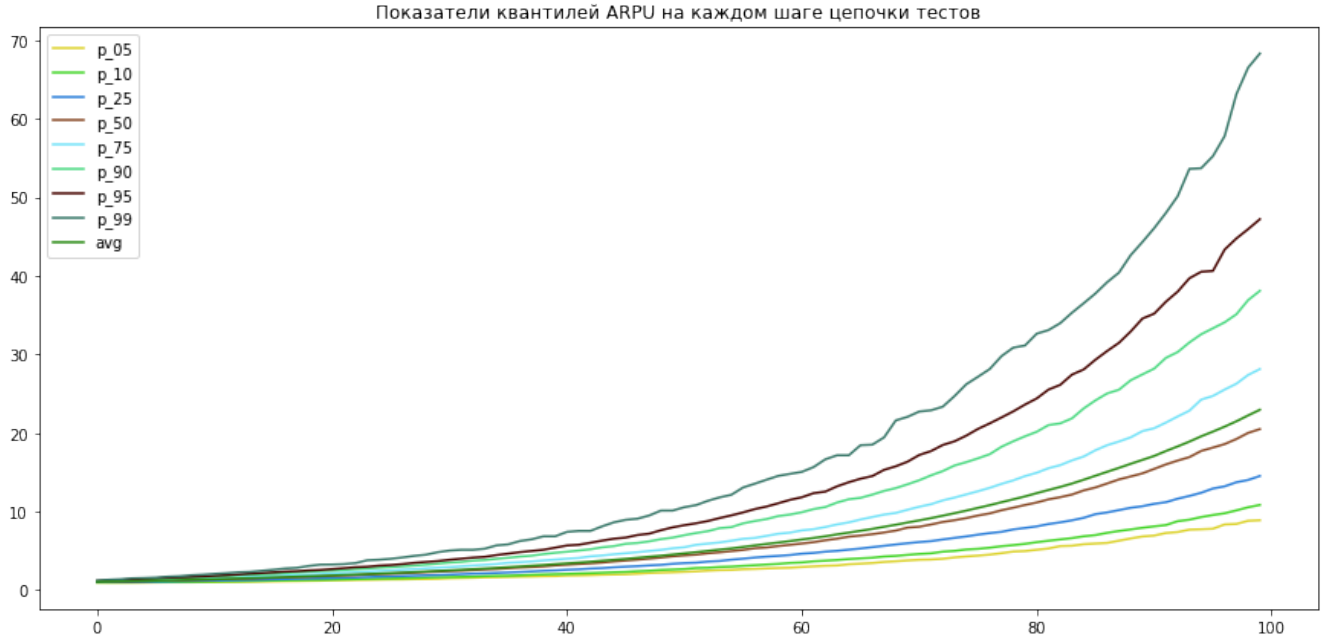
а) Принимаем всегда в бóльшую сторону



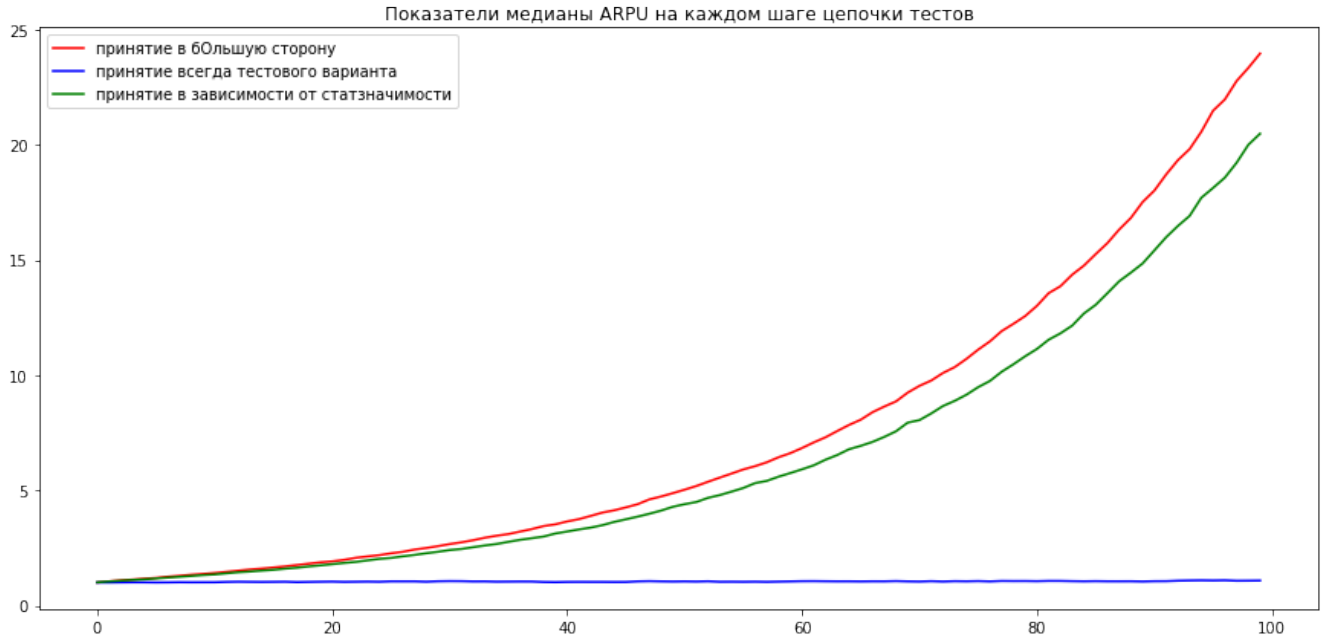
б) Принимаем всегда тестовый вариант



