План исследования:

Проект по A/B-тестированию

Цель — провести оценку результатов A/B-теста.Оценить корректность проведения теста, проанализировать результаты теста.

Структура 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 — все пользователи, зарегистрировавшиеся в интернет-магазине в период с 7 по 21 декабря 2020 года.
Структура файла:

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

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

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

final_ab_participants.csv — таблица участников тестов.
Структура файла:

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

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

1 Проверка и предобработка данных

Импорт библиотек и загрузка данных:

Ввод [1]:

```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from scipy import stats as st
import math as mth
import plotly.express as px
from plotly import graph_objects as go
```

Ввод [2]:

```
final_ab_events = pd.read_csv('https://code.s3.yandex.net/datasets/final_ab_events.csv')
```

Ввод [3]:

```
ab_project_marketing_events = pd.read_csv('https://code.s3.yandex.net/datasets/ab_project_marketing_events.csv')
```

Ввод [4]:

```
final_ab_new_users= pd.read_csv('https://code.s3.yandex.net/datasets/final_ab_new_users.csv')
```

Ввод [5]:

```
final_ab_participants = pd.read_csv('https://code.s3.yandex.net/datasets/final_ab_participants.csv')
```

Создадим функцию для вывода общей информации по датафрейму:первые строки, размер, типы данных и количество пропусков в столбцах, наименования столбцов отдельно:

Ввод [6]:

```
def describe(df: pd.DataFrame):
    display(df.head())
    display(f'Общая информация: {df.shape}')
    display(df.info())
    print('Дубликаты в массиве:', df.duplicated().sum(), 'в процентах:', round(df.duplicated().mean()*100, 2))
    display(f'Названия столбцов: {df.columns}')
```

Ввод [7]:

```
describe(final_ab_events)
```

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

```
'Общая информация: (440317, 4)'

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

None

Дубликаты в массиве: 0 в процентах: 0.0

"Названия столбцов: Index(['user_id', 'event_dt', 'event_name', 'details'], dtype='object')"
```

Ввод [8]:

```
describe(ab_project_marketing_events)
```

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

```
'Общая информация: (14, 4)'

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

None

Дубликаты в массиве: 0 в процентах: 0.0

"Названия столбцов: Index(['name', 'regions', 'start_dt', 'finish_dt'], dtype='object')"
```

Ввод [9]:

```
describe(final_ab_new_users)
```

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

```
'Общая информация: (61733, 4)'

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

None

Дубликаты в массиве: 0 в процентах: 0.0

"Названия столбцов: Index(['user_id', 'first_date', 'region', 'device'], dtype='object')"
```


Ввод [10]:

```
describe(final_ab_participants)
```

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

```
'Общая информация: (18268, 3)'

<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

Дубликаты в массиве: 0 в процентах: 0.0

"Названия столбцов: Index(['user_id', 'group', 'ab_test'], dtype='object')"
```

Полных дубликатов в массивах нет.

Пропуски в данных есть только по столбцу details массива final_ab_events — дополнительные данные о событии. Видимо не по всем событиям есть дополнительные данные:

Ввод [11]:

```
final_ab_events.groupby('event_name')['details'].unique()
```

Out[11]:

```
event_name
login                                [nan]
product_cart                         [nan]
product_page                         [nan]
purchase       [99.99, 9.99, 4.99, 499.99]
Name: details, dtype: object
```

В деталях содержатся данные по цене покупок, по остальным событиям - пропуски.

Названия всех столбцов корректны. Типы данных по столбцам с датами и временем поменяем:

Ввод [12]:

```
final_ab_events['event_dt'] = pd.to_datetime(final_ab_events['event_dt'])
```

Ввод [13]:

```
ab_project_marketing_events['start_dt'] = pd.to_datetime(ab_project_marketing_events['start_dt'])
```

Ввод [14]:

```
ab_project_marketing_events['finish_dt'] = pd.to_datetime(ab_project_marketing_events['finish_dt'])
```

Ввод [15]:

```
final_ab_new_users['first_date'] = pd.to_datetime(final_ab_new_users['first_date'])
```

Вывод:

- в массивах нет существенных пропусков и строк-дубликатов. Пропуски в данных есть только по столбцу details массива final_ab_events - он содержит данные только по цене покупок. На наше исследование это не повлияет;
- названия всех столбцов корректны;
- типы данных по столбцам с датами и временем изменены для удобства работы с ними.

2 Оценка корректности проведения теста

2.1 Проверка данных на соответствие техническому заданию

2.1.1 Соответствие временных интервалов

Проверим соответствие дат регистрации пользователей: "пользователи, зарегистрировавшиеся с 7 по 21 декабря 2020 года".

Объединим новых пользователей с участниками теста по id участников теста, чтобы проверить данные по региону:

Ввод [16]:

```
final_ab = final_ab_participants.merge(final_ab_new_users, on='user_id', how='left')
final_ab.shape
```

Out[16]:

```
(18268, 6)
```

Ввод [17]:

```
final_ab_new_users1 = final_ab.query('ab_test == "recommender_system_test"')
```

Ввод [18]:

```
min = final_ab_new_users1['first_date'].dt.date.min()
display('Минимальная дата регистрации пользователей {}'.format(min))
```

```
'Минимальная дата регистрации пользователей 2020-12-07'
```

Ввод [19]:

```
max = final_ab_new_users1['first_date'].dt.date.max()
display('Максимальная дата регистрации пользователей {}'.format(max))
```

```
'Максимальная дата регистрации пользователей 2020-12-21'
```

