

# Проведение А/В-тестирования интернет-магазина

## Описание проекта

Для оценки изменений, связанных с внедрением улучшенной рекомендательной системы в интернет-магазине, проведено А/В-тестирование.

## Техническое задание проекта

- Название теста: recommender\_system\_test ;
- Группы: А (контрольная), В (новая платёжная воронка);
- Дата запуска: 2020-12-07;
- Дата остановки набора новых пользователей: 2020-12-21;
- Дата остановки: 2021-01-04;
- Ожидаемое количество участников теста: 15% новых пользователей из региона EU;
- Назначение теста: тестирование изменений, связанных с внедрением улучшенной рекомендательной системы;
- Ожидаемый эффект: за 14 дней с момента регистрации в системе пользователи покажут улучшение каждой метрики не менее, чем на 5 процентных пунктов:
  - конверсии в просмотр карточек товаров — событие product\_page
  - просмотры корзины — product\_cart
  - покупки — purchase.

## Описание данных

Структура датасета ab\_project\_marketing\_events.csv - календарь маркетинговых событий на 2020 год:

- name — название маркетингового события;
- regions — регионы, в которых будет проводиться рекламная кампания;
- start\_dt — дата начала кампании;
- finish\_dt — дата завершения кампании.

Структура датасета final\_ab\_new\_users.csv - все пользователи, зарегистрировавшиеся в интернет-магазине в период с 7 по 21 декабря 2020 года:

- user\_id — идентификатор пользователя;
- first\_date — дата регистрации;

- `region` — регион пользователя;
- `device` — устройство, с которого происходила регистрация.

*Структура датасета `final_ab_events.csv` - все события новых пользователей в период с 7 декабря 2020 по 4 января 2021 года:*

- `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.

*Структура датасета `final_ab_participants.csv` - таблица участников тестов:*

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

## Цель исследования

- Проверить данные на соответствие ТЗ
- Провести исследовательский анализ
- Проанализировать результаты теста и дать рекомендации

## Декомпозиция

### Шаг 1. Загрузка данных

### Шаг 2. Предобработка данных

- Исследовать пропущенные значения;
- Исследовать соответствие типов;
- Исследовать дубликаты;
- Проверить корректность наименований колонок;
- Переименовать колонки, если это необходимо;
- Удалить дубликаты;
- Привести типы;
- Заменить пропущенные значения, если это возможно;

### Шаг 3. Проверка соответствия данных теста Техническому заданию:

- Выделить пользователей участвующих в тесте и проверить: период набора пользователей в тест и его соответствие требованиям технического задания;
- Проверить следующее условие: все ли попавшие в тест пользователи представляют целевой регион и составляет ли общее количество пользователей из целевого региона 15% от общего числа пользователей из целевого региона, зарегистрированных в период набора пользователей в тест;
- Проверить равномерность распределения пользователей по группам теста и корректность их формирования;
- Удостовериться, что нет пересечений с конкурирующим тестом и нет пользователей, участвующих в двух группах теста одновременно.

#### **Шаг 4. Изучить данные о пользовательской активности:**

- проверить, есть пользователи, которые не совершили событий после регистрации, изучите их количество и распределение между группами теста;
- Оставить только те события, которые были совершены в первые 14 дней с момента регистрации; принять решение, оставлять ли пользователей, не проживших 14 дней.
- Оценить, когда пользователи совершают свои первые события каждого вида, построить гистограмму возраста событий.
- Проверить, какого размера выборки потребуются для получения достоверного результата, при базовой конверсии 50%.

#### **Шаг 5. Исследовательский анализ данных**

- Распределение количества событий на пользователя в разрезе групп теста, сравнить её средние значения между собой у групп теста;
- Динамика количества событий в группах теста по дням: изучить распределение числа событий по дням и сравнить динамику групп теста между собой.
- Убедиться, что время проведения теста не совпадает с маркетинговыми и другими активностями. Настроить автоматическую проверку, выдающую список событий, пересекающихся с тестом. При необходимости оценить воздействие маркетинговых событий на динамику количества событий.
- Построить простые продуктовые воронки для двух групп теста с учетом логической последовательности совершения событий;
- Изучить изменение конверсии в продуктовой воронке тестовой группы, по сравнению с контрольной и ответить на вопрос: наблюдается ли ожидаемый эффект увеличения конверсии в группе В на 10 процентных пунктов, относительно конверсии в группе А?

#### **Шаг 6. Провести оценку результатов A/B-тестирования:**

- Проверить статистическую разницу долей z-критерием по событиям.

#### **Шаг 6. Общий вывод:**

- Общее заключение о корректности проведения теста и рекомендации.

#### **Шаг 1. Загрузка данных**

Импортируем необходимые библиотеки и загружаем данные из файлов в датафреймы:

```
In [1]: # необходимо обновить библиотеку для корректного отображения графиков
```

```
!pip install plotly==5.9.0
```

```
Requirement already satisfied: plotly==5.9.0 in c:\users\acer\anaconda3\lib\site-packages (5.9.0)
```

```
Requirement already satisfied: tenacity>=6.2.0 in c:\users\acer\anaconda3\lib\site-packages (from plotly==5.9.0) (8.0.1)
```

```
In [2]: import pandas as pd  
import datetime as dt
```

```
import numpy as np  
from scipy import stats as st  
import math as mth  
  
import plotly.express as px  
from plotly import graph_objects as go  
from plotly.subplots import make_subplots  
  
import warnings  
warnings.filterwarnings("ignore")
```

```
In [3]: try:  
    market_events = pd.read_csv('~/Desktop/учеба яндекс/A_B_tests_final/ab_project_marketing_events.csv')  
    users = pd.read_csv('/Desktop/учеба яндекс/final_project/final_ab_new_users.csv')  
    ab_events = pd.read_csv('/Desktop/учеба яндекс/final_project/final_ab_events.csv')  
    ab_participants = pd.read_csv('/Desktop/учеба яндекс/final_project/final_ab_participants.csv')  
except:  
    market_events = pd.read_csv('https://code.s3.yandex.net/datasets/ab_project_marketing_events.csv')  
    users = pd.read_csv('https://code.s3.yandex.net/datasets/final_ab_new_users.csv')  
    ab_events = pd.read_csv('https://code.s3.yandex.net/datasets/final_ab_events.csv')  
    ab_participants = pd.read_csv('https://code.s3.yandex.net/datasets/final_ab_participants.csv')
```

Используемые функции:

```
In [4]: # функция на вывод основных характеристик датасета
def data(data, name):
    # Выведем 5 строк датафреймов
    display(data.head(10).style.set_caption(name))
    # Выведем основные характеристики датафрейма (типы столбцов, пропущенные значения)
    print(data.info())
    print()
    # Проверим, присутствуют ли пропуски в датафрейме:
    print(f'Количество пропусков в датафрейме {name} - {data.isna().sum()}')
    print()
```

```
In [5]: # функция на вывод уникальных строковых значений
def unique_rows(data):
    for i in data.columns:
        if data[i].dtype == 'O' and i != 'user_id':
            print(f'Уникальные значения в столбце {i}')
            print("\n".join(map(str, data[i].unique()))), sep='\n'
            print()
```

Рассмотрим полученные данные из датафреймов:

```
In [6]: dataframes = [market_events, users, ab_events, ab_participants]
dataframes_names = list(['market_events', 'users', 'ab_events', 'ab_participants'])
```

```
In [7]: count = 0
for i in dataframes:
    print('Характеристики датафрейма', dataframes_names[count])
    data(i, dataframes_names[count])
    count += 1
```

Характеристики датафрейма market\_events

market\_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
5	Black Friday Ads Campaign	EU, CIS, APAC, N.America	2020-11-26	2020-12-01
6	Chinese New Year Promo	APAC	2020-01-25	2020-02-07
7	Labor day (May 1st) Ads Campaign	EU, CIS, APAC	2020-05-01	2020-05-03
8	International Women's Day Promo	EU, CIS, APAC	2020-03-08	2020-03-10
9	Victory Day CIS (May 9th) Event	CIS	2020-05-09	2020-05-11

```
<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    object  
 3   finish_dt   14 non-null    object  
dtypes: object(4)
memory usage: 576.0+ bytes
None
```

```
Количество пропусков в датафрейме market_events - name          0
regions      0
start_dt     0
finish_dt    0
dtype: int64
```

Характеристики датафрейма users

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
5	137119F5A9E69421	2020-12-07	N.America	iPhone
6	62F0C741CC42D0CC	2020-12-07	APAC	iPhone
7	8942E64218C9A1ED	2020-12-07	EU	PC
8	499AFACF904BBAE3	2020-12-07	N.America	iPhone
9	FFCEA1179C253104	2020-12-07	EU	Android

```
<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   object 
 2   region      61733 non-null   object 
 3   device       61733 non-null   object 
dtypes: object(4)
memory usage: 1.9+ MB
None
```

```
Количество пропусков в датафрейме users - user_id          0
first_date    0
region        0
device         0
dtype: int64
```

Характеристики датафрейма ab\_events

ab\_events

	user_id	event_dt	event_name	details
0	E1BDDCE0DAFA2679	2020-12-07 20:22:03	purchase	99.990000
1	7B6452F081F49504	2020-12-07 09:22:53	purchase	9.990000
2	9CD9F34546DF254C	2020-12-07 12:59:29	purchase	4.990000
3	96F27A054B191457	2020-12-07 04:02:40	purchase	4.990000
4	1FD7660FDF94CA1F	2020-12-07 10:15:09	purchase	4.990000
5	831887FE7F2D6CBA	2020-12-07 06:50:29	purchase	4.990000
6	6B2F726BFD5F8220	2020-12-07 11:27:42	purchase	4.990000
7	BEB37715AACF53B0	2020-12-07 04:26:15	purchase	4.990000
8	B5FA27F582227197	2020-12-07 01:46:37	purchase	4.990000
9	A92195E3CFB83DBD	2020-12-07 00:32:07	purchase	4.990000

```
<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   object  
 2   event_name  440317 non-null   object  
 3   details     62740 non-null    float64 
dtypes: float64(1), object(3)
memory usage: 13.4+ MB
None
```

Количество пропусков в датафрейме ab\_events - user\_id 0  
event\_dt 0  
event\_name 0  
details 377577  
dtype: int64

Характеристики датафрейма ab\_participants

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
5	4FF2998A348C484F	A	recommender_system_test
6	7473E0943673C09E	A	recommender_system_test
7	C46FE336D240A054	A	recommender_system_test
8	92CB588012C10D3D	A	recommender_system_test
9	057AB296296C7FC0	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
```

```
Количество пропусков в датафрейме ab_participants - user_id      0
group      0
ab_test    0
dtype: int64
```

#### Вывод:

- всего представлены четыре датасета - market\_events, users, ab\_events и ab\_participants;
- в датасетах market\_events, users, ab\_events и ab\_participants 14, 61733, 440317 и 18268 строк соответственно;
- 377577 пропусков обнаружено в датасете ab\_events, в столбце details (дополнительные данные о событиях);
- отметим, что в датасетах market\_events, users, ab\_events столбцы со временем относятся к типу данных object.

## Шаг 2. Предобработка данных

Приведем столбцы датасетов market\_events (столбцы start\_dt и finish\_dt), users (столбец first\_date) и ab\_events( event\_dt ) к типу времени datetime:

