Проект по A/B-тестированию

Задача: провести оценку результатов A/B-теста.

Цель: оценить корректность проведения теста и проанализировать его результаты.

Исходные данные:
Датасет с действиями пользователей, техническое задание и несколько вспомогательных датасетов.

Техническое задание:

- Название теста: `recommender_system_test` ;
- Группы: `A` (контрольная), `B` (новая платёжная воронка);
- Дата запуска: `2020-12-07` ;
- Дата остановки набора новых пользователей: `2020-12-21` ;
- Дата остановки: `2021-01-04` ;
- Аудитория: `15%` новых пользователей из региона `EU` ;
- Назначение теста: тестирование изменений, связанных с внедрением улучшенной рекомендательной системы;
- Ожидаемое количество участников теста: `6000` .
- Ожидаемый эффект: за `14` дней с момента регистрации в системе пользователи покажут улучшение каждой метрики не менее, чем на `10%` :
  - конверсии в просмотр карточек товаров — событие `product_page` ;
  - просмотры корзины — `product_cart` ;
  - покупки — `purchase` .

Данные:

- ab_project_marketing_events.csv
- final_ab_new_users.csv
- final_ab_events.csv
- final_ab_participants.csv

`/datasets/ab_project_marketing_events.csv` — календарь маркетинговых событий на 2020 год;

Структура файла:

- `name` — название маркетингового события;
- `regions` — регионы, в которых будет проводиться рекламная кампания;
- `start_dt` — дата начала кампании;
- `finish_dt` — дата завершения кампании.

`/datasets/final_ab_new_users.csv` — все пользователи, зарегистрировавшиеся в интернет-магазине в период с 7 по 21 декабря 2020 года;

Структура файла:

- `user_id` — идентификатор пользователя;
- `first_date` — дата регистрации;
- `region` — регион пользователя;
- `device` — устройство, с которого происходила регистрация.

`/datasets/final_ab_events.csv` — все события новых пользователей в период с 7 декабря 2020 по 4 января 2021 года;

Структура файла:

- `user_id` — идентификатор пользователя;
- `event_dt` — дата и время события;
- `event_name` — тип события;
- `details` — дополнительные данные о событии. Например, для покупок, purchase, в этом поле хранится стоимость покупки в долларах.

`/datasets/final_ab_participants.csv` — таблица участников тестов.

Структура файла:

- `user_id` — идентификатор пользователя;
- `ab_test` — название теста;
- `group` — группа пользователя.

Содержание

- 1 Исследование данных
  - 1.1 Пропуски и типы данных
  - 1.2 Дубликаты
- 2 Оценка корректности проведения теста
- 3 Исследовательский анализ данных
- 4 Оценка результатов A/B-тестирования

In [1]:

```
# Импорт библиотек
import pandas as pd
import plotly.express as px
from plotly import graph_objects as go
import matplotlib.pyplot as plt
import math
from scipy import stats as st
```

In [2]:

```
# Чтение файлов
try:
    df_ab_project_marketing_events = pd.read_csv('datasets/ab_project_marketing_events.csv',
                                                 parse_dates=['start_dt', 'finish_dt'])

    df_final_ab_new_users = pd.read_csv('datasets/final_ab_new_users.csv', parse_dates=['first_date'])

    df_final_ab_events = pd.read_csv('datasets/final_ab_events.csv', parse_dates=['event_dt'])

    df_final_ab_participants = pd.read_csv('datasets/final_ab_participants.csv')

except:
    df_ab_project_marketing_events = pd.read_csv('/datasets/ab_project_marketing_events.csv',
                                                 parse_dates=['start_dt', 'finish_dt'])

    df_final_ab_new_users = pd.read_csv('/datasets/final_ab_new_users.csv',
                                        parse_dates=['first_date'])

    df_final_ab_events = pd.read_csv('/datasets/final_ab_events.csv',
                                     parse_dates=['event_dt'])

    df_final_ab_participants = pd.read_csv('/datasets/final_ab_participants.csv')
```

1 Исследование данных

1.1 Пропуски и типы данных

In [3]:
```
# Информация о данных, пропусках
for df, name in zip([df_ab_project_marketing_events,
                     df_final_ab_new_users,
                     df_final_ab_events,
                     df_final_ab_participants],
                    ['df_ab_project_marketing_events',
                     'df_final_ab_new_users',
                     'df_final_ab_events',
                     'df_final_ab_participants']):
    print('Датасет', name)
    display(df.head())
    display(df.info())
    print('Уникальные значения:')
    display(df.nunique())
    print()
```

```
Датасет df_ab_project_marketing_events
```

	name	regions	start_dt	finish_dt
0	Christmas&New Year Promo	EU, N.America	2020-12-25	2021-01-03
1	St. Valentine's Day Giveaway	EU, CIS, APAC, N.America	2020-02-14	2020-02-16
2	St. Patric's Day Promo	EU, N.America	2020-03-17	2020-03-19
3	Easter Promo	EU, CIS, APAC, N.America	2020-04-12	2020-04-19
4	4th of July Promo	N.America	2020-07-04	2020-07-11

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 14 entries, 0 to 13
Data columns (total 4 columns):
 #   Column     Non-Null Count  Dtype         
---  ------     --------------  -----         
 0   name       14 non-null     object        
 1   regions    14 non-null     object        
 2   start_dt   14 non-null     datetime64[ns]
 3   finish_dt  14 non-null     datetime64[ns]
dtypes: datetime64[ns](2), object(2)
memory usage: 576.0+ bytes

None

Уникальные значения:

name         14
regions       6
start_dt     14
finish_dt    14
dtype: int64


Датасет df_final_ab_new_users
```

	user_id	first_date	region	device
0	D72A72121175D8BE	2020-12-07	EU	PC
1	F1C668619DFE6E65	2020-12-07	N.America	Android
2	2E1BF1D4C37EA01F	2020-12-07	EU	PC
3	50734A22C0C63768	2020-12-07	EU	iPhone
4	E1BDDCE0DAFA2679	2020-12-07	N.America	iPhone

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 61733 entries, 0 to 61732
Data columns (total 4 columns):
 #   Column      Non-Null Count  Dtype         
---  ------      --------------  -----         
 0   user_id     61733 non-null  object        
 1   first_date  61733 non-null  datetime64[ns]
 2   region      61733 non-null  object        
 3   device      61733 non-null  object        
dtypes: datetime64[ns](1), object(3)
memory usage: 1.9+ MB

None

Уникальные значения:

user_id       61733
first_date       17
region            4
device            4
dtype: int64


Датасет df_final_ab_events
```

	user_id	event_dt	event_name	details
0	E1BDDCE0DAFA2679	2020-12-07 20:22:03	purchase	99.99
1	7B6452F081F49504	2020-12-07 09:22:53	purchase	9.99
2	9CD9F34546DF254C	2020-12-07 12:59:29	purchase	4.99
3	96F27A054B191457	2020-12-07 04:02:40	purchase	4.99
4	1FD7660FDF94CA1F	2020-12-07 10:15:09	purchase	4.99