Даты регистрации пользователей соответствуют ТЗ.

final_ab_events — действия новых пользователей в период с 7 декабря 2020 по 4 января 2021 года. Проверим:

Ввод [20]:

```
min = final_ab_events['event_dt'].dt.date.min()
display('Минимальная дата действий пользователей {}'.format(min))
```

```
'Минимальная дата действий пользователей 2020-12-07'
```

Ввод [21]:

```
max = final_ab_events['event_dt'].dt.date.max()
display('Максимальная дата действий пользователей {}'.format(max))
```

```
'Максимальная дата действий пользователей 2020-12-30'
```

Даты укладываются в интервал, приведенный в ТЗ. Правда, предполагалось, что действия совершаются до 4 января, а по факту нет информации после 30 декабря. Похоже тест завершили на 5 дней раньше.

ab_project_marketing_events — календарь маркетинговых событий на 2020 год

Ввод [22]:

```
min = ab_project_marketing_events['start_dt'].dt.date.min()
display('Минимальная дата маркетинговых событий {}'.format(min))
```

```
'Минимальная дата маркетинговых событий 2020-01-25'
```

Ввод [23]:

```
max = ab_project_marketing_events['finish_dt'].dt.date.max()
display('Максимальная дата маркетинговых событий {}'.format(max))
```

```
'Максимальная дата маркетинговых событий 2021-01-07'
```

Видим, что маркетинговые события перешли на начало 2021-го.

2.1.2 Соответствие данных по тесту и его участникам

- название теста: `recommender_system_test` ;

- группы: A — контрольная, B — новая платёжная воронка.

Количество участников тестов:

Ввод [24]:

```
final_ab_participants['user_id'].nunique()
```

Out[24]:

```
16666
```

Участники по группам:

Ввод [25]:

```
final_ab_participants['group'].value_counts()
```

Out[25]:

```
A    9655
B    8613
Name: group, dtype: int64
```

Участники по тестам:

Ввод [26]:

```
final_ab_participants['ab_test'].value_counts()
```

Out[26]:

```
interface_eu_test          11567
recommender_system_test     6701
Name: ab_test, dtype: int64
```

Видим, что есть данные по 2 тестам: interface_eu_test и recommender_system_test.
Посчитаем количество участников нашего теста по группам:

Ввод [27]:

```
final_ab_participants.query('ab_test == "recommender_system_test"')['group'].value_counts()
```

Out[27]:

```
A    3824
B    2877
Name: group, dtype: int64
```

Проверим пересекаемость участников в группах нашего теста:

Ввод [28]:

```
users = final_ab_participants.query('ab_test == "recommender_system_test"').groupby('user_id').agg({'group' : 'nunique'}).query('group>1')
users
```

Out[28]:

user_id	group

Нет пользователей, которые одновременно находятся в группах A и B теста recommender_system_test.

Ввод [29]:

```
a = final_ab_participants.query('ab_test == "recommender_system_test" and group=="A"')['user_id'].nunique()
b = final_ab_participants.query('ab_test == "recommender_system_test" and group=="B"')['user_id'].nunique()
display('Группа A: {} Группа B: {}'.format(a, b))
```

```
'Группа A: 3824 Группа B: 2877'
```

Проверим пересекаемость пользователей по тестам:

Ввод [30]:

```
users_t = final_ab_participants.groupby('user_id').agg({'ab_test' : 'nunique'}).query('ab_test>1').reset_index()
users_t = users_t['user_id'].to_list()
len(users_t)
```

Out[30]:

```
1602
```

Проверим другим способом:

Ввод [31]:

```
inter = final_ab_participants.query('ab_test == "interface_eu_test"')['user_id'].unique()
```

Ввод [32]:

```
cross_test_rec= final_ab_participants.query('ab_test == "recommender_system_test" and user_id in @inter')['user_id'].nunique()
cross_test_rec
```

Out[32]:

```
1602
```

1602 пользователя участвуют одновременно в 2-х тестах. Некорректно проводить тест при таких исходных данных. Проверим далее, как распределены пользователи теста interface_eu_test по группам нашего теста. Если равномерно, то их влияние будет так же равномерно распределено на обе группы.

Ввод [33]:

```
cross_test_a = final_ab_participants.query('group=="A" and user_id in @users_t')['user_id'].nunique()
cross_test_a
```

Out[33]:

```
1258
```

Ввод [34]:

```
cross_test_b = final_ab_participants.query('group=="B" and user_id in @users_t')['user_id'].nunique()
cross_test_b
```

Out[34]:

```
1120
```

Ввод [35]:

```
a1 = cross_test_a/a
b1 = cross_test_b/b
```

Ввод [36]:

```
display('Доля второго теста в A: {} Доля второго теста в B: {}'.format(round((cross_test_a/a), 3), round((cross_test_b/b), 3)))
```

```
'Доля второго теста в A: 0.329 Доля второго теста в B: 0.389'
```

33 % пользователей второго теста попадают в группу А и 39 % в группу В. Не слишком равномерное распределение. Но если убрать этих пользователей, в нашем тесте останется около 5000 участников.

2.1.3 Регион и общее количество участников

- аудитория: 15% новых пользователей из региона EU;
- ожидаемое количество участников теста: 6000.