```
In [8]: market_events['start_dt'] = pd.to_datetime(market_events['start_dt'], format='%Y-%m-%d %H:%M:%S')
market_events['finish_dt'] = pd.to_datetime(market_events['finish_dt'], format='%Y-%m-%d %H:%M:%S')

users['first_date'] = pd.to_datetime(users['first_date'], format='%Y-%m-%d %H:%M:%S')

ab_events['event_dt'] = pd.to_datetime(ab_events['event_dt'], format='%Y-%m-%d %H:%M:%S')
```

Проверим, присутствуют ли явные дубликаты в датафрейме:

```
In [9]: count = 0
for i in dataframes:
    print(f'Всего дупликатов в датафрейме {dataframes_names[count]} - {i.duplicated().sum()}, '
          f'то есть {round(100 * i.duplicated().sum() / len(i), 2)} процентов от всех данных')
    count += 1
```

Всего дупликатов в датафрейме market\_events - 0, то есть 0.0 процентов от всех данных

Всего дупликатов в датафрейме users - 0, то есть 0.0 процентов от всех данных

Всего дупликатов в датафрейме ab\_events - 0, то есть 0.0 процентов от всех данных

Всего дупликатов в датафрейме ab\_participants - 0, то есть 0.0 процентов от всех данных

Проверим, присутствуют ли неявные дубликаты в датафреймах market\_events:

```
In [10]: unique_rows(market_events)
unique_rows(users)
unique_rows(ab_events)
unique_rows(ab_participants)
```

Уникальные значения в столбце name  
Christmas&New Year Promo  
St. Valentine's Day Giveaway  
St. Patric's Day Promo  
Easter Promo  
4th of July Promo  
Black Friday Ads Campaign  
Chinese New Year Promo  
Labor day (May 1st) Ads Campaign  
International Women's Day Promo  
Victory Day CIS (May 9th) Event  
CIS New Year Gift Lottery  
Dragon Boat Festival Giveaway  
Single's Day Gift Promo  
Chinese Moon Festival

Уникальные значения в столбце regions

EU, N.America  
EU, CIS, APAC, N.America  
N.America  
APAC  
EU, CIS, APAC  
CIS

Уникальные значения в столбце region

EU  
N.America  
APAC  
CIS

Уникальные значения в столбце device

PC  
Android  
iPhone  
Mac

Уникальные значения в столбце event\_name  
purchase  
product\_cart  
product\_page  
login

Уникальные значения в столбце group

A  
B

```
Уникальные значения в столбце ab_test
recommender_system_test
interface_eu_test
```

В датасетах не обнаружены неявные дупликаты.

Рассмотрим столбец details датасета ab\_events, в котором были обнаружены пропуски:

```
In [11]: print(f'Количество пропусков в столбце details - {ab_events["details"].isna().sum()}, '
           f'что составляет {round(100 * ab_events["details"].isna().sum() / len(ab_events), 2)} %')
```

Количество пропусков в столбце details - 377577, что составляет 85.75 %

```
In [12]: ab_events.query('details.isna()').pivot_table(index='event_name', values='details', aggfunc='count')
```

Out[12]:

details	
event_name	
login	0
product_cart	0
product_page	0

Судя по всему, заполнение столбца details происходит только при оплате - в столбец вносится сумма покупки. В остальных случаях (например, регистрация), данные не записываются. Следовательно, удалять пропуски в столбце не имеет смысла.

### Шаг 3. Проверка данных на соответствие ТЗ:

- период набора пользователей в тест и его соответствие требованиям технического задания;

```
In [13]: # проверим минимальную и максимальную дату столбца first_date датасета users
print(f'Начало периода - {min(users["first_date"])}, конец периода - {max(users["first_date"])}')
```

Начало периода - 2020-12-07 00:00:00, конец периода - 2020-12-23 00:00:00

По ТЗ все пользователи зарегистрировались в период с 7 по 21 декабря 2020 года. В данном случае, данных чуть больше.

Оставим данные до 21 декабря включительно:

```
In [14]: users = users[users["first_date"].dt.date <= dt.datetime.strptime('2020-12-21', '%Y-%m-%d').date()]
```

```
In [15]: # проверим минимальную и максимальную дату столбца event_dt датасета ab_events
print(f'Начало периода - {min(ab_events["event_dt"])}, конец периода - {max(ab_events["event_dt"])}')
```

Начало периода - 2020-12-07 00:00:33, конец периода - 2020-12-30 23:36:33

По ТЗ события новых пользователей совершились в период с 7 декабря 2020 по 4 января 2021. Отсутствуют данные за 5 дней. Причинами отсутствия могут быть неверная выгрузка данных, намеренная остановка теста ввиду набора достаточного количества данных.

- Проверим следующее условие: все ли попавшие в тест пользователи представляют целевой регион и составляет ли общее количество пользователей из целевого региона 15% от общего числа пользователей из целевого региона, зарегистрированных в период набора пользователей в тест;

Найдем идентификаторы участников теста recommender\_system\_test из датасета ab\_participants:

```
In [16]: eu = ab_participants.query('ab_test == "recommender_system_test"')['user_id']
```

Посчитаем соотношение количества участников теста ко всем пользователям из целевого региона, зарегистрированных в период набора:

```
In [17]: print('Итого получено следующее соотношение:',
      int(round(100 * int(users.query('user_id in @eu').pivot_table(index='region', values='device', aggfunc='count').iloc[2]) /
           int(users.query('region == "EU"').pivot_table(index='region', values='device', aggfunc='count').iloc[0]), 2)),
      '% количества участников теста ко всем пользователям из целевого региона')
```

Итого получено следующее соотношение: 15 % количества участников теста ко всем пользователям из целевого региона

Полученное соотношение соответствует ТЗ.

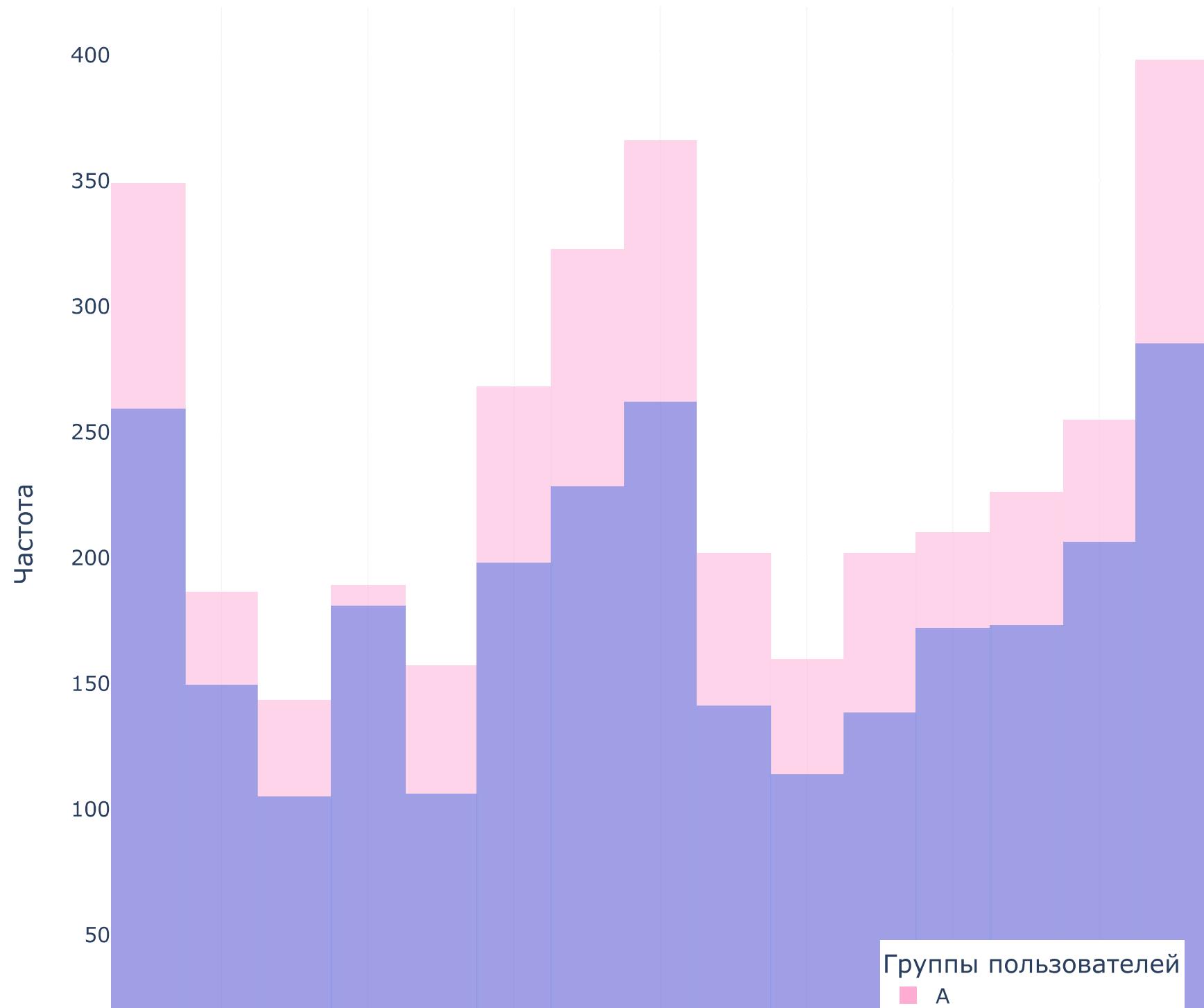
- оценить равномерность распределения пользователей по группам теста и корректность их формирования;

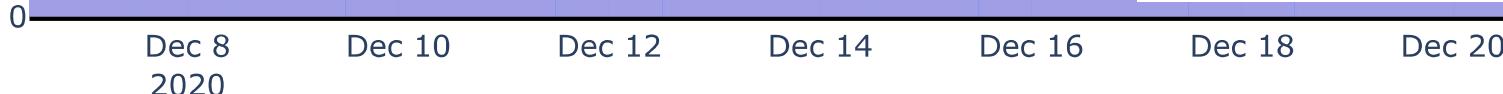
Оставим в датасете только пользователей из целевого региона:

```
In [18]: users = users.query('region == "EU"')
```

```
In [19]: (px.histogram(users.query('user_id in @eu').merge(ab_participants.query('ab_test == "recommender_system_test"'), on='user_id'),
                      x='first_date',
                      color='group',
                      color_discrete_map={'A': '#feacd2', 'B': 'royalblue'},
                      barmode="overlay")
          .update_layout(plot_bgcolor='rgba(0,0,0,0)',
                        autosize=False,
                        title={
                            'text': "Динамика набора пользователей в группы теста",
                            'y':0.98,
                            'x':0.5,
                            'xanchor': 'center',
                            'yanchor': 'top'
                        },
                        xaxis_title="Дата",
                        yaxis_title="Частота",
                        font=dict(
                            size=15),
                        width=950,
                        height=900,
                        legend=dict(
                            title='Группы пользователей',
                            yanchor="bottom",
                            y=0.01,
                            xanchor="left",
                            x=0.7
                        ))
          .update_xaxes(showline=True,
                        linewidth=2,
                        linecolor='black',
                        gridcolor='LightGrey',))
```

## Динамика набора пользователей в группы теста





Контрольная и тестовая группа представляют собой две выборки из генеральной совокупности и должны отражать ее основные характеристики. Судя по гистограмме, события распределены похожим образом по группам А и В, но в группе А регистраций происходит больше.

Сколько пользователей в каждой экспериментальной группе?

