Сборный проект

Описание проекта

Вы работаете в стартапе, который продаёт продукты питания. Нужно разобраться, как ведут себя пользователи вашего мобильного приложения. Изучите воронку продаж. Узнайте, как пользователи доходят до покупки. Сколько пользователей доходит до покупки, а сколько — «застревает» на предыдущих шагах? На каких именно? После этого исследуйте результаты A/A/B-эксперимента. Дизайнеры захотели поменять шрифты во всём приложении, а менеджеры испугались, что пользователям будет непривычно. Договорились принять решение по результатам A/A/B-теста. Пользователей разбили на 3 группы: 2 контрольные со старыми шрифтами и одну экспериментальную — с новыми. Выясните, какой шрифт лучше. Создание двух групп A вместо одной имеет определённые преимущества. Если две контрольные группы окажутся равны, вы можете быть уверены в точности проведенного тестирования. Если же между значениями A и A будут существенные различия, это поможет обнаружить факторы, которые привели к искажению результатов. Сравнение контрольных групп также помогает понять, сколько времени и данных потребуется для дальнейших тестов. В случае общей аналитики и A/A/B-эксперимента работайте с одними и теми же данными. В реальных проектах всегда идут эксперименты. Аналитики исследуют качество работы приложения по общим данным, не учитывая принадлежность пользователей к экспериментам.

Описание данных

Каждая запись в логе — это действие пользователя, или событие.

- EventName — название события;
- DeviceIDHash — уникальный идентификатор пользователя;
- EventTimestamp — время события;
- ExpId — номер эксперимента: 246 и 247 — контрольные группы, а 248 — экспериментальная.

Инструкция по выполнению проекта

Шаг 1. Откройте файл с данными и изучите общую информацию

Путь к файлу: /datasets/logs_exp.csv. Скачать датасет

Шаг 2. Подготовьте данные

1. Замените названия столбцов на удобные для вас;
2. Проверьте пропуски и типы данных. Откорректируйте, если нужно;
3. Добавьте столбец даты и времени, а также отдельный столбец дат;

Шаг 3. Изучите и проверьте данные

1. Сколько всего событий в логе?
2. Сколько всего пользователей в логе?
3. Сколько в среднем событий приходится на пользователя?
4. Данными за какой период вы располагаете? Найдите максимальную и минимальную дату. Постройте гистограмму по дате и 5. времени. Можно ли быть уверенным, что у вас одинаково полные данные за весь период? Технически в логи новых дней по некоторым пользователям могут «доезжать» события из прошлого — это может «перекашивать данные». Определите, с какого момента данные полные и отбросьте более старые. Данными за какой период времени вы располагаете на самом деле?
5. Много ли событий и пользователей вы потеряли, отбросив старые данные?
6. Проверьте, что у вас есть пользователи из всех трёх экспериментальных групп.

Шаг 4. Изучите воронку событий

1. Посмотрите, какие события есть в логах, как часто они встречаются. Отсортируйте события по частоте.
2. Посчитайте, сколько пользователей совершали каждое из этих событий. Отсортируйте события по числу пользователей. Посчитайте долю пользователей, которые хоть раз совершали событие
3. Предположите, в каком порядке происходят события. Все ли они выстраиваются в последовательную цепочку? Их не нужно учитывать при расчёте воронки.
4. По воронке событий посчитайте, какая доля пользователей проходит на следующий шаг воронки (от числа пользователей на предыдущем). То есть для последовательности событий A → B → C посчитайте отношение числа пользователей с событием B к

количеству пользователей с событием A, а также отношение числа пользователей с событием C к количеству пользователей с событием B.
5. На каком шаге теряете больше всего пользователей?
6. Какая доля пользователей доходит от первого события до оплаты?

