Оценка результатов A/B-теста по тестированию изменений, связанных с внедрением улучшенной рекомендательной системы

Оглавление

Описание данных и задачи

[к Оглавлению](#0.0)

Таблица **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.

Таблица **final_ab_new_users.csv** — пользователи, зарегистрировавшиеся 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.

Таблица **final_ab_events.csv** — действия новых пользователей в период с 7 декабря 2020 по 4 января 2021 года. Структура файла:

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

Таблица **final_ab_participants.csv** — таблица участников тестов. Структура файла:

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

Задача

Провести оценку результатов A/B-теста:

- Оценить, корректность проведения теста
- Проанализировать результаты теста

Чтобы оценить корректность проведения теста, проверить:

- пересечение тестовой аудитории с конкурирующим тестом,
- совпадение теста и маркетинговых событий, другие проблемы временных границ теста.

Техническое задание

[к Оглавлению](#0.0)

- 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 пользователи покажут улучшение каждой метрики не менее, чем на 10%:
  - конверсии в просмотр карточек товаров — событие `product_page` ,

- просмотры корзины — `product_cart` ,
- покупки — `purchase` .

1. Загрузка данных и подготовка их к анализу

[к Оглавлению](#0.0)

Импортируем библиотеки

In [1]:

```
import matplotlib.pyplot as plt
import pandas as pd
import seaborn as sns
from IPython.display import display
import numpy as np

import plotly.express as px
from plotly import graph_objects as go
import math as mth
from scipy import stats as st
```

In [2]:

```
ab_project_marketing_events = pd.read_csv('ab_project_marketing_events.cs
v')
final_ab_new_users = pd.read_csv('final_ab_new_users.csv')
final_ab_events = pd.read_csv('final_ab_events.csv')
final_ab_participants = pd.read_csv('final_ab_participants.csv')
```

In [3]:

```
# ab_project_marketing_events = pd.read_csv('/datasets/ab_project_marketin
g_events.csv')
# final_ab_new_users = pd.read_csv('/datasets/final_ab_new_users.csv')
# final_ab_events = pd.read_csv('/datasets/final_ab_events.csv')
# final_ab_participants = pd.read_csv('/datasets/final_ab_participants.cs
v')
```

In [4]:

```
for df in [ab_project_marketing_events,
           final_ab_new_users,
           final_ab_events, final_ab_participants]:
    display(df.head(), df.info(), df.describe())
    print('-----------------------------------')
```

```
<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
```

```
 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
```

	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

None

	name	regions	start_dt	finish_dt
count	14	14	14	14
unique	14	6	14	14
top	Single's Day Gift Promo	APAC	2020-03-17	2020-07-01
freq	1	4	1	1

```
----------------------------------
<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
```

	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

None

	user_id	first_date	region	device
count	61733	61733	61733	61733
unique	61733	17	4	4
top	EEA09A67995EDE53	2020-12-21	EU	Android
freq	1	6290	46270	27520

```
----------------------------------
<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
```

	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

```
None
```

	details
count	62740.000000
mean	23.877631
std	72.180465
min	4.990000
25%	4.990000
50%	4.990000
75%	9.990000
max	499.990000

```
----------------------------------
<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
```

	user_id	group	ab_test
0	D1ABA3E2887B6A73	A	recommender_system_test
1	A7A3664BD6242119	A	recommender_system_test
2	DABC14FDDFADD29E	A	recommender_system_test
3	04988C5DF189632F	A	recommender_system_test

4	482F14783456D21B	B	recommender_system_test

None

	user_id	group	ab_test
count	18268	18268	18268
unique	16666	2	2
top	E7F87F9FBA3269B1	A	interface_eu_test
freq	2	9655	11567

1.1. Преобразование типов

[к Оглавлению](#0.0)

Изменим столбцы с датами на тип `datetime`

Применим функцию для переименования столбцов

In [5]:

```
final_ab_new_users.columns = ['user_id', 'first_dt', 'region', 'device']
```