```
In [20]: group_one = (ab_participants.query('ab_test=="recommender_system_test"')
                  .pivot_table(index='group', values='user_id', aggfunc='count').reset_index())
group_one.columns = ['group', 'amount']
group_one['share'] = round(100 * group_one['amount'] / sum(group_one['amount']), 2)
display(group_one)
print('Всего пользователей в группе А и В - ', group_one['amount'].sum())
```

	group	amount	share
0	A	3824	57.07
1	B	2877	42.93

Всего пользователей в группе А и В - 6701

Видим, что пользователей из контрольной группы больше.

```
In [21]: print('Всего уникальных пользователей теста recommender_system_test - ',
           ab_participants.query('ab_test=="recommender_system_test"')['user_id'].nunique())
```

Всего уникальных пользователей теста recommender\_system\_test - 6701

Пересечений между группами А и В теста recommender\_system\_test нет.

Проверим, есть ли пересечение пользователей между конкурирующими тестами recommender\_system\_test и interface\_eu\_test:

```
In [22]: group_two = (ab_participants.query('ab_test=="interface_eu_test"')
                    .pivot_table(index='group', values='user_id', aggfunc='count').reset_index())
group_two.columns = ['group', 'amount']
group_two['share'] = round(100 * group_two['amount'] / sum(group_two['amount']), 2)
display(group_two)
print('Всего пользователей в группе А и В теста interface_eu_test - ', group_two['amount'].sum())
```

	group	amount	share
0	A	5831	50.41
1	B	5736	49.59

Всего пользователей в группе А и В теста interface\_eu\_test - 11567

```
In [23]: print('Всего уникальных пользователей теста recommender_system_test - ', ab_participants['user_id'].nunique())
```

Всего уникальных пользователей теста recommender\_system\_test - 16666

```
In [24]: if group_one['amount'].sum() + group_two['amount'].sum() > ab_participants['user_id'].nunique():
    print('Количество пользователей, которые попали в два теста одновременно - ',
          group_one['amount'].sum() + group_two['amount'].sum() - ab_participants['user_id'].nunique(),
          ', что составляет',
          round(100 * (group_one['amount'].sum() + group_two['amount'].sum() - ab_participants['user_id'].nunique()) /
                ab_participants['user_id'].nunique(), 2), '%')
```

Количество пользователей, которые попали в два теста одновременно - 1602 , что составляет 9.61 %

Таких пользователей достаточно много - 9.61%. Удалив их, може потерять в мощности теста. Контрольная группа отличается тем, что на нее не производится никакого воздействия/изменения в отличие от тестовой группы. Поэтому оценим влияние группы В конкурирующего теста на наши группы:

```
In [25]: # найдем отношение пересекающихся пользователей тестовых групп в двух тестах к общему количеству уникальных пользователей
# тестовой группы теста recommender_system_test
round((len(set(ab_participants.query('ab_test == "recommender_system_test" and group == "B"')['user_id']))
       .intersection(ab_participants.query('ab_test == "interface_eu_test" and group == "B"')['user_id'])) / 2877), 3)
```

Out[25]: 0.12

```
In [26]: # найдем отношение пересекающихся пользователей тестовой группы конкурирующего теста и контрольной группы нашего теста  
# к общему количеству уникальных пользователей контрольной группы теста recommender_system_test  
round((len(set(ab_participants.query('ab_test == "recommender_system_test" and group == "A")['user_id']))  
      .intersection(ab_participants.query('ab_test == "interface_eu_test" and group == "B")['user_id'])) / 3824), 3)
```

Out[26]: 0.115

Доли между собой практически равны, следовательно, конкурирующий тест практически одинаково влияет на пользователей двух групп нашего теста - значит, оставляем пересекающихся пользователей.

#### Вывод:

- Привели столбцы датасетов market\_events (столбцы start\_dt и finish\_dt), users (столбец first\_date) и ab\_events(event\_dt) к типу времени datetime
- явные и неявные дупликаты не обнаружены
- обнаружены пропуски (85,75%) в столбце details датасета ab\_events. Пропуски оставляем как есть, так как заполнение столбца details происходит только при оплате.
- в датасете users отфильтрованы значения по дате регистрации в соответствии с ТЗ - с 7 по 21 декабря 2020 г.
- в датасете ab\_events события совершаются с 7 по 30 декабря 2020, что не соответствует ТЗ. По ТЗ события новых пользователей совершались в период с 7 декабря 2020 по 4 января 2021. Отсутствуют данные за 5 дней. Причинами отсутствия могут быть неверная выгрузка данных, намеренная остановка теста ввиду набора достаточного количества данных.
- События распределены схожим образом по группам А и В, но в группе А регистраций происходит больше. Также пользователей больше в группе А.
- Пересечений между группами А и В теста recommender\_system\_test нет выявлено.
- Пересечения между группами двух тестов есть - принято решение сохранить пользователей.

#### Шаг 4. Изучение данных о пользовательской активности:

- активность пользователей: есть ли пользователи, которые не совершали событий после регистрации, изучите их количество и распределение между группами теста;

Перед исследованием объединим датасеты ab\_participants, users и ab\_events:

```
In [27]: new_data = (ab_participants.merge(users, how='left', on='user_id')
                           .query('ab_test == "recommender_system_test" and region == "EU"')
                           .merge(ab_events, how='left', on='user_id'))
new_data.head()
```

Out[27]:

	user_id	group	ab_test	first_date	region	device	event_dt	event_name	details
0	D1ABA3E2887B6A73	A	recommender_system_test	2020-12-07	EU	PC	2020-12-07 14:43:27	purchase	99.99
1	D1ABA3E2887B6A73	A	recommender_system_test	2020-12-07	EU	PC	2020-12-25 00:04:56	purchase	4.99
2	D1ABA3E2887B6A73	A	recommender_system_test	2020-12-07	EU	PC	2020-12-07 14:43:29	product_cart	NaN
3	D1ABA3E2887B6A73	A	recommender_system_test	2020-12-07	EU	PC	2020-12-25 00:04:57	product_cart	NaN
4	D1ABA3E2887B6A73	A	recommender_system_test	2020-12-07	EU	PC	2020-12-07 14:43:27	product_page	NaN

```
In [28]: # количество пользователей по тестам
(new_data.pivot_table(index='group', values='user_id', aggfunc='nunique')
 .reset_index())
```

Out[28]:

	group	user_id
0	A	3634
1	B	2717

```
In [29]: print('Всего пользователей, которые не совершали никаких действий после регистрации:',
           new_data.pivot_table(index='user_id', values='event_name', aggfunc='count').reset_index()
           .query('event_name == 0')['user_id'].nunique())
```

Всего пользователей, которые не совершали никаких действий после регистрации: 2870

Рассмотрим, как эти пользователи распределены по группам тестирования:

```
In [30]: (new_data.query('event_dt.isna() and event_name.isna()')
          .pivot_table(index='group', values='device', aggfunc='count')
          .reset_index())
```

Out[30]:

	group	device
0	A	1030
1	B	1840

```
In [31]: not_active_users = (new_data.pivot_table(index='group', values='user_id', aggfunc='nunique')
                           .reset_index().merge(new_data.query('event_dt.isna() and event_name.isna()')
                           .pivot_table(index='group', values='user_id', aggfunc='nunique')
                           .reset_index(), on='group'))
not_active_users.columns = ['group', 'all_users', 'not_active']
not_active_users['share'] = round(not_active_users['not_active'] / not_active_users['all_users'], 2)
not_active_users
```

Out[31]:

	group	all_users	not_active	share
0	A	3634	1030	0.28
1	B	2717	1840	0.68

Неактивные пользователи распределены неравномерно по датасету. Необходимо понять, почему так много пользователей попали в контрольную группу тестирования. При этом, так как пользователи не совершают никаких действий, нельзя оценить какие-либо изменения, поэтому удалим таких пользователей:

```
In [32]: new_data = new_data.query('~event_dt.isna() and ~event_name.isna()')
```

- проверить, что все участники теста имели возможность совершать события все 14 дней с момента регистрации, также оценить, когда пользователи совершают свои первые события каждого вида.

Добавим новый столбец - разницу в днях между датой совершения события и датой регистрации :