Шаг 5. Изучите результаты эксперимента

1. Сколько пользователей в каждой экспериментальной группе?
2. Есть 2 контрольные группы для A/A-эксперимента, чтобы проверить корректность всех механизмов и расчётов. Проверьте, находят ли статистические критерии разницу между выборками 246 и 247.
3. Выберите самое популярное событие. Посчитайте число пользователей, совершивших это событие в каждой из контрольных групп. Посчитайте долю пользователей, совершивших это событие. Проверьте, будет ли отличие между группами статистически достоверным. Проделайте то же самое для всех других событий (удобно обернуть проверку в отдельную функцию). Можно ли сказать, что разбиение на группы работает корректно?
4. Аналогично поступите с группой с изменённым шрифтом. Сравните результаты с каждой из контрольных групп в отдельности по каждому событию. Сравните результаты с объединённой контрольной группой. Какие выводы из эксперимента можно сделать?
5. Какой уровень значимости вы выбрали при проверке статистических гипотез выше? Посчитайте, сколько проверок статистических гипотез вы сделали. При уровне значимости 0.1 каждый десятый раз можно получать ложный результат. Какой уровень значимости стоит применить? Если вы хотите изменить его, проделайте предыдущие пункты и проверьте свои выводы.

Комментарии ревьюера ~~Молодец, что приводишь описание цели исследования, входных данных и структуру работы. Единственный момент - ссылки на подразделы, к сожалению, не работают. Можешь, пожалуйста, проверить, что пошло не так?~~ Сделано!

Комментарии студента: исправил.

Комментарии ревьюера v2 Супер, отлично :)

Шаг 1. Откройте файл с данными и изучите общую информацию

In [1]:

```
import pandas as pd
import numpy as np
import math as mth
import seaborn as sns
import matplotlib.pyplot as plt
from scipy import stats as st
from plotly import graph_objects as go
from pandas.plotting import register_matplotlib_converters
register_matplotlib_converters()
```

In [2]:

```
df = pd.read_csv('/datasets/logs_exp.csv')
```

In [3]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 244126 entries, 0 to 244125
Data columns (total 1 columns):
EventName DeviceIDHash EventTimestamp ExpId    244126 non-null object
dtypes: object(1)
memory usage: 1.9+ MB
```

In [4]:

```
df.head()
```

Out[4]:

EventName\tDeviceIDHash\tEventTimestamp\tExpId

	EventName\tDeviceIDHash\tEventTimestamp\tExpId
0	MainScreenAppear\t4575588528974610257\t1564029...
1	MainScreenAppear\t7416695313311560658\t1564053...
2	PaymentScreenSuccessful\t3518123091307005509\t...
3	CartScreenAppear\t3518123091307005509\t1564054...
4	PaymentScreenSuccessful\t6217807653094995999\t...

таблица выгружена некорректно, сначала преобразуем ее для дальнейшего разбора, воспользуемся разделителем

In [5]:

```
df = pd.read_csv('/datasets/logs_exp.csv', sep='\t')
```

Шаг 2. Подготовьте данные

Замените названия столбцов на удобные для вас;

In [6]:

```
df.head()
```

Out[6]:

	EventName	DeviceIDHash	EventTimestamp	ExpId
0	MainScreenAppear	4575588528974610257	1564029816	246
1	MainScreenAppear	7416695313311560658	1564053102	246
2	PaymentScreenSuccessful	3518123091307005509	1564054127	248
3	CartScreenAppear	3518123091307005509	1564054127	248
4	PaymentScreenSuccessful	6217807653094995999	1564055322	248

In [7]:

```
df = df.rename(columns={'EventName': 'event_name', 'DeviceIDHash': 'user_id', 'EventTimestamp': 'event_time', 'ExpId': 'exp_id'})
#df.columns = ['event_name', 'user_id', 'event_time', 'exp_id']
df.head()
```

Out[7]:

	event_name	user_id	event_time	exp_id
0	MainScreenAppear	4575588528974610257	1564029816	246
1	MainScreenAppear	7416695313311560658	1564053102	246
2	PaymentScreenSuccessful	3518123091307005509	1564054127	248
3	CartScreenAppear	3518123091307005509	1564054127	248
4	PaymentScreenSuccessful	6217807653094995999	1564055322	248

In [8]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 244126 entries, 0 to 244125
Data columns (total 4 columns):
event_name    244126 non-null object
user_id       244126 non-null int64
event_time    244126 non-null int64
exp_id        244126 non-null int64
dtypes: int64(3), object(1)
memory usage: 7.5+ MB
```