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 440317 entries, 0 to 440316
Data columns (total 4 columns):
 #   Column      Non-Null Count   Dtype  
---  ------      --------------   -----  
 0   user_id     440317 non-null  object 
 1   event_dt    440317 non-null  datetime64[ns]
 2   event_name  440317 non-null  object 
 3   details     62740 non-null   float64
dtypes: datetime64[ns](1), float64(1), object(2)
memory usage: 13.4+ MB

None

Уникальные значения:

user_id        58703
event_dt      267268
event_name         4
details            4
dtype: int64


Датасет df_final_ab_participants
```

	user_id	group	ab_test
0	D1ABA3E2887B6A73	A	recommender_system_test
1	A7A3664BD6242119	A	recommender_system_test
2	DABC14FDDFADD29E	A	recommender_system_test
3	04988C5DF189632E	A	recommender_system_test
4	482F14783456D21B	B	recommender_system_test

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 18268 entries, 0 to 18267
Data columns (total 3 columns):
 #   Column   Non-Null Count  Dtype 
---  ------   --------------  ----- 
 0   user_id  18268 non-null  object
 1   group    18268 non-null  object
 2   ab_test  18268 non-null  object
dtypes: object(3)
memory usage: 428.3+ KB

None

Уникальные значения:

user_id    16666
group          2
ab_test        2
dtype: int64
```

Заметка:

Датасет *df_final_ab_events* имеет пропуски в графе `details`, но вполне вероятно, что там пусто вследствие отсутствия покупок. Даты были преобразованы на этапе чтения файлов.

1.2 Дубликаты

In [4]:
```
# Явные дубликаты
print('df_ab_project_marketing_events:', df_ab_project_marketing_events.duplicated().sum())
print('df_final_ab_events:', df_final_ab_events.duplicated().sum())
print('df_final_ab_new_users:', df_final_ab_new_users.duplicated().sum())
print('df_final_ab_participants:', df_final_ab_participants.duplicated().sum())
```

```
df_ab_project_marketing_events: 0
df_final_ab_events: 0
df_final_ab_new_users: 0
df_final_ab_participants: 0
```

In [5]:
```
# df_ab_project_marketing_events
for col in ['name', 'regions']:
    print('Оригинал', col, ':', df_ab_project_marketing_events[col].nunique())
    print('Нижний регистр', col, ':', df_ab_project_marketing_events[col].str.lower().nunique())
    print()
```

```
Оригинал name : 14
Нижний регистр name : 14

Оригинал regions : 6
Нижний регистр regions : 6
```

In [6]:
```
# df_final_ab_events
for col in ['user_id', 'region', 'device']:
    print('Оригинал', col, ':', df_final_ab_new_users[col].nunique())
    print('Нижний регистр', col, ':', df_final_ab_new_users[col].str.lower().nunique())
    print()
```

```
Оригинал user_id : 61733
Нижний регистр user_id : 61733

Оригинал region : 4
Нижний регистр region : 4

Оригинал device : 4
Нижний регистр device : 4
```

In [7]:
```
# df_final_ab_new_users
for col in ['user_id', 'event_name']:
    print('Оригинал', col, ':', df_final_ab_events[col].nunique())
    print('Нижний регистр', col, ':', df_final_ab_events[col].str.lower().nunique())
    print()
```

```
Оригинал user_id : 58703
Нижний регистр user_id : 58703

Оригинал event_name : 4
Нижний регистр event_name : 4
```

In [8]:
```
# df_final_ab_participants
for col in ['user_id', 'group', 'ab_test']:
    print('Оригинал', col, ':', df_final_ab_participants[col].nunique())
    print('Нижний регистр', col, ':', df_final_ab_participants[col].str.lower().nunique())
    print()
```

```
Оригинал user_id : 16666
Нижний регистр user_id : 16666

Оригинал group : 2
Нижний регистр group : 2

Оригинал ab_test : 2
Нижний регистр ab_test : 2
```

Заметка:

Дубликатов нет

2 Оценка корректности проведения теста

In [9]:
```
test_start = pd.to_datetime('2020-12-07')
test_end = pd.to_datetime('2020-12-21')
```

In [10]:
```
df_final_ab_participants.group.unique()
```

Out[10]:
```
array(['A', 'B'], dtype=object)
```

In [11]:
```
display(df_final_ab_events.event_dt.min())
display(df_final_ab_events.event_dt.max())
```

```
Timestamp('2020-12-07 00:00:33')

Timestamp('2020-12-30 23:36:33')
```

In [12]:
```
# Получение данных о датах
def get_info(data, col):
    df = data.merge(df_final_ab_participants,
                how='inner',
                left_on=['user_id'],
                right_on=['user_id']
                )
    df = df.query('(ab_test == "recommender_system_test")') # выбираем нужные данные (group == "A") &

    display(df.head())

    display(df[col].min())
    display(df[col].max())
```

In [13]:
```
get_info(df_final_ab_new_users, 'first_date')
```

	user_id	first_date	region	device	group	ab_test
0	D72A72121175D8BE	2020-12-07	EU	PC	A	recommender_system_test
3	E6DE857AFBDC6102	2020-12-07	EU	PC	B	recommender_system_test
7	DD4352CDCF8C3D57	2020-12-07	EU	Android	B	recommender_system_test
10	831887FE7F2D6CBA	2020-12-07	EU	Android	A	recommender_system_test
12	4CB179C7F847320B	2020-12-07	EU	iPhone	B	recommender_system_test

```
Timestamp('2020-12-07 00:00:00')
```

```
Timestamp('2020-12-21 00:00:00')
```

Заметка:

все пользователи, зарегистрировавшиеся в интернет-магазине в период с 7 по 21 декабря 2020 года

Дата запуска: 2020-12-07;
Дата остановки набора новых пользователей: 2020-12-21;