в) Принимаем в зависимости от статзначимости



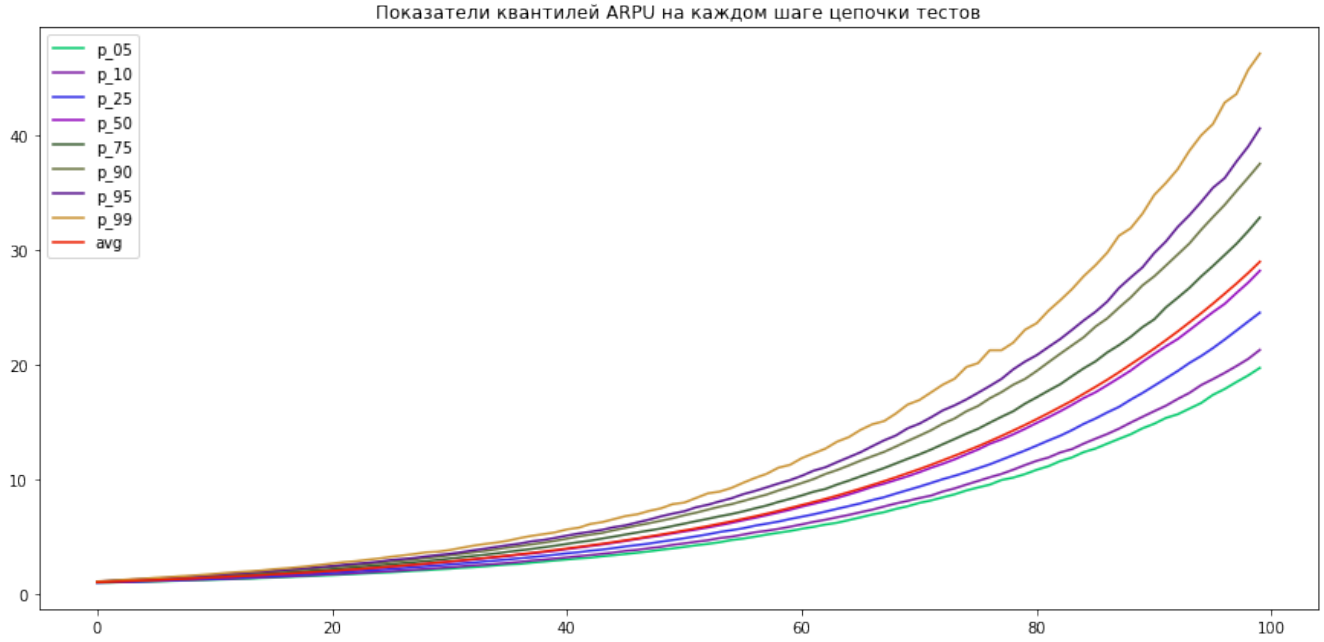
Смотрим на медианные значения для трех вариантов принятия решений:



