
Мини-тест по пройденному материалу

- Ссылка: <https://forms.gle/j4NHHS4usDHChsXC7>

FeedBack по занятию АВТ-2:

- Что понравилось:
 - Наличие практических примеров кода;
 - Наличие теоретического материала;
- Что хотелось бы улучшить:
 - Больше кода и примеров для практики вместо теории;
 - Конспект и план схема;
- Доходчивость и простота: среднее - 7, max - 27% и 27% ответов – 7 и 8;
- Полнота: 8, max – 40% ответов – 9;
- Новизна: 8,25, max – 37 ,5% ответов – 10.

ТЕХНОАТОМ: Продуктовая аналитика.
A/B-тестирование

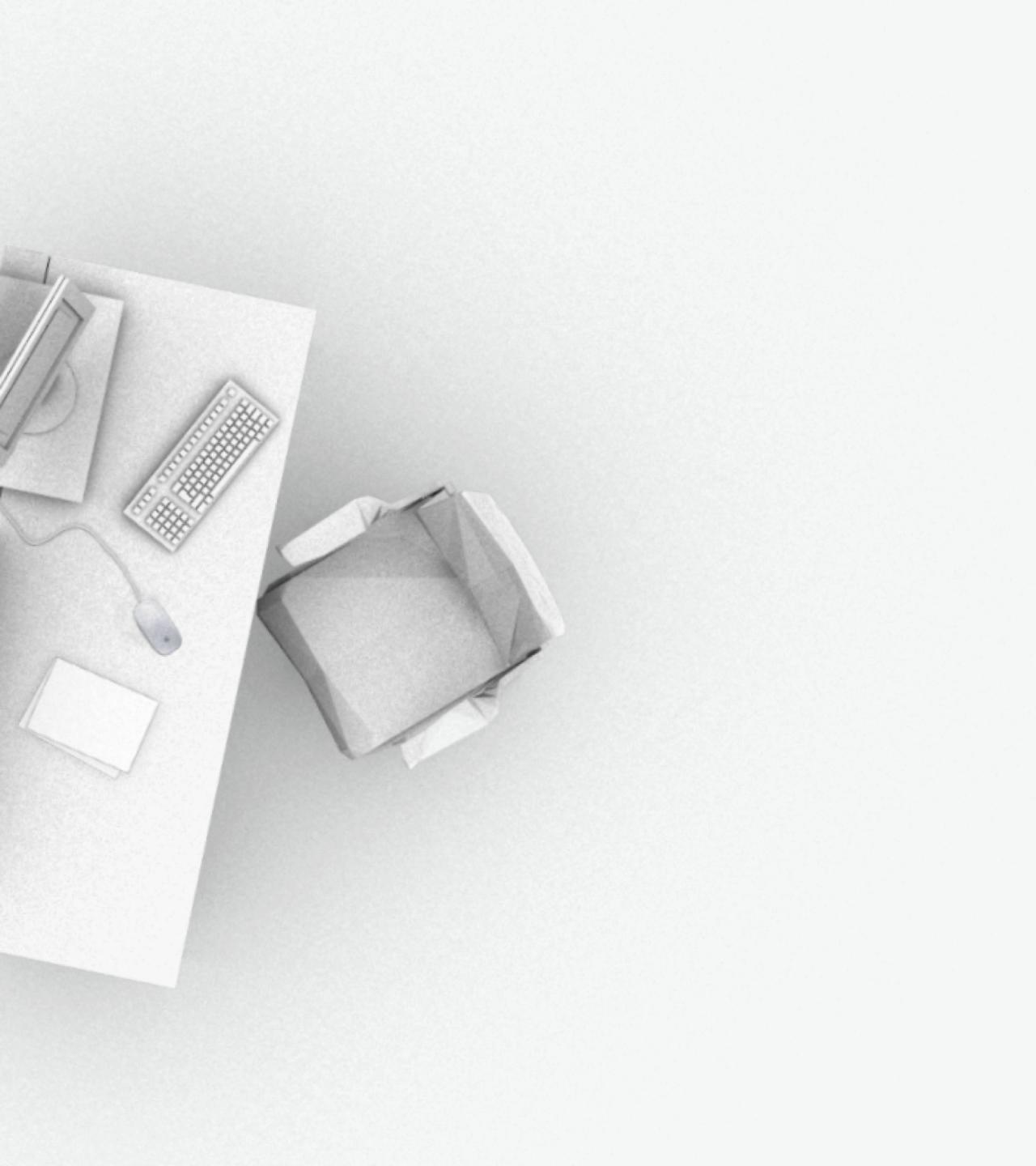
Лекция №3:

A/B-тестирование: основные библиотеки
Современные практики расчёта A/B-тестов



План лекции

- 1. Методы частотной нормализации.**
Теоретические основы. Центральная предельная теорема. Выборки без возвращения. Метод bootstrap
- 2. Программная реализация.**
Библиотечные модули. Библиотека Facebook Bootstrapped: основные функции
- 3. Альтернативные методы нормализации.** Типы метрик. Обзор методов нормализации ratio-метрик и метрик времени.
- 4. Домашние задания 6 и 7.** Разбор основных вопросов и замечаний.
Интерпретация результатов, работа с неоднозначностью в данных.