С датами все в порядке

In [14]:
```
get_info(df_final_ab_events, 'event_dt')
```

	user_id	event_dt	event_name	details	group	ab_test
12	831887FE7F2D6CBA	2020-12-07 06:50:29	purchase	4.99	A	recommender_system_test
13	831887FE7F2D6CBA	2020-12-09 02:19:17	purchase	99.99	A	recommender_system_test
14	831887FE7F2D6CBA	2020-12-07 06:50:30	product_cart	NaN	A	recommender_system_test
15	831887FE7F2D6CBA	2020-12-08 10:52:27	product_cart	NaN	A	recommender_system_test
16	831887FE7F2D6CBA	2020-12-09 02:19:17	product_cart	NaN	A	recommender_system_test

```
Timestamp('2020-12-07 00:05:57')
```

```
Timestamp('2020-12-30 12:42:57')
```

Заметка:

Все события новых пользователей в период с 7 декабря 2020 по 4 января 2021 года

Дата остановки: 2021-01-04;

В рамки укладываются

In [15]:
```
# Дополнительный фрейм для работы
df = (df_final_ab_new_users
                .merge(df_final_ab_participants,
                    how='inner',
                    left_on=['user_id'],
                    right_on=['user_id'])
                .query('(ab_test == "recommender_system_test")'))
```

In [16]:
```
print('Охват аудитории новых пользователей по региону EU: {:.2%}'
      .format(df
                .query('region == "EU"')
                .user_id.nunique()
                /
                df_final_ab_new_users.query('(region == "EU")&(first_date >= @test_start)&(first_date <= @test_end)')
                .user_id
                .nunique()
               )
     )
```

```
Охват аудитории новых пользователей по региону EU: 15.00%
```

Заметка:

Аудитория: 15% новых пользователей из региона EU;

У нас 15,00%

In [17]:
```
# Количество участников
df.user_id.nunique()
```

Out[17]: 6701

Заметка:

Ожидаемое количество участников теста: 6000

Количество участников чуть больше (6701)

Проверим:

Время проведения теста. Убедимся, что оно не совпадает с маркетинговыми и другими активностями.

Все события новых пользователей в период с 7 декабря 2020 по 4 января 2021 года

In [18]:
```
# Совпадение с маркетинговыми и другими активностями
df_ab_project_marketing_events.query('(start_dt >= "2020-12-07")|(finish_dt >= "2020-12-07")')
```

Out[18]:

	name	regions	start_dt	finish_dt
0	Christmas&New Year Promo	EU, N.America	2020-12-25	2021-01-03
10	CIS New Year Gift Lottery	CIS	2020-12-30	2021-01-07

In [ ]:
```
marketing_events.query('finish_dt >= "2020-12-07" and start_dt <="2021-01-04"')
```

Заметка:

Последние действия пользователей отмечаются 2020-12-30 12:42:57 .

В наш тест вошла одна промо акция Christmas&New Year Promo

Проверим:

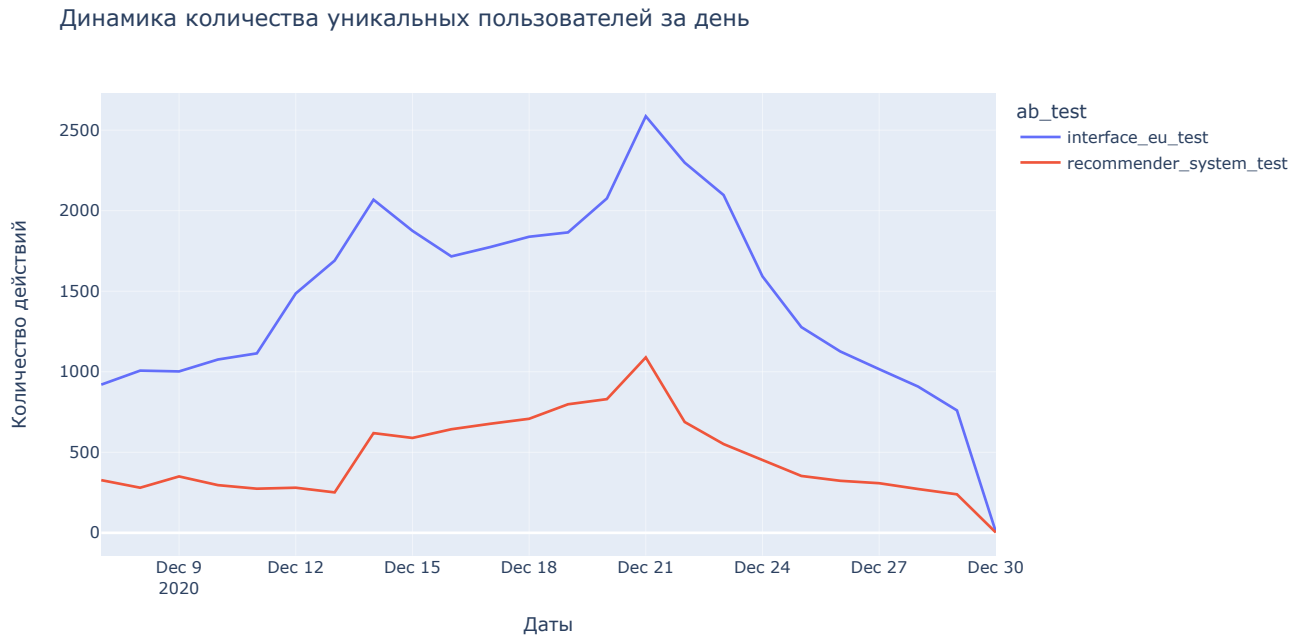
Аудитория теста: удостоверьться, что нет пересечений с конкурирующим тестом и нет пользователей, участвующих в двух группах теста одновременно. Проверим равномерность распределения пользователей по тестовым группам и правильность их формирования.