Проверьте пропуски и типы данных. Откорректируйте, если нужно;

In [9]:

In [9]:

```
df.duplicated().sum()
```

Out[9]:

413

In [10]:

```
df = df.drop_duplicates()
```

In [11]:

```
df.duplicated().sum()
```

Out[11]:

0

In [12]:

```
df.isna().sum()
```

Out[12]:

```
event_name    0
user_id       0
event_time    0
exp_id        0
dtype: int64
```

Добавьте столбец даты и времени, а также отдельный столбец дат;

In [13]:

```
df['event_time'] = pd.to_datetime(df['event_time'], unit='s')
df['event_date'] = df['event_time'].dt.date.astype('datetime64')
```

In [14]:

```
df.head()
```

Out[14]:

	event_name	user_id	event_time	exp_id	event_date
0	MainScreenAppear	4575588528974610257	2019-07-25 04:43:36	246	2019-07-25
1	MainScreenAppear	7416695313311560658	2019-07-25 11:11:42	246	2019-07-25
2	PaymentScreenSuccessful	3518123091307005509	2019-07-25 11:28:47	248	2019-07-25
3	CartScreenAppear	3518123091307005509	2019-07-25 11:28:47	248	2019-07-25
4	PaymentScreenSuccessful	6217807653094995999	2019-07-25 11:48:42	248	2019-07-25

Шаг 3. Изучите и проверьте данные

Сколько всего событий в логе?

In [15]:

```
print('Событий в логе:', df['event_name'].shape[0])
```

Событий в логе: 243713

Сколько всего пользователей в логе?

In [16]:

```
print('Пользователей в логе:', df['user_id'].nunique())
```

Пользователей в логе: 7551

Сколько в среднем событий приходится на пользователя?

In [17]:

```
print('В среднем событий на пользователя %d.' % (df.shape[0] / df['user_id'].nunique()))
```

В среднем событий на пользователя 32.

Данными за какой период вы располагаете? Найдите максимальную и минимальную дату. Постройте гистограмму по дате и времени. Можно ли быть уверенным, что у вас одинаково полные данные за весь период? Технически в логи новых дней по некоторым пользователям могут «доезжать» события из прошлого — это может «перекашивать данные». Определите, с какого момента данные полные и отбросьте более старые. Данными за какой период времени вы располагаете на самом деле?

In [18]:

```
print(min(df['event_time']))
print(max(df['event_time']))
```

2019-07-25 04:43:36
2019-08-07 21:15:17

In [19]:

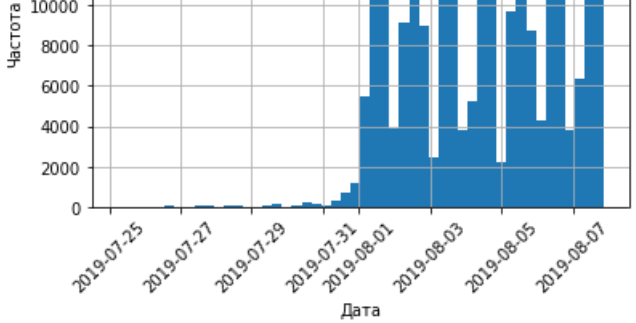
```
temp = df.groupby(by='event_date').agg({'event_name': 'count'}).reset_index()
temp['percent'] = temp['event_name'] / temp['event_name'].sum() * 100
temp['cum_sum'] = temp['percent'].cumsum()
temp
```

Out[19]:

	event_date	event_name	percent	cum_sum
0	2019-07-25	9	0.003693	0.003693
1	2019-07-26	31	0.012720	0.016413
2	2019-07-27	55	0.022568	0.038980
3	2019-07-28	105	0.043083	0.082064
4	2019-07-29	184	0.075499	0.157562
5	2019-07-30	412	0.169051	0.326614
6	2019-07-31	2030	0.832947	1.159561
7	2019-08-01	36141	14.829328	15.988889
8	2019-08-02	35554	14.588471	30.577359
9	2019-08-03	33282	13.656227	44.233586
10	2019-08-04	32968	13.527387	57.760973
11	2019-08-05	36058	14.795271	72.556244
12	2019-08-06	35788	14.684485	87.240730
13	2019-08-07	31096	12.759270	100.000000