In [6]:

```
def date_to(dataframes):
    for df in dataframes:
        for column in df.columns:
            if 'dt' in column:
                df[column] = pd.to_datetime(df[column], format='%Y-%m-%d')
date_to([ab_project_marketing_events, final_ab_new_users, final_ab_events
])
```

Проверим результат

In [7]:

```
for df in [ab_project_marketing_events,
           final_ab_new_users,
           final_ab_events]:
    display(df.head(), df.info())
    print('---------------------------------')
```

```
<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
```

	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

```
None

---------------------------------
<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_dt  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
```

	user_id	first_dt	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

```
None

---------------------------------
<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
```

	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

1	7B04521 06H 49504	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

```
None

---------------------------------
```

1.2. Пропущенные значения и дубликаты

[к Оглавлению](#0.0)

In [8]:

```
for df in [ab_project_marketing_events,
           final_ab_new_users,
           final_ab_events,
          final_ab_participants]:
    display(df.isnull().sum(), df.duplicated().sum())
    print('---------------------------------')
```

```
name          0
regions       0
start_dt      0
finish_dt     0
dtype: int64

0

---------------------------------

user_id      0
first_dt     0
region       0
device       0
dtype: int64

0

---------------------------------

user_id            0
event_dt           0
event_name         0
details       377577
dtype: int64

0

---------------------------------

user_id    0
group      0
ab_test    0
dtype: int64

0
```

```
----------------------------------
```

In [9]:

```
display(final_ab_events['event_name'].value_counts(), final_ab_events['details'].value_counts())
```

```
login           189552
product_page    125563
purchase         62740
product_cart     62462
Name: event_name, dtype: int64

4.99      46362
9.99       9530
99.99      5631
499.99     1217
Name: details, dtype: int64
```

Дубликатов не обнаружено.
Пропуски есть в таблице о действиях новых пользователей в столбце `details`, в котором отображены дополнительные данные для покупок `purchase` в долларах. Для других событий нет стоимости покупки и поэтому стоит Nan.

2. Исследовательский анализ данных (EDA)

[к Оглавлению](#0.0)

Приведем наши данные к техническому заданию

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 пользователи покажут улучшение каждой метрики не менее, чем на 10%:
  - конверсии в просмотр карточек товаров — событие `product_page` ,
  - просмотры корзины — `product_cart` ,
  - покупки — `purchase` .

Возьмем только нужный нам тест

In [10]:

```
final_system_test = final_ab_participants.query('ab_test == "recommender_system_test"')
display(final_system_test.info(), final_system_test['ab_test'].value_counts())
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 6701 entries, 0 to 6700
Data columns (total 3 columns):
 #   Column   Non-Null Count  Dtype 
---  ------   --------------  ----- 
 0   user_id  6701 non-null   object
 1   group    6701 non-null   object
 2   ab_test  6701 non-null   object
dtypes: object(3)
memory usage: 209.4+ KB

None

recommender_system_test    6701
Name: ab_test, dtype: int64
```

Посмотрим на одинаковых пользователей в разных тестах.

In [11]:

```
users = final_ab_participants.groupby(['user_id', 'group']).agg({'user_id': ['count']}).reset_index()
users.columns = ['user_id', 'group', 'count']
users = users.query('count > 1').sort_values(by='user_id')
users
```

Out[11]:

	user_id	group	count
2	001064FEAAB631A1	B	2
10	00341D8401F0F665	A	2
12	003B6786B4FF5B03	A	2
51	00EFA157F7B6E1C4	A	2
57	010DB4614355A4BB	B	2
...	...	...	...
17396	FF2174A1AA0EAD20	A	2
17403	FF44696E39039D29	A	2
17412	FF7BE2897FC0380D	B	2
17418	FF9A81323FA67D6E	B	2
17437	FFED90241D04503F	B	2

826 rows × 3 columns

Выберем первого пользователя и посмотрим в каких тестах от участвовал

In [12]:

```
final_ab_participants.query('user_id == "001064FEAAB631A1"')
```

Out[12]:

	user_id	group	ab_test
235	001064FEAAB631A1	B	recommender_system_test
17892	001064FEAAB631A1	B	interface_eu_test

In [13]:

```
print('Количество пользователей, которые участвовали в двух тестах {} или {:.1%}'
      .format(users['user_id'].count(), users['user_id'].count()/final_ab_participants['user_id'].count()))
```

```
Количество пользователей, которые участвовали в двух тестах 8
26 или 4.5%
```

Выберем тех пользователей, которые зарегистрировались в нужный нам промежуток времени.