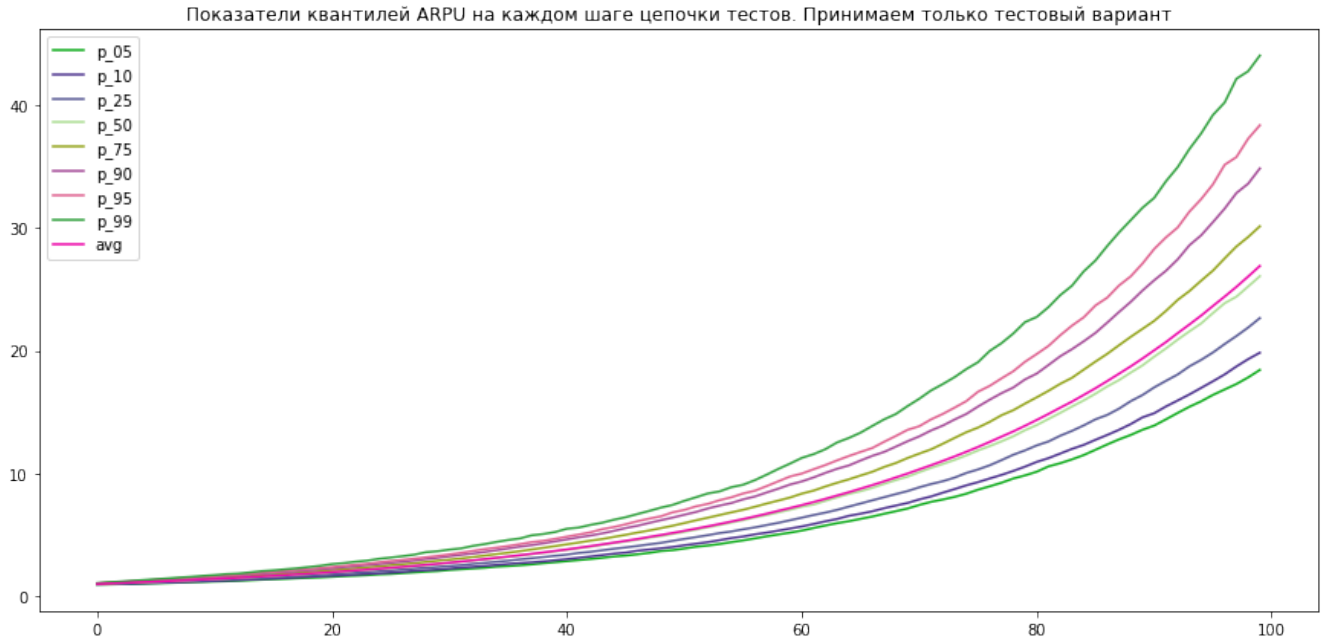
До этого мы рассматривали симметричные распределения фичей, но было бы странно думать, что геймдизайнеры делают абсолютно рандомные улучшения (с вероятностью 0.5 они улучшают ситуацию, с вероятностью 0.5 - ухудшают). Поэтому мы решили рассмотреть группу фичей, в которых геймдизайнеры уверены на 95%. Конкретные примеры я привести не могу, поскольку это NDA.