In [19]:
```
# Временный датафрейм
df_temp = (df_final_ab_events
               .merge(df_final_ab_participants,
                   how='inner',
                   left_on=['user_id'],
                   right_on=['user_id']))

df_temp['date'] = df_temp.event_dt.dt.date

df_temp.head()
```

Out[19]:

	user_id	event_dt	event_name	details	group	ab_test	date
0	96F27A054B191457	2020-12-07 04:02:40	purchase	4.99	B	interface_eu_test	2020-12-07
1	96F27A054B191457	2020-12-08 09:43:14	purchase	4.99	B	interface_eu_test	2020-12-08
2	96F27A054B191457	2020-12-09 00:44:10	purchase	4.99	B	interface_eu_test	2020-12-09
3	96F27A054B191457	2020-12-26 00:33:57	purchase	9.99	B	interface_eu_test	2020-12-26
4	96F27A054B191457	2020-12-07 04:02:41	product_page	NaN	B	interface_eu_test	2020-12-07

In [20]:
```
# Отрисовываем график
fig = px.line(df_temp.pivot_table(index=['ab_test', 'date'], values='user_id', aggfunc='nunique').reset_index(),
              x='date',
              y='user_id',
              title='Динамика количества уникальных пользователей за день',
              color='ab_test'
             )

#fig.update_xaxes(dtick="M1")

fig.update_layout(yaxis_title='Количество действий',
                  xaxis_title='Даты'
                  )

fig.show()
```



Заметка:

Два теста идут одновременно

In [21]:
```
# Проверим есть ли пересечение между типами тестов
if df_final_ab_participants.groupby('user_id')['ab_test'].nunique().max() > 1:
    print('Есть пересечение между тестами')
else:
    print('Нет пересечения между тестами')
```

```
Есть пересечение между тестами
```

In [22]:
```
# Проверим есть ли пользователи, которые участвовали в двух группах
if df_final_ab_participants.groupby('user_id')['group'].nunique().max() > 1:
    print('Некоторые пользователи участвовали в двух группах')
else:
    print('Нет пользователей, участвовавших в двух группах')
```

```
Некоторые пользователи участвовали в двух группах
```

In [23]:
```
# Создадим дф в котором будут содержаться информация об участниках двух тестов
two_tests = df_final_ab_participants\
.groupby('user_id')\
.agg({'ab_test': 'nunique'})\
.reset_index()\
.query('ab_test == 2')

print ('Количество пользователей учавствовавших в обоих тестах:', len(two_tests))
```

```
Количество пользователей учавствовавших в обоих тестах: 1602
```

Заметка:

Необходимо исключить этих пользователей, так как на них влияют два источника, а не один - выводы неоднозначны.

In [24]:
```
print('Количество до удаления:', df_final_ab_participants.shape[0])
df_final_ab_participants = df_final_ab_participants.query('user_id not in @two_tests["user_id"]')
print('Количество после удаления:', df_final_ab_participants.shape[0])
```

```
Количество до удаления: 18268
Количество после удаления: 15064
```

Заметка:

Повторим проверки после удаления данных

In [25]:
```
def get_info1():↔
```

In [26]:
```
1 get_info1()
```

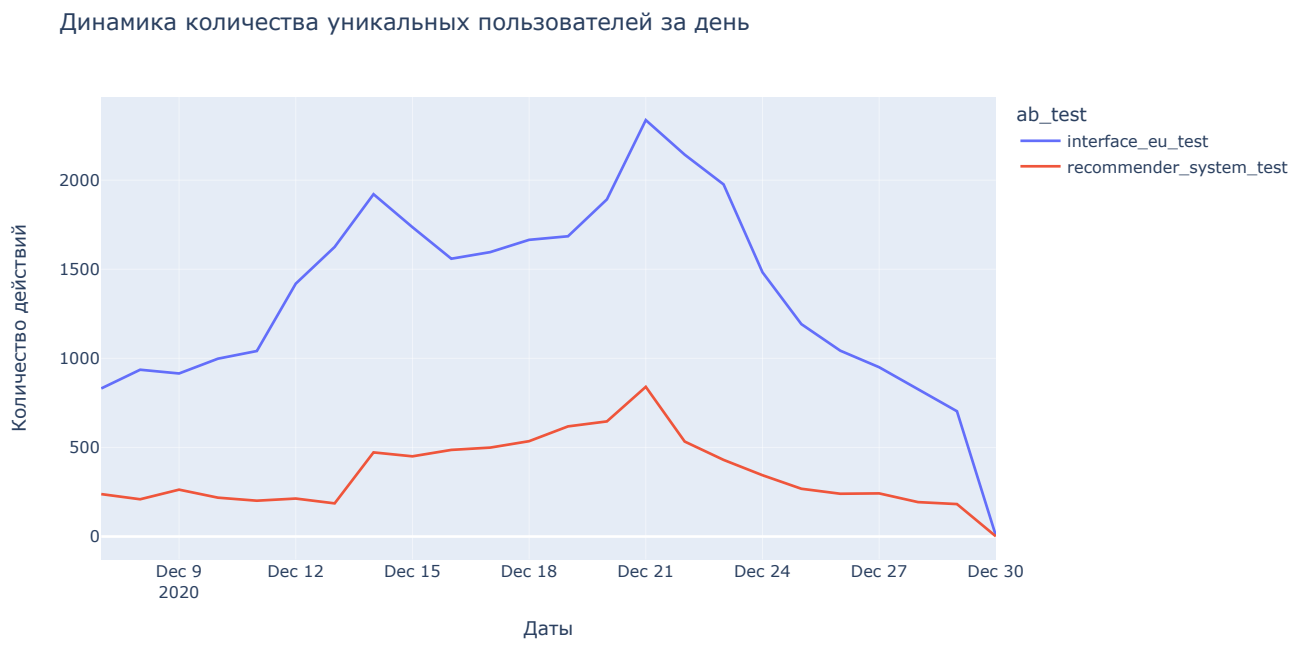
```
Группы: ['A' 'B']
Мин. дата: 2020-12-07 00:00:33
Макс. дата: 2020-12-30 23:36:33

Информация о df_final_ab_new_users
```

	user_id	first_date	region	device	group	ab_test
0	D72A72121175D8BE	2020-12-07	EU	PC	A	recommender_system_test
6	831887FE7F2D6CBA	2020-12-07	EU	Android	A	recommender_system_test
8	4CB179C7F847320B	2020-12-07	EU	iPhone	B	recommender_system_test
9	29C92313A98B1176	2020-12-07	APAC	Android	B	recommender_system_test
11	7D1BFB181017EB46	2020-12-07	CIS	PC	B	recommender_system_test

```
Мин. дата: 2020-12-07 00:00:00
Макс. дата: 2020-12-21 00:00:00
None

Информация о df_final_ab_events
```

	user_id	event_dt	event_name	details	group	ab_test
12	831887FE7F2D6CBA	2020-12-07 06:50:29	purchase	4.99	A	recommender_system_test
13	831887FE7F2D6CBA	2020-12-09 02:19:17	purchase	99.99	A	recommender_system_test
14	831887FE7F2D6CBA	2020-12-07 06:50:30	product_cart	NaN	A	recommender_system_test
15	831887FE7F2D6CBA	2020-12-08 10:52:27	product_cart	NaN	A	recommender_system_test
16	831887FE7F2D6CBA	2020-12-09 02:19:17	product_cart	NaN	A	recommender_system_test

```
Мин. дата: 2020-12-07 00:16:00
Макс. дата: 2020-12-30 06:42:52
None

Охват аудитории новых пользователей по региону EU: 11.22%

Количество участников 5099
```




```
Нет пересечения между тестами
Нет пользователей, участвовавших в двух группах

Количество пользователей учавствовавших в обоих тестах: 0

Доли участников
```

group	user_id	%
A	2903	56.93
B	2196	43.07

```
Всего пользователей во фрейме: 15064
```

Заметка:

- охват аудитории уменьшился;

- пересечений больше нет;
- участников в двух группах теста нет;
- доли различны ~14%;
- даты остались в порядке.