In [14]:

```
final_ab_new_users['first_dt'].value_counts().sort_index()
```

Out[14]:

```
2020-12-07    5595
2020-12-08    3239
2020-12-09    2101
2020-12-10    3076
2020-12-11    2390
2020-12-12    3963
2020-12-13    4691
2020-12-14    5654
2020-12-15    3043
2020-12-16    2110
2020-12-17    3048
2020-12-18    3365
2020-12-19    3617
2020-12-20    4288
2020-12-21    6290
2020-12-22    3083
2020-12-23    2180
Name: first_dt, dtype: int64
```

In [15]:

```
final_ab_users = final_ab_new_users.query('first_dt <= "2020-12-21"')
final_ab_users['first_dt'].value_counts().sort_index();
```

Объединим таблицу участников тестирования и зарегистрировавшихся пользователей

In [16]:

```
final_ab_test = final_system_test.merge(final_ab_events, on = 'user_id', h
ow='left')
display(final_ab_test.head(), final_ab_test.info());
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 27724 entries, 0 to 27723
Data columns (total 6 columns):
 #   Column      Non-Null Count  Dtype         
---  ------      --------------  -----         
 0   user_id     27724 non-null  object        
 1   group       27724 non-null  object        
 2   ab_test     27724 non-null  object        
 3   event_dt    24698 non-null  datetime64[ns]
 4   event_name  24698 non-null  object        
 5   details     3331 non-null   float64       
dtypes: datetime64[ns](1), float64(1), object(4)
memory usage: 1.5+ MB
```

	user_id	group	ab_test	event_dt	event_name	det
0	D1ABA3E2887B6A73	A	recommender_system_test	2020-12-07 14:43:27	purchase	9
1	D1ABA3E2887B6A73	A	recommender_system_test	2020-12-25 00:04:56	purchase	
2	D1ABA3E2887B6A73	A	recommender_system_test	2020-12-07 14:43:29	product_cart	
3	D1ABA3E2887B6A73	A	recommender_system_test	2020-12-25 00:04:57	product_cart	
4	D1ABA3E2887B6A73	A	recommender_system_test	2020-12-07 14:43:27	product_page	

```
None
```

Проверим на пропуски

In [17]:

```
final_ab_test.isnull().sum()
```

Out[17]:

```
user_id           0
group             0
ab_test           0
event_dt       3026
event_name     3026
details       24393
dtype: int64
```

Пропуски появились у тип события и даты покупки. Удалим не нужные строки

In [18]:

```
final_ab_test = final_ab_test.dropna(subset=['event_name'])
final_ab_test.isnull().sum()
```

Out[18]:

```
user_id           0
group             0
ab_test           0
event_dt          0
event_name        0
details       21367
dtype: int64
```

Добавим данные из таблицы с пользователями, которые зарегистрировались в нужное нам время.

In [19]:

```
final_ab_test = final_ab_test.merge(final_ab_users, on = 'user_id', how='l
eft')
final_ab_test.head()
```

Out[19]:

	user_id	group	ab_test	event_dt	event_name	det
0	D1ABA3E2887B6A73	A	recommender_system_test	2020-12-07 14:43:27	purchase	9
1	D1ABA3E2887B6A73	A	recommender_system_test	2020-12-25 00:04:56	purchase	
2	D1ABA3E2887B6A73	A	recommender_system_test	2020-12-07 14:43:29	product_cart	
3	D1ABA3E2887B6A73	A	recommender_system_test	2020-12-25 00:04:57	product_cart	
4	D1ABA3E2887B6A73	A	recommender_system_test	2020-12-07 14:43:27	product_page	