```
In [33]: new_data['age_event'] = (new_data['event_dt'].dt.date - new_data['first_date'].dt.date).dt.days
```

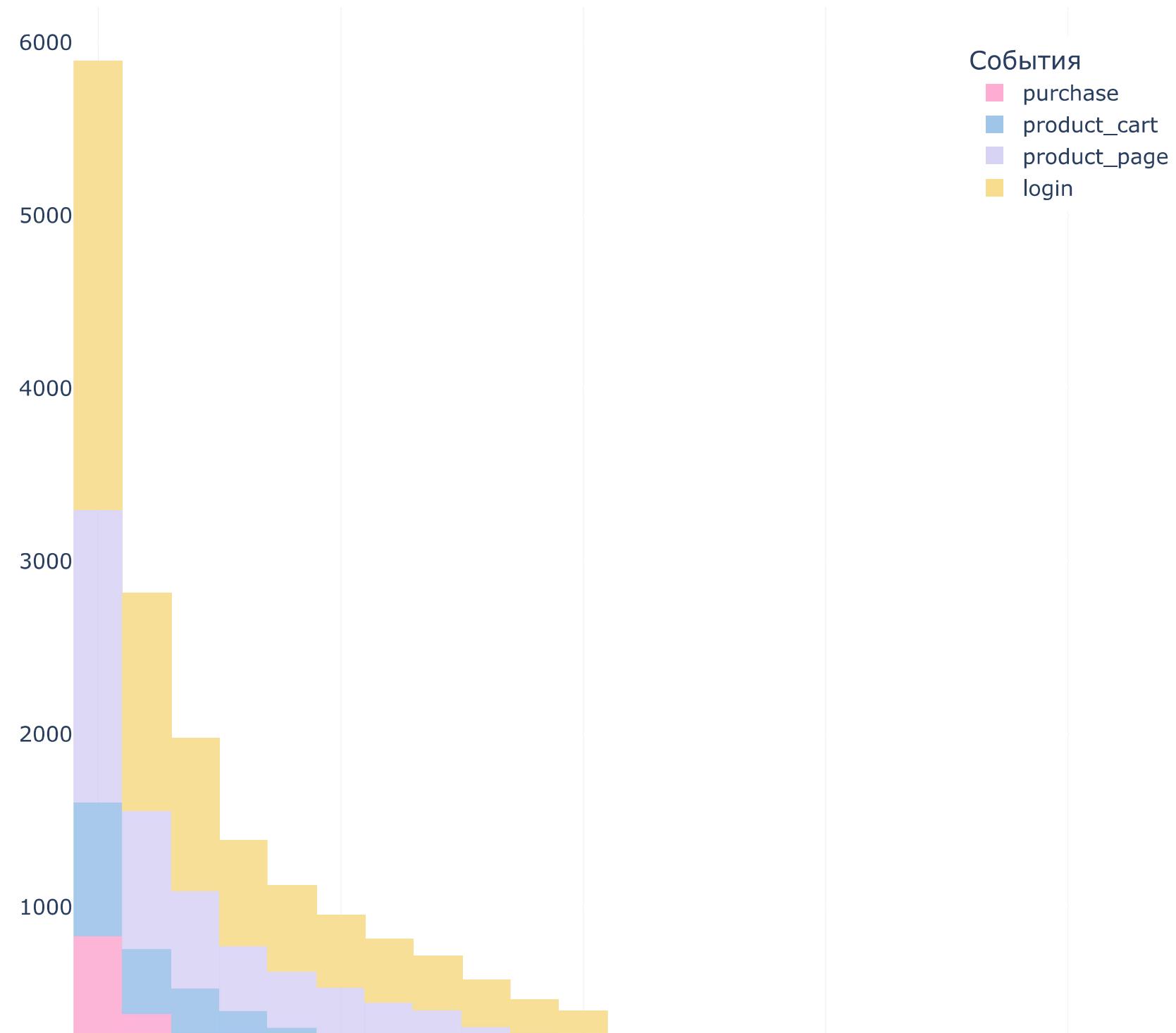
Оценим возраст событий для каждого действия, совершенного пользователем:

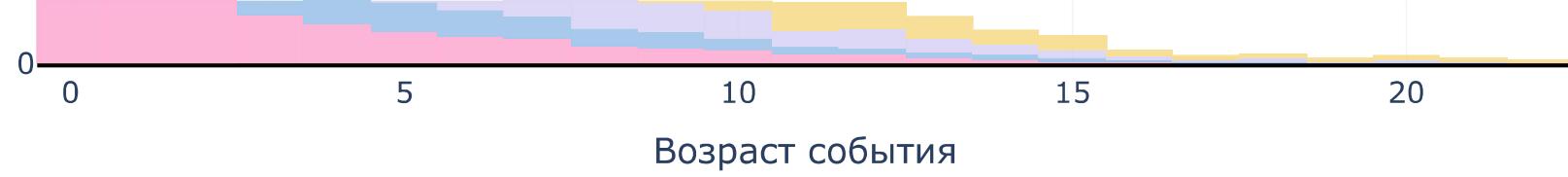
```
In [34]: print('Всего пользователей, не прожившие 14 дней -',
      new_data.query('age_event > 14')['user_id'].nunique())
```

Всего пользователей, не прожившие 14 дней - 255

```
In [35]: px.histogram(new_data.query('group == "A"'),
                     x='age_event', color='event_name',
                     color_discrete_map={'purchase': '#feacd2', 'product_cart': '#9fc5e8', 'product_page':'#d7d3f4', 'login':'#f9dd8e'},
                     opacity=0.9,
                     ) \
.update_layout(plot_bgcolor='rgba(0,0,0,0)',
               autosize=False,
               title={
                   'text': "Возраст событий в группе А",
                   'y':0.98,
                   'x':0.5,
                   'xanchor': 'center',
                   'yanchor': 'top'
               },
               xaxis_title="Возраст события",
               yaxis_title="",
               font=dict(
                   size=15),
               width=950,
               height=900,
               legend=dict(
                   title='События',
                   yanchor="bottom",
                   y=0.81,
                   xanchor="left",
                   x=0.8
               )) \
.update_xaxes(showline=True,
               linewidth=2,
               linecolor='black',
               gridcolor='LightGrey',)
```

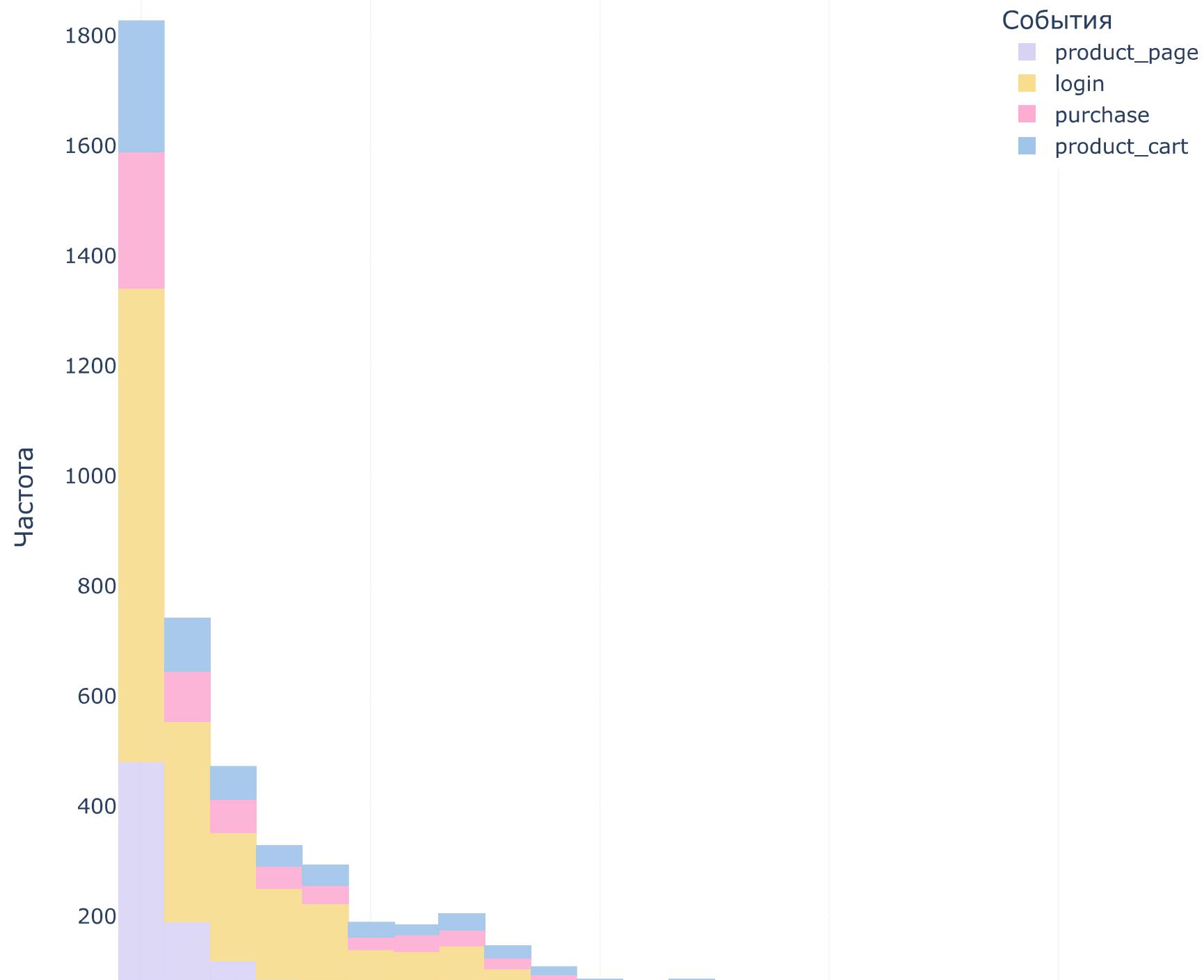
## Возраст событий в группе А

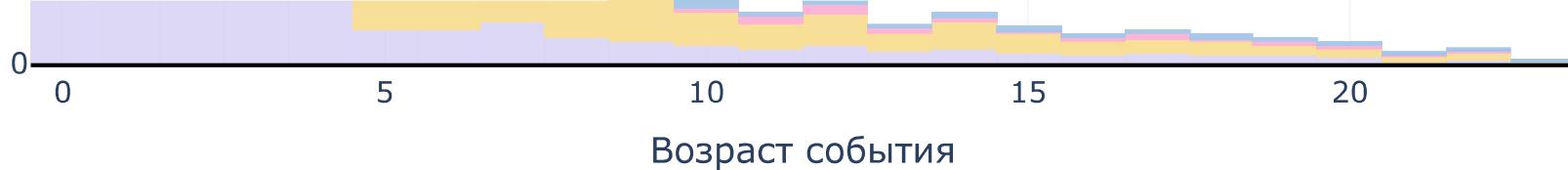




```
In [36]: px.histogram(new_data.query('group == "B"'),
                     x='age_event', color='event_name',
                     color_discrete_map={'purchase': '#feacd2', 'product_cart': '#9fc5e8', 'product_page':'#d7d3f4', 'login':'#f9dd8e'},
                     opacity=0.9,
                     ) \
.update_layout(plot_bgcolor='rgba(0,0,0,0)',
               autosize=False,
               title={
                   'text': "Возраст событий в группе В",
                   'y':0.98,
                   'x':0.5,
                   'xanchor': 'center',
                   'yanchor': 'top'
               },
               xaxis_title="Возраст события",
               yaxis_title="Частота",
               font=dict(
                   size=15),
               width=950,
               height=900,
               legend=dict(
                   title='События',
                   yanchor="bottom",
                   y=0.81,
                   xanchor="left",
                   x=0.8
               )) \
.update_xaxes(showline=True,
               linewidth=2,
               linecolor='black',
               gridcolor='LightGrey',)
```

## Возраст событий в группе В





Большая часть пользователей из группы А и В совершают события ранее 14 дней. Наиболее активно пользователи совершают события в первые 5 дней. Оставим тех пользователей, которые 14 дней не прожили.

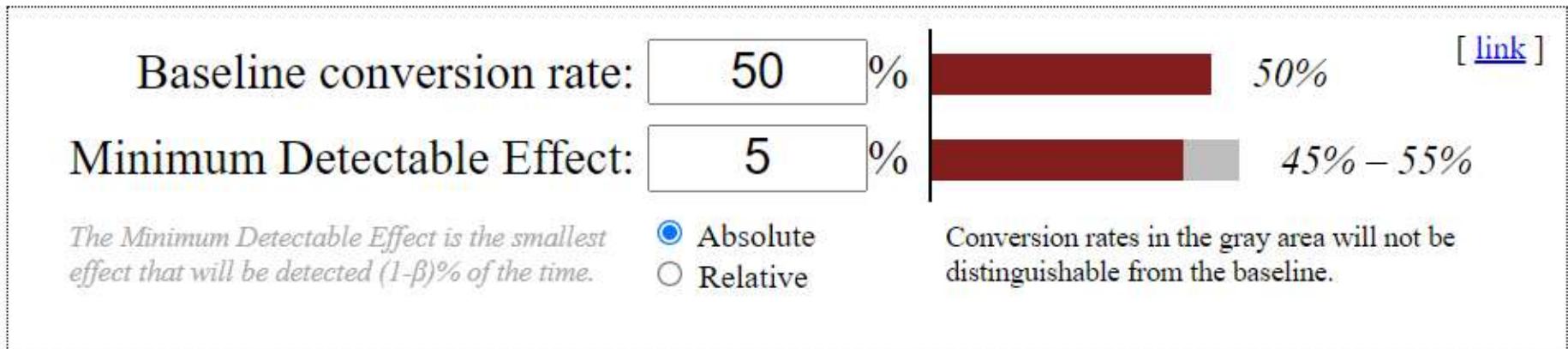
Отфильтруем датасет new\_data так, чтобы в него вошли только те события, которые были совершены в первые 14 дней с момента регистрации:

```
In [37]: new_data = new_data.query('age_event < 15')
```

- Проверим, какого размера выборки потребуются для получения достоверного результата, при базовой конверсии 50%

Для этого воспользуемся калькулятором, указав базовую конверсию в 50%:

*Question:* How many subjects are needed for an A/B test?



*Sample size:*

1,567

per variation

```
In [38]: print('Количество пользователей в группе А:', new_data.query('group == "A"')['user_id'].nunique())
print('Количество пользователей в группе В:', new_data.query('group == "B"')['user_id'].nunique())
```

Количество пользователей в группе А: 2604

Количество пользователей в группе В: 877

Значит, тестовая группа не соответствует требованиям. Возможно, при такой выборке получить достоверный результат не получится.

**Вывод:**

- Исключены неактивные пользователи из контрольной и тестовой группы. Пользователи распределены неравномерно по датасету. Необходимо понять, почему так много неактивных пользователей попали в контрольную группу тестирования.
- Сохранены пользователи, которые не прожили 14 дней ввиду того, что большая часть пользователей совершают события в пределах 14 дней.
- Исключены события, которые совершены позднее 14 дней.

## Шаг 5. Исследовательский анализ данных

- Распределение количества событий на пользователя в разрезе групп теста:

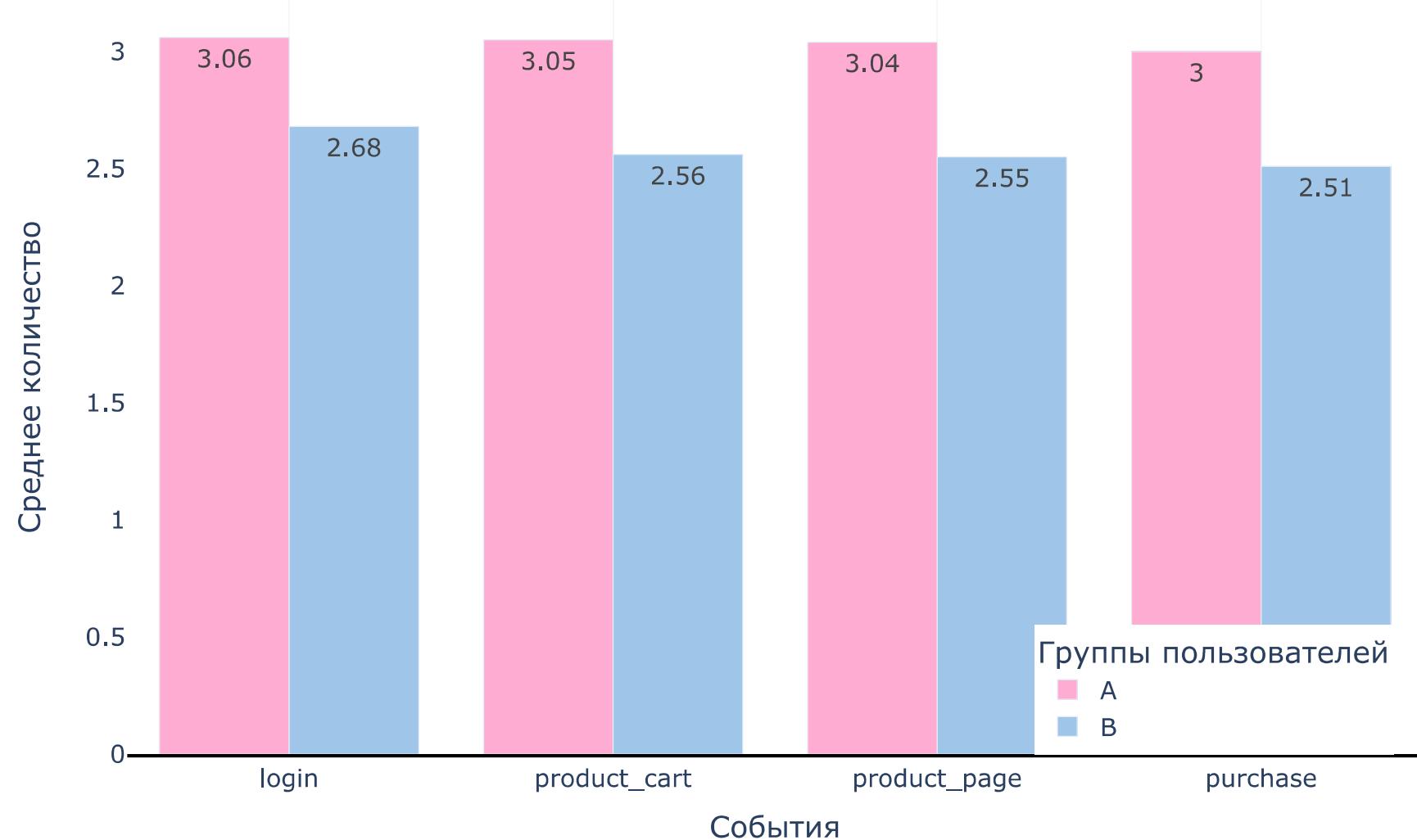
```
In [39]: events_users = new_data.groupby(['user_id', 'event_name', 'group'], as_index=False).agg({'device':'count'})
```

```
In [40]: events_users = events_users.groupby(['group', 'event_name'], as_index=False).agg({'device':'mean'})  
events_users.columns = ['group', 'event_name', 'mean_value']  
events_users['mean_value'] = round(events_users['mean_value'], 2)
```

In [41]:

```
(px.bar(events_users,
        x='event_name',
        y='mean_value',
        color='group',
        color_discrete_map={'A': '#feacd2', 'B': '#9fc5e8'},
        text_auto=True,
        barmode='group')
.update_layout(plot_bgcolor='rgba(0,0,0,0)')
.update_xaxes(showline=True, linewidth=2, linecolor='black', gridcolor='LightGrey')
.update_layout(plot_bgcolor='rgba(0,0,0,0)',
    autosize=False,
    width=950,
    height=600,
    title = {
        'text': "Среднее количество событий на пользователя",
        'y':0.98,
        'x':0.5,
        'xanchor': 'center',
        'yanchor': 'top'
    },
    legend=dict(
        title='Группы пользователей',
        yanchor="bottom",
        y=0,
        xanchor="left",
        x=0.7
    ),
    xaxis_title="События",
    yaxis_title="Среднее количество",
    font=dict(
        size=15),
))
```

## Среднее количество событий на пользователя



В группе А в среднем на пользователя приходится около 3 событий, в группе В данный показатель чуть ниже - около 2.5-2.6 события.

- Распределение по устройствам:

```
In [42]: device_a = new_data.query('group == "A"').pivot_table(index='device', values='user_id', aggfunc='nunique').reset_index()
device_a['share'] = round(device_a['user_id'] / device_a['user_id'].sum(), 2)

device_b = new_data.query('group == "B"').pivot_table(index='device', values='user_id', aggfunc='nunique').reset_index()
device_b['share'] = round(device_b['user_id'] / device_b['user_id'].sum(), 2)

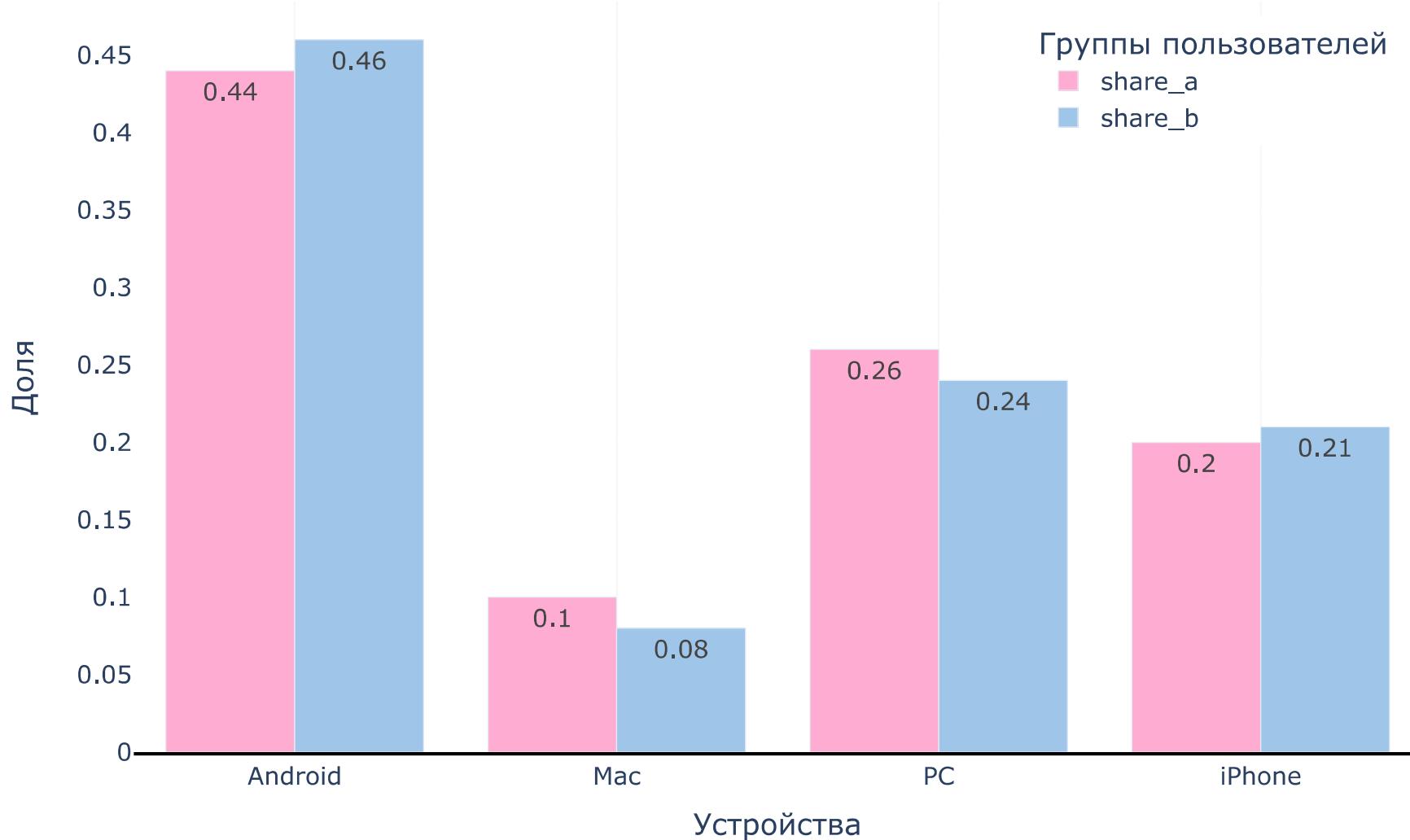
devices = device_a.merge(device_b, on='device').rename(columns={'user_id_x': 'users_a',
                                                               'share_x': 'share_a',
                                                               'user_id_y': 'users_b',
                                                               'share_y': 'share_b'})
devices
```

Out[42]:

	device	users_a	share_a	users_b	share_b
0	Android	1139	0.44	405	0.46
1	Mac	255	0.10	74	0.08
2	PC	689	0.26	212	0.24
3	iPhone	521	0.20	186	0.21

