Содержание

A/B-тестирование

Постановка задачи

Ваша задача — провести оценку результатов A/B-теста. В вашем распоряжении есть датасет с действиями пользователей, техническое задание и несколько вспомогательных датасетов.

- Оцените корректность проведения теста
- Проанализируйте результаты теста

Чтобы оценить корректность проведения теста, проверьте:

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

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` .

Описание данных

`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.

`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.

`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.

`final_ab_participants.csv` — таблица участников тестов.

Структура файла:

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

Как сделать задание?

- Опишите цели исследования
- Исследуйте данные:
  - Требуется ли преобразование типов?
  - Опишите природу пропущенных значений и дубликатов, если их обнаружите.
- Оцените корректность проведения теста. Обратите внимание на:
  - Соответствие данных требованиям технического задания. Проверьте корректность всех пунктов технического задания.
  - Время проведения теста. Убедитесь, что оно не совпадает с маркетинговыми и другими активностями.
  - Аудиторию теста. Удостоверьтесь, что нет пересечений с конкурирующим тестом и нет пользователей, участвующих в двух группах теста одновременно. Проверьте равномерность распределения по тестовым группам и правильность их формирования.
- Проведите исследовательский анализ данных:
  - Количество событий на пользователя одинаково распределены в выборках?
  - Как число событий в выборках распределено по дням?
  - Как меняется конверсия в воронке в выборках на разных этапах?
  - Какие особенности данных нужно учесть, прежде чем приступать к A/B-тестированию?
- Оцените результаты A/B-тестирования
  - Что можно сказать про результаты A/B-тестирования?
  - Проверьте статистическую разницу долей z-критерием.
- Опишите выводы по этапу исследовательского анализа данных и по проведённой оценке результатов A/B-тестирования. Сделайте общее заключение о корректности проведения теста.

1 Загрузка данных, изучение общей информации

1.1 Импорт библиотек

Импорт библиотек

Ввод [1]:

```
# импортируем библиотеки
import pandas as pd               # Библиотека Pandas
import matplotlib.pyplot as plt  # Библиотека для визуализации
import numpy as np                # Библиотека для математических вычислений
import seaborn as sns             # Библиотека для визуализации данных
import plotly.express as px       # Библиотека для визуализации данных
import datetime as dt             # Библиотека для преобразования к типу данных "дата"
import math as mth                # Библиотека для математических вычислений
import scipy.stats as st          # Библиотека для высокоуровневых математических вычислений
from plotly import graph_objects as go # Библиотека для интерактивной визуализации

                                  # Библиотека для геоаналитики
#from folium import Map, Marker, Choropleth  # импортируем карту и маркер
#from folium.plugins import MarkerCluster   # импортируем кластер
#import json                        # библиотека для работы с  json
#from geopy.distance import geodesic as GD  #библиотека для геодезических расчетов
#from sqlalchemy import create_engine

```

1.2 Считывание данных из исходного файла

Ввод [2]:

```
# считываем  данные и сохраняем в переменные

try:
    ab_project_marketing_events = pd.read_csv('ab_project_marketing_events.csv')
    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')

except:
    ab_project_marketing_events = pd.read_csv('https://code.s3.yandex.net/datasets/ab_project_marketing_events.csv')
    final_ab_new_users = pd.read_csv('https://code.s3.yandex.net/datasets/final_ab_new_users.csv')
    final_ab_events = pd.read_csv('https://code.s3.yandex.net/datasets/final_ab_events.csv')
    final_ab_participants = pd.read_csv('https://code.s3.yandex.net/datasets/final_ab_participants.csv')


```

1.3 Первичный обзор данных

Ввод [3]:

```
# перечень датасетов и их имен
dataset_list = {'Календарь маркетинговых событий на 2020 год' : ab_project_marketing_events ,
                'Пользователи, зарегистрировавшиеся с 7 по 21 декабря 2020 года' : final_ab_new_users ,
                'Действия новых пользователей в период с 7 Дек 2020 по 4 Янв 2021' : final_ab_events ,
                'Таблица участников тестов': final_ab_participants }
```

Ввод [4]:

```
for  name, ds in dataset_list.items():
    print()
    print('Название датасета:', name)
    print()
    ds.info()
    display(ds)
    print('*'*80)
```

```
Название датасета: Календарь маркетинговых событий на 2020 год

<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
```

	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
10	CIS New Year Gift Lottery	CIS	2020-12-30	2021-01-07
11	Dragon Boat Festival Giveaway	APAC	2020-06-25	2020-07-01
12	Single's Day Gift Promo	APAC	2020-11-11	2020-11-12
13	Chinese Moon Festival	APAC	2020-10-01	2020-10-07

```
********************************************************************************

Название датасета: Пользователи, зарегистрировавшиеся с 7 по 21 декабря 2020 года

<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
...	...	...	...	...
61728	1DB53B933257165D	2020-12-20	EU	Android
61729	538643EB4527ED03	2020-12-20	EU	Mac
61730	7ADEE837D5D8CBBD	2020-12-20	EU	PC
61731	1C7D23927835213F	2020-12-20	EU	iPhone
61732	8F04273BB2860229	2020-12-20	EU	Android

61733 rows × 4 columns

```
********************************************************************************

Название датасета: Действия новых пользователей в период с 7 Дек 2020 по 4 Янв 2021

<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
...	...	...	...	...
440312	245E85F65C358E08	2020-12-30 19:35:55	login	NaN
440313	9385A108F5A0A7A7	2020-12-30 10:54:15	login	NaN
440314	DB650B7559AC6EAC	2020-12-30 10:59:09	login	NaN
440315	F80C9BDDEA02E53C	2020-12-30 09:53:39	login	NaN
440316	7AEC61159B672CC5	2020-12-30 11:36:13	login	NaN

440317 rows × 4 columns

```
********************************************************************************

Название датасета: Таблица участников тестов

<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	04988C5DF189632E	A	recommender_system_test
4	482F14783456D21B	B	recommender_system_test
...	...	...	...
18263	1D302F8688B91781	B	interface_eu_test
18264	3DE51B726983B657	A	interface_eu_test
18265	F501F79D332BE86C	A	interface_eu_test
18266	63FBE257B05F2245	A	interface_eu_test
18267	79F9ABFB029CF724	B	interface_eu_test

18268 rows × 3 columns

```
********************************************************************************
```

2. Предобработка данных

2.1 Приведение данных к формату даты/времени

В датасетах необходимо выполнить преобразование форматов для данных даты/времени в соответствующих столбцах.

Ввод [5]:

```
ab_project_marketing_events['start_dt'] = pd.to_datetime(ab_project_marketing_events['start_dt'], format='%Y-%m-%d')
ab_project_marketing_events['finish_dt'] = pd.to_datetime(ab_project_marketing_events['finish_dt'], format='%Y-%m-%d')

final_ab_new_users['first_date'] = pd.to_datetime(final_ab_new_users['first_date'], format='%Y-%m-%d')

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

2.2 Обзор данных

Ввод [6]:

```
# Функция  обзора данных.

def meet_dataset (dataset):

    print()
    print('Общая информация о датасете')
    display(dataset.info())
    print()

    print('Общие статистические данные')
    display(dataset.describe())
    print('\n')

    try:
        print('Общие гистограммы для  столбцов датасета')
        dataset.hist (figsize=(15,10))
        plt.show()
        print('\n','\n')
    except:
        print()

    print ('Количество дубликатов -', dataset.duplicated().sum())
    print ('Доля дубликатов в датасете, %:', round( 100*dataset.duplicated().sum()/dataset.shape[0], 2))
    print()

    #print ('Количество пропусков -', dataset.isna().sum(),'\n','\n')

    #количество пропусков по столбцам
    data_gap= pd.DataFrame(dataset.isna().sum()).reset_index(drop=False)
    # обозначение имени столбца
    data_gap.columns=['Столбец','Количество пропусков (isna)']
    # относительное количество пропусков по столбцам в %
    data_gap['Количество пропусков (isna) в %'] = round(data_gap['Количество пропусков (isna)']/ len(dataset) * 100, 2)

    data_gap_2 = pd.DataFrame(dataset.isnull().sum()).reset_index(drop=False)
    data_gap_2.columns=['Столбец','Количество пропусков (isnull)']
    data_gap_2['Количество пропусков (isnull) в %'] = round(data_gap_2['Количество пропусков (isnull)']/ len(dataset) * 100, 2)

    data_gap =   data_gap.merge(data_gap_2, on = 'Столбец', how= 'left')


    #display (data_gap.style.background_gradient('coolwarm'))
    display (data_gap.style.background_gradient())

    print()

    #Приведение названий столбцов к нижнему регистру
    #dataset.columns = [x.lower().replace(' ', '_') for x in dataset.columns.values]

    print ('Датасет:')

    return display(dataset.head(20))
```

Обзор датасета

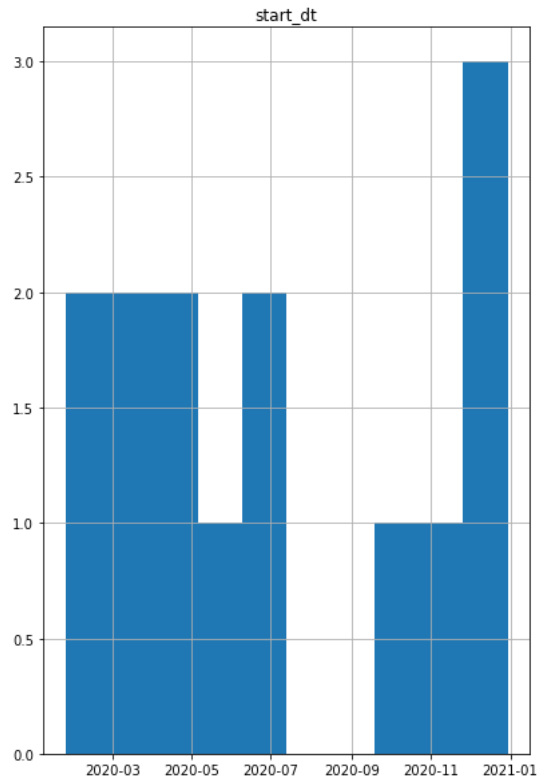
Ввод [7]:

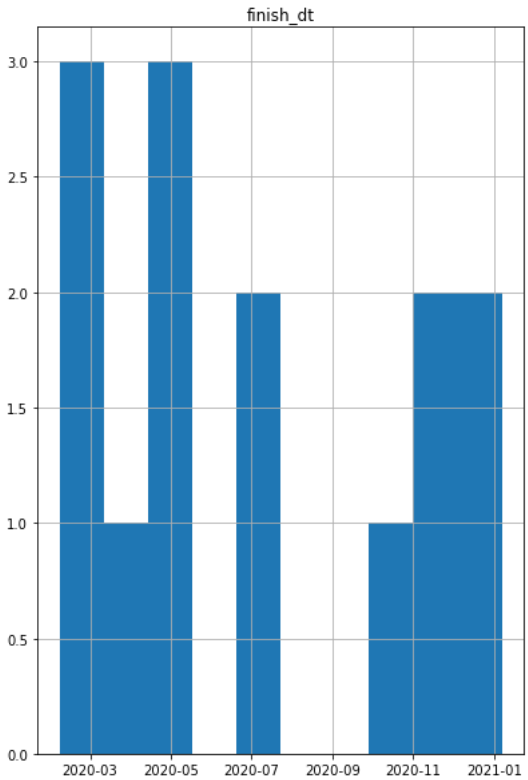
```
for  name, ds in dataset_list.items():
    print()
    print('Название датасета:', name)
    print()
    meet_dataset(ds)
    print('*'*80)

```

```
Название датасета: Календарь маркетинговых событий на 2020 год


Общая информация о датасете
<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


Общие статистические данные

C:\Users\МиО\AppData\Local\Temp\ipykernel_12548\2376842602.py:11: FutureWarning: Treating datetime data as categorical rath
er than numeric in `.describe` is deprecated and will be removed in a future version of pandas. Specify `datetime_is_numeri
c=True` to silence this warning and adopt the future behavior now.
  display(dataset.describe())
C:\Users\МиО\AppData\Local\Temp\ipykernel_12548\2376842602.py:11: FutureWarning: Treating datetime data as categorical rath
er than numeric in `.describe` is deprecated and will be removed in a future version of pandas. Specify `datetime_is_numeri
c=True` to silence this warning and adopt the future behavior now.
  display(dataset.describe())
```

	name	regions	start_dt	finish_dt
count	14	14	14	14
unique	14	6	14	14
top	Christmas&New Year Promo	APAC	2020-12-25 00:00:00	2021-01-03 00:00:00
freq	1	4	1	1
first	NaN	NaN	2020-01-25 00:00:00	2020-02-07 00:00:00
last	NaN	NaN	2020-12-30 00:00:00	2021-01-07 00:00:00

```
Общие гистограммы для  столбцов датасета
```