3 Исследовательский анализ данных

Создадим общий фрейм

In [27]:
```
# Дополнительный фрейм для работы
df_for_work = (df_final_ab_new_users.query('(first_date >= @test_start)&(first_date <= @test_end)')
               .merge(df_final_ab_events,
                   how='left',
                   left_on=['user_id'],
                   right_on=['user_id'])
               )

df_for_work.head()
```

Out[27]:

	user_id	first_date	region	device	event_dt	event_name	details
0	D72A72121175D8BE	2020-12-07	EU	PC	2020-12-07 21:52:10	product_page	NaN
1	D72A72121175D8BE	2020-12-07	EU	PC	2020-12-07 21:52:07	login	NaN
2	F1C668619DFE6E65	2020-12-07	N.America	Android	2020-12-07 16:38:09	product_page	NaN
3	F1C668619DFE6E65	2020-12-07	N.America	Android	2020-12-08 02:02:34	product_page	NaN
4	F1C668619DFE6E65	2020-12-07	N.America	Android	2020-12-23 14:35:41	product_page	NaN

In [28]:
```
# Дополнительный фрейм для работы
df_for_work = (df_for_work
               .merge(df_final_ab_participants,
                   how='left',
                   left_on=['user_id'],
                   right_on=['user_id'])
               )

df_for_work = df_for_work.query('ab_test == "recommender_system_test"').reset_index(drop=True)

display(df_for_work.head())
print('Записей:', df_for_work.shape[0])
print('Количество пользователей:', df_for_work.user_id.nunique())
```

	user_id	first_date	region	device	event_dt	event_name	details	group	ab_test
0	D72A72121175D8BE	2020-12-07	EU	PC	2020-12-07 21:52:10	product_page	NaN	A	recommender_system_test
1	D72A72121175D8BE	2020-12-07	EU	PC	2020-12-07 21:52:07	login	NaN	A	recommender_system_test
2	831887FE7F2D6CBA	2020-12-07	EU	Android	2020-12-07 06:50:29	purchase	4.99	A	recommender_system_test
3	831887FE7F2D6CBA	2020-12-07	EU	Android	2020-12-09 02:19:17	purchase	99.99	A	recommender_system_test
4	831887FE7F2D6CBA	2020-12-07	EU	Android	2020-12-07 06:50:30	product_cart	NaN	A	recommender_system_test

```
Записей: 21115
Количество пользователей: 5099
```

In [29]:
```
df_for_work['date_event'] = df_for_work.event_dt.dt.date
df_for_work['date_event'] = pd.to_datetime(df_for_work['date_event'])
df_for_work['dt_delta'] = df_for_work['date_event'] - df_for_work['first_date']

display(df_for_work.head())
print('Количество записей:', df_for_work.shape[0])
```

	user_id	first_date	region	device	event_dt	event_name	details	group	ab_test	date_event	dt_delta
0	D72A72121175D8BE	2020-12-07	EU	PC	2020-12-07 21:52:10	product_page	NaN	A	recommender_system_test	2020-12-07	0 days
1	D72A72121175D8BE	2020-12-07	EU	PC	2020-12-07 21:52:07	login	NaN	A	recommender_system_test	2020-12-07	0 days
2	831887FE7F2D6CBA	2020-12-07	EU	Android	2020-12-07 06:50:29	purchase	4.99	A	recommender_system_test	2020-12-07	0 days
3	831887FE7F2D6CBA	2020-12-07	EU	Android	2020-12-09 02:19:17	purchase	99.99	A	recommender_system_test	2020-12-09	2 days
4	831887FE7F2D6CBA	2020-12-07	EU	Android	2020-12-07 06:50:30	product_cart	NaN	A	recommender_system_test	2020-12-07	0 days