Количество участников по тестам:

Ввод [37]:

```
final_ab['ab_test'].value_counts()
```

Out[37]:

```
interface_eu_test          11567
recommender_system_test     6701
Name: ab_test, dtype: int64
```

Общее количество участников нашего теста пока что больше 6000, что соответствует ТЗ. Уберем из массива строки второго теста:

Ввод [38]:

```
final_ab = final_ab.query('ab_test == "recommender_system_test"')
```

Проверим участников теста по принадлежности к региону:

Ввод [39]:

```
test_eu = final_ab.query('region == "EU"')['user_id'].nunique()
test_eu
```

Out[39]:

```
6351
```

Чуть больше, чем ожидалось в ТЗ.

Для расчета процента от общего количества узнаем, сколько новых участников (не учитывая их принадлежность к нашему тесту) из региона EU:

Ввод [40]:

```
total_eu = final_ab_new_users.query('region == "EU"')['user_id'].nunique()
total_eu
```

Out[40]:

```
46270
```

Ввод [41]:

```
perc_test_eu = round((test_eu*100/total_eu), 2)
perc_test_eu
```

Out[41]:

```
13.73
```

Только 13,7 % новых пользователей из региона EU участвуют в нашем тестировании.

2.1.4 Улучшение метрик за 14 дней

Ожидаемый эффект: за 14 дней с момента регистрации пользователи покажут улучшение каждой метрики не менее, чем на 10%

Присоединим данные по событиям (метрикам) и их времени:

Ввод [42]:

```
final_ab_ev = final_ab.merge(final_ab_events, on='user_id', how='left')
final_ab_ev.shape
```

Out[42]:

```
(27724, 9)
```

Строк стало больше, т.к. наши пользователи могли совершить каждое действие не по одному разу.

Посчитаем лайфтайм пользователей с момента регистрации и удалим события пользователей, совершенные более, чем через 14 дней с этого момента:

Ввод [43]:

```
final_ab_ev['life_event'] = (final_ab_ev['event_dt'] - final_ab_ev['first_date']).dt.days
final_ab_ev.head()
```

Out[43]:

	user_id	group	ab_test	first_date	region	device	event_dt	event_name	details	life_event
0	D1ABA3E2887B6A73	A	recommender_system_test	2020-12-07	EU	PC	2020-12-07 14:43:27	purchase	99.99	0.0
1	D1ABA3E2887B6A73	A	recommender_system_test	2020-12-07	EU	PC	2020-12-25 00:04:56	purchase	4.99	18.0
2	D1ABA3E2887B6A73	A	recommender_system_test	2020-12-07	EU	PC	2020-12-07 14:43:29	product_cart	NaN	0.0
3	D1ABA3E2887B6A73	A	recommender_system_test	2020-12-07	EU	PC	2020-12-25 00:04:57	product_cart	NaN	18.0
4	D1ABA3E2887B6A73	A	recommender_system_test	2020-12-07	EU	PC	2020-12-07 14:43:27	product_page	NaN	0.0

Ввод [44]:

```
final_ab_ev = final_ab_ev.query('life_event <= 14')
final_ab_ev
```

Out[44]:

	user_id	group	ab_test	first_date	region	device	event_dt	event_name	details	life_event
0	D1ABA3E2887B6A73	A	recommender_system_test	2020-12-07	EU	PC	2020-12-07 14:43:27	purchase	99.99	0.0
2	D1ABA3E2887B6A73	A	recommender_system_test	2020-12-07	EU	PC	2020-12-07 14:43:29	product_cart	NaN	0.0
4	D1ABA3E2887B6A73	A	recommender_system_test	2020-12-07	EU	PC	2020-12-07 14:43:27	product_page	NaN	0.0
6	D1ABA3E2887B6A73	A	recommender_system_test	2020-12-07	EU	PC	2020-12-07 14:43:27	login	NaN	0.0
8	A7A3664BD6242119	A	recommender_system_test	2020-12-20	EU	iPhone	2020-12-20 15:46:06	product_page	NaN	0.0
...	...	...	...	...	...	...	...	...	...	...
27719	6715343AFBA285AE	B	recommender_system_test	2020-12-07	CIS	Android	2020-12-07 10:12:15	login	NaN	0.0
27720	6715343AFBA285AE	B	recommender_system_test	2020-12-07	CIS	Android	2020-12-08 22:51:16	login	NaN	1.0
27721	6715343AFBA285AE	B	recommender_system_test	2020-12-07	CIS	Android	2020-12-09 02:28:03	login	NaN	2.0
27722	6715343AFBA285AE	B	recommender_system_test	2020-12-07	CIS	Android	2020-12-10 22:55:14	login	NaN	3.0
27723	6715343AFBA285AE	B	recommender_system_test	2020-12-07	CIS	Android	2020-12-14 12:40:41	login	NaN	7.0

24070 rows × 10 columns

Видим, что наш массив опять сократился

Ввод [45]:

```
final_ab_ev.query('region == "EU"').groupby('group')['user_id'].nunique()
```

Out[45]:

```
group
A    2604
B     877
Name: user_id, dtype: int64
```

После того, как мы убрали события пережившие 14 дней с момента регистрации, количество пользователей в группах существенно сократилось, особенно в группе B. Увеличилась разница в количестве участников групп.