```
In [43]: px.bar(devices,
    x='device',
    y=['share_a', 'share_b'],
    #color='group',
    color_discrete_map={'share_a': '#feacd2', 'share_b': '#9fc5e8'},
    text_auto=True,
    barmode='group') \
    .update_layout(plot_bgcolor='rgba(0,0,0,0)') \
    .update_xaxes(showline=True, linewidth=2, linecolor='black', gridcolor='LightGrey') \
    .update_layout(plot_bgcolor='rgba(0,0,0,0)',
    autosize=False,
    width=950,
    height=600,
    title = {
        'text': "Распределение устройств по группам",
        'y':0.98,
        'x':0.5,
        'xanchor': 'center',
        'yanchor': 'top'
    },
    legend=dict(
        title='Группы пользователей',
        yanchor="bottom",
        y=0.81,
        xanchor="left",
        x=0.7
    ),
    xaxis_title="Устройства",
    yaxis_title="Доля",
    font=dict(
        size=15),
    )
```

## Распределение устройств по группам

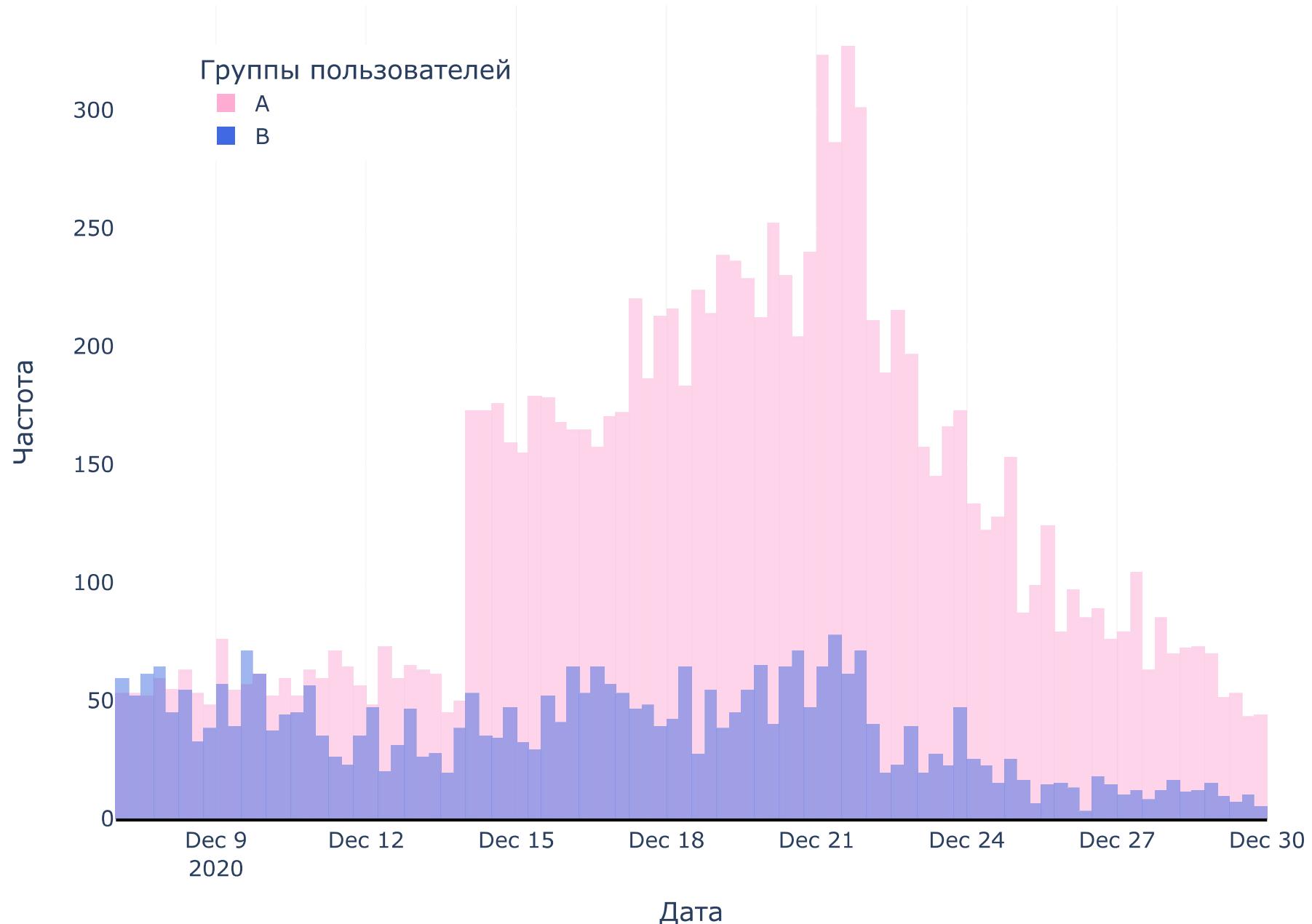


Почти у половины пользователей группы А и В (44% и 46%) - Android. Типом устройства Mac пользуются не так активно - всего около 10% в обеих группах. У iPhone 20% и 21% в группе А и В соответственно. В дальнейшем можно проанализировать, почему пользователей с Android в два раза больше пользователей iPhone, связано ли это с интерфейсом приложения, наличием багов и т.д.

- Динамика количества событий в группах теста по дням: изучим распределение числа событий по дням и сравним динамику групп теста между собой.

```
In [44]: px.histogram(new_data.pivot_table(index=['group', 'event_dt'], values='user_id', aggfunc='count').reset_index(),
                     x='event_dt', color='group',
                     color_discrete_map={'A': '#feacd2', 'B': 'royalblue'},
                     barmode="overlay") \
.update_layout(plot_bgcolor='rgba(0,0,0,0)',
               autosize=False,
               title={
                   'text': "Распределение числа событий по дням",
                   'y':0.98,
                   'x':0.5,
                   'xanchor': 'center',
                   'yanchor': 'top'
               },
               xaxis_title="Дата",
               yaxis_title="Частота",
               font=dict(
                   size=15),
               width=950,
               height=700,
               legend=dict(
                   title='Группы пользователей',
                   yanchor="bottom",
                   y=0.81,
                   xanchor="left",
                   x=0.07
               )) \
.update_xaxes(showline=True,
               linewidth=2,
               linecolor='black',
               gridcolor='LightGrey',)
```

# Распределение числа событий по дням



На графике видно, как повлияло удаление неактивных пользователей: в группе В событий много меньше по сравнению с группой А. Резкий подъем количества событий у группы А наблюдается с 14 декабря - можно предположить, что на это могли повлиять новогодние/рождественские праздники - люди совершают больше действий, покупок и т.д. До 21 декабря у групп с каждым днем все больше событий, после этого количество событий

снижается.

- Убедимся, что время проведения теста не совпадает с маркетинговыми и другими активностями.

Оставим в датасете с маркетинговыми событиями market\_events только те события, которые относятся к целевому региону:

```
In [45]: list = []

count = 0
for i in market_events['regions']:
    if 'EU' in i:
        list.append(market_events.iloc[count])
    count += 1
market_events_new = pd.DataFrame(list)
market_events_new
```

Out[45]:

	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
5	Black Friday Ads Campaign	EU, CIS, APAC, N.America	2020-11-26	2020-12-01
7	Labor day (May 1st) Ads Campaign	EU, CIS, APAC	2020-05-01	2020-05-03
8	International Women's Day Promo	EU, CIS, APAC	2020-03-08	2020-03-10

Зададим два условия - начало маркетингового события меньше или равно дате максимального события датасета new\_data (30 декабря) и конец маркетингового события больше или равно дате минимального события датасета new\_data (7 декабря):

```
In [46]: a = market_events_new['start_dt'] <= max(new_data['event_dt'])
b = market_events_new['finish_dt'] >= min(new_data['event_dt'])
a_b = (a & b).reset_index()
a_b.columns = ['index', 'bool']
```

```
In [47]: count = 0
for i in a_b['bool']:
    if i:
        print('Маркетинговые события, пересекающиеся с датой проведения теста: ',
              market_events_new.iloc[count]['name'])
    count += 1
```

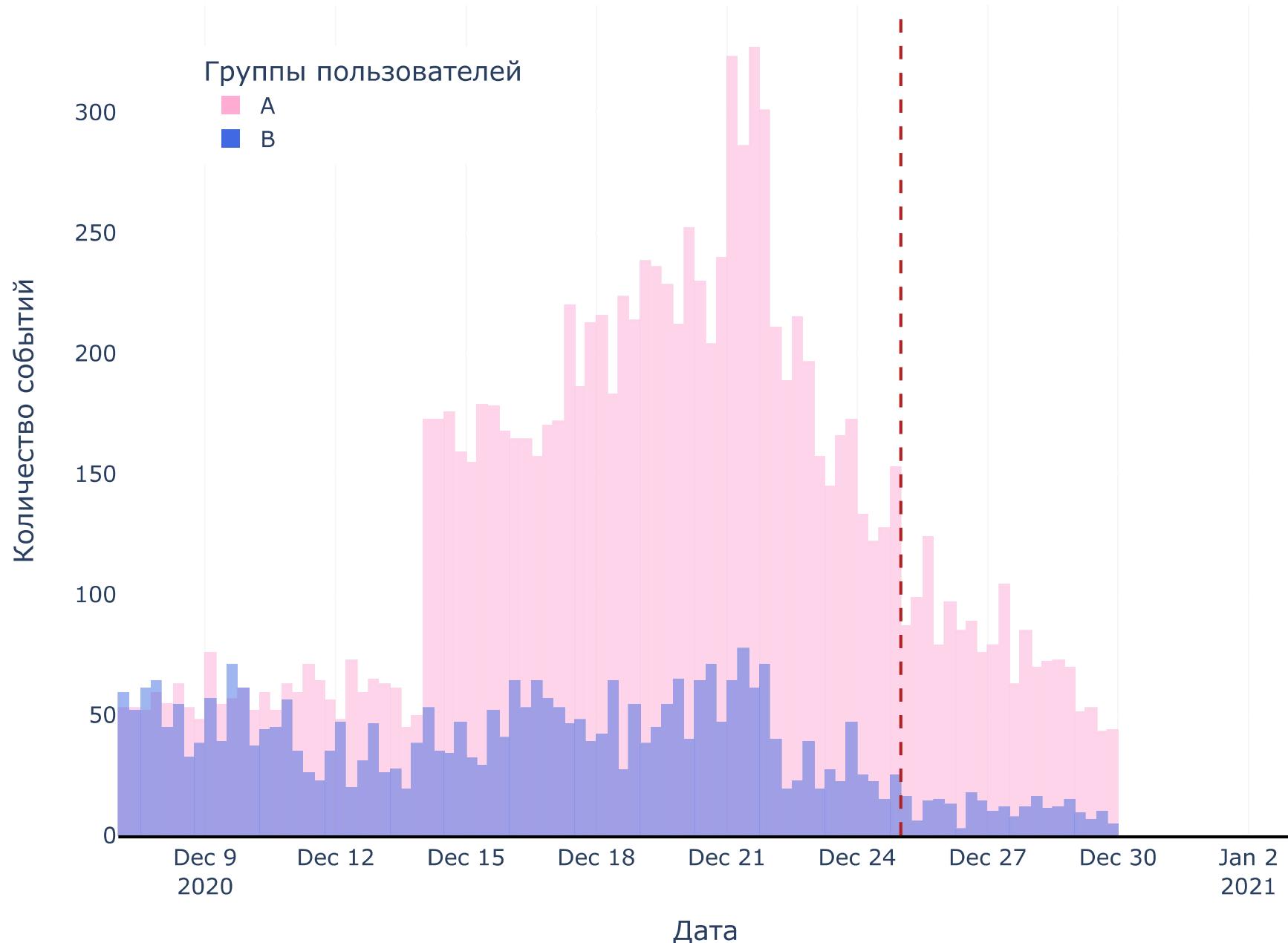
Маркетинговые события, пересекающиеся с датой проведения теста: Christmas&New Year Promo

В момент проведения теста попадает одно маркетинговое событие: Christmas&New Year Promo