Количество дубликатов - 0
Доля дубликатов в датасете, %: 0.0

	Столбец	Количество пропусков (isna)	Количество пропусков (isna) в %	Количество пропусков (isnull)	Количество пропусков (isnull) в %
0	name	0	0.000000	0	0.000000
1	regions	0	0.000000	0	0.000000
2	start_dt	0	0.000000	0	0.000000
3	finish_dt	0	0.000000	0	0.000000

Датасет:

	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
10	CIS New Year Gift Lottery	CIS	2020-12-30	2021-01-07
11	Dragon Boat Festival Giveaway	APAC	2020-06-25	2020-07-01
12	Single's Day Gift Promo	APAC	2020-11-11	2020-11-12
13	Chinese Moon Festival	APAC	2020-10-01	2020-10-07

****

Название датасета: Пользователи, зарегистрировавшиеся с 7 по 21 декабря 2020 года

Общая информация о датасете
```
<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
```

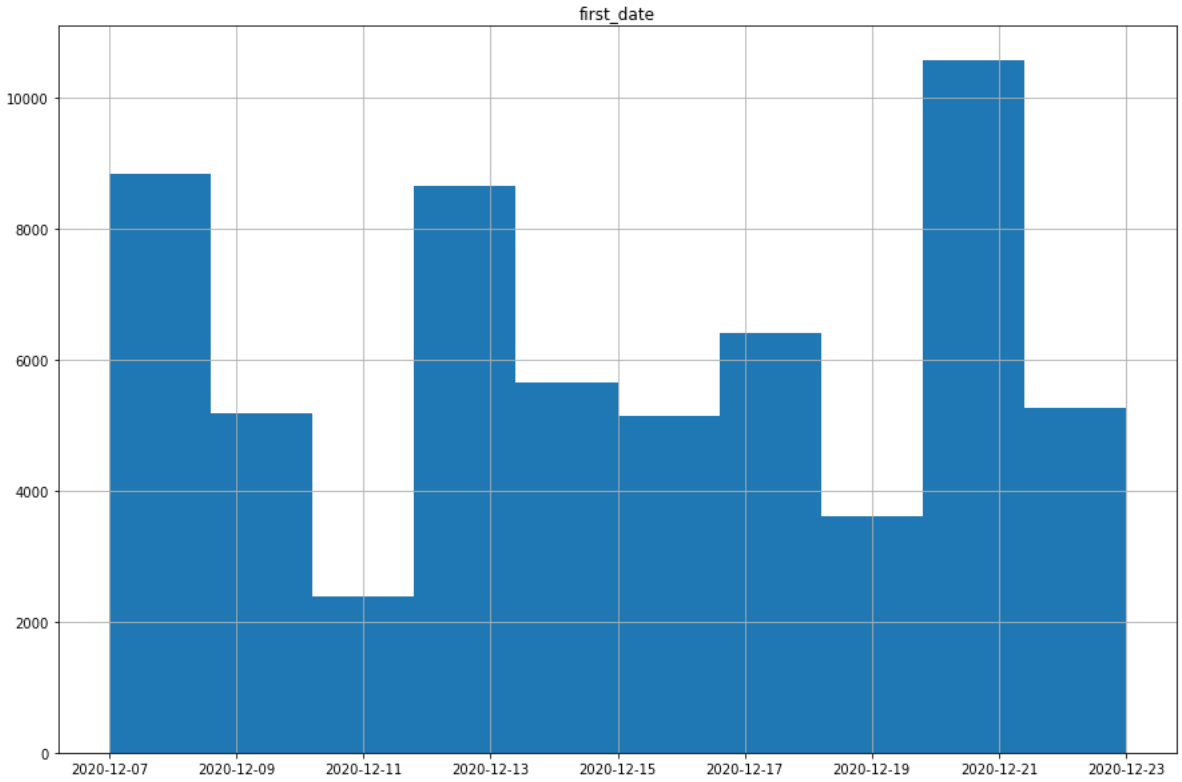
None

Общие статистические данные

```
C:\Users\МиО\AppData\Local\Temp\ipykernel_12548\2376842602.py:11: FutureWarning: Treating datetime data as categorical rath
er than numeric in `.describe` is deprecated and will be removed in a future version of pandas. Specify `datetime_is_numeri
c=True` to silence this warning and adopt the future behavior now.
  display(dataset.describe())
```

	user_id	first_date	region	device
count	61733	61733	61733	61733
unique	61733	17	4	4
top	D72A72121175D8BE	2020-12-21 00:00:00	EU	Android
freq	1	6290	46270	27520
first	NaN	2020-12-07 00:00:00	NaN	NaN
last	NaN	2020-12-23 00:00:00	NaN	NaN

Общие гистограммы для столбцов датасета




```
Количество дубликатов - 0
Доля дубликатов в датасете, %: 0.0
```

	Столбец	Количество пропусков (isna)	Количество пропусков (isna) в %	Количество пропусков (isnull)	Количество пропусков (isnull) в %
0	user_id	0	0.000000	0	0.000000
1	first_date	0	0.000000	0	0.000000
2	region	0	0.000000	0	0.000000
3	device	0	0.000000	0	0.000000

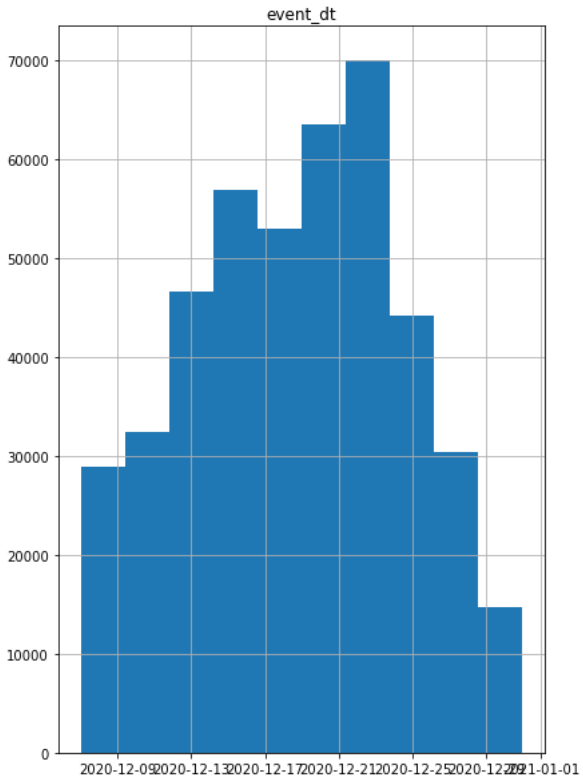
```
Датасет:
```

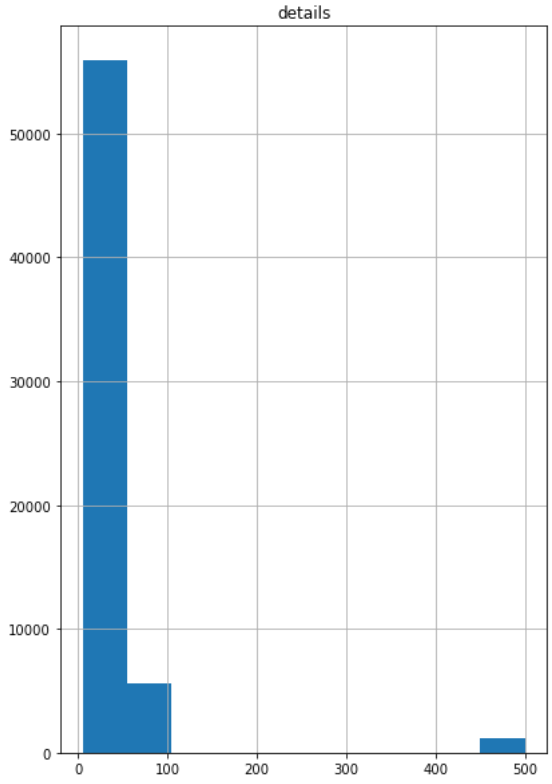
	user_id	first_date	region	device
0	D72A72121175D8BE	2020-12-07	EU	PC
1	F1C668619DFE6E65	2020-12-07	N.America	Android
2	2E1BF1D4C37EA01F	2020-12-07	EU	PC
3	50734A22C0C63768	2020-12-07	EU	iPhone
4	E1BDDCE0DAFA2679	2020-12-07	N.America	iPhone
5	137119F5A9E69421	2020-12-07	N.America	iPhone
6	62F0C741CC42D0CC	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
10	5EB159DA9DC94DBA	2020-12-07	APAC	PC
11	084A22B980BA8169	2020-12-07	EU	Android
12	8ACC2420471B31E4	2020-12-07	EU	PC
13	E6DE857AFBDC6102	2020-12-07	EU	PC
14	5BE017E9C8CC42F8	2020-12-07	EU	Android
15	7B6452F081F49504	2020-12-07	EU	iPhone
16	0FC21E6F8FAA8DEC	2020-12-07	EU	PC
17	9CD9F34546DF254C	2020-12-07	N.America	iPhone
18	96F27A054B191457	2020-12-07	EU	iPhone
19	1FD7660FDF94CA1F	2020-12-07	EU	Android

```
********************************************************************************

Название датасета: Действия новых пользователей в период с 7 Дек 2020 по 4 Янв 2021


Общая информация о датасете
<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


Общие статистические данные
```

	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

```
Общие гистограммы для  столбцов датасета
```






```
Количество дубликатов - 0
Доля дубликатов в датасете, %: 0.0
```

	Столбец	Количество пропусков (isna)	Количество пропусков (isna) в %	Количество пропусков (isnull)	Количество пропусков (isnull) в %
0	user_id	0	0.000000	0	0.000000
1	event_dt	0	0.000000	0	0.000000
2	event_name	0	0.000000	0	0.000000
3	details	377577	85.750000	377577	85.750000

Датасет:

	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
5	831887FE7F2D6CBA	2020-12-07 06:50:29	purchase	4.99
6	6B2F726BFD5F8220	2020-12-07 11:27:42	purchase	4.99
7	BEB37715AACF53B0	2020-12-07 04:26:15	purchase	4.99
8	B5FA27F582227197	2020-12-07 01:46:37	purchase	4.99
9	A92195E3CFB83DBD	2020-12-07 00:32:07	purchase	4.99
10	E2E76A8B3389127C	2020-12-07 15:39:32	purchase	4.99
11	354D653172FF2A2D	2020-12-07 15:45:11	purchase	4.99
12	7FCD34F47C13A9AC	2020-12-07 22:06:13	purchase	9.99
13	0313C457F07C339E	2020-12-07 13:10:48	purchase	9.99
14	214902A9BAAF3422	2020-12-07 19:31:44	purchase	4.99
15	AE6B95B9A7C6380E	2020-12-07 19:09:12	purchase	4.99
16	EC9AFCC18F1CF6D9	2020-12-07 19:19:00	purchase	4.99
17	3C5DD0288AC4FE23	2020-12-07 19:42:40	purchase	4.99
18	649ECF69EC552A56	2020-12-07 06:19:31	purchase	4.99
19	9C31B4124B3AE217	2020-12-07 21:38:36	purchase	4.99