In [20]:

```
ax = df['event_time'].hist(bins=50)
plt.title('Гистограмма по дате и времени')
plt.ylabel("Частота")
plt.xlabel("Дата")
plt.xticks(rotation=45)
plt.show()
```



фактически мы располагаем данными с 1.08 по 07.08

In [21]:

```
df_filt = df.query('event_date >= "2019-08-01" and event_date <= "2019-08-07"')
```

In [22]:

```
ax = df_filt['event_time'].hist(bins=50)
plt.title('Гистограмма по дате и времени')
plt.ylabel("Частота")
plt.xlabel("Дата")
plt.xticks(rotation=45)
plt.show()
```

Много ли событий и пользователей вы потеряли, отбросив старые данные?

потери пользователей

In [23]:

```
lost_users = round((1-(df_filt['user_id'].nunique()/df['user_id'].nunique()))*100, 3)
print(f'Было потеряно: {lost_users} % пользователей')
```

Было потеряно: 0.225 % пользователей

потери событий

In [24]:

```
lost_events = round(((1-df_filt['event_name'].shape[0]/df['event_name'].shape[0])*100), 3)
print(f'Было потеряно: {lost_events} % событий')
```

Было потеряно: 1.16 % событий

Проверьте, что у вас есть пользователи из всех трёх экспериментальных групп.

In [25]:

```
users = df_filt.groupby('exp_id')['user_id'].nunique()
users
```

Out[25]:

```
exp_id
246    2484
247    2513
248    2537
Name: user_id, dtype: int64
```

Шаг 4. Изучите воронку событий

Посмотрите, какие события есть в логах, как часто они встречаются. Отсортируйте события по частоте.

In [26]:

```
df_events = (df_filt
             .pivot_table(index='event_name',
                          values='user_id',
                          aggfunc=['count', 'nunique']
                         )
             .reset_index()
            )
df_events.columns
```

Out[26]:

```
MultiIndex([('event_name',          ''),
            (     'count',   'user_id'),
            (   'nunique', 'event_name'),
            (   'nunique',   'user_id')],
           )
```

преобразуем и оставим только нужное

In [27]:

```
df_events.columns = ['event_name', 'count_uid','name', 'nunique_uid']
df_events.drop('name', axis=1, inplace=True)
df_events = df_events.sort_values(by='count_uid', ascending=False)
```

In [28]:

```
df_events
```

Out[28]:

	event_name	count_uid	nunique_uid
1	MainScreenAppear	117328	7419
2	OffersScreenAppear	46333	4593
0	CartScreenAppear	42303	3734
3	PaymentScreenSuccessful	33918	3539
4	Tutorial	1005	840

Посчитайте, сколько пользователей совершали каждое из этих событий. Отсортируйте события по числу пользователей. Посчитайте долю пользователей, которые хоть раз совершали событие

In [29]:

```
number_of_people = df_filt.groupby('event_name')['user_id'].nunique().sort_values(ascending=False).to_frame().reset_index()\
    .rename(columns={'user_id': 'total_users'})
number_of_people['percent'] = np.round(100* number_of_people['total_users'] / df_filt['user_id'].nunique())
number_of_people
```

Out[29]:

	event_name	total_users	percent

	event_name	total_users	percent
0	MainScreenAppear	7419	98.0
1	OffersScreenAppear	4593	61.0
2	CartScreenAppear	3734	50.0
3	PaymentScreenSuccessful	3539	47.0
4	Tutorial	840	11.0

Предположите, в каком порядке происходят события. Все ли они выстраиваются в последовательную цепочку? Их не нужно учитывать при расчёте воронки.