1. Методы частотной нормализации

Центральная предельная теорема

Пусть X_1, \dots, X_n есть бесконечная последовательность независимых одинаково распределённых случайных величин, имеющих конечное математическое ожидание μ и дисперсию σ^2 .

Пусть также $S_n = \sum_{i=1}^n X_i$, тогда при $n \rightarrow \infty$

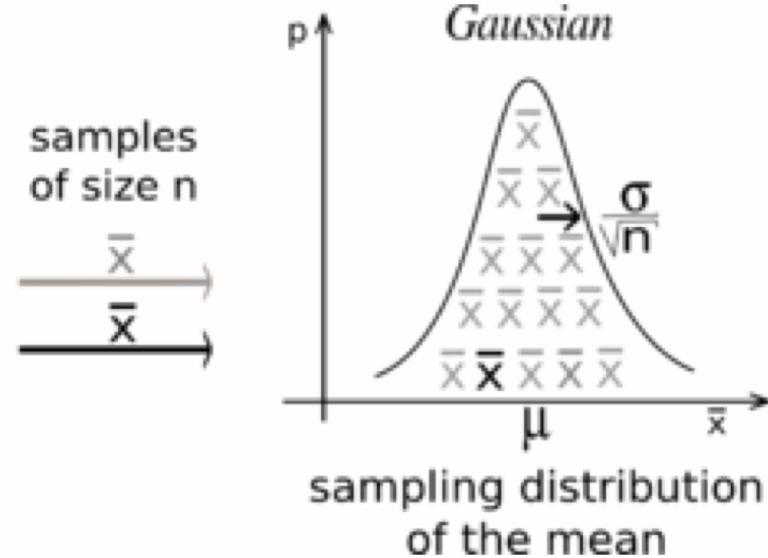
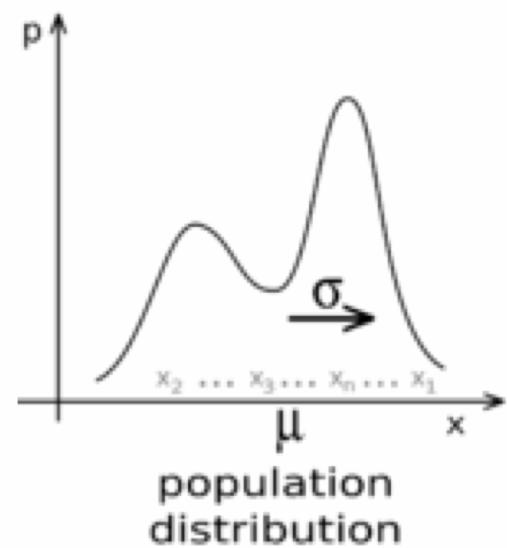
$$\frac{S_n - \mu n}{\sigma \sqrt{n}} \rightarrow N(\mu = 0, \sigma^2 = 1)$$

Или для суммы $\overline{X}_n = \frac{1}{n} \sum_{i=1}^n X_i$ первых n величин:

$$\sqrt{n} \frac{\overline{X}_n - \mu}{\sigma} \rightarrow N(\mu = 0, \sigma^2 = 1)$$