****

Название датасета: Таблица участников тестов

Общая информация о датасете

```
<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

Общие статистические данные

	user_id	group	ab_test
count	18268	18268	18268
unique	16666	2	2
top	0FDFDA0B2DEC2D91	A	interface_eu_test
freq	2	9655	11567

Общие гистограммы для столбцов датасета

Количество дубликатов - 0
Доля дубликатов в датасете, %: 0.0

	Столбец	Количество пропусков (isna)	Количество пропусков (isna) в %	Количество пропусков (isnull)	Количество пропусков (isnull) в %
0	user_id	0	0.000000	0	0.000000
1	group	0	0.000000	0	0.000000
2	ab_test	0	0.000000	0	0.000000

Датасет:

	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
10	B3A2485649E4A012	A	recommender_system_test
11	E9FA12FAE3F5769C	B	recommender_system_test
12	FDD0A81654D707	A	recommender_system_test
13	547E99A7BDB0FCE9	A	recommender_system_test
14	66FC298441D50783	A	recommender_system_test
15	2DEC89AE4BDE6F87	A	recommender_system_test
16	CD74C3D22FFC09C2	B	recommender_system_test
17	19F4A16B875E04EE	A	recommender_system_test
18	BFE782851A612ED3	B	recommender_system_test
19	DFF78DC7022709TE	A	recommender_system_test

2.3 Дополнительное знакомство и предобработка данных

2.3.1 Календарь маркетинговых событий на 2020 год (`ab_project_marketing_events.csv`)

Добавим в календарь событий запись с периодом проведения A/B- теста

Ввод [8]:
```
ab_project_marketing_events
```

Out[8]:

	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
10	CIS New Year Gift Lottery	CIS	2020-12-30	2021-01-07
11	Dragon Boat Festival Giveaway	APAC	2020-06-25	2020-07-01
12	Single's Day Gift Promo	APAC	2020-11-11	2020-11-12
13	Chinese Moon Festival	APAC	2020-10-01	2020-10-07

****

Ввод [9]:

```
ab_project_marketing_events.loc[len(ab_project_marketing_events.index )] = ['AB Test', 'EU', '2020-12-07','2021-01-04']
ab_project_marketing_events.loc[len(ab_project_marketing_events.index )] = ['AB Test (user selection stage)', 'EU', '2020-12-07','202

ab_project_marketing_events['start_dt'] = pd.to_datetime(ab_project_marketing_events['start_dt'], format='%Y-%m-%d')
ab_project_marketing_events['finish_dt'] = pd.to_datetime(ab_project_marketing_events['finish_dt'], format='%Y-%m-%d')

#проверка корректно добавленной строки
display(ab_project_marketing_events.info())
display(ab_project_marketing_events)
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 16 entries, 0 to 15
Data columns (total 4 columns):
 #   Column     Non-Null Count  Dtype         
---  ------     --------------  -----         
 0   name       16 non-null     object        
 1   regions    16 non-null     object        
 2   start_dt   16 non-null     datetime64[ns]
 3   finish_dt  16 non-null     datetime64[ns]
dtypes: datetime64[ns](2), object(2)
memory usage: 640.0+ bytes
```

None

	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
10	CIS New Year Gift Lottery	CIS	2020-12-30	2021-01-07
11	Dragon Boat Festival Giveaway	APAC	2020-06-25	2020-07-01
12	Single's Day Gift Promo	APAC	2020-11-11	2020-11-12
13	Chinese Moon Festival	APAC	2020-10-01	2020-10-07
14	AB Test	EU	2020-12-07	2021-01-04
15	AB Test (user selection stage)	EU	2020-12-07	2020-12-21

Добавим отдельный столбец с маркером EU- региона

Ввод [10]:

```
def marking (region):

    if 'EU' in region:
        result = 'EU'
    else:
        result = 'other'

    return result
```

Ввод [11]:

```
ab_project_marketing_events['marker'] = ab_project_marketing_events['regions'].apply(marking)
```

Выведем диаграмму Ганта для мероприятий и выделим отдельные из них, затрагивающие регион EU

Ввод [12]:

```
fig = px.timeline(ab_project_marketing_events, x_start="start_dt", x_end="finish_dt", y="name", color = 'marker')
fig.update_yaxes(autorange="reversed")
fig.update_layout(title= 'Диаграмма Ганта для маркетинговых событий 2020 года',
                  xaxis_title= 'Дата',
                  yaxis_title= 'События'  )
fig.show()
```

1. С периодом проведения A/B теста совпадает только 'Christmas&New Year Promo', но дата начала этой кампании (25 декабря) позже даты окончания отбора пользователей для теста (21 декабря). Поэтому можно предположить, что на привлечение клиентов она не влияет. Возможно косвенное влияние на дальнейшие этапы воронки.
2. Перед проведением теста была кампания 'Black Friday Ads Campaign', но она закончилась 1 декабря, за 6 дней до начала запуска теста.

2.3.2 Пользователи, зарегистрировавшиеся с 7 по 21 декабря 2020 года (`final_ab_new_users.csv`)

Ввод [13]:

```
print("Перечень уникальных устройств, с которых регистрировались  пользователи")
print(final_ab_new_users['device'].unique())
print()
```

```
Перечень уникальных устройств, с которых регистрировались  пользователи
['PC' 'Android' 'iPhone' 'Mac']
```

Ввод [14]:

```
print('Начальная дата регистрации новых пользователей:', final_ab_new_users['first_date'].min() )
print('Конечная дата регистрации новых пользователей:', final_ab_new_users['first_date'].max() )
```

```
Начальная дата регистрации новых пользователей: 2020-12-07 00:00:00
Конечная дата регистрации новых пользователей: 2020-12-23 00:00:00
```

Последние пользователи регистрировались на 2 дня позже даты остановки набора новых пользователей- 21 декабря 2020. Удалим таких пользователей.

Ввод [15]:

```
final_ab_new_users = final_ab_new_users[final_ab_new_users['first_date']<='2020-12-21']
#print('Начальная дата регистрации новых пользователей:', final_ab_new_users['first_date'].min() )
#print('Конечная дата регистрации новых пользователей:', final_ab_new_users['first_date'].max() )
```

2.3.3 Действия новых пользователей в период с 7 Дек 2020 по 4 Янв 2021 (`final_ab_events.csv`)

Ввод [16]:

```
print('Начальная дата действий новых пользователей:', final_ab_events['event_dt'].min() )
print('Конечная дата действий новых пользователей:', final_ab_events['event_dt'].max() )
print()
```

```
Начальная дата действий новых пользователей: 2020-12-07 00:00:33
Конечная дата действий новых пользователей: 2020-12-30 23:36:33
```

Ввод [17]:

```
print('Уникальные события, совершаемые пользователями:')
print( final_ab_events['event_name'].unique())
print()
```

```
Уникальные события, совершаемые пользователями:
['purchase' 'product_cart' 'product_page' 'login']
```

2.3.4 Таблица участников тестов (final_ab_participants.csv)

Проверка групп на изолированность

Ввод [18]:

```
print('Уникальные группы, на которые разделены пользователи:')
print( final_ab_participants['group'].unique())
print()
```

```
Уникальные группы, на которые разделены пользователи:
['A' 'B']
```

Ввод [19]:

```
print('Проверка групп на изолированность.')
print()
user_count_start = final_ab_participants['user_id'].nunique()

#реестр пользователей в каждой группе
id_A = final_ab_participants[final_ab_participants['group']=='A']['user_id']
id_B = final_ab_participants[final_ab_participants['group']=='B']['user_id']

# реестр пользователей в обоих группах
id_A_B = np.intersect1d(id_A, id_B)

#print('Перечень пересекающихся пользователей в обеих группах:')
#print(id_A_B)
#print()
print('Количество пересекающихся пользователей в обеих группах:', len(id_A_B))
print()

print('Количество  изначально  уникальных пользователей в датасете:', user_count_start)
ratio =  round(100* len(id_A_B) /user_count_start ,2)
print(f'Доля пересекающихся пользователей от уникальных пользователей: {ratio} %')
```

```
Проверка групп на изолированность.

Количество пересекающихся пользователей в обеих группах: 776

Количество  изначально  уникальных пользователей в датасете: 16666
Доля пересекающихся пользователей от уникальных пользователей: 4.66 %
```

Удалим пользователей из пересекающихся групп

Ввод [20]:

```
#удаление пересечений групп
final_ab_participants = final_ab_participants[~final_ab_participants['user_id'].isin(id_A_B)]

print('Количество уникальных пользователей в датасете (итоговое):', final_ab_participants['user_id'].nunique())
ratio =  round(100* (final_ab_participants['user_id'].nunique()) /(user_count_start) ,2)
print(f'Размер очищенного датасета  относительно исходного : {ratio} %')
```

```
Количество уникальных пользователей в датасете (итоговое): 15890
Размер очищенного датасета  относительно исходного : 95.34 %
```

Проверка на пересечение тестов

Ввод [21]:

```
print('Перечень уникальных тестов: ')
print(final_ab_participants['ab_test'].unique())
```

```
Перечень уникальных тестов:
['recommender_system_test' 'interface_eu_test']
```

Ввод [22]:

```
print('Количество участников теста recommender_system_test')
print( final_ab_participants[final_ab_participants['ab_test']=='recommender_system_test']['user_id'].nunique() )
```

```
Количество участников теста recommender_system_test
5925
```

Ввод [23]:

```
print('Проверка тестов на изолированность.')
print()

#реестр пользователей в каждой группе
id_recommender = final_ab_participants[final_ab_participants['ab_test']=='recommender_system_test']['user_id'].unique()
id_interface = final_ab_participants[final_ab_participants['ab_test']=='interface_eu_test']['user_id'].unique()

# реестр пользователей в обоих группах
id_both_test = np.intersect1d(id_recommender, id_interface)

#print('Перечень пересекающихся пользователей в обоих тестах:')
#print(id_both_test)
#print()
print('Количество пересекающихся пользователей в обоих тестах:', len(id_both_test))
print()
print('Количество изначально  уникальных пользователей в датасете:', user_count_start)
ratio =  round(100* len(id_both_test) /(user_count_start) ,2)
print(f'Доля пересекающихся в пользователей от уникальных пользователей: {ratio} %')
```

```
Проверка тестов на изолированность.

Количество пересекающихся пользователей в обоих тестах: 826

Количество изначально  уникальных пользователей в датасете: 16666
Доля пересекающихся в пользователей от уникальных пользователей: 4.96 %
```

Распределение пользователей из пересекающихся тестов по группам:

Ввод [24]:

```
part_a =  final_ab_participants[final_ab_participants['group']=='A'].query('user_id in @id_both_test')['user_id'].nunique()
part_b =  final_ab_participants[final_ab_participants['group']=='B'].query('user_id in @id_both_test')['user_id'].nunique()
#whole_ab = part_a +part_b
#whole_ab

print(f'Пользователи группы A, попавшие в оба теста - {part_a}, что составляет {round(100*part_a/(part_a+part_b),2 )} % от всех польз
print(f'Пользователи группы B, попавшие в оба теста - {part_b}, что составляет {round(100*part_b/(part_a+part_b),2 )} % от всех польз
```

```
Пользователи группы A, попавшие в оба теста - 482, что составляет 58.35 % от всех пользователей с пересечениями тестов
Пользователи группы B, попавшие в оба теста - 344, что составляет 41.65 % от всех пользователей с пересечениями тестов
```

Доли не одинаковые, влияние на тестовую и контрольную группы не идентичное, поэтому для исключения ошибочных оценок, пользователей, попавших в оба теста, следует удалить из датасета.

Ввод [25]:

```
#удаление пересечений тестов
final_ab_participants = final_ab_participants[~final_ab_participants['user_id'].isin(id_both_test)]

print('Количество уникальных пользователей в датасете (итоговое):', final_ab_participants['user_id'].nunique())
ratio =  round(100* (final_ab_participants['user_id'].nunique()) /(user_count_start) ,2)
print(f'Размер очищенного датасета  относительно исходного : {ratio} %')
```

```
Количество уникальных пользователей в датасете (итоговое): 15064
Размер очищенного датасета  относительно исходного : 90.39 %
```

2.3.5 Итоговая таблица участников теста

Сформируем перечень участников теста 'recommender_system_test'

Ввод [26]:

```
# перечень пользователей из места recommender_system_test
test_users_list =  final_ab_participants[final_ab_participants['ab_test']=='recommender_system_test']['user_id']
print('Количество участников теста  -recommender_system_test- ',  len(test_users_list))
```

```
Количество участников теста  -recommender_system_test-  5099
```

!!! Участников теста меньше необходимых 6000

Сформируем перечень пользователей из EU региона

Ввод [27]:

```
# новые пользователи из EU региона

new_users_eu   = final_ab_new_users[final_ab_new_users['region']=='EU']
new_users_eu_list = new_users_eu['user_id']
#display(new_users_eu)

print('Количество пользователей из EU')
new_users_eu_cnt = new_users_eu['user_id'].nunique()
print(new_users_eu_cnt)


```

```
Количество пользователей из EU
42340
```

Ввод [28]:

```
total_usr_eu = np.intersect1d( new_users_eu['user_id'].unique(), test_users_list  )
print('Количество пользователей из EU, участвующих в тесте', len(total_usr_eu) )
print('Доля аудитории теста от всех пользователей из EU, %:',
         round(100 *len(total_usr_eu)/ new_users_eu_cnt ,2) )

```