In [20]:

```
final_ab_test.isnull().sum()
```

Out[20]:

```
user_id           0
group             0
ab_test           0
event_dt          0
event_name        0
details       21367
first_dt          0
region            0
device            0
```

```
device                 0
dtype: int64
```

In [21]:

```
print('После предобратотки в таблице осталось {} записей или {:.1%} от изн
ачального количества'
      .format(final_ab_test['user_id'].count(),
              (final_ab_test['user_id'].count())/final_ab_events['user_id'
].count()))
```

```
После предобратотки в таблице осталось 24698 записей или 5.6%
от изначального количества
```

In [22]:

```
print('В таблице осталось {} уникальных пользователя, {:.1%} от ожидаемого
количества в 6000 человек'
      .format(final_ab_test['user_id'].nunique(), (final_ab_test['user_id'
].nunique())/6000))
```

```
В таблице осталось 3675 уникальных пользователя, 61.3% от ожи
даемого количества в 6000 человек
```

Посмотрим, сколько уникальных пользователей входит в каждую группу.

In [23]:

```
ab_user = final_ab_test.groupby('group')['user_id'].agg(['count']).reset_i
ndex()
ab_user['procent'] = ((ab_user['count']/ab_user['count'].sum())*100).round
(1)
ab_user.sort_values(by='procent', ascending=False)
ab_user
```

Out[23]:

	group	count	procent
0	A	19304	78.2
1	B	5394	21.8

В группе А в 3.5 раза больше пользователей чем в группе В.

Тест не возможен.

- Группы не равны. Не равномерное количество пользователей в выборках исказят результаты тестирования и мы можем завершить тест слишком рано, как только расчеты показали, что есть статистическая разница между откликами или обратное: продолжить тестирование, когда нужный размер выборки уже набран, а ожидаемой разницы в откликах нет.
- После обработки ушло 94 % действий пользователей.
- Ожидаемого количества человек не достигли.
- Есть пользователи, которые участвовали в обоих тестах, их доля составляет 4.5%

2.1. Изменение конверсии в воронке на разных этапах

[к Оглавлению](#0.0)

Посмотрим на количество событий и уникальных пользователей, которые есть в таблице и изменение конверсии на разных этапах.

In [24]:

```
event = final_ab_test.groupby('event_name').agg({'event_name': ['count']})
.reset_index()
event.columns = ('event_name', "count")
event = event.sort_values(by='count', ascending=False)
event
```

Out[24]:

	event_name	count
0	login	11190
2	product_page	6930
3	purchase	3331
1	product_cart	3247

In [25]:

```
nun_user = final_ab_test.groupby('event_name')['user_id'].agg(['nunique'])
.reset_index()
nun_user;
```

События должны происходить в следующем порядке:

Login - пользователь входит на сайт;
Product_page - предложение о товаре (экран с товаром);
Product_card - переход в корзину;
Purshase - экран успешной оплаты заказа.
Этапы `Product_card` и `Purshase` имеют разное количество заходов пользователей, причем покупок больще чем переходов в корзину. Следовательно покупки совершаются по одному товару миную корзину. Надо расставить события в нужный порядок.

Так же посмотрим на конверсию пользователей на этапах воронки

In [26]:

```
event_user = event.merge(nun_user, on = 'event_name', how='left')
event_user.columns = ['event_name', 'count', 'n_users']
event_user['procent'] = (event_user['n_users'] / final_ab_test['user_id'].
nunique()*100).round(1)
event_user['step'] = pd.Series([0, 1, 3, 2])
event_user = event_user.sort_values(by='step', ascending=True)
event_user
```

Out[26]:

	event_name	count	n_users	procent	step
0	login	11190	3675	100.0	0
1	product_page	6930	2303	62.7	1
3	product_cart	3247	1079	29.4	2
2	purchase	3331	1128	30.7	3

In [27]:

```
fig = go.Figure(go.Funnel(
    y = event_user['event_name'],
    x = event_user['n_users'],
    textinfo = "value+percent initial"))
fig.update_layout(title_text='Воронка событий', yaxis_title="События")
fig.show()
```

- При переходе с первого этапа на второй теряется 37.3% пользователей. Переходит 62.7% пользователей.
- При переходе со второго этапа на третий теряется 53.1% пользователей. Переходит 46.9% пользователей.
- При переходе с третьего на четвертый этап увеличивается количество на 4.5% пользователей. Дошло до оплаты 51.4% пользователей.