1. MainScreenAppear
2. OffersScreenAppear
3. CartScreenAppear
4. PaymentScreenSuccessful

Tutorial нельзя логически отнести к чему-тоЮ думаю тожно выбросить

In [30]:

```
number_of_people = number_of_people[number_of_people['event_name'] != 'Tutorial']
```

In [31]:

```
number_of_people
```

Out[31]:

	event_name	total_users	percent
0	MainScreenAppear	7419	98.0
1	OffersScreenAppear	4593	61.0
2	CartScreenAppear	3734	50.0
3	PaymentScreenSuccessful	3539	47.0

По воронке событий посчитайте, какая доля пользователей проходит на следующий шаг воронки (от числа пользователей на предыдущем). То есть для последовательности событий A → B → C посчитайте отношение числа пользователей с событием B к количеству пользователей с событием A, а также отношение числа пользователей с событием C к количеству пользователей с событием B.

In [32]:

```
fig = go.Figure()
fig = go.Figure(go.Funnel(
    y = number_of_people['event_name'],
    x = number_of_people['total_users']))
fig.show()
```

На каком шаге теряете больше всего пользователей?

главный экран - offer screen appear

Какая доля пользователей доходит от первого события до оплаты?

47,0%

> * близжайший % потери пользователей к главному экрану - это туториал, но в последствии мы его исключаем, так что на главном экране теряется больше всего пользователей * долю пользователей от главного экрана до оплаты - иправил

Шаг 5. Изучите результаты эксперимента

Сколько пользователей в каждой экспериментальной группе?

In [33]:

```
people = df_filt.groupby('exp_id')['user_id'].nunique()
people
```

Out[33]:

```
exp_id
246    2484
247    2513
248    2537
Name: user_id, dtype: int64
```

Есть 2 контрольные группы для A/A-эксперимента, чтобы проверить корректность всех механизмов и расчётов. Проверьте, находят ли статистические критерии разницу между выборками 246 и 247.

In [34]:

```
aa_246 = df_filt[(df_filt['exp_id'] == 246) & (df_filt['event_name'])]
aa_247 = df_filt[(df_filt['exp_id'] == 247) & (df_filt['event_name'])]
aa_248 = df_filt[(df_filt['exp_id'] == 248) & (df_filt['event_name'])]
```

In [35]:

```
funnel_246 = aa_246.groupby('event_name')['user_id'].nunique().sort_values(ascending=False).to_frame().reset_index()\
    .rename(columns={'user_id': 'total_users'})
funnel_247 = aa_247.groupby('event_name')['user_id'].nunique().sort_values(ascending=False).to_frame().reset_index()\
    .rename(columns={'user_id': 'total_users'})
funnel_248 = aa_248.groupby('event_name')['user_id'].nunique().sort_values(ascending=False).to_frame().reset_index()\
    .rename(columns={'user_id': 'total_users'})
```

In [36]:

```
funnel_247['total_users'].sum() / funnel_246['total_users'].sum() * 100
```

Out[36]:

```
99.09441805225653
```

In [37]:

```
funnel_247['total_users'] / funnel_246['total_users'] * 100
```

Out[37]:

```
0    101.061224
1     98.573281
2     97.788310
3     96.500000
4    101.798561
Name: total_users, dtype: float64
```

In [38]:

```
fig = go.Figure()

fig.add_trace(go.Funnel(
    name = 'aa 246',
    y = funnel_246['event_name'],
    x = funnel_246['total_users'],
    ))

fig.add_trace(go.Funnel(
    name = 'aa 247',
    y = funnel_247['event_name'],
    x = funnel_247['total_users'],
    ))

fig.add_trace(go.Funnel(
    name = 'aa 248',
    y = funnel_248['event_name'],
    x = funnel_248['total_users'],
    ))

fig.show()
```

разницы между группами нет

Выберите самое популярное событие. Посчитайте число пользователей, совершивших это событие в каждой из контрольных групп. Посчитайте долю пользователей, совершивших это событие. Проверьте, будет ли отличие между группами статистически достоверным. Проделайте то же самое для всех других событий (удобно обернуть проверку в отдельную функцию). Можно ли сказать, что разбиение на группы работает корректно?