Центральная предельная теорема

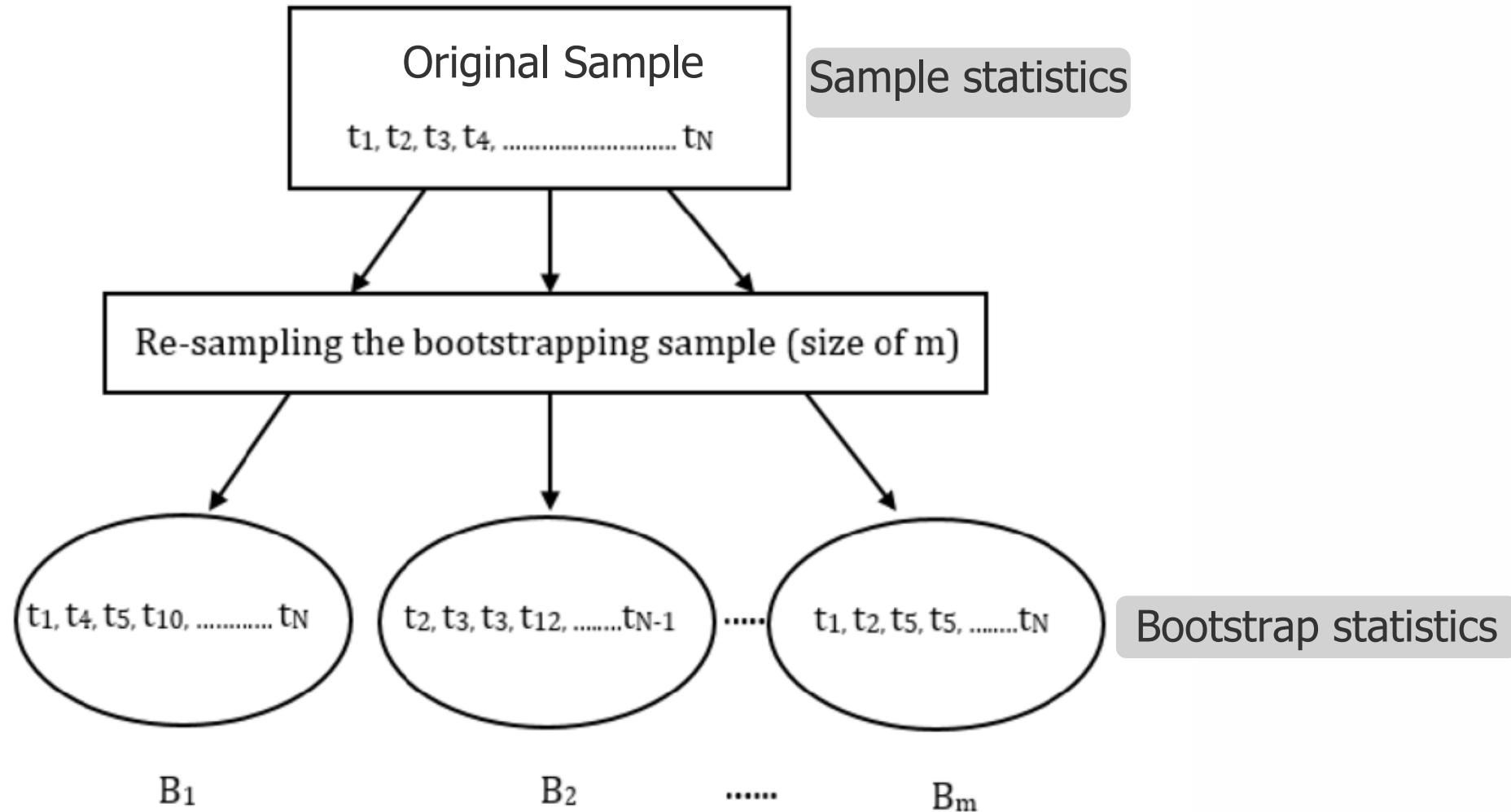
«Сумма большого количества независимых случайных величин, имеющих примерно одинаковые масштабы (ни одно из слагаемых не вносит в сумму определяющего вклада), имеет распределение, близкое к нормальному.»



Выборки без возвращения

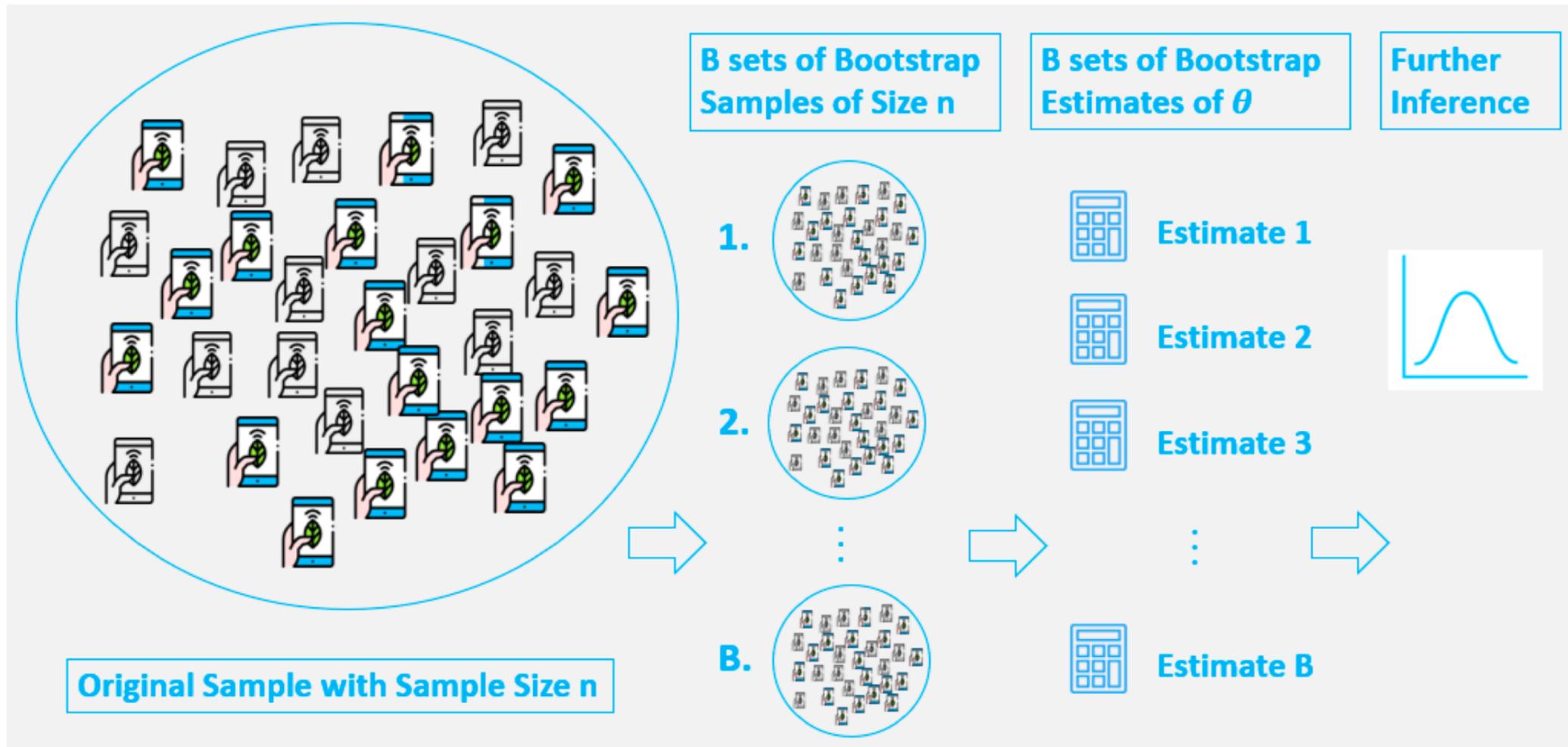
```
SELECT
    group_id, hash(user_id)%100 AS hash_id,
    SUM(metric_value) AS metric_value
FROM logs.front_log
    WHERE event_date >= '2020-01-01' AND event_date <= '2020-01-01'
GROUP BY group_id, hash_id
```