Выделим пользователей групп теста в отдельные массивы.
Для проверки улучшения метрик построим воронки для групп A и B по цепочке: login - product_page - product_cart - purchase:

Ввод [46]:

```
gr_a = final_ab_ev.query('group == "A" and region == "EU"')
gr_a['user_id'].nunique()
```

Out[46]:

```
2604
```

Ввод [47]:

```
gr_b = final_ab_ev.query('group == "B" and region == "EU"')
gr_b['user_id'].nunique()
```

Out[47]:

```
877
```

Посчитаем уникальных пользователей группы A на первом шаге воронки - логине пользователей системы:

Ввод [48]:

```
funn_a = gr_a.query('event_name == "login"')['user_id']
fun_a1 = funn_a.nunique()
display('Количество пользователей шага login: {}'.format(fun_a1))
```

```
'Количество пользователей шага login: 2604'
```

Добавим пользователей первого шага в список, чтобы посчитать конверсию относительно него:

Ввод [49]:

```
a = funn_a.tolist()
```

Узнаем, сколько залогинившихся пользователей дошло до каждого шага будущей воронки:

Ввод [50]:

```
fun_a2 = gr_a.query('user_id in @a and event_name=="product_page"')['user_id'].nunique()
display('Количество пользователей шага product_page: {}'.format(fun_a2))
```

```
'Количество пользователей шага product_page: 1685'
```

Ввод [51]:

```
fun_a3 = gr_a.query('user_id in @a and event_name=="product_cart"')['user_id'].nunique()
display('Количество пользователей шага product_cart: {}'.format(fun_a3))
```

```
'Количество пользователей шага product_cart: 782'
```

Ввод [52]:

```
fun_a4 = gr_a.query('user_id in @a and event_name=="purchase"')['user_id'].nunique()
display('Количество пользователей шага purchase: {}'.format(fun_a4))
```

```
'Количество пользователей шага purchase: 833'
```

Те же расчеты проведем для группы B:

Ввод [53]:

```
funn_b = gr_b.query('event_name == "login"')['user_id']
fun_b1 = funn_b.nunique()
display('Количество пользователей шага login: {}'.format(fun_b1))
```

```
'Количество пользователей шага login: 876'
```


Ввод [54]:

```
b = funn_b.tolist()
```

Ввод [55]:

```
fun_b2 = gr_b.query('user_id in @b and event_name=="product_page"')['user_id'].nunique()
display('Количество пользователей шага product_page: {}'.format(fun_b2))
```

```
'Количество пользователей шага product_page: 493'
```

Ввод [56]:

```
fun_b3 = gr_b.query('user_id in @b and event_name=="product_cart"')['user_id'].nunique()
display('Количество пользователей шага product_cart: {}'.format(fun_b3))
```

```
'Количество пользователей шага product_cart: 244'
```

Ввод [57]:

```
fun_b4 = gr_b.query('user_id in @b and event_name=="purchase"')['user_id'].nunique()
display('Количество пользователей шага purchase: {}'.format(fun_b4))
```

```
'Количество пользователей шага purchase: 248'
```

Создадим таблицу для построения воронок по группам

Ввод [58]:

```
data = {'event_name': ['login', 'product_page', 'product_cart', 'purchase'], 'gr_a': [1754, 1128, 545, 577], 'gr_b': [659, 376, 194, 190]}
data = pd.DataFrame(data)
data
```

Out[58]:

	event_name	gr_a	gr_b
0	login	1754	659
1	product_page	1128	376
2	product_cart	545	194
3	purchase	577	190

Ввод [59]:

```
fig = go.Figure()

fig.add_trace(go.Funnel(
    name = 'группа A - контрольная',
    y = data['event_name'],
    x = data['gr_a'],
    textinfo = "value+percent initial+percent previous"))

fig.add_trace(go.Funnel(
    name = 'группа B - новая',
    orientation = "h",
    y = data['event_name'],
    x = data['gr_b'],
    textposition = "inside",
    textinfo = "value+percent initial+percent previous"))
fig.update_layout(
            title={
            'text' : 'Воронка группы A — контрольная, B — новая платёжная воронка',
            'x':0.50,
            'y':0.88,
            'xanchor': 'center'
        })

fig.show()
```

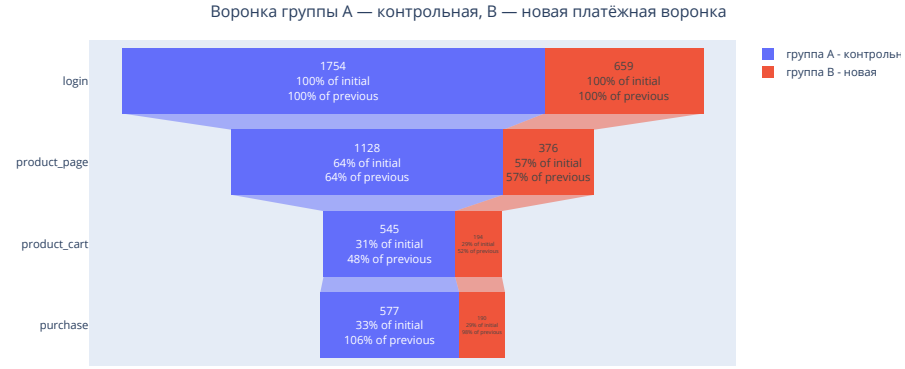
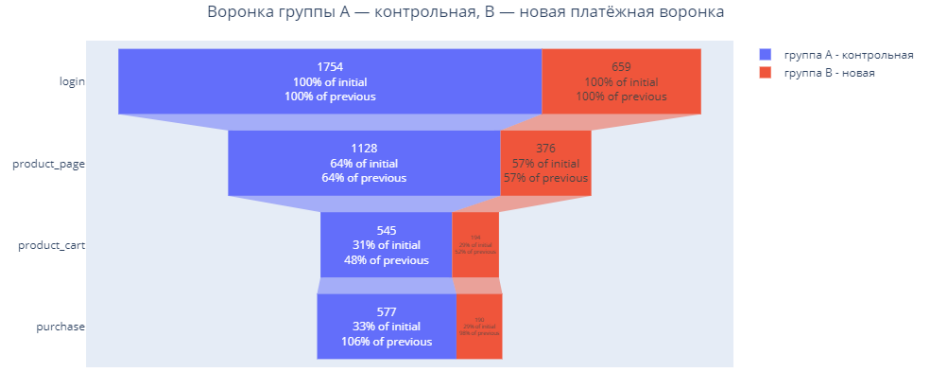