```
In [48]: fig = px.histogram(new_data.pivot_table(index=['group', 'event_dt'], values='user_id', aggfunc='count').reset_index(),
                         x='event_dt', color='group',
                         color_discrete_map={'A': '#feacd2', 'B': 'royalblue'},
                         barmode="overlay")
fig.update_layout(plot_bgcolor='rgba(0,0,0,0)',
                  autosize=False,
                  title={
                      'text': "Распределение числа событий и даты маркетингового события",
                      'y':0.98,
                      'x':0.5,
                      'xanchor': 'center',
                      'yanchor': 'top'
                  },
                  xaxis_title="Дата",
                  yaxis_title="Количество событий",
                  font=dict(
                      size=15),
                  width=950,
                  height=700,
                  legend=dict(
                      title='Группы пользователей',
                      yanchor="bottom",
                      y=0.81,
                      xanchor="left",
                      x=0.07
                  ))
fig.update_xaxes(showline=True,
                  linewidth=2,
                  linecolor='black',
                  gridcolor='LightGrey')

# добавим даты начала маркетингового события Christmas&New Year Promo
fig.add_vline(x=market_events_new.query('name == "Christmas&New Year Promo"')['start_dt'].iloc[0],
              line_dash = 'dash',
              line_color = 'firebrick')
fig.add_vline(x=market_events_new.query('name == "Christmas&New Year Promo"')['finish_dt'].iloc[0],
              line_dash = 'dash',
              line_color = 'firebrick')
```

# Распределение числа событий и даты маркетингового события



В период проведения маркетингового события Christmas&New Year Promo на графике не замечено сильное изменение активности пользователей. Событие началось после предрождественской подготовки, пик активности пользователей пришелся на 21 декабря и после этого активность снижалась.

- Построим простые продуктовые воронки для двух групп теста с учетом логической последовательности совершения событий

Логическая цепочка действий такова: login -> product\_page -> product\_cart -> purchase.

В разрезе уникальных пользователей построим воронки:

```
In [49]: funnel_a = (new_data.query('group == "A"').pivot_table(index='event_name', values='user_id', aggfunc='nunique')
                  .sort_values(by='user_id', ascending=False).reset_index())

list = []

step = 1
for i in range(4):
    if i == 0:
        list.append(100 * funnel_a['user_id'][i] / funnel_a['user_id'][i])
    else:
        list.append(100 * funnel_a['user_id'][step] / funnel_a['user_id'][step - 1])
    step += 1
funnel_a['conv_step'] = list
funnel_a['conv_step'] = round(funnel_a['conv_step'], 2)
funnel_a = funnel_a.reindex([0, 1, 3, 2])
funnel_a
```

Out[49]:

	event_name	user_id	conv_step
0	login	2604	100.00
1	product_page	1685	64.71
3	product_cart	782	93.88
2	purchase	833	49.44

Отметим, что событий purchase больше событий product\_cart - 833 против 782.

```
In [50]: funnel_b = (new_data.query('group == "B"').pivot_table(index='event_name', values='user_id', aggfunc='nunique')
                     .sort_values(by='user_id', ascending=False).reset_index())

list = []

step = 1
for i in range(4):
    if i == 0:
        list.append(100 * funnel_b['user_id'][i] / funnel_b['user_id'][i])
    else:
        list.append(100 * funnel_b['user_id'][step] / funnel_b['user_id'][step - 1])
    step += 1
funnel_b['conv_step'] = list
funnel_b['conv_step'] = round(funnel_b['conv_step'], 2)
funnel_b = funnel_b.reindex([0, 1, 3, 2])
funnel_b
```

Out[50]:

	event_name	user_id	conv_step
0	login	876	100.00
1	product_page	493	56.28
3	product_cart	244	97.99
2	purchase	249	50.51

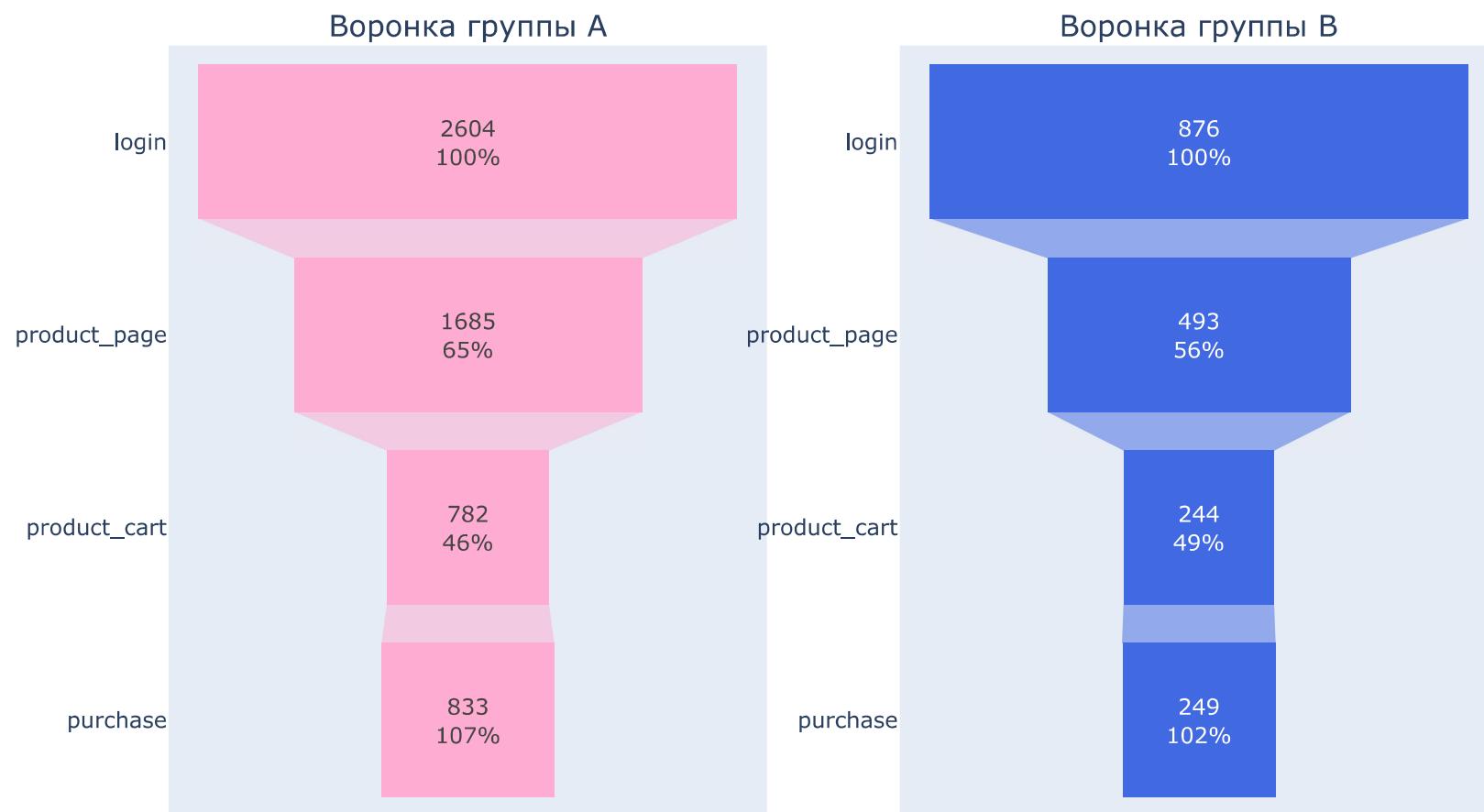
Также отметим, что событий purchase больше событий product\_cart - 249 против 244.