Разделим на группы A и B

Посмотрим как меняестя конверсия в воронке на разных этапах для двух групп.

In [28]:

```
group_A = final_ab_test.query('group == "A"')
group_B = final_ab_test.query('group == "B"')
```

In [29]:

```
funnel_A = (group_A.groupby('event_name')['user_id'].nunique().sort_values
(ascending=False).to_frame().reset_index()
              .rename(columns={'user_id': 'total_users'}))
funnel_A['step'] = pd.Series([0, 1, 3, 2])
funnel_A = funnel_A.sort_values(by='step', ascending=True)


funnel_B = (group_B.groupby('event_name')['user_id'].nunique().sort_values
(ascending=False).to_frame().reset_index()
              .rename(columns={'user_id': 'total_users'}))
funnel_B['step'] = pd.Series([0, 1, 3, 2])
funnel_B = funnel_B.sort_values(by='step', ascending=True)
funnel_B
```

Out[29]:

	event_name	total_users	step
0	login	928	0
1	product_page	523	1
3	product_cart	255	2
2	purchase	256	3

In [30]:

```
fig = go.Figure()

fig.add_trace(go.Funnel(
    name = 'group_A',
    y = funnel_A['event_name'],
    x = funnel_A['total_users'],
    textinfo = "value+percent initial"))

fig.add_trace(go.Funnel(
    name = 'group_B',
    y = funnel_B['event_name'],
    x = funnel_B['total_users'],
    textinfo = "value+percent initial"))

fig.show()
```

Потери пользователей в группе B выше чем в группе A. В группе B совершили покупку на 4% меньше пользователей чем в A.

2.2. Распределение количества событий на пользователя в выборках

[к Оглавлению](#0.0)

In [31]:

```
print('Всего событий %d, типов событий %d' % (final_ab_test.shape[0], fina
l_ab_test['event_name'].nunique()))
```

```
Всего событий 24698, типов событий 4
```

In [32]:

```
print('Всего пользователей в логе %d' % (final_ab_test['user_id'].nunique
()))
```

```
Всего пользователей в логе 3675
```

In [33]:

```
print('В среднем на пользователя приходится %d событий' % (final_ab_test[
'user_id'].value_counts().mean()))
print('По медиане на пользователя приходится %d событий' % (final_ab_test[
'user_id'].value_counts().median()))
```

```
print('По моде на пользователя приходится %d событий' % (final_ab_test['us
er_id'].value_counts().mode()))
```

```
В среднем на пользователя приходится 6 событий
По медиане на пользователя приходится 6 событий
По моде на пользователя приходится 6 событий
```

2.3. В выборках встречаются одни и те же пользователи?

[к Оглавлению](#0.0)

In [34]:

```
users = final_system_test.groupby(['user_id', 'group']).agg({'user_id': [
'count']}).reset_index()
users.columns = ['user_id', 'group', 'count']
users = users.query('count > 1').sort_values(by='user_id')
users
```

Out[34]:

user_id	group	count

В группах A и B нашего теста нет одинаковых пользователей.