фото для графиков plotly:



Судя по данным созданной воронки - метрики новых пользователей за 14 дней с момента регистрации не только не улучшились на 10 %, но даже стали хуже на всех шагах после логина. Конверсия группы B на каждом шаге меньше, чем конверсия контрольной группы.
В группе A конверсия в покупки больше, чем в корзину. В группе B разница между конверсиями этих метрик тоже ничтожно мала. Видимо, пользователям удобнее покупать в 1 клик. На платформе нестрогая воронка продаж, и можно приобрести продукт, минуя некоторые этапы.

Вывод

- даты ab_project_marketing_events — календаря маркетинговых событий на 2020 год - переходят на 2021 год. Это совпадает с окончанием нашего теста. Необходимо будет проверить пересечение дат;
- есть данные по 2 тестам: interface_eu_test и recommender_system_test. 1602 пользователя участвуют в них одновременно. Доля второго теста в группе A recommender_system_test: 0.33. Доля второго теста в B recommender_system_test: 0.39. Доли участников второго теста в группах нашего теста не одинаковы, решено пока не исключать этих участников, чтобы не уменьшить значительно количество пользователей в тесте. И проверить остальные данные;
- всего 6351 пользователь из региона EU относится к тесту recommender_system_test, это 14% новых пользователей из региона EU. По ТЗ ожидалось 6000 и 15 %. Если убрать пользователей, участвующих в другом тесте, это количество существенно сократится;
- после того, как мы убрали события пережившие 14 дней с момента регистрации (по условиям ТЗ: "за 14 дней с момента регистрации пользователи покажут улучшение каждой метрики не менее, чем на 10%"), количество пользователей в группах существенно сократилось, особенно в группе B. Увеличилась разница в количестве участников групп. Пользователи распределены неравномерно, проверим это на этапе исследовательского анализа;
- судя по данным созданной воронки - метрики новых пользователей за 14 дней с момента регистрации не только не улучшились на 10 %, но даже стали хуже на всех шагах после логина. Конверсия группы B на каждом шаге меньше, чем конверсия контрольной группы.

2.2 Время проведения теста, пересечение с маркетинговыми активностями

Дата запуска теста: 2020-12-07; дата остановки: 2021-01-04. Проверим, попадают ли события календаря маркетинговых активностей на это время:

Ввод [60]:

```
ab_project_marketing_events.query('start_dt <= "2021-01-04" and start_dt >=" 2020-12-07"')
```

Out[60]:

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

Аж 2 промо компании попадают на даты нашего теста, что может существенно повлиять на активность пользователей в эти даты.
Кроме того - предпразничные и праздничные даты - не лучшее время для тестирования улучшения рекомендательной системы, т.к. нельзя будет определить в случае улучшений - связаны они с изменениями в системе или с праздничным повышением спроса. Это нужно учитывать при изучении абсолютных данных.

Рекомендуется перезапустить тест после праздников.

3 Исследовательский анализ данных

3.1 Равномерность распределения признаков по тестовым группам

Проверим распределение пользователей групп по регионам:

Ввод [61]:

```
gr_a1 = final_ab_ev.query('group == "A"')
gr_a1['region'].value_counts()
```

Out[61]:

```
EU           17977
N.America      689
APAC           160
CIS            121
Name: region, dtype: int64
```

Ввод [62]:

```
gr_b1 = final_ab_ev.query('group == "B"')
gr_b1['region'].value_counts()
```

Out[62]:

```
EU           4851
N.America     129
CIS            78
APAC           65
Name: region, dtype: int64
```

До очистки данных и приведения их в соответствие ТЗ пользователей из Европы в группе А было практически в 4 раза больше, чем в группе В. После всех изменений на этапе логина сохранилось примерно такое же соотношение пользователей групп.

Распределение по устройствам:

Ввод [63]:

```
a_dev = gr_a.groupby(['user_id'])['device'].count().reset_index()
```

Ввод [64]:

```
gr_a['device'].value_counts()
```

Out[64]:

```
Android    7758
PC         4724
iPhone     3672
Mac        1823
Name: device, dtype: int64
```

Ввод [65]:

```
b_dev = gr_b.groupby(['user_id'])['device'].count().reset_index()
```

Ввод [66]:

```
gr_b['device'].value_counts()
```

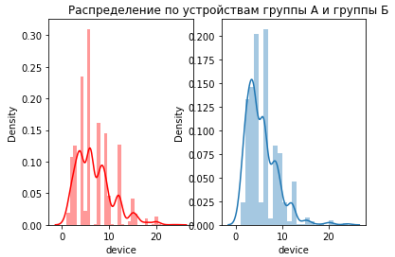
Out[66]:

```
Android    2236
PC         1122
iPhone     1101
Mac         392
Name: device, dtype: int64
```


Ввод [67]:

```
fig, axes = plt.subplots(1, 2)

sns.distplot(a_dev['device'], ax=axes[0], color = 'red')
sns.distplot(b_dev['device'], ax=axes[1])
plt.title('Распределение по устройствам группы А и группы Б', x = 0.05)
plt.show()
```

```
/opt/conda/lib/python3.9/site-packages/seaborn/distributions.py:2557: FutureWarning:

`distplot` is a deprecated function and will be removed in a future version. Please adapt your code to use either `displot`
(a figure-level function with similar flexibility) or `histplot` (an axes-level function for histograms).

/opt/conda/lib/python3.9/site-packages/seaborn/distributions.py:2557: FutureWarning:

`distplot` is a deprecated function and will be removed in a future version. Please adapt your code to use either `displot`
(a figure-level function with similar flexibility) or `histplot` (an axes-level function for histograms).
```



Видим неравномерность распределения данных по группам, разную плотность, максимумы. Последовательность популярных устройств при этом сохранена.