```
Количество пользователей из EU, участвующих в тесте 4749
Доля аудитории теста от всех пользователей из EU, %: 11.22
```

Количество пользователей стало еще меньше от необходимых по ТЗ 6 тысяч и доля новых пользователей из региона EU тоже меньше заданных 15%.

Объединим таблицы пользователей и действий.

Ввод [29]:

```
complete_user = final_ab_participants[final_ab_participants['ab_test']=='recommender_system_test'].merge(final_ab_events.query('user_
complete_user = complete_user.sort_values(by='user_id')
display(complete_user.head())
```

	user_id	group	ab_test	event_dt	event_name	details
18006	000ABE35EE11412F	A	recommender_system_test	NaT	NaN	NaN
16328	0010A1C096941592	A	recommender_system_test	2020-12-21 21:05:24	product_page	NaN
16327	0010A1C096941592	A	recommender_system_test	2020-12-19 04:34:38	product_page	NaN
16333	0010A1C096941592	A	recommender_system_test	2020-12-23 11:52:09	login	NaN
16331	0010A1C096941592	A	recommender_system_test	2020-12-19 04:34:37	login	NaN

Добавим информацию о дате регистрации и устройстве, с которого проходила регистрация.

Ввод [30]:

```
complete_user = complete_user.merge(final_ab_new_users, how = 'left')
complete_user = complete_user.sort_values(by='user_id').reset_index(drop = True)
display(complete_user.head())
```

	user_id	group	ab_test	event_dt	event_name	details	first_date	region	device
0	000ABE35EE11412F	A	recommender_system_test	NaT	NaN	NaN	2020-12-08	EU	PC
1	0010A1C096941592	A	recommender_system_test	2020-12-23 11:52:10	product_page	NaN	2020-12-17	EU	Android
2	0010A1C096941592	A	recommender_system_test	2020-12-17 21:07:27	product_page	NaN	2020-12-17	EU	Android
3	0010A1C096941592	A	recommender_system_test	2020-12-21 21:05:23	purchase	4.99	2020-12-17	EU	Android
4	0010A1C096941592	A	recommender_system_test	2020-12-19 04:34:37	purchase	4.99	2020-12-17	EU	Android

Заполним пропуски в некоторых столбцах

Ввод [31]:

```
# заполним пропуски в столбце с действиям маркером "only_registration"
# заполним пропуски в столбце с датой действия значением даты регистрации
complete_user['event_name'] =complete_user['event_name'].fillna('inactive')
complete_user['event_dt'] =complete_user['event_dt'].fillna(complete_user['first_date'])
complete_user['event_dt'] = pd.to_datetime(complete_user['event_dt'], format='%Y-%m-%d')

display(complete_user.head(15))
```

	user_id	group	ab_test	event_dt	event_name	details	first_date	region	device
0	000ABE35EE11412F	A	recommender_system_test	2020-12-08 00:00:00	inactive	NaN	2020-12-08	EU	PC
1	0010A1C096941592	A	recommender_system_test	2020-12-23 11:52:10	product_page	NaN	2020-12-17	EU	Android
2	0010A1C096941592	A	recommender_system_test	2020-12-17 21:07:27	product_page	NaN	2020-12-17	EU	Android
3	0010A1C096941592	A	recommender_system_test	2020-12-21 21:05:23	purchase	4.99	2020-12-17	EU	Android
4	0010A1C096941592	A	recommender_system_test	2020-12-19 04:34:37	purchase	4.99	2020-12-17	EU	Android
5	0010A1C096941592	A	recommender_system_test	2020-12-17 21:07:27	purchase	4.99	2020-12-17	EU	Android
6	0010A1C096941592	A	recommender_system_test	2020-12-23 11:52:09	purchase	9.99	2020-12-17	EU	Android
7	0010A1C096941592	A	recommender_system_test	2020-12-17 21:07:27	login	NaN	2020-12-17	EU	Android
8	0010A1C096941592	A	recommender_system_test	2020-12-19 04:34:37	login	NaN	2020-12-17	EU	Android
9	0010A1C096941592	A	recommender_system_test	2020-12-23 11:52:09	login	NaN	2020-12-17	EU	Android
10	0010A1C096941592	A	recommender_system_test	2020-12-19 04:34:38	product_page	NaN	2020-12-17	EU	Android
11	0010A1C096941592	A	recommender_system_test	2020-12-21 21:05:24	product_page	NaN	2020-12-17	EU	Android
12	0010A1C096941592	A	recommender_system_test	2020-12-21 21:05:23	login	NaN	2020-12-17	EU	Android
13	001C05E87D336C59	A	recommender_system_test	2020-12-10 00:00:00	inactive	NaN	2020-12-10	EU	iPhone
14	003DF44D7589BBD4	A	recommender_system_test	2020-12-19 16:16:03	login	NaN	2020-12-17	EU	Android

Ввод [32]:

```
print('Количество уникальных пользователей объединенного датасета:', complete_user['user_id'].nunique())

print('Количество уникальных пользователей, совершивших хотя бы одно действие:',
      complete_user[complete_user['event_name']!='inactive']['user_id'].nunique())
print('Доля активных от всех:', round(100* (complete_user[complete_user['event_name']!='inactive']['user_id'].nunique())/complete_use
```

```
Количество уникальных пользователей объединенного датасета: 5099
Количество уникальных пользователей, совершивших хотя бы одно действие: 2594
Доля активных от всех: 50.87 %
```

Кроме того, что исходное количество пользователей меньше необходимого по ТЗ, так еще и пользователи не отличаются активностью.

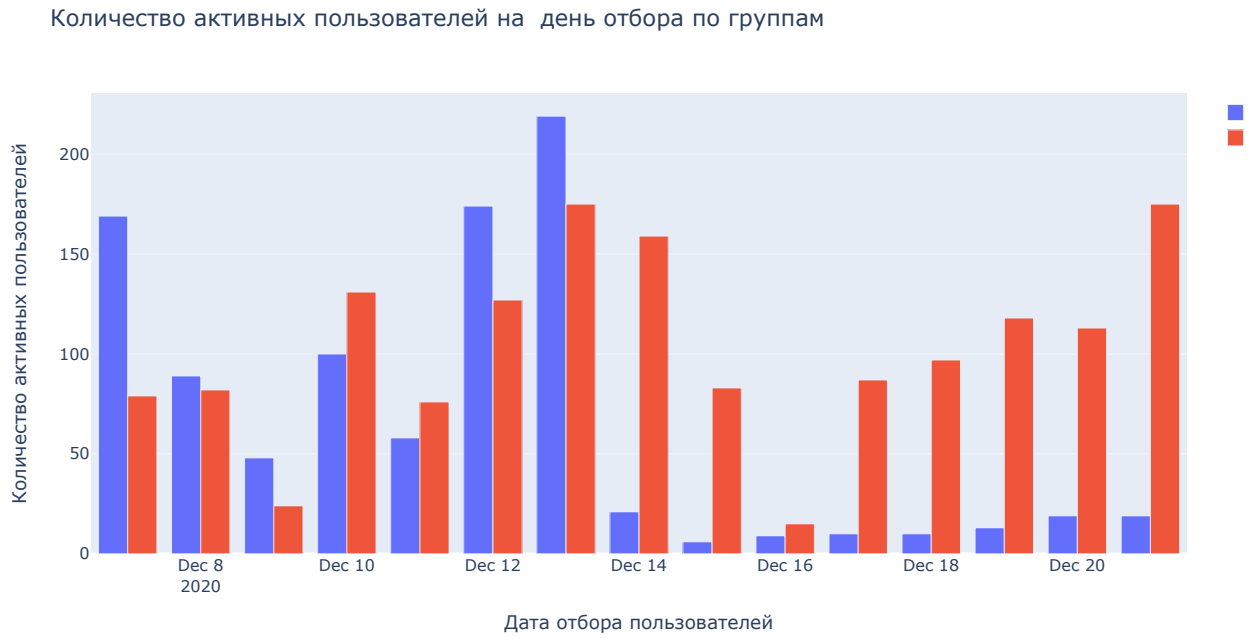
Посмотрим на динамику набора активных пользователей.

Ввод [33]:

```
# обрезка даты/времени до даты
complete_user['date_cut'] = complete_user['event_dt'].dt.date

#complete_user

# подсчет количества пользователей  в группе A
day_users_A =  complete_user[(complete_user['group']=='A') & (complete_user['event_name']=='inactive')].groupby('first_date')['user_i
day_users_A['group'] = 'A'
#day_events_A

# подсчет количества пользователей в группе B
day_users_B =  complete_user[(complete_user['group']=='B') & (complete_user['event_name']=='inactive')].groupby('first_date')['user_i
day_users_B['group'] = 'B'
#day_events_B


# объединение данных
day_users= pd.concat([day_users_A,day_users_B],axis=0 , sort= False)
day_users.columns=['data', 'users_count_per_day', 'group']

day_users = day_users.reset_index(drop= True)

```

Ввод [34]:

```
fig = go.Figure(data=[
    go.Bar(name='A', x=day_users[day_users['group']=="A"]['data'],
                     y=day_users[day_users['group']=="A"]['users_count_per_day']),

    go.Bar(name='B', x=day_users[day_users['group']=="B"]['data'],
                     y=day_users[day_users['group']=="B"]['users_count_per_day']),

])


fig.update_layout(title= 'Количество активных пользователей на  день отбора по группам',
                      xaxis_title='Дата отбора пользователей',
                      yaxis_title='Количество активных пользователей')
# Change the bar mode
#fig.update_layout(barmode='group')
fig.show()

```



1. Набор неравномерный как для разных групп, так и в течение периода отбора участников теста. Для обеих групп заметен пик количества активных пользователей 12-14 декабря.
2. Активные пользователи контрольной группы А добавлялись в перечень участников теста более активно в первой половине теста с резким спадом после 13 декабря.
3. Тестовая группа В показывает бОльшую, чем у группы А равномерность отбора, но и она недостаточно равномерна.

Посмотрим на неактивных пользователей.

Ввод [35]:

```
# подсчет количества пользователей  в группе A
day_users_A =  complete_user[(complete_user['group']=='A') & (complete_user['event_name']!='inactive')].groupby('first_date')['user_i
day_users_A['group'] = 'A'
#day_events_A

# подсчет количества пользователей в группе B
day_users_B =  complete_user[(complete_user['group']=='B') & (complete_user['event_name']!='inactive')].groupby('first_date')['user_i
day_users_B['group'] = 'B'
#day_events_B


# объединение данных
day_users= pd.concat([day_users_A,day_users_B],axis=0 , sort= False)
day_users.columns=['data', 'users_count_per_day', 'group']

day_users = day_users.reset_index(drop= True)

```

Ввод [36]:

```
fig = go.Figure(data=[
    go.Bar(name='A', x=day_users[day_users['group']=="A"]['data'],
                     y=day_users[day_users['group']=="A"]['users_count_per_day']),

    go.Bar(name='B', x=day_users[day_users['group']=="B"]['data'],
                     y=day_users[day_users['group']=="B"]['users_count_per_day']),