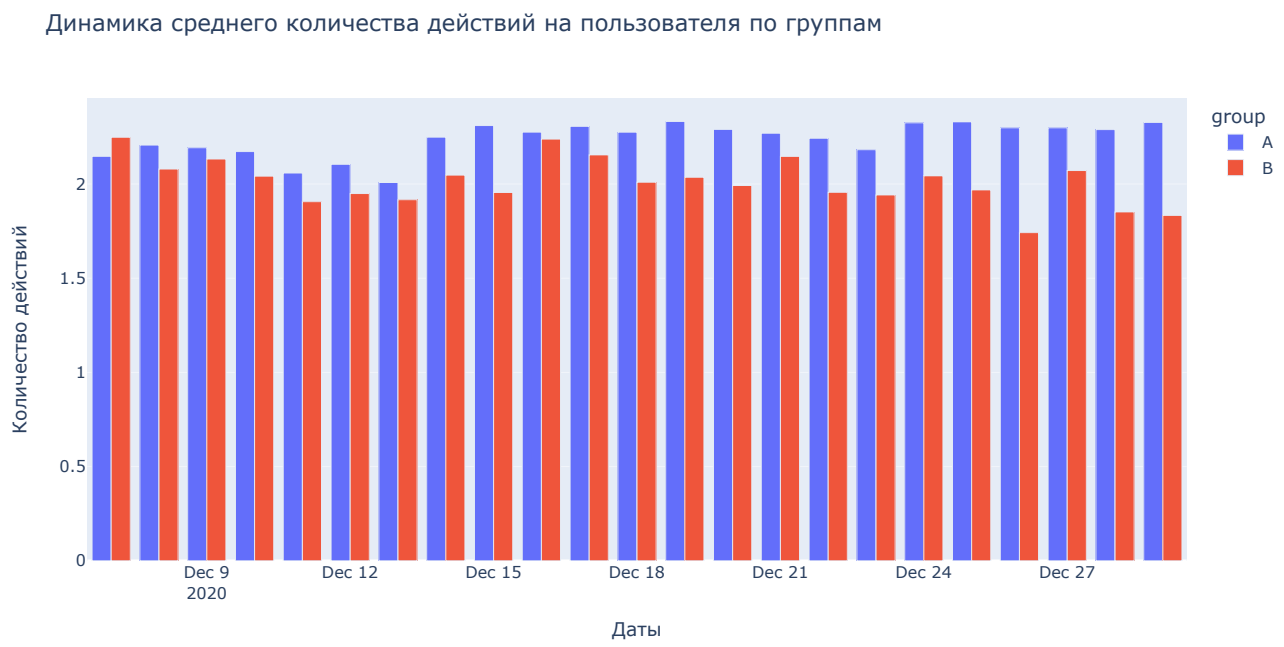
```
Количество записей: 21115
```

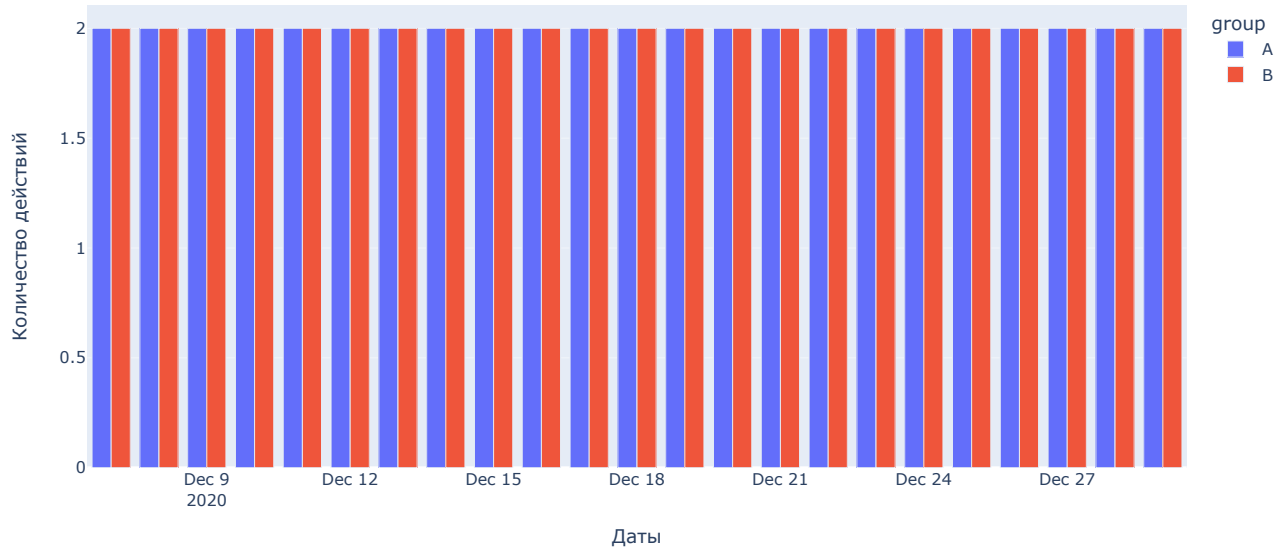
In [30]:
```
# Оставим записи с действиями за две недели после регистрации
df_for_work = df_for_work[df_for_work['dt_delta'] <= pd.Timedelta(days=14)]

display(df_for_work.head())
print('Количество записей:', df_for_work.shape[0])
```

	user_id	first_date	region	device	event_dt	event_name	details	group	ab_test	date_event	dt_delta
0	D72A72121175D8BE	2020-12-07	EU	PC	2020-12-07 21:52:10	product_page	NaN	A	recommender_system_test	2020-12-07	0 days
1	D72A72121175D8BE	2020-12-07	EU	PC	2020-12-07 21:52:07	login	NaN	A	recommender_system_test	2020-12-07	0 days
2	831887FE7F2D6CBA	2020-12-07	EU	Android	2020-12-07 06:50:29	purchase	4.99	A	recommender_system_test	2020-12-07	0 days
3	831887FE7F2D6CBA	2020-12-07	EU	Android	2020-12-09 02:19:17	purchase	99.99	A	recommender_system_test	2020-12-09	2 days
4	831887FE7F2D6CBA	2020-12-07	EU	Android	2020-12-07 06:50:30	product_cart	NaN	A	recommender_system_test	2020-12-07	0 days

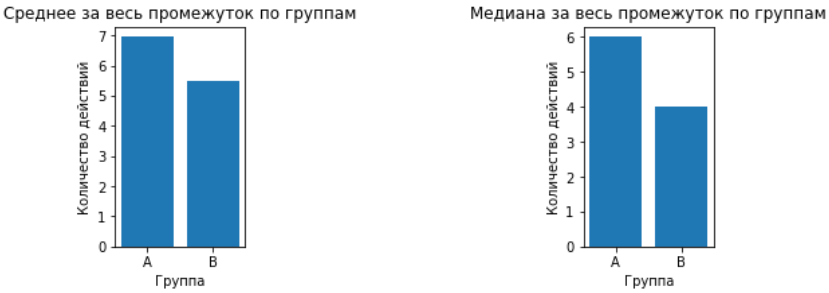
```
Количество записей: 18329
```

In [31]:
```
# Распределение в выборках количества событий на пользователя↔
```





In [32]:
```
# Среднее за весь промежуток по группам
plt.figure(figsize=(8,3))
ax1 = plt.subplot(1, 4, 1)
df_temp = df_for_work.pivot_table(index=['group', 'user_id'], values='event_dt', aggfunc='count')
df_temp = df_temp.pivot_table(index=['group'],
                              values='event_dt',
                              aggfunc='mean').reset_index()
plt.bar(data=df_temp,
        x='group',
        height='event_dt'
        )

plt.ylabel('Количество действий')
plt.xlabel('Группа')
plt.title("Среднее за весь промежуток по группам")

ax2 = plt.subplot(1, 4, 4)
df_temp = df_for_work.pivot_table(index=['group', 'user_id'], values='event_dt', aggfunc='count')
df_temp = df_temp.pivot_table(index=['group'],
                              values='event_dt',
                              aggfunc='median').reset_index()
plt.bar(data=df_temp,
        x='group',
        height='event_dt'
        )

plt.ylabel('Количество действий')
plt.xlabel('Группа')
plt.title("Медиана за весь промежуток по группам")

plt.show()
```



Заметка:

Среднее количество действий находится в пределах 2 шт., медианное = 2 шт.

Если посмотреть на весь промежуток, то количество вырастет и станет сильнее разниться.