Равномерность распределения событий по пользователям групп:

Ввод [68]:

```
a_ev = gr_a.groupby(['user_id'])['event_name'].count().reset_index()
a_ev.describe()
```

Out[68]:

	event_name
count	2604.00000
mean	6.90361
std	3.84470
min	1.00000
25%	4.00000
50%	6.00000
75%	9.00000
max	24.00000

Ввод [69]:

```
b_ev = gr_b.groupby(['user_id'])['event_name'].count().reset_index()
b_ev.describe()
```

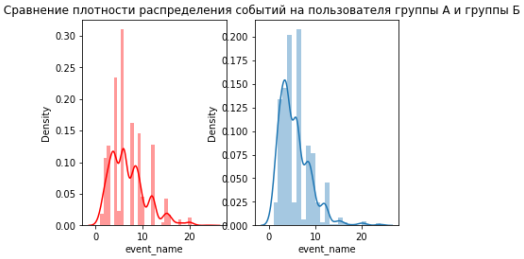
Out[69]:

	event_name
count	877.000000
mean	5.531357
std	3.314281
min	1.000000
25%	3.000000
50%	4.000000
75%	8.000000
max	24.000000

Ввод [70]:

```
fig, axes = plt.subplots(1, 2)

sns.distplot(a_ev['event_name'], ax=axes[0], color = 'red')
sns.distplot(b_ev['event_name'], ax=axes[1])
plt.title('Сравнение плотности распределения событий на пользователя группы А и группы Б', x = 0.05)
plt.show()
```

```
/opt/conda/lib/python3.9/site-packages/seaborn/distributions.py:2557: FutureWarning:

`distplot` is a deprecated function and will be removed in a future version. Please adapt your code to use either `displot`
(a figure-level function with similar flexibility) or `histplot` (an axes-level function for histograms).

/opt/conda/lib/python3.9/site-packages/seaborn/distributions.py:2557: FutureWarning:

`distplot` is a deprecated function and will be removed in a future version. Please adapt your code to use either `displot`
(a figure-level function with similar flexibility) or `histplot` (an axes-level function for histograms).
```



События распределены также, как и устройства. Это связано с самим первоначальным делением пользователей на группы.

Ввод [71]:

```
gr_a['event_name'].value_counts()
```

Out[71]:

```
login           7968
product_page    5125
purchase        2499
product_cart    2385
Name: event_name, dtype: int64
```

Ввод [72]:

```
gr_b['event_name'].value_counts()
```

Out[72]:

```
login           2345
product_page    1257
product_cart     625
purchase         624
Name: event_name, dtype: int64
```

3.2 Распределение событий по датам и группам

Ввод [73]:

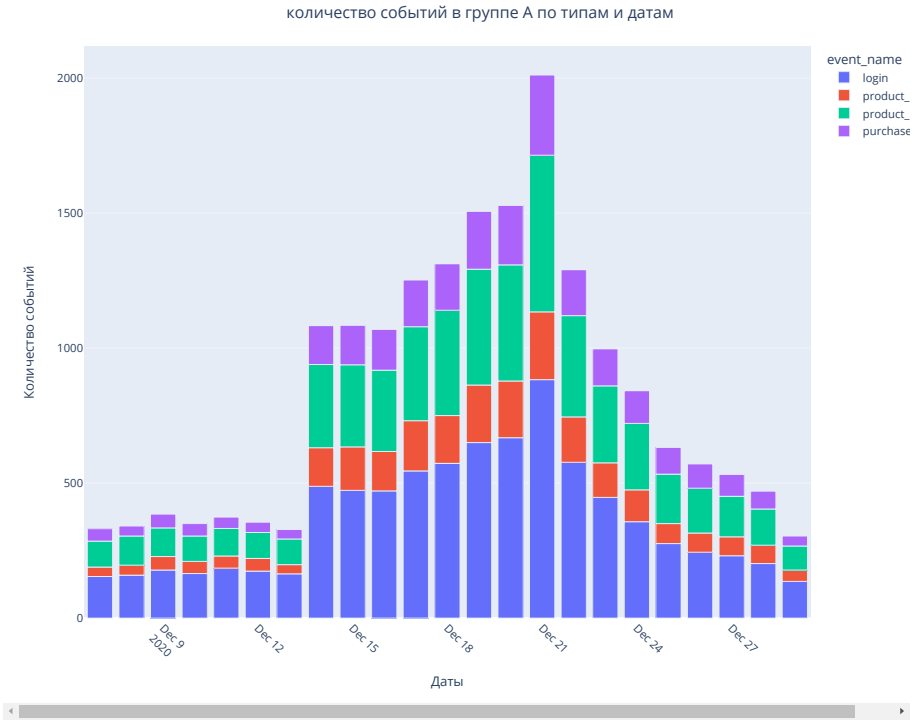
```
final_ab_ev['day'] = final_ab_ev['event_dt'].dt.date
```