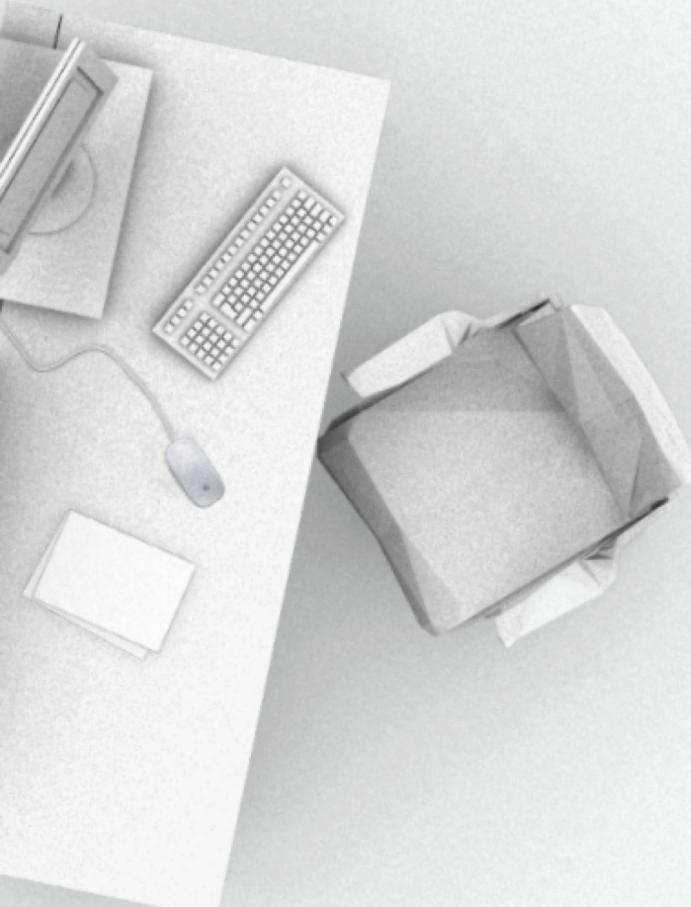
Bootstrap



#(

Bootstrap: физическая интерпретация





2. Программная реализация

Библиотека Facebook Bootstrapped

Библиотека позиционирует себя как инструмент для конструирования доверительных интервалов и расчёта статистической значимости;

Репозиторий проекта: <https://github.com/facebookincubator/bootstrapped/>

Фактически, метод представляет собой конструктор размещений с возвращением, то есть выбор случайным образом, с повторениями элементов из генеральной совокупности (если таковой можно назвать исходную выборку). На выходе формируются средние (или медианы, суммы, etc.) от средних для каждой из сформированных подвыборок.

Библиотека Facebook Bootstrapped

Полезные модули:

```
from bootstrapped import bootstrap as bs
from bootstrapped import compare_functions as bs_compare
from bootstrapped import stats_functions as bs_stats
```

Библиотека Facebook Bootstrapped

Модуль bootstrap:

```
from bootstrapped import bootstrap as bs
```

Класс основных функций конструирования подвыборок и обсчёта статистической значимости;

bs.bootstrap()

bs.bootstrap_ab()

Библиотека Facebook Bootstrapped

Метод `bs.bootstrap()`:

- реализует механизм формирования подвыборок;
- по умолчанию возвращает *lower bound* (5 перцентиль) и *upper bound* (95 перцентиль);
- чтобы вернуть дискретное распределение в этом диапазоне, необходимо установить параметр *return_distribution = True*, его генерирует вспомогательная функция *generate_distributions()*

Библиотека Facebook Bootstrapped

Метод `generate_distributions()`:

- можно задать число итераций при помощи параметра *num_iterations*, в которых будет осуществляться формирование подвыборок, и количество подвыборок *iteration_batch_size* для каждой итерации;
- на выходе `generate_distributions()` будет сформирована выборка размером, равным числу итераций *num_iterations*, элементы которой будут представлять собой среднее от значений выборок *iteration_batch_size*, рассчитанных на каждой итерации;

Библиотека Facebook Bootstrapped

Метод `generate_distributions()`:

- Пример: пусть исходная выборка имеет размер 2 000 000; $num_iterations = 10\ 000$, $iteration_batch_size = 300$. Тогда на каждой из 10 000 итераций в памяти будет храниться 300 списков по 2 000 000 элементов.
- при больших объемах выборок данные могут перестать влезать в память, поэтому в таких случаях желательно уменьшать значение $iteration_batch_size$;
- функция также позволяет производить параллельные вычисления на нескольких ядрах процессора, на нескольких thread'ах, устанавливая необходимое число при помощи параметра $num_threads$

Библиотека Facebook Bootstrapped

Метод `bs.bootstrap_ab()`:

- реализует механизм формирования подвыборок;
- по умолчанию возвращает *lower bound* (5 перцентиль) и *upper bound* (95 перцентиль);
- чтобы вернуть дискретное распределение в этом диапазоне, необходимо установить параметр *return_distribution = True*, его генерирует вспомогательная функция *generate_distributions()*

Библиотека Facebook Bootstrapped: **bootstrap()**

```
from bootstrapped import bootstrap as bs

# Bootstrap Method
data_a = data[data['group'] == 'experiment_buckets']
data_b = data[data['group'] == 'control_buckets']

bs_ab_estims = bs.bootstrap_ab(data_a.groupby(data['user_id']).target.sum().values,
                               data_b.groupby(data['user_id']).target.sum().values,
                               bs_stats.mean,
                               bs_compare.percent_change, num_iterations=5000, alpha=0.10,
                               iteration_batch_size=100, scale_test_by=1, num_threads=4)
```

Библиотека Facebook Bootstrapped: **bootstrap_ab()**

```
from bootstrapped import bootstrap as bs

# Bootstrap Method
data_a = data[data['group'] == 'experiment_buckets']
data_b = data[data['group'] == 'control_buckets']

bs_data_a = bs.bootstrap(data_a.groupby(data['user_id']).target.sum().values,
                         stat_func=bs_stats.mean, num_iterations=10000,
                         iteration_batch_size=300,
                         return_distribution=True)
bs_data_b = bs.bootstrap(data_b.groupby(data['user_id']).target.sum().values,
                         stat_func=bs_stats.mean, num_iterations=10000,
                         iteration_batch_size=300,
                         return_distribution=True)
```

Библиотека Facebook Bootstrapped

Модуль `bs_compare`:

```
from bootstrapped import compare_functions as bs_compare
```

Класс функций, предоставляющий инструментарий для формирования метрик для оценки и сравнения;

Библиотека Facebook Bootstrapped

Модуль **bs_compare**:

```
bs_compare.difference()  
bs_compare.percent_change()  
bs_compare.ratio()  
bs_compare.percent_difference()
```

```
difference = (test_stat - ctrl_stat)  
percent_change = (test_stat - ctrl_stat) * 100.0 / ctrl_stat  
ratio = test_stat / ctrl_stat  
percent_difference = (test_stat - ctrl_stat) / ((test_stat + ctrl_stat) / 2.0) * 100.0
```

Библиотека Facebook Bootstrapped

Модуль **bs_stats**:

```
from bootstrapped import stats_functions as bs_stats
```

Класс функций, реализующих способы агрегации исследуемой метрики;

```
bs_stats.mean  
bs_stats.sum  
bs_stats.median  
bs_stats.std
```