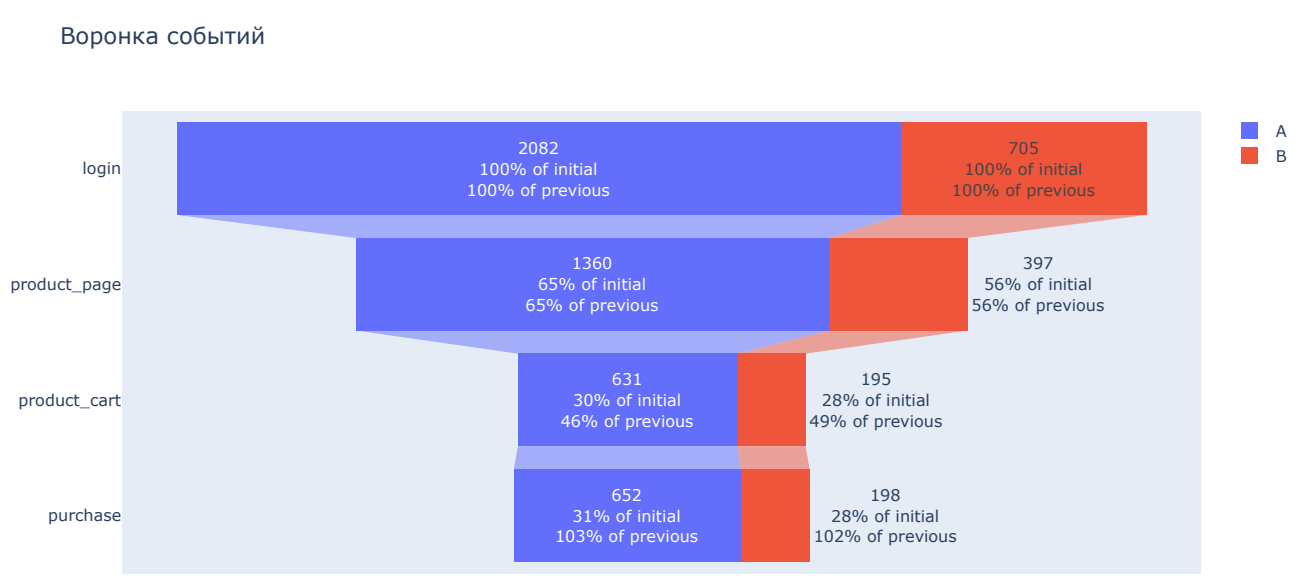
In [33]:
```
# Распределение числа событий в выборках по дням↔
```

In [34]:
```
# Для упорядочивания воронки событий созжаним датафрейм
group_events_count = df_for_work.groupby(['event_name','group']).agg({'user_id':'nunique'}).reset_index()

# Изменим индексы для порядка действий
new_index = [0, 1, 4, 5,  2, 3, 6, 7]

group_events_count = group_events_count.reindex(new_index)

group_events_count
```

Out[34]:

	event_name	group	user_id
0	login	A	2082
1	login	B	705
4	product_page	A	1360
5	product_page	B	397
2	product_cart	A	631
3	product_cart	B	195
6	purchase	A	652
7	purchase	B	198

In [35]:
```
# Строим воронку событий по уникальным пользователям↔
```



Заметка:

Часть заказов минует `product_cart` , переходя сразу к `purchase` .

Наибольшие потери конверсии наблюдаются на шаге 3 - переход с `product_page` к `product_cart` .

Особенности данных:

(для A/B- тестирования)

- отсутствуют данные о действиях пользователей с 31.12.2020 по 04.01.2021;
- в период проведения теста для интересующих нас пользователей параллельно проходила акция `Christmas&New Year Promo` ;
- во время иследования проходило другое тестирование. Все пользователи попавшие в оба теста были удалены, т.к. их действия не будут репрезентативными;
- без учета исключенных пользователей аудитория теста равна 15%, что соответствует ТЗ, однако после исключения не репрезентативных пользователей доля снизилась до 11.22%, что является нарушением ТЗ;
- медианы по количеству совершенных действий у пользователей в группах различаются (A=6, B=4), однако в разрезе времени они примерно одинаковы.

? Заметка ?

Возможно, стоило исключать пользователей только группы B?

4 Оценка результатов A/B-тестирования

Подготовим данные к тесту.

Для удобства работы по строкам, создадим датафрейм с группами в виде столбцов и строк - событий.

In [36]:
```
# Датафрейм воронки для теста - различия в одинаковых событиях
event_group_test = df_for_work.pivot_table(index='event_name',
                                           columns='group',
                                           values='user_id',
                                           aggfunc='nunique'
                                          ).reset_index()

event_group_test
```

Out[36]:

group	event_name	A	B
0	login	2082	705
1	product_cart	631	195
2	product_page	1360	397
3	purchase	652	198

In [37]:
```
# Упорядочим события (как в воронке)
new_index = [0, 2, 1, 3]

event_group_test = event_group_test.reindex(new_index)

event_group_test
```

Out[37]:

group	event_name	A	B
0	login	2082	705
2	product_page	1360	397
1	product_cart	631	195
3	purchase	652	198

При проведении Z-теста в знаменателях пропорций успеха указывается размер группы. Для более короткого обращения запишем в датафрейм.

In [38]:
```
users_bygroup = df_for_work.groupby('group')['user_id'].nunique()
users_bygroup
```

Out[38]:
```
group
A    2082
B     706
Name: user_id, dtype: int64
```

Гипотезы:

Наличие изменения конверсии группы B по отношению к группе A:

H_0: Среднее количество пользователей, совершивших значимое событие в группах A и B, равно.

H_1: Среднее количество пользователей, совершивших значимое событие в группах A и B, различается.