Ввод [74]:

```
dateA = final_ab_ev.query('group == "A"').groupby(['day', 'event_name']).agg({'user_id':'nunique'}).reset_index()
dateB = final_ab_ev.query('group == "B"').groupby(['day', 'event_name']).agg({'user_id':'nunique'}).reset_index()
```

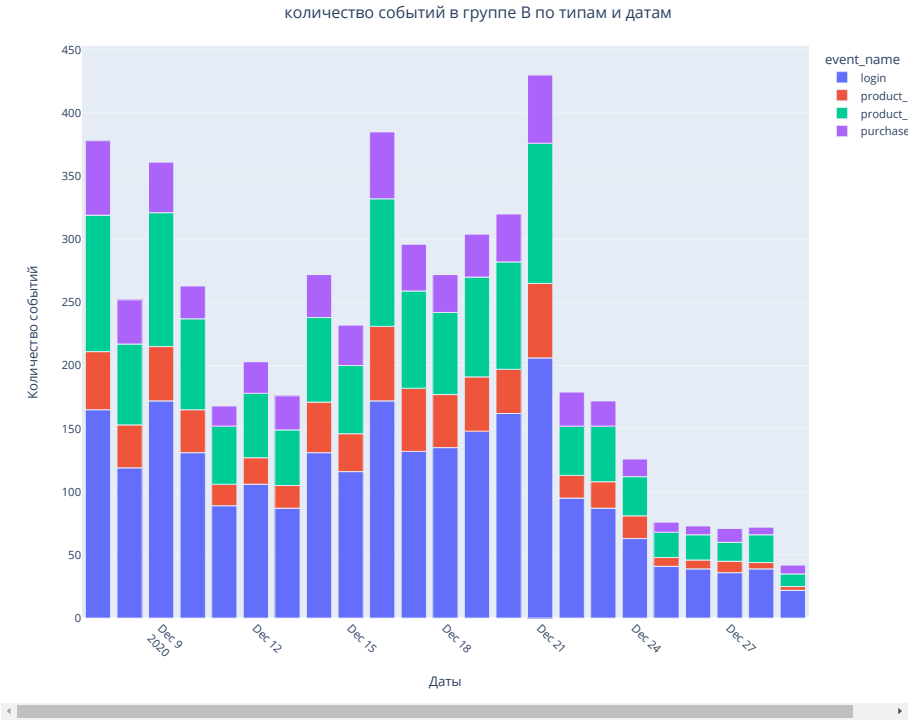

Ввод [75]:

```
fig = px.bar(dateA, x = 'day', y = 'user_id',
             color = 'event_name')
fig.update_layout(title="количество событий в группе А по типам и датам", title_x = 0.5, width=1000, height=750)
fig.update_xaxes(title_text='Даты', tickangle=45)
fig.update_yaxes(title_text='Количество событий')
fig.show()
```



Ввод [76]:

```
fig = px.bar(dateB, x = 'day', y = 'user_id',
             color = 'event_name')
fig.update_layout(title="количество событий в группе B по типам и датам", title_x = 0.5, width=1000, height=750)
fig.update_xaxes(title_text='Даты', tickangle=45)
fig.update_yaxes(title_text='Количество событий')
fig.show()
```



Проверим количество регистраций в группах по датам:

Ввод [77]:

```
final_ab_ev['first_date'] = final_ab_ev['first_date'].dt.date
```

Ввод [78]:

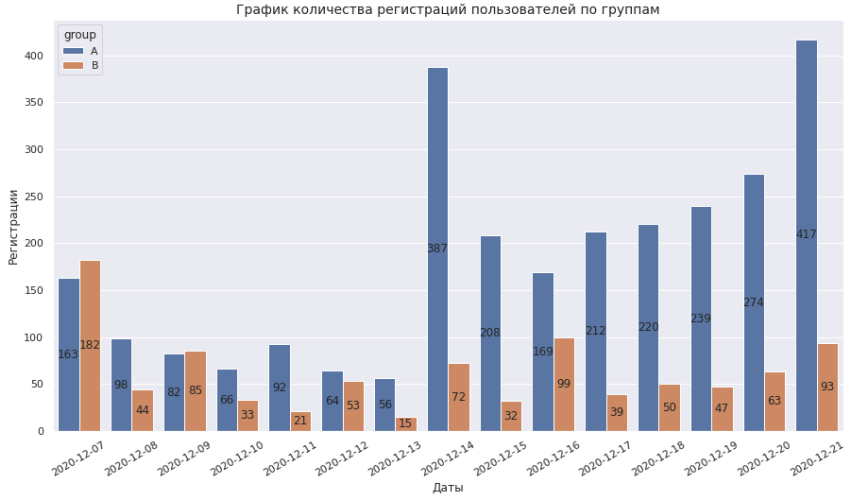
```
reg_ev = final_ab_ev.groupby(['first_date', 'group']).agg({'event_name': 'count'}).reset_index()
```

Ввод [79]:

```
reg_date = final_ab_ev.groupby(['first_date', 'group']).agg({'user_id': 'nunique'}).reset_index()
```