- **H0 - между выборками 246 и 247 нет различимой разницы**
- **H1 - выборки 246 и 247 отличаются между друг другом**

далее проверяем между собой:

- **А-группа - 246 , Б-группа - 248**
- **А-группа - 247 , Б-группа - 248**

- **А-группа - 247 , Б-группа - 248**
- **АА-группа - 246,247 (объединение), Б-группа -248**

In [39]:

```
df_filt = df_filt[df_filt['event_name'] != 'Tutorial']
```

In [40]:

```
pop_funnel = df_filt.pivot_table(index='event_name', columns='exp_id',values='user_id',aggfunc='nunique')\
                    .sort_values(246,ascending=False)
pop_funnel['246+247'] = pop_funnel[246] + pop_funnel[247]
pop_funnel
```

Out[40]:

exp_id	246	247	248	246+247
event_name				
MainScreenAppear	2450	2476	2493	4926
OffersScreenAppear	1542	1520	1531	3062
CartScreenAppear	1266	1238	1230	2504
PaymentScreenSuccessful	1200	1158	1181	2358

Самое популярное событие - MainScreenAppear

In [41]:

```
users = people.to_frame().reset_index()
users.loc[3] = ['246+247', 4997]
```

In [42]:

```
users = users.set_index(users.columns[0])
users
```

Out[42]:

exp_id	user_id
246	2484
247	2513
248	2537
246+247	4997

воспользуемся z-критерием

In [43]:

```
def z_test(exp1, exp2, event, alpha):
    p1_ev = pop_funnel.loc[event, exp1]
    p2_ev = pop_funnel.loc[event, exp2]
    p1_us = users.loc[exp1, 'user_id']
    p2_us = users.loc[exp2, 'user_id']
    p1 = p1_ev / p1_us
    p2 = p2_ev / p2_us
    difference = p1 - p2
    p_combined = (p1_ev + p2_ev) / (p1_us + p2_us)
    z_value = difference / mth.sqrt(p_combined * (1 - p_combined) * (1 / p1_us + 1 / p2_us))
    distr = st.norm(0, 1)
    p_value = (1 - distr.cdf(abs(z_value))) * 2
    print('Проверка для  {} и {}, событие: {}, p-значение: {p_value:.2f}'.format(exp1, exp2, event, p_value=p_value))
    if (p_value < alpha):
        print("Отвергаем нулевую гипотезу")
    else:
        print("Не получилось отвергнуть нулевую гипотезу")
```

In [44]:

```
for event in pop_funnel.index:
    z_test(246, 247, event, 0.05)
    print()
```

Проверка для 246 и 247, событие: MainScreenAppear, p-значение: 0.76
Не получилось отвергнуть нулевую гипотезу

Проверка для 246 и 247, событие: OffersScreenAppear, p-значение: 0.25
Не получилось отвергнуть нулевую гипотезу

Проверка для 246 и 247, событие: CartScreenAppear, p-значение: 0.23
Не получилось отвергнуть нулевую гипотезу

Проверка для 246 и 247, событие: PaymentScreenSuccessful, p-значение: 0.11
Не получилось отвергнуть нулевую гипотезу

Вывод об отличии сделать нельзя

Аналогично поступите с группой с изменённым шрифтом. Сравните результаты с каждой из контрольных групп в отдельности по каждому событию. Сравните результаты с объединённой контрольной группой. Какие выводы из эксперимента можно сделать?

In [45]:

```
for event in pop_funnel.index:
    z_test(246, 248, event, 0.05)
    print()
```

Проверка для 246 и 248, событие: MainScreenAppear, p-значение: 0.29
Не получилось отвергнуть нулевую гипотезу

Проверка для 246 и 248, событие: OffersScreenAppear, p-значение: 0.21
Не получилось отвергнуть нулевую гипотезу

Проверка для 246 и 248, событие: CartScreenAppear, p-значение: 0.08
Не получилось отвергнуть нулевую гипотезу

Проверка для 246 и 248, событие: PaymentScreenSuccessful, p-значение: 0.21
Не получилось отвергнуть нулевую гипотезу