Функция Z-теста:

Z-тест будет проходить по этапам (событиям) воронки, в следствие чего будет выполняться несколько раз (4 теста одновременно), что вводит дополнительную погрешность

Заметка: применим метод Шидака для расчёта требуемого уровня значимости

In [39]:
```
# Задаём параметры
alpha = 0.05
print('Заданный уровень значимости: α =', alpha)

m = event_group_test.shape[0]
print('Число тестов: m =', m)

alpha_by_shidok = 1 - (1 - alpha) ** (1 / m)
print('Требуемый уровень значимости по методу Шидака:', alpha_by_shidok)
```

```
Заданный уровень значимости: α = 0.05
Число тестов: m = 4
Требуемый уровень значимости по методу Шидака: 0.012741455098566168
```

In [40]:
```python
# Функция для проведения Z-теста
def z_test(group1, group2, alpha):
    for i in event_group_test.index:

        print('Действие: {}'.format(event_group_test['event_name'][i]))

        # Пропорция успехов в первой группе:
        p1 = event_group_test[group1][i] / users_bygroup[group1]

        # Пропорция успехов во второй группе:
        p2 = event_group_test[group2][i] / users_bygroup[group2]

        print('Группа', group1, ':',
              event_group_test[group1][i],
              users_bygroup[group1])
        print('Группа', group2, ':',
              event_group_test[group2][i],
              users_bygroup[group2]
             )

        # Пропорция успехов в комбинированном датасете:
        p_combined = (
            (event_group_test[group1][i] + event_group_test[group2][i]) /
                      (users_bygroup[group1] + users_bygroup[group2])
        )

        # Разница пропорций в датасетах
        difference = p1 - p2

        # Считаем статистику в ст.отклонениях стандартного нормального распределения
        z_value = difference / \
        math.sqrt(p_combined * (1 - p_combined) * (1/users_bygroup[group1] + 1/users_bygroup[group2]))

        # Задаем стандартное нормальное распределение (среднее 0, ст.отклонение 1)
        distr = st.norm(0, 1)

        p_value = (1 - distr.cdf(abs(z_value))) * 2

        print('{} p-значение: {}'.format(event_group_test['event_name'][i], p_value))

        if (p_value < alpha):
            print("ОТВЕРГАЕМ нулевую гипотезу: между группами есть значимая разница")
        else:
            print("НЕ получилось отвергнуть нулевую гипотезу, нет оснований считать группы разными")

        print('')
```

In [41]:
```python
# Запускаем Z-тест
z_test("A", "B", alpha_by_shidok)
```

```
Действие: login
Группа A : 2082 2082
Группа B : 705 706
login p-значение: 0.08587401754779211
НЕ получилось отвергнуть нулевую гипотезу, нет оснований считать группы разными

Действие: product_page
Группа A : 1360 2082
Группа B : 397 706
product_page p-значение: 1.5371909704686715e-05
ОТВЕРГАЕМ нулевую гипотезу: между группами есть значимая разница

Действие: product_cart
Группа A : 631 2082
Группа B : 195 706
product_cart p-значение: 0.1766337419130104
НЕ получилось отвергнуть нулевую гипотезу, нет оснований считать группы разными

Действие: purchase
Группа A : 652 2082
Группа B : 198 706
purchase p-значение: 0.10281767567786759
НЕ получилось отвергнуть нулевую гипотезу, нет оснований считать группы разными
```

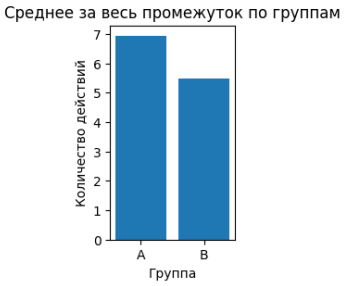
Итоги:

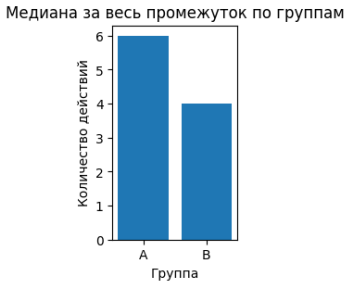
- отсутствуют данные о действиях пользователей с 31.12.2020 по 04.01.2021;
- в период проведения теста проходила акция Christmas&New Year Promo;
- параллельно проходило другое тестирование. Пользователи, попавшие в обе группы были удалены т.к. их действия не будут репрезентативными;
- после удаления осталось количество пользователей сократилось (требуется 6000) - нарушение ТЗ;
- с исключенными пользователями аудитория теста равна 15% (EU) - соответствует ТЗ, однако после исключении не репрезентативных пользователей доля падает до 11,2% - нарушение ТЗ;
- доли количества участников теста в группах A и B;

Доли участников

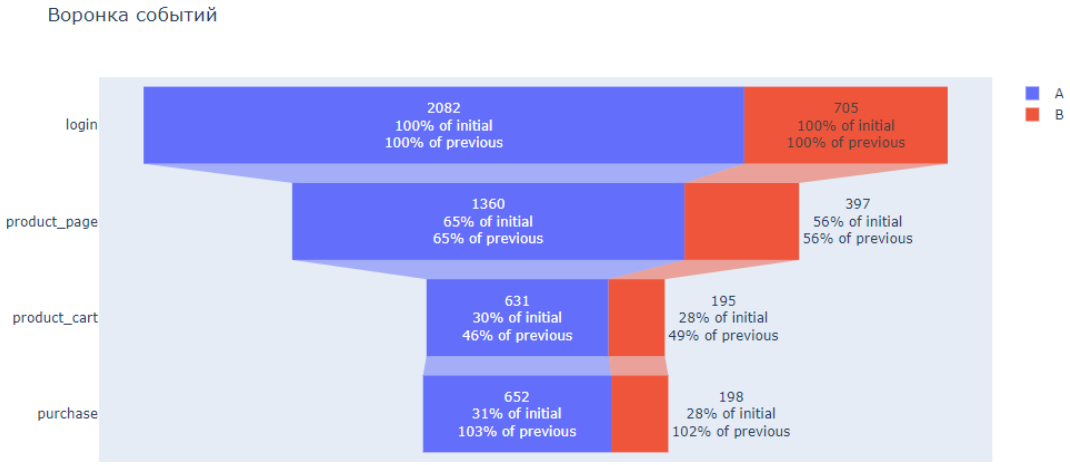
group	user_id	%
A	2903	56.93
B	2196	43.07

- количество совершенных действий у пользователей в группах различаются, если брать медиану за все дней;





- в группе А до конца воронки (совершения целевого действия - покупки) доходят на 3% больше пользователей, чем в группе В;



Результаты множественного тестирования

Был применён метод Шидака для расчёта требуемого уровня значимости.

По результатам множественного тестирования, по 3-м действиям (`login` , `product_cart` , `purchase`) не удалось отвергнуть нулевую гипотезу (нет оснований считать, что доли пользователей разные). По действию `product_page` отвергнули гипотезу (доли разные).

Исходя из произведенного тестирования изменения, связанные с внедрением улучшенной рекомендательной системы, не дали нужного результата, однако результаты **нельзя назвать корректными**.

В ходе проверки данных было выявлено множество ошибок, нарушений ТЗ, что не дает уверенности в проведенном тестировании.

Рекоммендации:

- подготовить данные и провести тест повторно, с учетом сделанных замечаний;
- стоит проверить механизм распределения по группам.