Посмотрим на результаты:

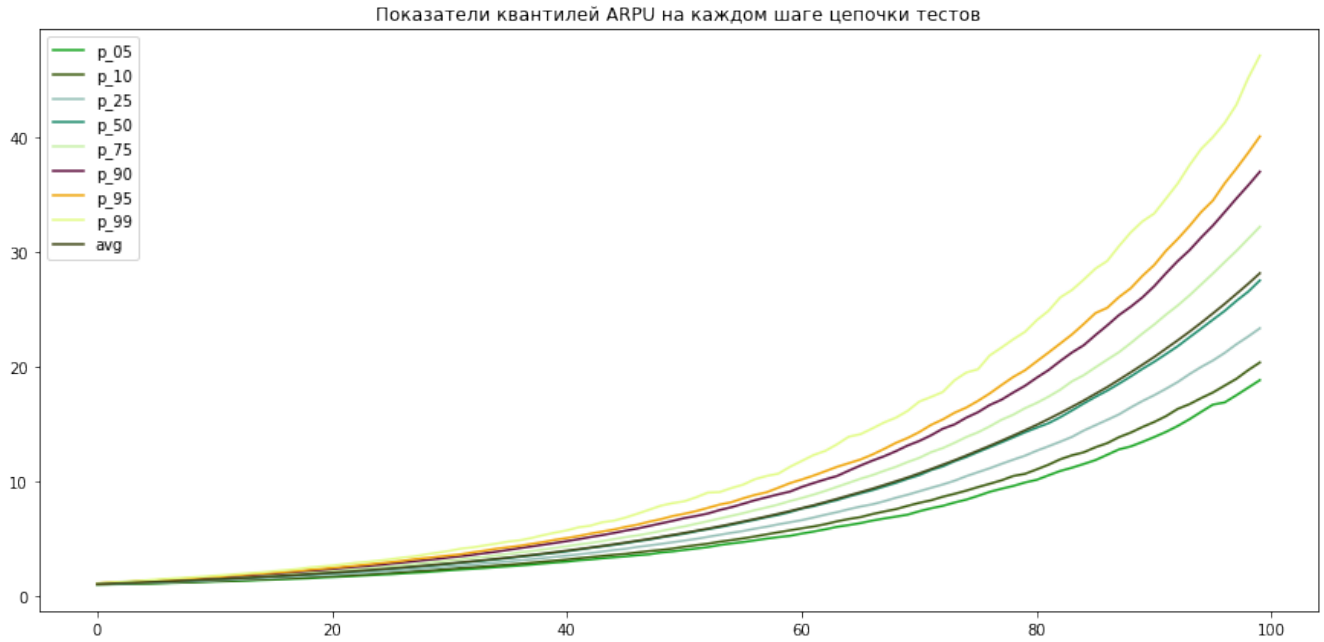
а) Принимаем всегда в бóльшую сторону



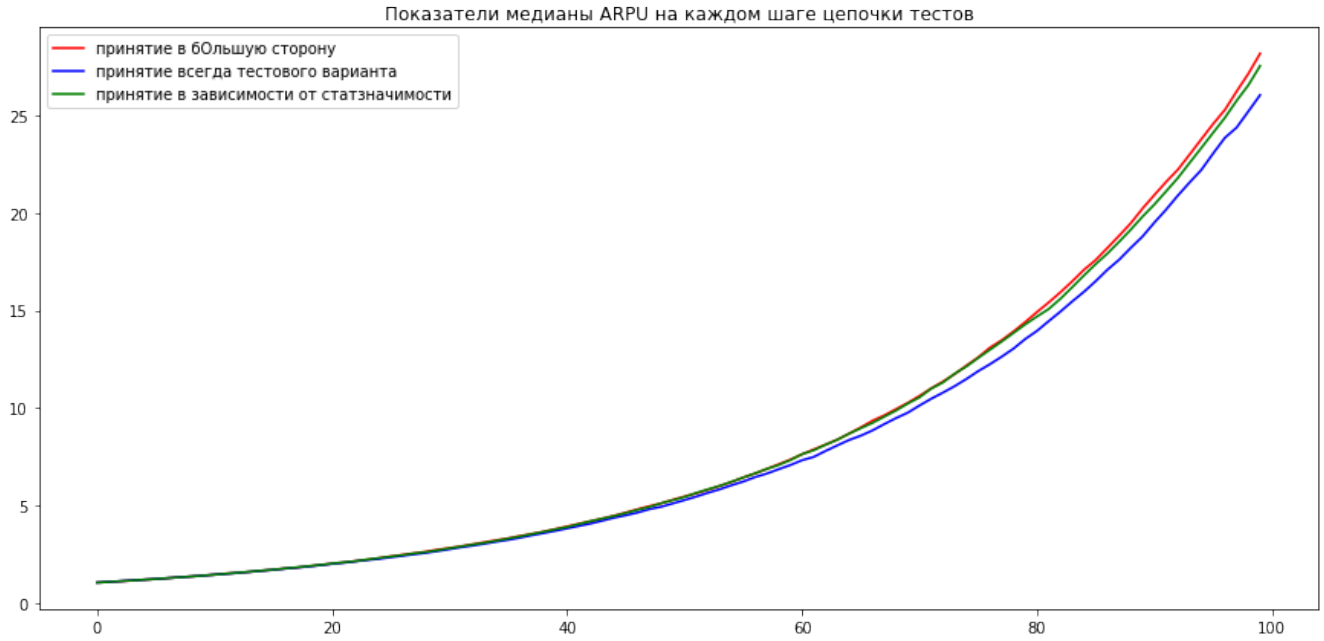
б) Принимаем всегда тестовый вариант



в) Принимаем в зависимости от статзначимости



Смотрим на медианные значения для трех вариантов принятия решений:



В целом для такой группы фичей можно сделать вывод, что проводить A/B тест не обязательно, можно сэкономить и время, и деньги и сразу принимать изменения, потому что проведение A/B теста может повлечь за собой ошибку в принятии решений из-за рандома.

3. Заключение

В данной работе рассмотрены важные для команды исследования в области A/B тестов.
Начальная задача, связанная с применимостью статистических тестов, важна, поскольку с помощью нее мы можем понять, насколько грамотно мы верифицируем результаты тестов. В первом пункте мы поняли, что с использованием теста на пропорции `proportions_ztest` на наших данных проблем не возникает, поэтому мы можем быть уверены в корректности его использования. С использованием же `ttest_ind` в A/B тестах с платежами нужно быть аккуратнее, а также внимательнее анализировать его результаты. Эти исследования показали нам новые направления для изучения и новые подходы к проведению A/B тестов на наших данных.