```
In [51]: fig = make_subplots(rows=1, cols=2, subplot_titles=("Воронка группы А", "Воронка группы В"))

fig.add_trace(
    go.Funnel(
        y = funnel_a['event_name'],
        x = funnel_a['user_id'],
        textposition = "inside",
        hoverinfo = "percent total",
        textinfo = "value+percent previous",
        marker = {"color": "#feacd2"}
    ),
    row=1, col=1
)

fig.add_trace(
    go.Funnel(
        y = funnel_b['event_name'],
        x = funnel_b['user_id'],
        textposition = "inside",
        hoverinfo = "percent total",
        textinfo = "value+percent previous",
        marker = {"color": "royalblue"}
    ),
    row=1, col=2
)

fig.update_layout(showlegend=False, height=600, width=900)
fig.show()
```



Судя по всему, существует так называется "быстрая покупка", когда пользователь может приобрести товар, не пользуясь корзиной.

Конверсия переходов к событию `product_page` и `purchase` у группы В снизилась по сравнению с группой А. Конверсия в событие `product_cart` возрасла с 46 по 49%.

Ожидаемый эффект увеличения конверсии в группе В на 10 процентных пунктов, относительно конверсии в группе А не наблюдается ни у какого события.

## **Вывод:**

- В группе А в среднем на пользователя приходится около 3 событий, в группе В данный показатель чуть ниже - около 2.5-2.6 события.
- Почти у половины пользователей группы А и В (44% и 46%) - Android. Типом устройства Mac пользуются не так активно - всего около 10% в обеих группах. У iPhone 20% и 21% в группе А и В соответственно. В дальнейшем можно проанализировать, почему пользователей с Android в два раза больше пользователей iPhone, связано ли это с интерфейсом приложения, наличием багов и т.д.
- В группе В событий много меньше по сравнению с группой А. Резкий подъем количества событий у группы А наблюдается с 14 декабря - можно предположить, что на это могли повлиять новогодние/рождественские праздники - люди совершают больше действий, покупок и т.д. До 21 декабря у групп с каждым днем все больше событий, после этого количество событий снижается.
- В момент проведения теста попадает одно маркетинговое событие: Christmas&New Year Promo. В период проведения маркетингового события Christmas&New Year Promo на графике не замечено сильное изменение активности пользователей. Сильного эффекта данное маркетинговое событие не оказало.
- Судя по всему, существует так называемая "быстрая покупка", когда пользователь может приобрести товар, не пользуясь корзиной. Конверсия переходов к событию product\_page и purchase у группы В снизилась по сравнению с группой А. Конверсия в событие product\_cart возросла с 46 по 49%. Ожидаемый эффект увеличения конверсии в группе В на 10 процентных пунктов, относительно конверсии в группе А не наблюдается ни у какого события.

## **Шаг 6. Оценка результатов A/B-тестирования:**

- Проверим статистическую разницу долей z-критерием.

Запишем функцию для проведения статистического теста:

```
In [52]: def stat_analysis(group_one, group_two, group_one_all, group_two_all):
    alpha = 0.05 / 3 # критический уровень статистической значимости у четом поправки Бонферони

    successes = np.array([group_one, group_two])
    trials = np.array([group_one_all, group_two_all])

    # пропорция в первой группе:
    p1 = successes[0] / trials[0]

    # пропорция во второй группе:
    p2 = successes[1] / trials[1]

    # пропорция в комбинированном датасете:
    p_combined = (successes[0] + successes[1]) / (trials[0] + trials[1])

    # разница пропорций в датасетах
    difference = p1 - p2

    z_value = difference / mth.sqrt(p_combined * (1 - p_combined) * (1 / trials[0] + 1 / trials[1]))

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

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

    print('p-значение: ', p_value)

    if p_value < alpha:
        print('Отвергаем нулевую гипотезу: между долями есть значимая разница')
    else:
        print(
            'Не получилось отвергнуть нулевую гипотезу, нет оснований считать доли разными'
        )
```

- Оценим статистическую разницу между группами перехода login -> product\_page. Постановка гипотез:
  - Нулевая гипотеза: статистически значимой разницы между конверсиями в просмотры страницы товара у двух групп нет
  - Альтернативная гипотеза: статистически значимая разница между конверсиями в просмотры страницы товара есть

```
In [53]: group_one = funnel_a.query('event_name == "product_page"')['user_id']
group_two = funnel_b.query('event_name == "product_page"')['user_id']
group_one_all = funnel_a.query('event_name == "login"')['user_id']
group_two_all = funnel_b.query('event_name == "login"')['user_id']

stat_analysis(group_one, group_two, group_one_all, group_two_all)
```

р-значение: [8.195976e-06]

Отвергаем нулевую гипотезу: между долями есть значимая разница

- Оценим статистическую разницу между группами перехода product\_page -> product\_cart. Постановка гипотез:
  - Нулевая гипотеза: статистически значимой разницы между конверсиями перехода к корзине у двух групп нет
  - Альтернативная гипотеза: статистически значимая разница между конверсиями перехода к корзине товара есть

```
In [54]: group_one = funnel_a.query('event_name == "product_cart"')['user_id']
group_two = funnel_b.query('event_name == "product_cart"')['user_id']
group_one_all = funnel_a.query('event_name == "product_page"')['user_id']
group_two_all = funnel_b.query('event_name == "product_page"')['user_id']

stat_analysis(group_one, group_two, group_one_all, group_two_all)
```

р-значение: [0.2276722]

Не получилось отвергнуть нулевую гипотезу, нет оснований считать доли разными

- Оценим статистическую разницу между группами перехода purchase -> product\_cart. Постановка гипотез:
  - Нулевая гипотеза: статистически значимой разницы между конверсиями к покупке у двух групп нет
  - Альтернативная гипотеза: статистически значимая разница между конверсиями к покупке товара есть

```
In [55]: group_one = funnel_a.query('event_name == "product_cart"')['user_id']
group_two = funnel_b.query('event_name == "product_cart"')['user_id']
group_one_all = funnel_a.query('event_name == "purchase"')['user_id']
group_two_all = funnel_b.query('event_name == "purchase"')['user_id']

stat_analysis(group_one, group_two, group_one_all, group_two_all)
```

р-значение: [0.01012766]

Отвергаем нулевую гипотезу: между долями есть значимая разница

**Вывод:**

- Гипотеза 1. Постановка гипотезы:

- Нулевая гипотеза: статистически значимой разницы между конверсиями в просмотры страницы товара у двух групп нет
- Альтернативная гипотеза: статистически значимая разница между конверсиями в просмотры страницы товара есть
- Гипотеза 2. Постановка гипотезы:
  - Нулевая гипотеза: статистически значимой разницы между конверсиями перехода к корзине у двух групп нет
  - Альтернативная гипотеза: статистически значимая разница между конверсиями перехода к корзине товара есть
- Гипотеза 3. Постановка гипотезы:
  - Нулевая гипотеза: статистически значимой разницы между конверсиями к покупке у двух групп нет
  - Альтернативная гипотеза: статистически значимая разница между конверсиями к покупке товара есть

В результате удалось выявить зависимости:

- Гипотеза 1 не подтверждена: Отвергаем нулевую гипотезу, между долями есть значимая разница.
- Гипотеза 2 подтверждена: Не получилось отвергнуть нулевую гипотезу, нет оснований считать доли разными.
- Гипотеза 3 не подтверждена: Отвергаем нулевую гипотезу, между долями есть значимая разница.

## **Шаг 6. Общий вывод:**

Был проведен анализ, чтобы проверить данные на соответствие ТЗ, провести исследовательский анализ, а также проанализировать результаты теста после введения улучшенной рекомендательной системы.

Перед анализом проведена предобработка данных :

- Привели столбцы датасетов market\_events (столбцы start\_dt и finish\_dt), users (столбец first\_date) и ab\_events(event\_dt) к типу времени datetime
- явные и неявные дубликаты не обнаружены
- обнаружены пропуски (85,75%) в столбце details датасета ab\_events. Пропуски оставляем как есть, так как заполнение столбца details происходит только при оплате.
- в датасете users отфильтрованы значения по дате регистрации в соответствии с ТЗ - с 7 по 21 декабря 2020 г.
- в датасете ab\_events события совершаются с 7 по 30 декабря 2020, что не соответствует ТЗ. По ТЗ события новых пользователей совершались в период с 7 декабря 2020 по 4 января 2021. Отсутствуют данные за 5 дней. Причинами отсутствия могут быть неверная выгрузка данных, намеренная остановка теста ввиду набора достаточного количества данных.
- События распределены схожим образом по группам А и В, но в группе А регистраций происходит больше. Также пользователей больше в группе А.
- Пересечений между группами А и В теста recommender\_system\_test нет выявлено.
- Пересечения между группами двух тестов есть - принято решение сохранить пользователей.
- Исключены неактивные пользователи из контрольной и тестовой группы. Пользователи распределены неравномерно по датасету. Необходимо понять, почему так много неактивных пользователей попали в контрольную группу тестирования.
- Сохранены пользователи, которые не прожили 14 дней ввиду того, что большая часть пользователей совершают события в пределах 14 дней.
- Исключены события, которые совершены позднее 14 дней.
- Тестовая группа не соответствует требованиям по размерам выборки. При такой выборке получить достоверный результат не получится.

В ходе исследования выявлены следующие закономерности:

- В группе А в среднем на пользователя приходится около 3 событий, в группе В данный показатель чуть ниже - около 2.5-2.6 события.
- Почти у половины пользователей группы А и В (44% и 46%) - Android. Типом устройства Mac пользуются не так активно - всего около 10% в обеих группах. У iPhone 20% и 21% в группе А и В соответственно.
- В группе В событий много меньше по сравнению с группой А. Резкий подъем количества событий у группы А наблюдается с 14 декабря - можно предположить, что на это могли повлиять новогодние/рождественские праздники - люди совершают больше действий, покупок и т.д. До 21 декабря у групп с каждым днем все больше событий, после этого количество событий снижается.
- В момент проведения теста попадает одно маркетинговое событие: Christmas&New Year Promo. В период проведения маркетингового события Christmas&New Year Promo на графике не замечено сильное изменение активности пользователей. Сильного эффекта данное маркетинговое событие не оказалось.
- Судя по всему, существует так называется "быстрая покупка", когда пользователь может приобрести товар, не пользуясь корзиной. Конверсия переходов к событию product\_page и purchase у группы В снизилась по сравнению с группой А. Конверсия в событие product\_cart возросла с 46 по 49%. Ожидаемый эффект увеличения конверсии в группе В на 10 процентных пункта, относительно конверсии в группе А не наблюдается ни у какого события.

Перед проведением исследования были поставлены несколько гипотез:

- Гипотеза 1. Постановка гипотезы:
  - Нулевая гипотеза: статистически значимой разницы между конверсиями в просмотры страницы товара у двух групп нет
  - Альтернативная гипотеза: статистически значимая разница между конверсиями в просмотры страницы товара есть
- Гипотеза 2. Постановка гипотезы:
  - Нулевая гипотеза: статистически значимой разницы между конверсиями перехода к корзине у двух групп нет
  - Альтернативная гипотеза: статистически значимая разница между конверсиями перехода к корзине товара есть
- Гипотеза 3. Постановка гипотезы:
  - Нулевая гипотеза: статистически значимой разницы между конверсиями к покупке у двух групп нет
  - Альтернативная гипотеза: статистически значимая разница между конверсиями к покупке товара есть

В результате удалось выявить зависимости:

- Гипотеза 1 не подтверждена: Отвергаем нулевую гипотезу, между долями есть значимая разница.
- Гипотеза 2 подтверждена: Не получилось отвергнуть нулевую гипотезу, нет оснований считать доли разными.
- Гипотеза 3 не подтверждена: Отвергаем нулевую гипотезу, между долями есть значимая разница.

## Рекомендации

- Необходимо сформировать достаточный размер выборки в тестовой группе для достоверного проведения тестирования, при условии, что пользователи тестовой группы совершают события.
- По возможности, проводить A/B-тестирование в периодах, не пересекающихся с праздниками, маркетинговыми событиями, чтобы исключить их возможное влияние на активность пользователей.
- Провести повторное A/B-тестирование, учитывая вышеуказанные рекомендации.

Также в дальнейшем можно проанализировать, почему пользователей с Android в два раза больше пользователей iPhone, связано ли это с интерфейсом приложения, наличием багов и т.д.