In [46]:

```
for event in pop_funnel.index:
    z_test(247, 248, event, 0.05)
    print()
```

Проверка для 247 и 248, событие: MainScreenAppear, p-значение: 0.46
Не получилось отвергнуть нулевую гипотезу

Проверка для 247 и 248, событие: OffersScreenAppear, p-значение: 0.92
Не получилось отвергнуть нулевую гипотезу

Проверка для 247 и 248, событие: CartScreenAppear, p-значение: 0.58
Не получилось отвергнуть нулевую гипотезу

Проверка для 247 и 248, событие: PaymentScreenSuccessful, p-значение: 0.74
Не получилось отвергнуть нулевую гипотезу

In [47]:

```
for event in pop_funnel.index:
    z_test('246+247', 248, event, 0.05)
    print()
```

Проверка для 246+247 и 248, событие: MainScreenAppear, p-значение: 0.29
Не получилось отвергнуть нулевую гипотезу

Проверка для 246+247 и 248, событие: OffersScreenAppear, p-значение: 0.43
Не получилось отвергнуть нулевую гипотезу

Проверка для 246+247 и 248, событие: CartScreenAppear, p-значение: 0.18
Не получилось отвергнуть нулевую гипотезу

Проверка для 246+247 и 248, событие: PaymentScreenSuccessful, p-значение: 0.60
Не получилось отвергнуть нулевую гипотезу

Видим что контрольные группы одинаковые между собой

Какой уровень значимости вы выбрали при проверке статистических гипотез выше? Посчитайте, сколько проверок статистических гипотез вы сделали. При уровне значимости 0.1 каждый десятый раз можно получать ложный результат. Какой уровень значимости стоит применить? Если вы хотите изменить его, проделайте предыдущие пункты и проверьте свои выводы.

- **p_value берем 5%**
- **16тестов провели - 4 aa, 12 аб теста**
- **по воронке видно что различий между контрольными группами нет, так что менять не вижу смысла**

In [48]:

```
for event in pop_funnel.index:
    z_test(246, 247, event, 0.01)
    print()
```

Проверка для 246 и 247, событие: MainScreenAppear, p-значение: 0.76
Не получилось отвергнуть нулевую гипотезу

Проверка для 246 и 247, событие: OffersScreenAppear, p-значение: 0.25
Не получилось отвергнуть нулевую гипотезу

Проверка для 246 и 247, событие: CartScreenAppear, p-значение: 0.23
Не получилось отвергнуть нулевую гипотезу

Проверка для 246 и 247, событие: PaymentScreenSuccessful, p-значение: 0.11
Не получилось отвергнуть нулевую гипотезу

In [49]:

```
for event in pop_funnel.index:
    z_test(246, 248, event, 0.01)
    print()
```

Проверка для 246 и 248, событие: MainScreenAppear, p-значение: 0.29
Не получилось отвергнуть нулевую гипотезу

Проверка для 246 и 248, событие: OffersScreenAppear, p-значение: 0.21
Не получилось отвергнуть нулевую гипотезу

Проверка для 246 и 248, событие: CartScreenAppear, p-значение: 0.08
Не получилось отвергнуть нулевую гипотезу

Проверка для 246 и 248, событие: PaymentScreenSuccessful, p-значение: 0.21
Не получилось отвергнуть нулевую гипотезу

In [50]:

```
for event in pop_funnel.index:
    z_test(247, 248, event, 0.01)
    print()
```

Проверка для 247 и 248, событие: MainScreenAppear, p-значение: 0.46
Не получилось отвергнуть нулевую гипотезу

Проверка для 247 и 248, событие: OffersScreenAppear, p-значение: 0.92
Не получилось отвергнуть нулевую гипотезу

Проверка для 247 и 248, событие: CartScreenAppear, p-значение: 0.58
Не получилось отвергнуть нулевую гипотезу

Проверка для 247 и 248, событие: PaymentScreenSuccessful, p-значение: 0.74
Не получилось отвергнуть нулевую гипотезу

In [51]:

In [51]:

```
for event in pop_funnel.index:
    z_test('246+247', 248, event, 0.01)
    print()
```