2.4. Распределение количества событий по дням

[к Оглавлению](#0.0)

Добавим в нашу таблицу столбцы с данными по дням и часам

In [35]:

```
final_ab_test['event_day'] = final_ab_test['event_dt'].astype('datetime64
[D]')
final_ab_test['event_hours'] = final_ab_test['event_dt'].astype('datetime6
4[h]')
```

In [36]:

```
final_ab_test['event_day'].value_counts()
```

Out[36]:

```
2020-12-21    2441
2020-12-20    1848
2020-12-19    1810
2020-12-18    1584
2020-12-17    1548
2020-12-22    1506
2020-12-16    1454
2020-12-14    1355
```

```
2020-12-15    1316
2020-12-23    1201
2020-12-24    1042
2020-12-25     789
2020-12-09     746
2020-12-26     733
2020-12-07     709
2020-12-27     691
2020-12-10     613
2020-12-28     603
2020-12-08     593
2020-12-12     558
2020-12-11     542
2020-12-29     508
2020-12-13     504
2020-12-30       4
Name: event_day, dtype: int64
```

In [37]:

```
df_date = final_ab_test.groupby('event_day').agg({'event_day': ['count']})
.reset_index()
df_date.columns = ('date', "count")
df_date = df_date.sort_values(by='date', ascending=False)
fig = go.Figure(data=[go.Bar(x=df_date['date'], y=df_date['count'], text=d
f_date['count'],
                             textposition='auto')])
fig.update_layout(barmode='group', title_text='Гистограмма по числу событи
й в день',
                  xaxis_title="День", yaxis_title="Количество событий",)
fig.show()
```

In [38]:

```
df_date = final_ab_test.query('event_day == "2020-12-21"').groupby('event_
hours').agg({'event_hours': ['count']}).reset_index()
df_date.columns = ('date', "count")
df_date = df_date.sort_values(by='date', ascending=False)
fig = go.Figure(data=[go.Bar(x=df_date['date'], y=df_date['count'], text=d
f_date['count'],
                             textposition='auto')])
fig.update_layout(barmode='group', title_text='Гистограмма по числу событи
й в час',
                  xaxis_title="Час", yaxis_title="Количество событий",)
fig.show()
```

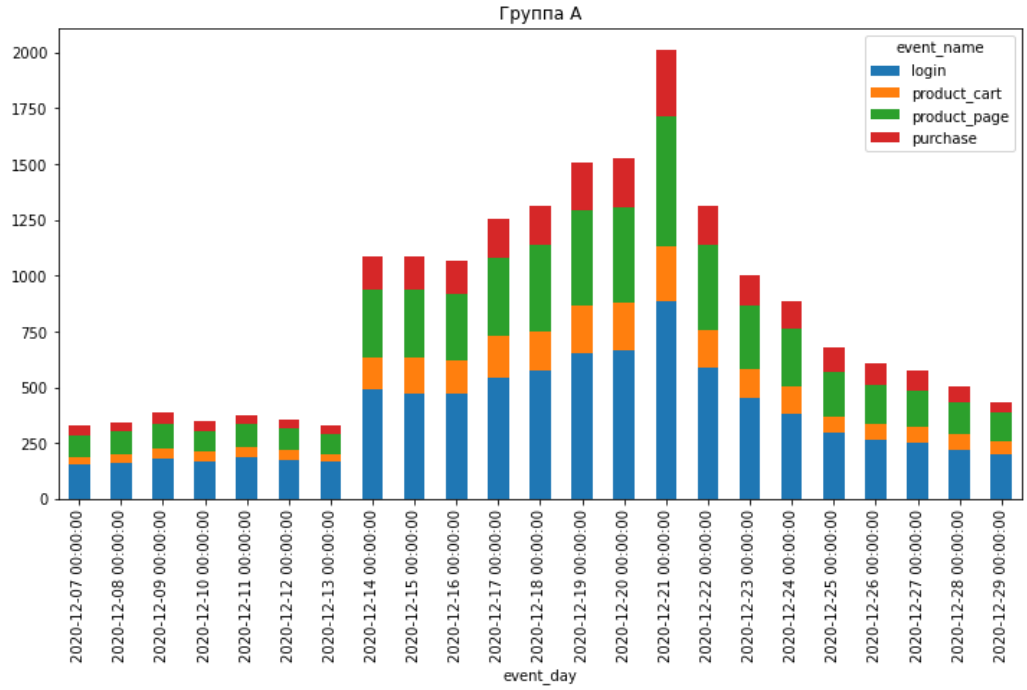
Обновим группа A и B добавив столбец день