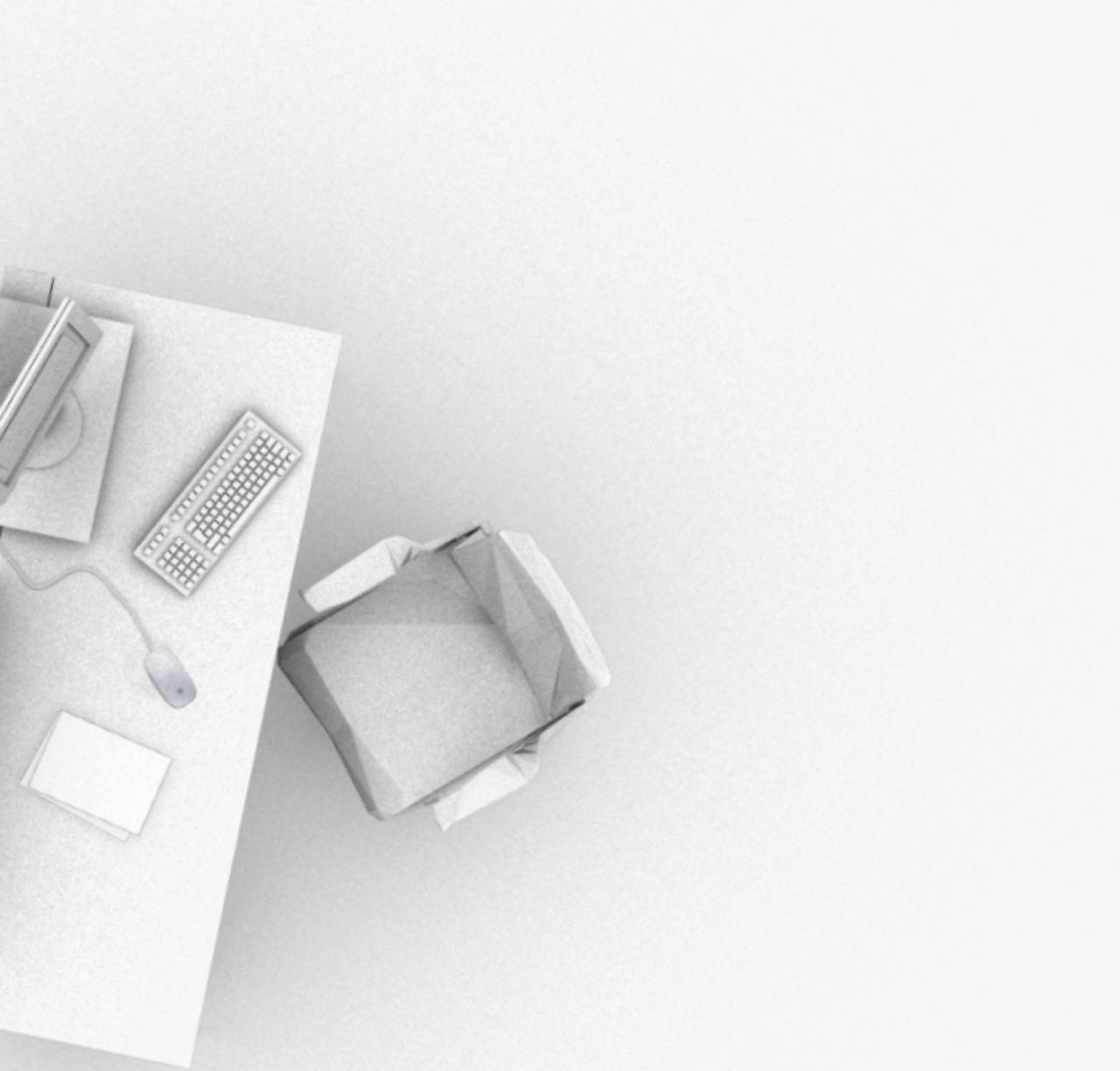
Библиотека Facebook Bootstrapped

В качестве *stat_func* можно применять и custom'ную пользовательскую функцию, например:

```
def test_func(test_stat, ctrl_stat):
    return (test_stat - ctrl_stat)/test_stat

bs.bootstrap_ab(test.values, control.values,
                stats_functions.mean,
                test_func, num_iterations=5000,
                alpha=0.05, iteration_batch_size=100,
                scale_test_by=1, num_threads=4)
```

Возвращает медиану, 0.05 и 0.95 процентили значения статистики.



3. Альтернативные методы нормализации

Типы метрик

- Агрегированные метрики в рамках групп:
 - DAU, WAU, goal visits, app installs, app uninstalls,
 - Оборот, выручка (GMV), прибыль (revenue), costs,
- На пользователя (user-based):
 - CAC, CPU, LTV, goals per users, orders per user
- На сессию (visit-based):
 - Goals per session, session time, page depth per session, churn rate, отказы (доля сессий < 30 секунд),
- На целевое действие:
 - CPO, CPC, CPA, CPM, CPI – по сути, ratio-метрики из 2-х агрегированных метрик:
$$\frac{\sum(costs)}{\sum(orders||clicks||actions||1000\ clicks||installs)}$$

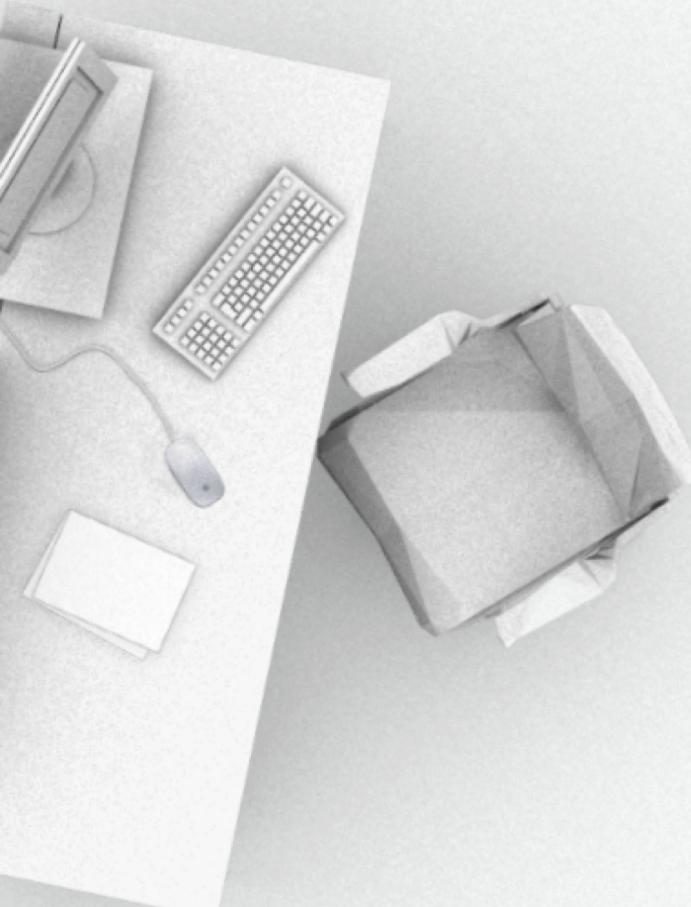
Обзор методов нормализации ratio-метрик и метрик времени