Проверка для 246+247 и 248, событие: MainScreenAppear, p-значение: 0.29
Не получилось отвергнуть нулевую гипотезу

Проверка для 246+247 и 248, событие: OffersScreenAppear, p-значение: 0.43
Не получилось отвергнуть нулевую гипотезу

Проверка для 246+247 и 248, событие: CartScreenAppear, p-значение: 0.18
Не получилось отвергнуть нулевую гипотезу

Проверка для 246+247 и 248, событие: PaymentScreenSuccessful, p-значение: 0.60
Не получилось отвергнуть нулевую гипотезу

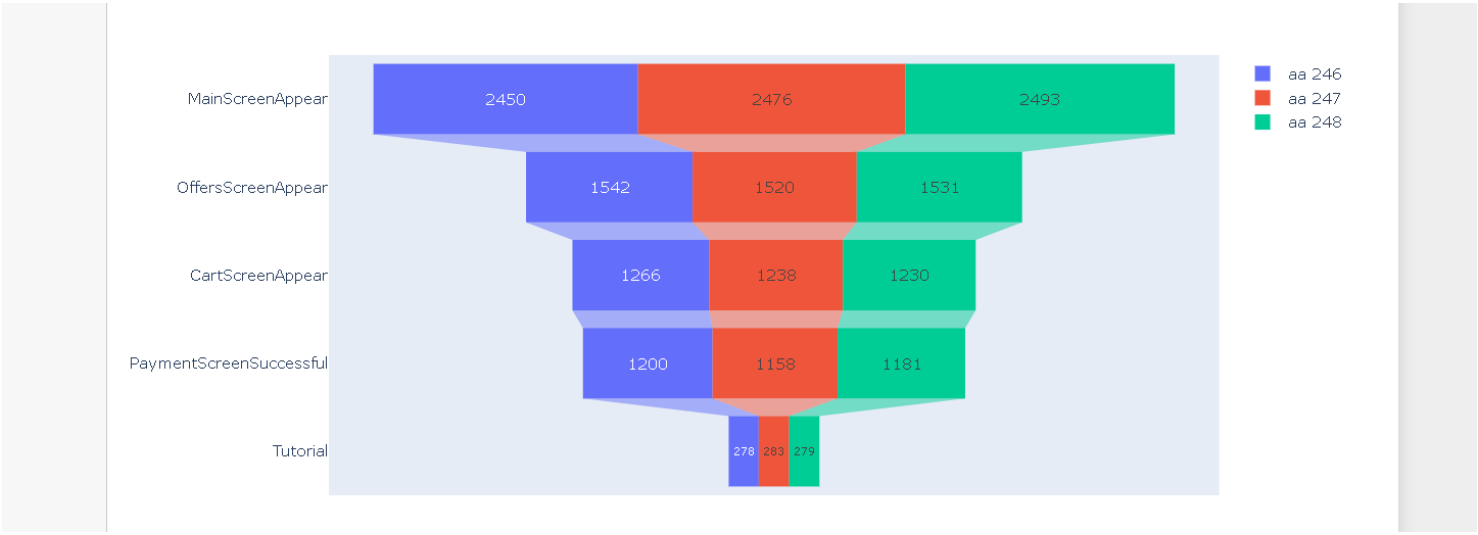
результаты аналогичны

Общий вывод

- **Самое распространенное событие – это просмотр главной страницы 'main_screen'**
- **Больше всего пользователей теряется при переходе с главной страницы к offer screen appear**
- **От просмотра главной страницы до успешной покупки доходит 47,0 % пользователей**
- **Мы выявили что данные собирались всего неделю**

Основной вывод

проверка гипотез не показала существенных изменений поведения пользователея до и после изменения шрифта, построение воронки на графике также не выявило существенных отличий



Изменение шрифта не влияет на конверсию пользователей

Рекомендации

думаю что данные являются непрезентативной выдачей, т.к. собраны всего за неделю

- **Рекомендуется собрать данные хотя бы еще за неделю чтобы опираться на более объемные данные**

подправил рекомендации)