Исследование "A/B-тестов без проведения A/B-тестирования" дало возможность нашей команде проводить аналитику не только "на будущее", но и по уже имеющимся данным, если возникает такая необходимость. Это важно, ведь иногда могут быть ситуации (как, например, описано в работе), когда нужно сравнить две группы игроков, отличающихся только одной характеристикой, но при этом нет возможности разделить их и запустить A/B-тест (нехватка времени/ресурсов/etc).

В последней части работы исследованы различные варианты принятия решений и влияние фичей на них. Найдена группа улучшений, для которых не обязательно проводить A/B тестирование и можно сохранить время и ресурсы, что несомненно важно для команды.

Список литературы

[1] Anas Shallah Jing Zhou Jiannan Lu. *All about sample-size calculations for A/B testing: Novel extensions and practical guide*. https://arxiv.org/abs/2305.16459. 2023.

[2] Elena Kulinskaya. *On two-sided p-values for non-symmetric distributions*. https://arxiv.org/pdf/0810.2124.pdf. 2008.

[3] Manuel J. A. Eugster Shafi Kamalbasha. *Bayesian A/B Testing for Business Decisions*. https://arxiv.org/abs/2003.02769. 2020.

[4] Соколиков Е.А Мальцев А.С. *Проверка применимости критерия с помощью моделирования (равномерное распределение p-value)*. http://www.statmod.ru/wiki/_media/study:fall2020:probmodel:pvalue.pdf. 2019.