In [39]:

```
group_A = final_ab_test.query('group == "A"')
group_B = final_ab_test.query('group == "B"')
```

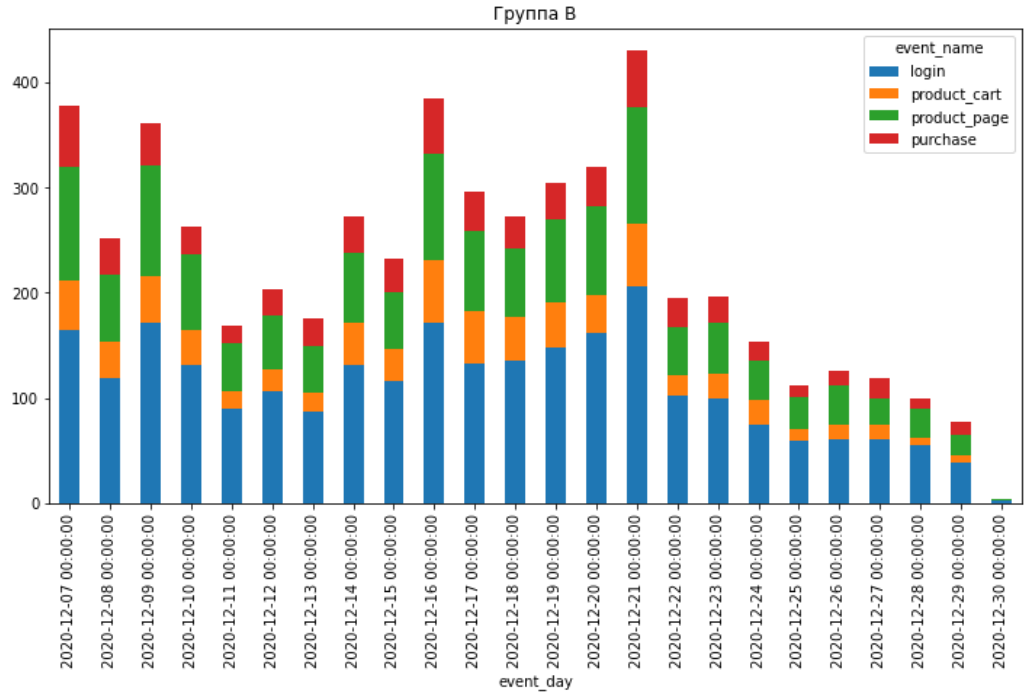
In [40]:

```
group_A.pivot_table(index='event_day', values='user_id',columns='event_nam
e',aggfunc='count').plot.bar(stacked=True,figsize=(12,6))
plt.title("Группа A")
plt.show()
```



In [41]:

```
group_B.pivot_table(index='event_day', values='user_id',columns='event_nam
e',aggfunc='count').plot.bar(stacked=True,figsize=(12,6))
plt.title("Группа B")
plt.show()
```



Пользователи разных групп ведут себя по разному. отличия как в днях активности так и в количестве событий за день.

В группе В количество событий в разы меньше чем в группе А

2.5. Особенность данных перед A/B-тестированием

[к Оглавлению](#0.0)

Перед тем как начать A/B-тест, нужно убедиться, что:

```
- на результаты не влияют аномалии и выбросы в генеральной совокупн
ости;
- инструмент «деления» трафика работает безошибочно;
- данные отправляются в системы аналитики корректно.
```