- Линеаризация:
 - Метод преобразования на основе функции аппроксимации Тейлора: $f: R^n \rightarrow R^m$
 - Есть ratio-метрика: $\mathcal{R}(U) = \frac{\sum_{u \in U} X(u)}{\sum_{u \in U} Y(u)}$
 - исходя из некоторой метрики $L_{X,Y,\alpha}(u) = X(u) - \alpha Y(u)$ $\forall u \in U$ получаем линеаризованную метрику:

$$\mathcal{L}(U) = \text{avg}_{u \in U} X(u) - \alpha \cdot \text{avg}_{u \in U} Y(u)$$

Обзор методов нормализации ratio-метрик и метрик времени

- Дельта-метод:
 - Если существует последовательность случайных величин X_n , удовлетворяющая $\sqrt{n}[X_n - \theta] \xrightarrow{D} \mathcal{N}(0, \sigma^2)$
 - то верно:
$$\sqrt{n}[g(X_n) - g(\theta)] \xrightarrow{D} \mathcal{N}(0, \sigma^2 [g'(\theta)]^2)$$
 - Для $g(X_n)$, такой, что существует $g'(\theta) = 0$ и полиномиально ограниченная случайной величиной



5. Разбор домашнего задания

Домашние задания 6

Наиболее частые примечания к ДЗ:

1) способы более элегантно посчитать моду:

```
distribution = np.array([...])
```

```
max(set(distribution), key = list(distribution).count) # 1
```

```
unique, counts = np.unique(distribution, return_counts=True) # 2
```

```
unique[np.argsort(-counts)][0] data = ps.Series(distribution) data.mode()
```

2) различие в результатах подсчёта дисперсии и среднеквадратичного отклонения из-за того, что в одном из способов неучтено смещение, в другом же, напротив, учтено: `var(ddof=1)`

Домашние задания 6

Наиболее частые примечания к ДЗ:

3) расчёт медианы – динамический:

(median[499] + median[500])/2 – хорошо, но можно лучше, с учетом возможности динамического задания числа случайных значений, генерируемых в процессе инициализации структуры **pd.Series()**:

Алгоритм:

- 1) определить, является ли размер массива чётным или нечётным числом;
- 2) если кол-во элементов в массиве нечётное, выбрать в качестве медианы центральный элемент массива;
- 3) если чётное, то в качестве медианы выбрать среднее двух смежных центральных элементов аналогично Вашему решению, но с динамическим индексом через **len[arr]**: **(arr[len(arr)/2] + arr[len(arr)/2 - 1])/2**

Домашние задания 6

Наиболее частые примечания к ДЗ:

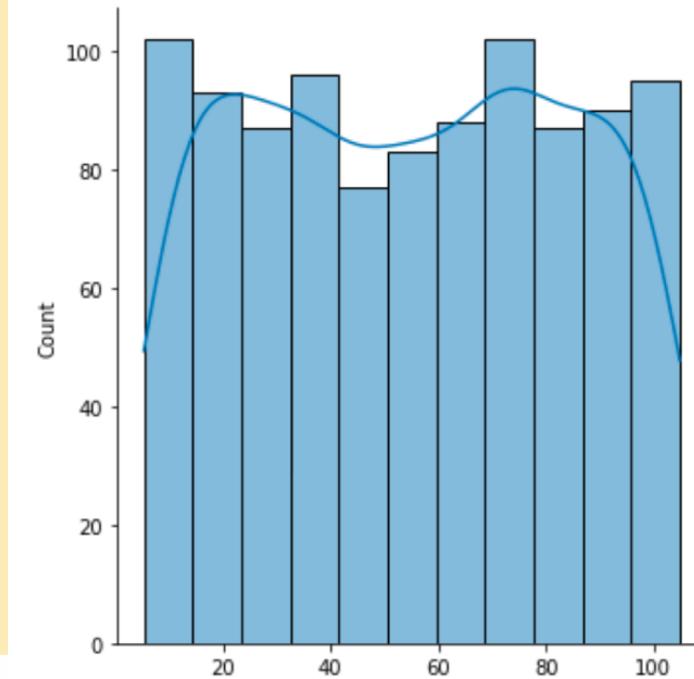
4) циклы инициализации, например:

```
pd.Series(data=[random.randint(a,b)  
               for _ in tmp])
```

можно реализовать без цикла:

```
sns.displot(pd.Series(data=np.random.rand(1000)*100+5), kde=True)
```

```
<seaborn.axisgrid.FacetGrid at 0x7f49ab5b9668>
```



Домашние задания 6

Наиболее частые примечания к ДЗ:

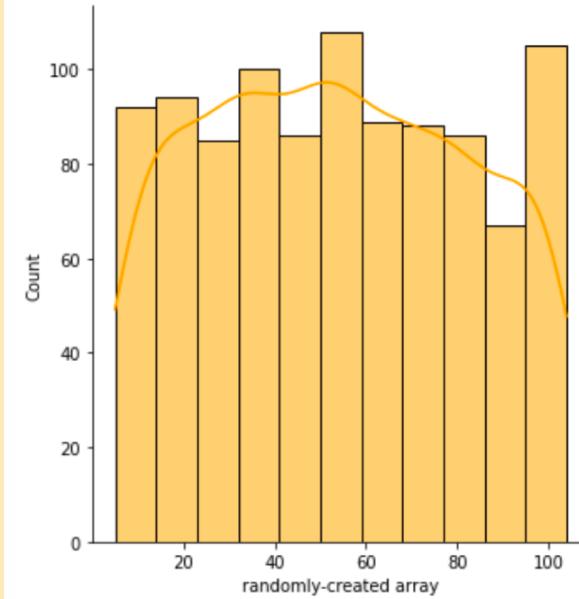
5) использование deprecated-функций:

seaborn.distplot()

FutureWarning: 'distplot' is a deprecated function and will be removed in a future version.