])


fig.update_layout(title= 'Количество неактивных пользователей на  день отбора по группам',
                      xaxis_title='Дата отбора пользователей',
                      yaxis_title='Количество неактивных пользователей')
# Change the bar mode
#fig.update_layout(barmode='group')
fig.show()

```

1. Набор неравномерный как для разных групп, так и в течение периода отбора участников теста. Система сплитования пользователей сбоит и здесь.
2. Неактивные пользователи контрольной группы А заметно пополнили перечень участников теста во второй половине теста с резким ростом после 13 декабря.
3. Тестовая группа В показывает бОльшую, чем у группы А равномерность отбора, но и она недостаточно равномерна.

2.4 Выводы по предобработке данных

1. **Календарь маркетинговых событий на 2020 год** (`ab_project_marketing_events.csv`)

- Названия столбцов в корректном формате
- Данные приведены в корректный формат.
- Дубликатов строк и пропусков в данных не обнаружено
- В датасете имеется 14 записей о маркетинговых событиях 2020 года
- Начальная дата событий - 25 января 2020
- Финальная дата событий - 07 января 2021
- Распределение дат событий неоднородно в течение года.
- Представлено 4 региона, на которые распространяется действие маркетинговых кампаний: EU, CIS, APAC, N.America и 6 сочетаний этих регионов. Данные по регионам обозначены как строки данных, для последующей обработки данных, необходимо выполнить разделение строк.

2. **Пользователи, зарегистрировавшиеся с 7 по 21 декабря 2020 года** (`final_ab_new_users.csv`)

- Названия столбцов в корректном формате
- Данные приведены в корректный формат.
- Дубликатов строк и пропусков в данных не обнаружено
- В датасете имеется 61 733 записей о пользователях
- Начальная дата - 07 декабря 2020
- Финальная дата - **23 (!)** декабря 2020 , на 2 дня позже необходимого 21 числа. Данных пользователей удалили из рассмотрения.
- Распределение дат регистраций неоднородно в течение периода регистрации.
- Перечень уникальных устройств, с которых регистрировались пользователи: 'PC', 'Android', 'iPhone', 'Mac'

3. **Действия новых пользователей в период с 7 Дек 2020 по 4 Янв 2021** (`final_ab_events.csv`)

- Названия столбцов в корректном формате
- Данные приведены в корректный формат.
- Дубликатов строк в данных не обнаружено.
- Пропуски в данных обнаружены в столбце `details` - 85.75% от всего датасета.
- В датасете имеется 440 317 записей о пользователях
- Начальная дата - 07 декабря 2020
- Финальная дата - **30 (!)** декабря 2020 , на 5 дней раньше необходимого 4 Янв 2021.
- Распределение дат событий неоднородно в течение периода.
- Стоимость покупок (столбец `details`) содержит данные в диапазоне 4,99 - 499,99, со средним 23,88 и медианой 4,99
- Уникальные события, совершаемые пользователями: 'purchase', 'product_cart', 'product_page', 'login'.

4. **Таблица участников тестов** (`final_ab_participants.csv`)

- Названия столбцов в корректном формате
- Данные приведены в корректный формат.
- Дубликатов строк и пропусков в данных не обнаружено
- В датасете имеется 18 268 записей о пользователях
- Пользователи разделены на группы A и B.
- Часть пользователей была в обеих группах и в обоих тестах, таких пользователей удалили из датасета для корректной оценки результатов теста.
- Учитывая явную неоднородность добавления различных пользователей по группам, следует внимательно изучить работу системы сплитования трафика.

5. **Итоговая таблица участников теста. Соответствие ТЗ.** После всех фильтраций сформирована итоговая таблица с данными пользователей по тестам.

    - Количество уникальных пользователей объединенного датасета: 5099
    - Количество уникальных пользователей, совершивших хотя бы одно действие: 2594
    - Доля активных от всех: 50.87 %

ТЗ

- 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` .

Выполнение отборочных требований ТЗ

- количество пользователей меньше от необходимых 6000 участников
- доля пользователей, участвующих в тесте, из региона EU тоже меньше заданных 15%
- последняя дата регистрации нового пользователя - 30 декабря - раньше запланированного останова набора на 5 дней

3 Исследовательский анализ данных (EDA)

3.1 Распределение пользователей по группам

Ввод [37]:

```
print('Количество уникальных пользователей группы A:',  complete_user[ complete_user['group']=='A']['user_id'].nunique())
print('Доля пользователей группы A от всех:',
    round(100*(complete_user[complete_user['group']=='A']['user_id'].nunique())/ complete_user['user_id'].nunique() ,2),'%')

print()
print('Количество уникальных пользователей группы B:',  complete_user[ complete_user['group']=='B']['user_id'].nunique())
print('Доля пользователей группы B от всех:',
    round(100*(complete_user[complete_user['group']=='B']['user_id'].nunique())/ complete_user['user_id'].nunique() ,2),'%')

print()
print('Относительное различие количества пользователей в группах:')
print( round(100*(complete_user[complete_user['group']=='B']['user_id'].nunique())/ complete_user[complete_user['group']=='A']['user_
```

```
Количество уникальных пользователей группы A: 2903
Доля пользователей группы A от всех: 56.93 %

Количество уникальных пользователей группы B: 2196
Доля пользователей группы B от всех: 43.07 %

Относительное различие количества пользователей в группах:
74.65 %
```

Изначально неоднородный состав пользователей. Посчитаем распределение пользователей по группам на каждом этапе и отобразим воронкой.

Ввод [38]:

```
ratio_ab =  complete_user.pivot_table(index = ['event_name','group'] ,  values= ['user_id'], aggfunc= {'nunique'}).reset_index()
ratio_ab.columns= ['stage', 'group','nunique' ]
ratio_ab.loc[len(ratio_ab.index)] = ['total users', 'A',  complete_user[ complete_user['group']=='A']['user_id'].nunique() ]
ratio_ab.loc[len(ratio_ab.index)] = ['total users', 'B',  complete_user[ complete_user['group']=='B']['user_id'].nunique()]

# определим порядок стадий

for i  in ratio_ab.index :
    if    ratio_ab.loc[i,'stage'] == 'total users':
          ratio_ab.loc[i,'stage_n'] = 1

    elif  ratio_ab.loc[i,'stage'] == 'inactive':
          ratio_ab.loc[i,'stage_n'] = 2

    elif  ratio_ab.loc[i,'stage'] == 'login':
          ratio_ab.loc[i,'stage_n'] = 3

    elif  ratio_ab.loc[i,'stage'] == 'product_page':
          ratio_ab.loc[i,'stage_n'] = 4

    elif  ratio_ab.loc[i,'stage'] == 'product_cart':
          ratio_ab.loc[i,'stage_n'] = 5

    elif  ratio_ab.loc[i,'stage'] == 'purchase':
          ratio_ab.loc[i,'stage_n'] = 6

ratio_ab = ratio_ab.sort_values(by = 'stage_n').reset_index(drop= True)
ratio_ab
```

Out[38]:

	stage	group	nunique	stage_n
0	total users	A	2903	1.0
1	total users	B	2196	1.0
2	inactive	A	964	2.0
3	inactive	B	1541	2.0
4	login	A	1939	3.0
5	login	B	655	3.0
6	product_page	A	1265	4.0
7	product_page	B	367	4.0
8	product_cart	A	589	5.0
9	product_cart	B	184	5.0
10	purchase	A	613	6.0
11	purchase	B	191	6.0

Ввод [39]:

```
fig = px.funnel(ratio_ab, x= 'nunique', y='stage', color='group',
                title='Воронка количества пользователей по типам действий ')

fig.show()
```

1. **Важно!!!** признаки 'бездейственный' `inactive` и 'залогинившийся' `login` по времени одномоментны и дополняют друг друга до полного числа пользователей в тесте.
2. Заметно, что контрольная группа A более активна на всех шагах воронки.

3.2 Количество событий на пользователя

Определим количество событий на пользователя для каждой группы в отдельности

Ввод [40]:

```
# подсчет количества событий на пользователя в группе A
user_events_A =  complete_user[complete_user['group']=='A'].groupby('user_id')['event_name'].count().reset_index()
user_events_A['group'] = 'A'
#user_events_A

# подсчет количества событий на пользователя в группе B
user_events_B =  complete_user[complete_user['group']=='B'].groupby('user_id')['event_name'].count().reset_index()
user_events_B['group'] = 'B'
#user_events_B

# объединение данных
user_events= pd.concat([user_events_A,user_events_B],axis=0 , sort= False)
user_events.columns=['user_id', 'event_count_per_user', 'group']

user_events
```

Out[40]:

	user_id	event_count_per_user	group
0	000ABE35EE11412F	1	A
1	0010A1C096941592	12	A
2	001C05E87D336C59	1	A
3	003DF44D7589BBD4	15	A
4	00505E15A9D81546	5	A
...	...	...	...
2191	FF547172851FB1EE	1	B
2192	FF6B409A73E7B239	1	B
2193	FF8CF7057415EB29	4	B
2194	FFB3F647898BA928	1	B
2195	FFC2C5F898D1245B	1	B

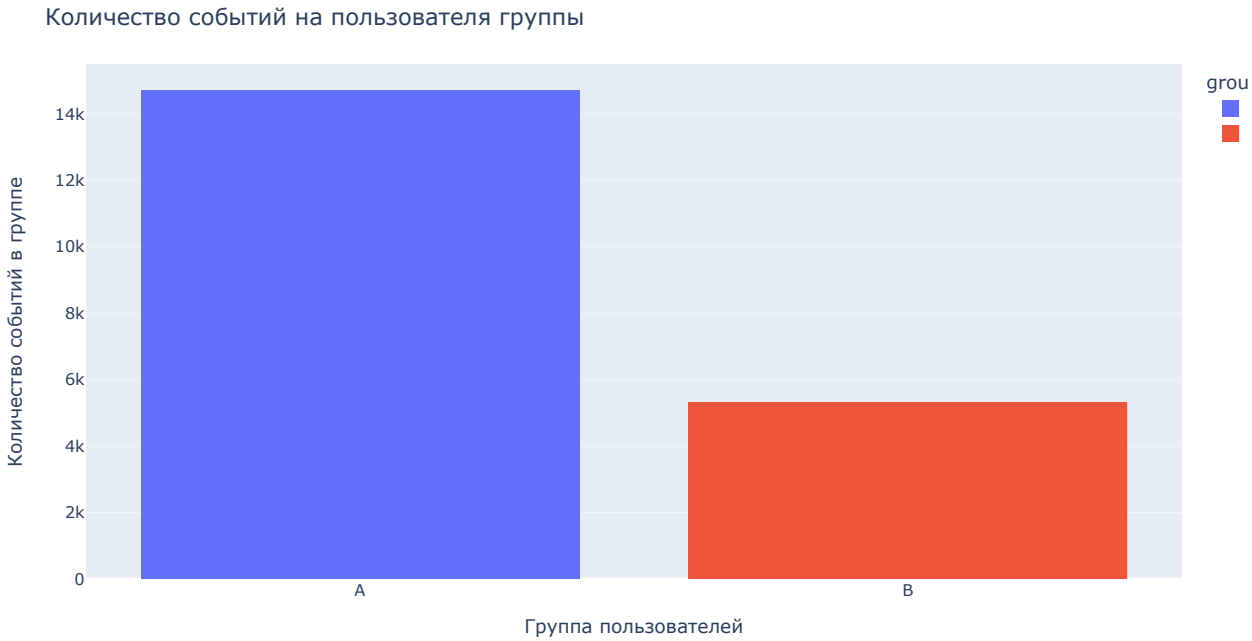
5099 rows × 3 columns

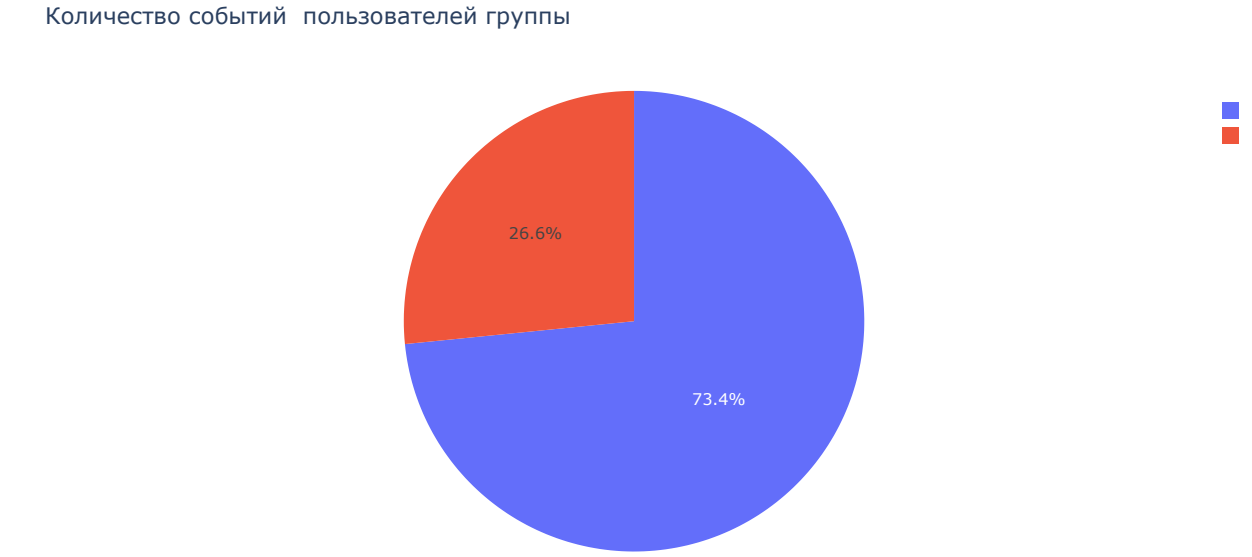
Ввод [41]:

```
for g in user_events['group'].unique():
    fig = px.histogram(user_events[user_events['group']== g ], x="event_count_per_user", marginal="box")

    fig.update_layout(title= f'Количество событий на пользователя группы {g}',
                      xaxis_title='Количество событий на пользователя',
                      yaxis_title='Количество пользователей')
    fig.show()
```

Ввод [42]:

```
fig = px.histogram(user_events, y="event_count_per_user", x ='group'  , color ='group')

fig.update_layout(title= f'Количество событий на пользователя группы',
                      xaxis_title='Группа пользователей',
                      yaxis_title='Количество событий в группе')
fig.show()

fig = px.pie(user_events, values="event_count_per_user"  , names ='group' ,   color ='group',
             title= f'Количество событий  пользователей группы')
fig.show()
```





Посмотрим на количество событий каждого типа в группах

Ввод [43]:

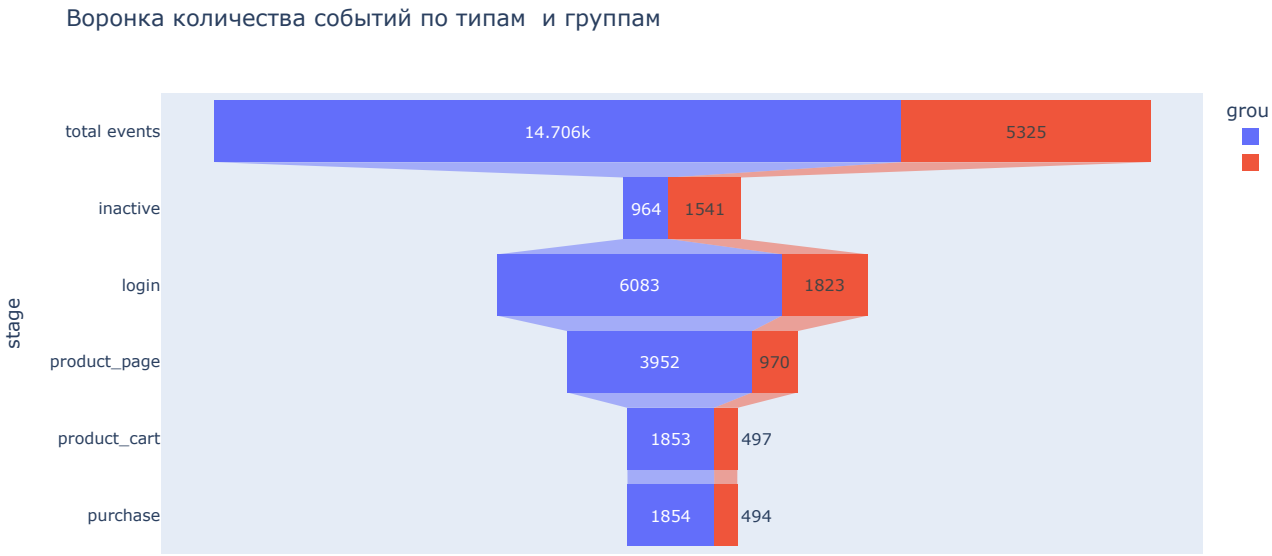
```
ratio_ab =  complete_user.pivot_table(index = ['event_name','group'] ,  values= ['user_id'], aggfunc= {'count'}).reset_index()
ratio_ab.columns= ['stage', 'group','nunique' ]
ratio_ab.loc[len(ratio_ab.index)] = ['total events', 'A',  complete_user[ complete_user['group']=='A']['user_id'].count() ]
ratio_ab.loc[len(ratio_ab.index)] = ['total events', 'B',  complete_user[ complete_user['group']=='B']['user_id'].count()]

# определим порядок стадий

for i  in ratio_ab.index :
    if    ratio_ab.loc[i,'stage'] == 'total events':
          ratio_ab.loc[i,'stage_n'] = 1

    elif  ratio_ab.loc[i,'stage'] == 'inactive':
          ratio_ab.loc[i,'stage_n'] = 2

    elif  ratio_ab.loc[i,'stage'] == 'login':
          ratio_ab.loc[i,'stage_n'] = 3

    elif  ratio_ab.loc[i,'stage'] == 'product_page':
          ratio_ab.loc[i,'stage_n'] = 4

    elif  ratio_ab.loc[i,'stage'] == 'product_cart':
          ratio_ab.loc[i,'stage_n'] = 5

    elif  ratio_ab.loc[i,'stage'] == 'purchase':
          ratio_ab.loc[i,'stage_n'] = 6

ratio_ab = ratio_ab.sort_values(by = 'stage_n').reset_index(drop= True)
ratio_ab
```

Out[43]:

	stage	group	nunique	stage_n
0	total events	A	14706	1.0
1	total events	B	5325	1.0
2	inactive	A	964	2.0
3	inactive	B	1541	2.0
4	login	A	6083	3.0
5	login	B	1823	3.0
6	product_page	A	3952	4.0
7	product_page	B	970	4.0
8	product_cart	A	1853	5.0
9	product_cart	B	497	5.0
10	purchase	A	1854	6.0
11	purchase	B	494	6.0

Ввод [44]:

```
fig = px.funnel(ratio_ab, x= 'nunique', y='stage', color='group',
                title='Воронка количества событий по типам  и группам ')

fig.show()
```



1. **Важно!!!** признак 'бездейственный' `inactive` дополняет остальные события до полного числа событий в тесте.
2. Пользователи группы B не только малочисленны, но и менее активны и совершают кратно меньше действий, чем пользователи контрольной группы, пользователи которой совершают больше действий каждого типа.

3.3 Распределение событий по дням

Определим количество событий в день для каждой группы в отдельности

Ввод [45]:

```
#complete_user

# подсчет количества событий на пользователя в группе A
day_events_A =  complete_user[complete_user['group']=='A'].groupby('date_cut')['event_name'].count().reset_index()
day_events_A['group'] = 'A'
#day_events_A

# подсчет количества событий на пользователя в группе B
day_events_B =  complete_user[complete_user['group']=='B'].groupby('date_cut')['event_name'].count().reset_index()
day_events_B['group'] = 'B'
#day_events_B


# объединение данных
day_events= pd.concat([day_events_A,day_events_B],axis=0 , sort= False)
day_events.columns=['data', 'event_count_per_day', 'group']

day_events = day_events.reset_index(drop= True)
#day_events
```

Посмотрим на распределение количества событий по дням:

Ввод [46]:

```
fig = go.Figure(data=[
    go.Bar(name='A', x=day_events[day_events['group']=="A"]['data'],
                     y=day_events[day_events['group']=="A"]['event_count_per_day'],
                     textposition='auto'),

    go.Bar(name='B', x=day_events[day_events['group']=="B"]['data'],
                     y=day_events[day_events['group']=="B"]['event_count_per_day'],
                     textposition='auto'),

])


fig.update_layout(title= 'Количество событий в день по группам',
                      xaxis_title='Дата',
                      yaxis_title='Количество событий')
# Change the bar mode
fig.update_layout(barmode='group')
fig.show()

```

1. Заметна увеличивающаяся разница между количеством событий в группах по мере продолжения теста.
2. В группах пик активности приходится на 21 декабря - дату окончания набора новых пользователей.
3. Для группы B спад активности после окончания набора пользователей более резкий, чем у контрольной группы.

События и дни лайфтайма

Для каждого пользователя определим лайфтам и среднее количество дней лайфтайма для каждого типа события.

Ввод [47]:

```
# определение номера дня события
complete_user['date_cut'] = pd.to_datetime(complete_user['date_cut'], format= '%Y-%m-%d')

complete_user['event_day'] = complete_user['date_cut'] - complete_user['first_date']

complete_user_lifitime = complete_user.groupby('user_id')['event_day'].max().reset_index()
complete_user_lifitime.columns = ['user_id','lifetime']

complete_user = complete_user.merge(complete_user_lifitime, on='user_id', how= 'left' )
complete_user['event_day'] = complete_user['event_day'].dt.days
complete_user['lifetime'] = complete_user['lifetime'].dt.days
complete_user.head(10)
```

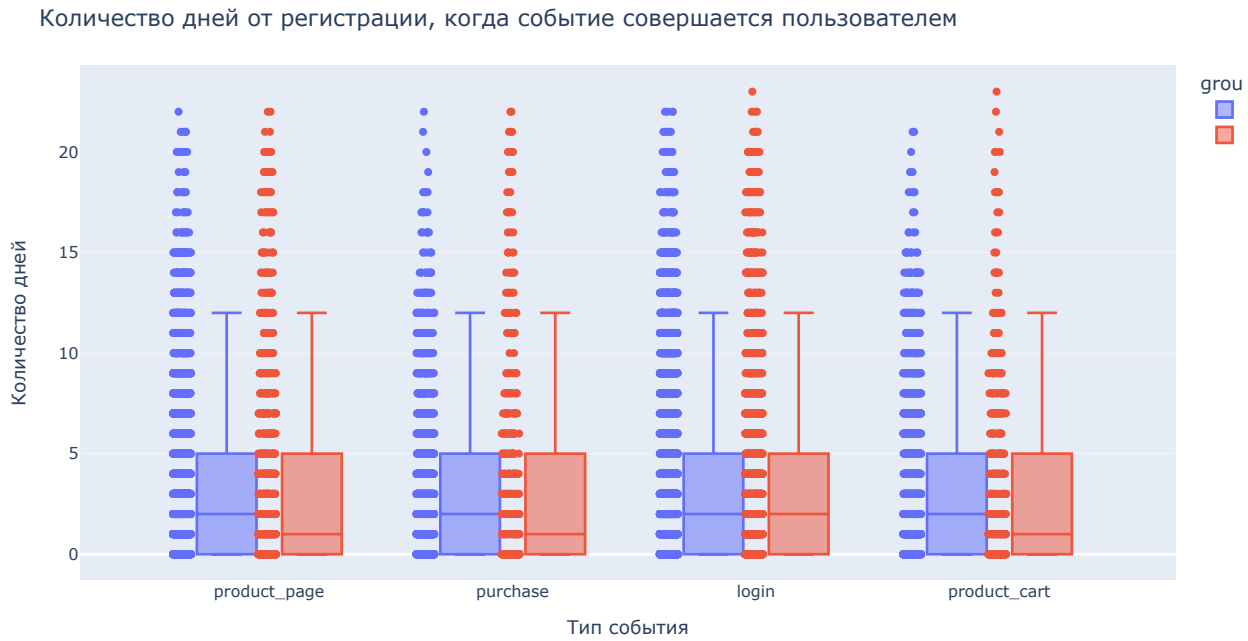
Out[47]:

	user_id	group	ab_test	event_dt	event_name	details	first_date	region	device	date_cut	event_day	lifetime
0	000ABE35EE11412F	A	recommender_system_test	2020-12-08 00:00:00	inactive	NaN	2020-12-08	EU	PC	2020-12-08	0	0
1	0010A1C096941592	A	recommender_system_test	2020-12-23 11:52:10	product_page	NaN	2020-12-17	EU	Android	2020-12-23	6	6
2	0010A1C096941592	A	recommender_system_test	2020-12-17 21:07:27	product_page	NaN	2020-12-17	EU	Android	2020-12-17	0	6
3	0010A1C096941592	A	recommender_system_test	2020-12-21 21:05:23	purchase	4.99	2020-12-17	EU	Android	2020-12-21	4	6
4	0010A1C096941592	A	recommender_system_test	2020-12-19 04:34:37	purchase	4.99	2020-12-17	EU	Android	2020-12-19	2	6
5	0010A1C096941592	A	recommender_system_test	2020-12-17 21:07:27	purchase	4.99	2020-12-17	EU	Android	2020-12-17	0	6
6	0010A1C096941592	A	recommender_system_test	2020-12-23 11:52:09	purchase	9.99	2020-12-17	EU	Android	2020-12-23	6	6
7	0010A1C096941592	A	recommender_system_test	2020-12-17 21:07:27	login	NaN	2020-12-17	EU	Android	2020-12-17	0	6
8	0010A1C096941592	A	recommender_system_test	2020-12-19 04:34:37	login	NaN	2020-12-17	EU	Android	2020-12-19	2	6
9	0010A1C096941592	A	recommender_system_test	2020-12-23 11:52:09	login	NaN	2020-12-17	EU	Android	2020-12-23	6	6

Посмотрим на среднее количество дней от регистрации, когда событие совершается пользователем:

Ввод [48]:

```
fig = px.box(complete_user[complete_user['event_name']!='inactive'],
                     x="event_name", y="event_day", points="all", color = 'group')

fig.update_layout(title= 'Количество дней от регистрации, когда событие совершается пользователем',
                      xaxis_title='Тип события',
                      yaxis_title='Количество дней')
fig.show()

```



1. Для обеих групп заметно похожее распределение активности пользователей в течение лайфтайма- основная масса активностей происходит в период от 2-3 до 12 дней.
2. В группе А медианно действия совершаются на второй день регистрации, в группе В - на третий.
3. Максимальный шлейф от регистрации до действия для группы А выше - 23 дня против 22 дней у группы В.

3.4 Конверсия

Учитывая, что основная активность пользователей происходит за 12 дней, то для оценки конверсии выполним отбор событий со шлейфом до 14 дней от даты регистрации (по ТЗ).

Ввод [49]:

```
complete_user_14  = complete_user[complete_user['event_day']<= 14]
complete_user_14.head()
```

Out[49]:

	user_id	group	ab_test	event_dt	event_name	details	first_date	region	device	date_cut	event_day	lifetime
0	000ABE35EE11412F	A	recommender_system_test	2020-12-08 00:00:00	inactive	NaN	2020-12-08	EU	PC	2020-12-08	0	0
1	0010A1C096941592	A	recommender_system_test	2020-12-23 11:52:10	product_page	NaN	2020-12-17	EU	Android	2020-12-23	6	6
2	0010A1C096941592	A	recommender_system_test	2020-12-17 21:07:27	product_page	NaN	2020-12-17	EU	Android	2020-12-17	0	6
3	0010A1C096941592	A	recommender_system_test	2020-12-21 21:05:23	purchase	4.99	2020-12-17	EU	Android	2020-12-21	4	6
4	0010A1C096941592	A	recommender_system_test	2020-12-19 04:34:37	purchase	4.99	2020-12-17	EU	Android	2020-12-19	2	6

Определим долю отобранных пользователей и событий для каждой группы

Ввод [50]:

```
for event in complete_user[complete_user['event_name']!='inactive']['event_name'].unique():
    print()
    event_cnt = complete_user[(complete_user['event_name']== event) & (complete_user['group']=='A') ] ['user_id'].count()
    print(f'Количество событий {event}  в группе A всего: {event_cnt}')
    event_cnt_14  = complete_user_14[(complete_user_14['event_name']== event) & (complete_user_14['group']=='A') ] ['user_id'].count(
    print(f'Количество событий {event}  в группе A за 14 дней: {event_cnt_14}')
    print(f'Доля событий {event}  за 14-дневный период от событий {event} за весь период:  { round( 100* event_cnt_14/event_cnt ,2) }
    print()

    event_cnt = complete_user[(complete_user['event_name']== event) & (complete_user['group']=='B') ] ['user_id'].count()
    print(f'Количество событий {event}  в группе B всего: {event_cnt}')
    event_cnt_14  = complete_user_14[(complete_user_14['event_name']== event) & (complete_user_14['group']=='B') ] ['user_id'].count(
    print(f'Количество событий {event}  в группе B за 14 дней: {event_cnt_14}')
    print(f'Доля событий {event}  за 14-дневный период от событий {event} за весь период:  { round( 100* event_cnt_14/event_cnt ,2) }
    print()
    print()
```

```
Количество событий product_page  в группе A всего: 3952
Количество событий product_page  в группе A за 14 дней: 3873
Доля событий product_page  за 14-дневный период от событий product_page за весь период:  98.0 %

Количество событий product_page  в группе B всего: 970
Количество событий product_page  в группе B за 14 дней: 921
Доля событий product_page  за 14-дневный период от событий product_page за весь период:  94.95 %



Количество событий purchase  в группе A всего: 1854
Количество событий purchase  в группе A за 14 дней: 1827
Доля событий purchase  за 14-дневный период от событий purchase за весь период:  98.54 %

Количество событий purchase  в группе B всего: 494
Количество событий purchase  в группе B за 14 дней: 467
Доля событий purchase  за 14-дневный период от событий purchase за весь период:  94.53 %



Количество событий login  в группе A всего: 6083
Количество событий login  в группе A за 14 дней: 5963
Доля событий login  за 14-дневный период от событий login за весь период:  98.03 %

Количество событий login  в группе B всего: 1823
Количество событий login  в группе B за 14 дней: 1731
Доля событий login  за 14-дневный период от событий login за весь период:  94.95 %



Количество событий product_cart  в группе A всего: 1853
Количество событий product_cart  в группе A за 14 дней: 1826
Доля событий product_cart  за 14-дневный период от событий product_cart за весь период:  98.54 %

Количество событий product_cart  в группе B всего: 497
Количество событий product_cart  в группе B за 14 дней: 479
Доля событий product_cart  за 14-дневный период от событий product_cart за весь период:  96.38 %
```

- Не больше 2% пользователей для группы A и не больше 6% пользователей группы B остаются за бортом 14-дневного периода.
- Конверсию для этапов будем определять только для активностей, не старше 14 дней.

Общая конверсия каждого этапа

Посмотрим на количество событий каждого типа в группах, рассчитаем общую конверсию, конверсию в шаг.

Ввод [51]:

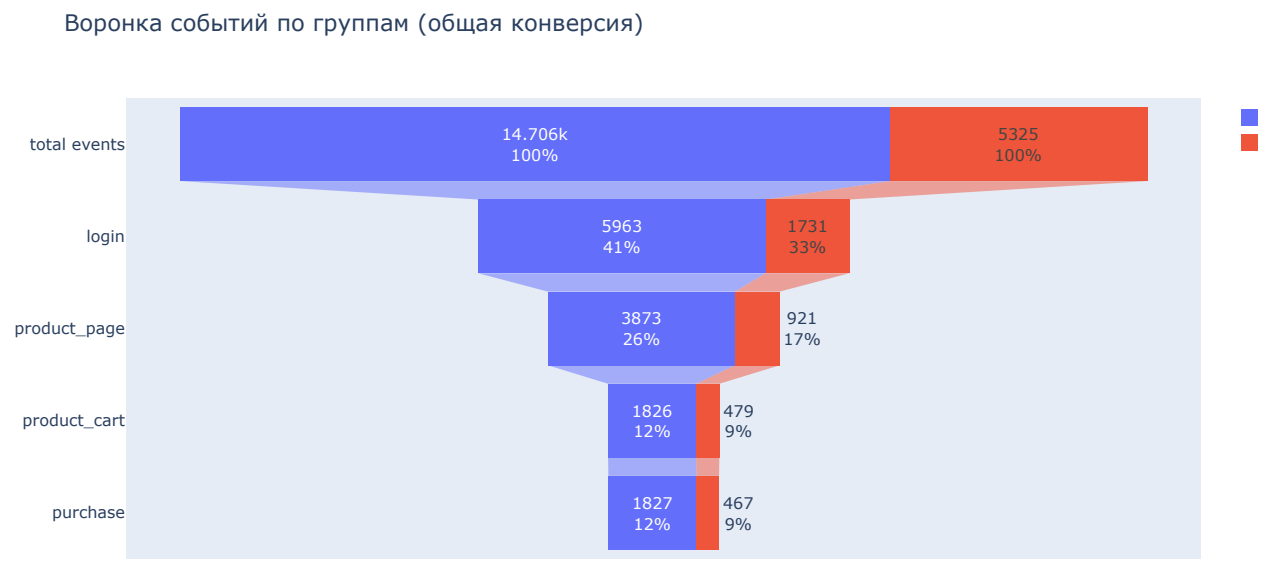
```
ratio_a =  complete_user_14[(complete_user_14['event_name']!='inactive') & (complete_user_14['group']=='A')].pivot_table(index = ['ev
                                                                                                                          values= ['user_id'], aggfunc= {'count'}
ratio_a.columns= ['stage', 'group','event_count' ]
ratio_a.loc[len(ratio_a.index)] = ['total events', 'A',  complete_user[ complete_user['group']=='A']['user_id'].count() ]

ratio_a['conversion'] = round(100* ratio_a['event_count']/ complete_user[ complete_user['group']=='A']['user_id'].count()   ,2)
ratio_a = ratio_a.sort_values(by='conversion', ascending = False).reset_index(drop= True)
ratio_a['conversion_per_step'] = round(100* ratio_a['event_count']/(ratio_a.shift(1,
                                                fill_value=complete_user[ complete_user['group']=='A']['user_id'].count())['event

ratio_b =  complete_user_14[(complete_user_14['event_name']!='inactive') & (complete_user_14['group']=='B')].pivot_table(index = ['ev
                                                                                                                        values= ['user_id'], aggfunc= {'count'}).reset_ind
ratio_b.columns= ['stage', 'group','event_count' ]
ratio_b.loc[len(ratio_a.index)] = ['total events', 'B',  complete_user[ complete_user['group']=='B']['user_id'].count() ]

ratio_b['conversion'] = round(100* ratio_b['event_count']/ complete_user[ complete_user['group']=='B']['user_id'].count()   ,2)
ratio_b = ratio_b.sort_values(by='conversion', ascending = False).reset_index(drop= True)
ratio_b['conversion_per_step'] = round(100* ratio_b['event_count']/(ratio_b.shift(1,
                                                fill_value=complete_user[ complete_user['group']=='B']['user_id'].count())['event




ratio_ab = pd.concat([ratio_a,ratio_b],axis=0 , sort= False)
ratio_ab = ratio_ab.reset_index(drop= True)

# определим порядок стадий

for i  in ratio_ab.index :
    if    ratio_ab.loc[i,'stage'] == 'total events':
          ratio_ab.loc[i,'stage_n'] = 1

    elif  ratio_ab.loc[i,'stage'] == 'login':
          ratio_ab.loc[i,'stage_n'] = 2

    elif  ratio_ab.loc[i,'stage'] == 'product_page':
          ratio_ab.loc[i,'stage_n'] = 3

    elif  ratio_ab.loc[i,'stage'] == 'product_cart':
          ratio_ab.loc[i,'stage_n'] = 4

    elif  ratio_ab.loc[i,'stage'] == 'purchase':
          ratio_ab.loc[i,'stage_n'] = 5

ratio_ab = ratio_ab.sort_values(by = 'stage_n').reset_index(drop= True)
ratio_ab
```

Out[51]:

	stage	group	event_count	conversion	conversion_per_step	stage_n
0	total events	A	14706	100.00	100.00	1.0
1	total events	B	5325	100.00	100.00	1.0
2	login	A	5963	40.55	40.55	2.0
3	login	B	1731	32.51	32.51	2.0
4	product_page	A	3873	26.34	64.95	3.0
5	product_page	B	921	17.30	53.21	3.0
6	product_cart	A	1826	12.42	47.15	4.0
7	product_cart	B	479	9.00	52.01	4.0
8	purchase	A	1827	12.42	100.05	5.0
9	purchase	B	467	8.77	97.49	5.0

Ввод [52]:

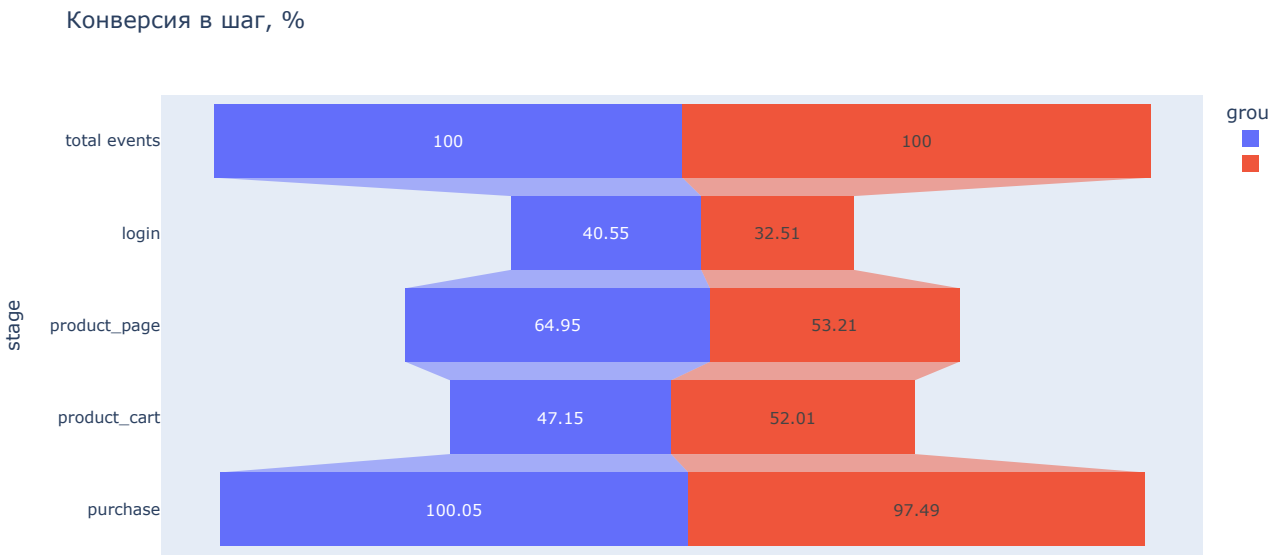
```
fig = go.Figure()
for group in ['A', 'B']:
    funnel = ratio_ab.query('group ==@group')
    fig.add_trace(go.Funnel(name = group, y = funnel['stage'], x = funnel['event_count'],
                            orientation = "h", textinfo = "value+percent initial"))
    fig.update_layout(title='Воронка событий по группам (общая конверсия)')
fig.show()
```



Определим конверсию в шаг

Ввод [53]:

```
fig = px.funnel(ratio_ab, x= 'conversion_per_step', y='stage', color='group',
                title='Конверсия в шаг, %')

fig.show()
```



Проверка ожидаемого эффекта от теста: 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` .

Ввод [54]:

```
print()
delta_page = round(100* (ratio_ab.loc[5,'conversion'] -ratio_ab.loc[4,'conversion']  )/ (ratio_ab.loc[4,'conversion'])  ,2)
print('Прирост конверсии в просмотр карточек товаров  (B-A)/A, %:', delta_page )

print()
delta_cart = round(100* (ratio_ab.loc[7,'conversion'] -ratio_ab.loc[6,'conversion']  )/ (ratio_ab.loc[6,'conversion'])  ,2)
print('Прирост конверсии в просмотр корзины (B-A)/A   , %:', delta_cart)

print()
delta_purchase = round(100* (ratio_ab.loc[9,'conversion'] -ratio_ab.loc[8,'conversion']  )/ (ratio_ab.loc[9,'conversion'])  ,2)
print('Прирост конверсии в покупки (B-A)/A   , %:', delta_purchase)

```

```
Прирост конверсии в просмотр карточек товаров  (B-A)/A, %: -34.32

Прирост конверсии в просмотр корзины (B-A)/A   , %: -27.54

Прирост конверсии в покупки (B-A)/A   , %: -41.62
```

1. Очень слабая конверсия в обеих группах даже на этапе `login` - 41% для группы A и всего треть для группы B.
2. Не менее слабая конверсия пользователей в покупателей, до этапа оплаты `purchase` доходит всего 9% пользователей группы B. Контрольная группа показывает немного лучший результат -12%.
3. Заметно почти одинаковое количество событий на этапах корзины и покупки- здесь проблем не наблюдается. В некоторых случаях покупка совершается помимо корзины (в группе A покупок 1827, а действий с корзиной 1826).
4. Во всех случаях общей конверсии группа B обнаруживает результат сильно хуже контрольной группы.
5. У группы B конверсия в шаг в просмотр корзины лучше (незначительно), чем у контрольной группы. Во всех остальных случаях, конверсия в шаг у группы B хуже, чем у контрольной.

4 Оценка результатов A/B тестирования

4.1 Проверка статистической разницы долей

Подготовим списки с общим количеством уникальных пользователей и количеством пользователей для каждого события

Ввод [55]:

```
 # Создадим  сводную таблицу с количеством пользователей в каждом событии для каждой группы
user_counts = (
    complete_user_14[complete_user_14['event_name']!='inactive']
    .groupby(['group', 'event_name']).agg({'user_id':'nunique'})
    .pivot_table(index='event_name', columns='group', values='user_id')
    .reset_index()
    .sort_values('A', ascending=False)
    .reset_index(drop =True)
)
print('Количество пользователей для каждого события в каждой группе:')
display(user_counts)
print()


# Расчитаем общее количество пользователей в каждой группе
total_user = complete_user_14.groupby('group').agg({'user_id':'nunique'}).reset_index()
print('Общее количество пользователей в каждой группе:')
display(total_user)
print()

# Подготовим списки с количеством пользователей для каждого события в каждой группе

print('Списки с количеством пользователей для каждого события в каждой группе:')
print()

# Контрольная  группа A:
A_list = [total_user.iloc[0]['user_id'], user_counts.iloc[0]['A'], user_counts.iloc[1]['A'], user_counts.iloc[2]['A'], user_counts.il
print('A_list :', A_list)
print()

# Экспериментальная группа B:
B_list = [total_user.iloc[1]['user_id'], user_counts.iloc[0]['B'], user_counts.iloc[1]['B'], user_counts.iloc[2]['B'], user_counts.il
print('B_list :', B_list)
print()

```

Количество пользователей для каждого события в каждой группе:

group	event_name	A	B
0	login	1939	654
1	product_page	1265	367
2	purchase	613	191
3	product_cart	589	184

Общее количество пользователей в каждой группе:

group	user_id
0	A	2903
1	B	2196

Списки с количеством пользователей для каждого события в каждой группе:

A_list : [2903, 1939, 1265, 613, 589]

B_list : [2196, 654, 367, 191, 184]

Подготовим функцию расчёта статистически значимой разницы между долями двух генеральных совокупностей

Ввод [56]:

```
# Функция для расчёта статистически значимой разницы между долями двух генеральных совокупностей
def find_stat_value(first_list, second_list, alpha, compare_count ):

    # Список с названием события в приложении
    event_name_list = ['login','product_page','product_cart','purchase'  ]
    # Основной цикл функции, который вычисляет p-value и проверяет нулевую гипотезу для каждого события
    for x in range(0, len(first_list)-1):
        x1 = first_list[x+1]  # Второе значение в первой группе
        x2 = second_list[x+1] # Второе значение во второй группе
        y1 = first_list[x]    # Первое значение в первой группе
        y2 = second_list[x]   # Первое значение во второй группе

        # alpha = 0.05 по умолчанию -   Указание критического уровня статистической значимости

        # поправка Бонферрони на множественное сравнение

        bonferroni_alpha = alpha/ compare_count

        successes = np.array([x1, x2])
        trials = np.array([y1, y2])
        p1 = successes[0]/trials[0] # Расчёт значения пропорции в первой группе
        p2 = successes[1]/trials[1] # Расчёт значения пропорции во второй группе
        # Расчёт совокупной пропорции в комбинированном датасете
        p_combined = (successes[0] + successes[1]) / (trials[0] + trials[1])
        difference = p1 - p2  # Расчёт разницы пропорций между группами
        # Расчёт статистики в ст.отклонениях стандартного нормального распределения
        z_value = difference / mth.sqrt(p_combined * (1 - p_combined) * (1/trials[0] + 1/trials[1]))
        distr = st.norm(0, 1) # Указание стандартного отклонения в нормальном распределении
        p_value = 1 - distr.cdf(z_value) # Расчёт значения статистической разницы между группами
        # Вывод полученных результатов на экран
        print('* Событие:',  event_name_list[x])
        print('Количество пользователей группы A для данного события', x1 )
        print('Количество пользователей группы A для предыдущего события', y1 )

        print('Количество пользователей группы B для данного события', x2 )
        print('Количество пользователей группы B для предыдущего события', y2 )
        print('* p-значение: ', p_value)

        print('* alpha: ', bonferroni_alpha)
        # Сравнение полученного p-value с установленным уровнем статистической значимости
        if (p_value < bonferroni_alpha):
            print("* Результат:", "Нулевая гипотеза отвергнута: между долями есть значимая разница")
        else:
            print("* Результат:", "Нулевая гипотеза не отвергнута: нет оснований считать доли разными")
        print('   ')
```

Зададим нулевые и альтернативные гипотезы и проведем проверку статзначимости различий.

- Нулевая гипотеза сформулирована следующим образом: **Доли групп "A" и "B" не отличаются друг от друга**
- Альтернативная гипотеза сформулирована следующим образом: **Доли групп "A" и "B" отличаются друг от друга**

В качестве границы критического уровня статистической значимости **alpha** принимаем **5%**.

Ввод [57]:

```
find_stat_value(A_list, B_list,   alpha = 0.05, compare_count =4 )
```

```
* Событие: login
Количество пользователей группы A для данного события 1939
Количество пользователей группы A для предыдущего события 2903
Количество пользователей группы B для данного события 654
Количество пользователей группы B для предыдущего события 2196
* p-значение:  0.0
* alpha:  0.0125
* Результат: Нулевая гипотеза отвергнута: между долями есть значимая разница

* Событие: product_page
Количество пользователей группы A для данного события 1265
Количество пользователей группы A для предыдущего события 1939
Количество пользователей группы B для данного события 367
Количество пользователей группы B для предыдущего события 654
* p-значение:  1.4731955949809361e-05
* alpha:  0.0125
* Результат: Нулевая гипотеза отвергнута: между долями есть значимая разница

* Событие: product_cart
Количество пользователей группы A для данного события 613
Количество пользователей группы A для предыдущего события 1265
Количество пользователей группы B для данного события 191
Количество пользователей группы B для предыдущего события 367
* p-значение:  0.8867594746207059
* alpha:  0.0125
* Результат: Нулевая гипотеза не отвергнута: нет оснований считать доли разными

* Событие: purchase
Количество пользователей группы A для данного события 589
Количество пользователей группы A для предыдущего события 613
Количество пользователей группы B для данного события 184
Количество пользователей группы B для предыдущего события 191
* p-значение:  0.5623175570409542
* alpha:  0.0125
* Результат: Нулевая гипотеза не отвергнута: нет оснований считать доли разными
```

Несмотря на различное количество пользователей на первых шагах (Login и product_page), к этапам просмотра корзины и покупки приходит пропорционально идентичное количество пользователей, а количественные отличия объясняются изначально неравными выборками.

4.2 Особенности подготовки к A/B тестированию, комментарии по результатам A/B тестирования

Для корректного проведения A/B теста необходимо:

- Проверить, что на результаты теста не повлияют другие тесты, марекитинговые кампании и прочие внешние события, влияющие прямо или косвенно. **!!!** В нашем случае тест проводился после ноябрьской распродажи и, частично, во время рождественских распродаж, к тому же параллельно с другим тестом, что прямо или косвенно влияло на результат как группы A, так и группы B - пользователи были набраны до начала распродаж, но они могли проявлять не учтенные системой посторонние активности, да и кошелек у покупателей не бездонный и нельзя не учитывать ограниченность возможностей покупателей при приобретении товаров.
- Проверить, что система разделения трафика работает корректно. **!!!** Значительная часть пользователей оказалась в обеих группах, что говорит о некорректной работе по сплитованию трафика.
- Предусмотреть вторую контрольную группу для проверки системы разделения трафика и корректности системы сбора данных. **!!!** Такой группы нет, есть данные A/B тестирования, а желательно A/A/B.
- Проверить обеспечение необходимого количества пользователей **!!!** Из необходимых по ТЗ 6000 пользователей, пригодными к анализу оказались меньше 4800.
- Проверить, что длительность проведенения теста достаточно **!!!** По этому параметру A/B тест корректный - большинство пользователей проявляет активность в заданный ТЗ 14-дневный период.

5 Общий вывод по результатам A/B тестирования

1. A/B тестирование проведено некорректно. Следует провести тест повторно, так как результатам доверять нет возможости.
2. Выявленные проблемы:
    - Тест проводился после ноябрьской распродажи и, частично, во время рождественских распродаж, к тому же параллельно с другим тестом, что прямо или косвенно влияло на результат как группы A, так и группы B.
    - Значительная часть пользователей оказалась в обеих группах, что говорит о некорректной работе по сплитованию трафика.
    - Не предусмотрена вторая контрольная группа для проверки системы разделения трафика и корректности системы сбора данных.
    - Из необходимых по ТЗ 6000 пользователей, пригодными к анализу оказались меньше 4800, также меньшей необходимой по ТЗ оказалась доля пользователей из региона EU.
    - Ожидаемый эффект от введения рекомендательной системы (за 14 дней с момента регистрации пользователи покажут улучшение каждой метрики не менее, чем на 10%) оказался недостигнут, - конверсия группы B хуже, чем A, но из-за неравномерности распределеления трафика сделать однозначные выводы нельзя, проверка не показывает статзначимости отличия на всех шагах воронки.