Перед A/B тестом проводят A/A-тест. Если трафик и инструмент проведения A/A-теста не подвели, различий в показателях не будет и поможет определить длительность теста и методику анализа данных.
Критерии успешного A/A-теста:

```
- Количество пользователей в различных группах различается не боле
е, чем на 1%;
- Для всех групп фиксируют и отправляют в системы аналитики данные
 об одном и том же;
- Различие ключевых метрик по группам не превышает 1% и не имеет ст
атистической значимости;
- Попавший в одну из групп посетитель остаётся в этой группе до кон
ца теста. Если пользователь видит разные версии исследуемой страниц
ы в ходе одного исследования, неизвестно, какая именно повлияла на
 его решения. Значит, и результаты такого теста нельзя интерпретиро
вать однозначно.
```

3. Оценка результатов A/B-тестирования

[к Оглавлению](#0.0)

3.1. Результаты A/B-тестирования

[к Оглавлению](#0.0)

Исходные таблицы содержат данные только о двух группах: А и В. Скрорей всего A/A-тестирование не было проведено либо проводилось некорректно. Считать результаты A/B тестирования верными нельзя.

3.2. Проверка статистической разницы долей z-критерием

[к Оглавлению](#0.0)

In [42]:

```
final_ab_test.groupby('group')['user_id'].nunique().to_frame()
```

Out[42]:

group	user_id
A	2747
B	928

In [43]:

```
user_event = final_ab_test.pivot_table(index = 'group', columns = 'event_name', values = 'user_id',
                                       aggfunc = 'nunique').reset_index()

user_event
```

Out[43]:

	event_name	group	login	product_cart	product_page	purchase
0		A	2747	824	1780	872
1		B	928	255	523	256

Уберем индекс у таблицы

In [44]:

```
user_events = user_event.set_index(user_event.columns[0])
user_events
```

Out[44]:

event_name	login	product_cart	product_page	purchase
group				
A	2747	824	1780	872
B	928	255	523	256

Проверим гипотезу о равенстве долей при помощи Z-критерия. Для этого напишем функцию.

Проведем сравнение двух групп A и B и убедимся, что тест был проведен корректно и между ними нет стат. значимых различий.

Нулевая гипотеза - между долями нет стат. значимой разницы.
Альтернативная гипотеза - между долями есть стат. значимая разница.

In [45]:

```
def st_test(p1_ev, p2_ev, p1_us, p2_us):
    p1 = p1_ev / p1_us
    p2 = p2_ev / p2_us
    p_combined = (p1_ev + p2_ev) / (p1_us + p2_us)

    difference = p1 - p2
    z_value = difference / mth.sqrt(p_combined * (1 - p_combined) * (1 / p
1_us + 1 / p2_us))
    distr = st.norm(0, 1)
    p_value = (1 - distr.cdf(abs(z_value))) * 2

    return p_value
```

In [46]:

```
p_value = st_test(2747, 928, user_events.loc['A'].sum(), user_events.loc[
'B'].sum())
print('p-значение: ', p_value)

alpha = .05
if (p_value < alpha):
    print("Принимаем альтернативную гипотезу: между долями есть значимая р
азница")
else:
    print("Оставляем нулевую гипотезу, нет оснований считать доли разными"
)
```

```
p-значение:  0.014261076220458024
Принимаем альтернативную гипотезу: между долями есть значимая
разница
```

Мы подтвердили, что между долями есть стат. значимая разница

4. Вывод по этапу иследовательского анализа и по проведеной оценке результатов A/B-тестирования

[к Оглавлению](#0.0)

По этапу иследовательского анализа и по проведеной оценке результатов A/B-тестирования выявлены следующие факты:

- Даты были в не правильном формате
- Дубликатов не обнаружено
- Пропуски есть в таблице о действиях новых пользователей в столбце `details`, в котором отображены дополнительные данные для покупок `purchase` в долларах. Для других событий нет стоимости покупки и поэтому стоит Nan
- Есть пользователи, которые участвовали в двух тестах 826 человек или 4.5%

- Есть пользователи, дата регистрации которых позже 21 декабря, что не соответсвует условию
- После обработки ушло 94 % действий пользователей
- Не было набрано ожидаемое количество пользователей
- Распределение не равномерное - количество юзеров в группе A превосходит группу B
- Не было проведено A/A-тестирование для выявления проблем с данными