Deprecated-функции лучше не использовать - в production такие решения могут в дальнейшем падать, если создатели библиотеки удалят или перестанут поддерживать deprecated-функционал; желательно использовать актуальные версии

```
sns.distplot(data=s, kde=True, color='orange').set(xlabel='randomly-created array')  
<seaborn.axisgrid.FacetGrid at 0x7f858c5872e8>
```



Домашние задания 6

Дополнительные материалы по KDE:

<https://medium.com/intel-student-ambassadors/kernel-density-estimation-with-python-using-sklearn-c50b3c337871>

<https://jakevdp.github.io/blog/2013/12/01/kernel-density-estimation/>

https://www.statsmodels.org/stable/examples/notebooks/generated/kernel_density.html

https://scikit-learn.org/stable/auto_examples/neighbors/plot_kde_1d.html

Дополнительные материалы по unidip:

<https://www.kdd.org/kdd2016/subtopic/view/skinny-dip-clustering-in-a-sea-of-noise>

<https://pypi.org/project/unidip/>

Домашние задания 7

- Ещё раз обговорим поля:
 - *metric_value* – некоторая целевая метрика, результат целевых действий в рамках bucket'a – например, просмотров рекламного баннера, телефона на сайте classified-сервиса
 - или покупок в retail/FMCG;
 - *users* – кол-во пользователей, совершивших целевое действие, *visits* – кол-во
 - целевых сессий,
 - *churn_users* и *churn_visits* – кол-во пользователей в рамках bucket'a, отказавшихся
 - сделать заказ, и сессий, не закончившихся успешным целевым действием;

Домашние задания 7

- Неоднозначность в логах:
 - visits == 0, но users == 1;
 - неоднозначность в данных, о которой мы говорили на первой лекции - в логировании возможны ошибки, погрешности, которую, в идеале, лучше обнаружить до финализации данных: **COUNT(NULL)** - не были залогированы идентификаторы **visit_id** => при расчёте получили 0;
 - несколько вариантов решения:
 - убрать эти записи, как ошибочные (потерять 5% наблюдений);
 - довериться одной из колонок: фактические данные о кол-ве визитов не восстановить, однако можно сделать аппроксимацию (подумайте, как?) их количества, а с users работать как с точной метрикой

15
баллов

Срок сдачи:
09.12.2020

Домашнее задание

АВТ-3

#037

Полезные ссылки и литература в помощь:

- Татьяна Мелехина: «Лекции по теории вероятностей и математической статистике»
- Wes McKinney: “Python for Data Analysis: Data Wrangling with Pandas, NumPy, and IPython”
- <https://pandas.pydata.org/>
- <https://matplotlib.org/>
- <https://jupyter.org/>

Домашнее задание: АВТ-3. Описание case'a

1. На сайте проводится некоторый эксперимент. Время проведения эксперимента – 2 недели. Всего на сайте DAU около 500 000 и WAU около 850 000. Всего за время работы на сайте образовалась аудитория из около 1 500 000 не уходящих в отток посетителей.
2. В файле <https://cloud.mail.ru> – данные о результатах эксперимента. В колонке `group_id` вы видите bucket'ы с номерами от 1 до 16. К группе 'A', контрольной, относятся bucket'ы 1..8, к группе 'B', экспериментальной – 9..16. Колонки:
metric_value – некоторая целевая метрика в рамках bucket'a –
например, просмотров рекламного баннера, телефона на сайте classified-сервиса
или покупок в retail/FMCG;
users – кол-во пользователей, совершивших целевое действие, *visits* – кол-во
целевых сессий,
churn_users и *churn_visits* – кол-во пользователей в рамках bucket'a, отказавшихся
сделать заказ, и сессий, не закончившихся успешным целевым действием.
3. Менеджера продукта интересует, какие позитивные и негативные эффекты
вызвало нововведение. Продумайте метрики, которые стоит исследовать в
рамках имеющихся данных для ответа на вопрос менеджера.

Домашнее задание: АВТ-3

1. Загрузите данные из файла в структуру pandas.DataFrame().
2. Проведите оценку равенства дисперсий исследуемых метрик в группах и исследуйте распределение на нормальность. Какими критериями вы воспользовались и почему?
3. Оцените статистическую значимость различий исследуемых метрик в выборках при помощи библиотеки bootstrapped.
4. Изобразите гистограммы и диаграммы размаха.
5. Результат работы пришлите в формате IPython Jupyter Notebook на сервисе Google Colab.

Thanks for attention