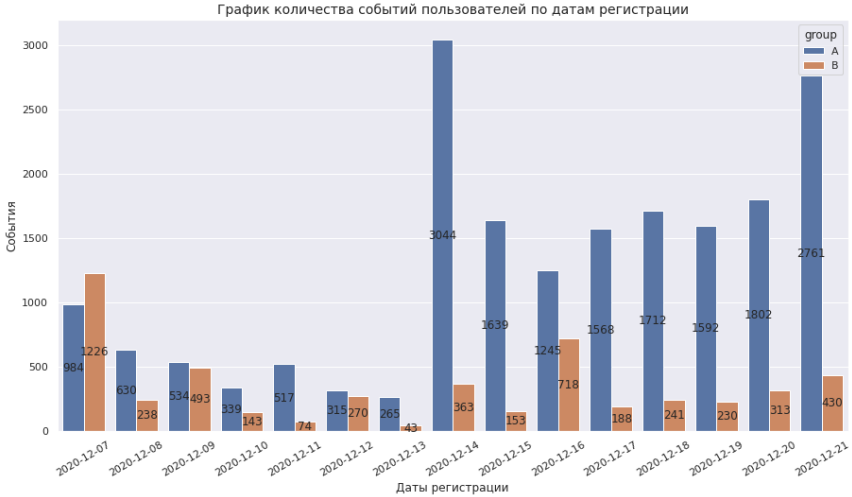

Ввод [80]:

```
sns.set(rc={'figure.figsize':(15, 8)})
sns.set_palette('deep')
ax = sns.barplot(x='first_date', y='user_id', hue='group', data=reg_date)
ax.set_title('График количества регистраций пользователей по группам', fontsize=14)
plt.xlabel('Даты')
plt.ylabel('Регистрации')
plt.xticks(rotation=30)
for p in ax.patches:
    width, height = p.get_width(), p.get_height()
    x, y = p.get_xy()
    ax.text(x+width/2,
    y+height/2,
    '{:.0f}'.format(height),
    horizontalalignment='center',
    verticalalignment='center')
plt.show()
```



Ввод [81]:

```
sns.set(rc={'figure.figsize':(15, 8)})
sns.set_palette('deep')
ax = sns.barplot(x='first_date', y='event_name', hue='group', data=reg_ev)
ax.set_title('График количества событий пользователей по датам регистрации', fontsize=14)
plt.xlabel('Даты регистрации')
plt.ylabel('События')
plt.xticks(rotation=30)
for p in ax.patches:
    width, height = p.get_width(), p.get_height()
    x, y = p.get_xy()
    ax.text(x+width/2,
    y+height/2,
    '{:.0f}'.format(height),
    horizontalalignment='center',
    verticalalignment='center')
plt.show()
```



Вывод: Типы событий по датам в обеих группах распределены пропорционально их количеству. А вот количество событий в группах по датам весьма неравномерно.
В группе А похоже были какие-то проблемы с привлечением, доступом пользователей в систему до 14.12, после чего наблюдается резкий всплеск активности пользователей.
В группе Б активность в первые дни 7-10 декабря идет волнами, но достаточна высока. Спад приходится на 11-13.12.
После 14.12 активность повышается опять. Возможно, были технические проблемы в районе 10-13.12 в обеих группах. В последние дни тестирования, начиная с 22.12 - активность пользователей снижается в обеих группах.

События в группе А 13.12 практически сошли на нет. И зарегистрировано всего 15 пользователей. Набор пользователей идет неравномерно, больше всего пользователей группы А набирается во второй половине временного интервала теста, когда пользователи уже не успеют прожить 14 дней в системе.

4 Оценка результатов A/B-тестирования

При проверке данных на соответствие ТЗ мы обнаружили, что внедрение улучшенной рекомендательной системы не привело к улучшению каждой метрики не менее, чем на 10% за 14 дней с момента регистрации пользователей:

- конверсия в product_page группы А 64 %, группы В 57 %;
- конверсия в product_cart группы А 31 %, группы В 29 %;
- конверсия в purchase группы А 33 %, группы В 29 %.
  Группы распределены недостаточно равномерно. Стоит перенабрать группы и запустить тест заново.

Убедимся в этом, проверив статистическую разницу долей z-критерием. Подготовим таблицу шагов для расчета:

Ввод [82]:

```
test = data.drop(index=0)
test
```

Out[82]:

	event_name	gr_a	gr_b
1	product_page	1128	376
2	product_cart	545	194
3	purchase	577	190

Проведем проверку гипотезы о равенстве долей пользователей, переходящих на следующий шаг воронки, при помощи Z-теста для группы А и В. Проверим, находят ли статистические критерии разницу между этими группами. Создадим функцию для проверки гипотезы о равенстве пропорций двух генеральных совокупностей по их выборкам. Нулевая гипотеза (h0): между долями нет значимой разницы, альтернативная: между долями есть значимая разница. Примем уровень значимости alpha = 0.05.

Ввод [83]:

```
def zval_test(gr1, gr2, alpha):
    for i in test.index:
        # пропорция успехов в первой группе:
        p1 = test[gr1][i] / fun_a1
        # пропорция успехов во второй группе:
        p2 = test[gr2][i] / fun_b1

        print(test[gr1][i], test[gr2][i], fun_a1, fun_b1)
        # пропорция успехов в комбинированном датасете:
        p_combined = ((test[gr1][i] + test[gr2][i]) /
                      (fun_a1 + fun_b1))
        # разница пропорций в датасетах
        difference = p1 - p2
        # считаем статистику в ст.отклонениях стандартного нормального распределения
        z_value = difference / mth.sqrt(p_combined * (1 - p_combined) *
                                        (1/fun_a1 + 1/fun_b1))
        # задаем стандартное нормальное распределение (среднее 0, ст.отклонение 1)
        distr = st.norm(0, 1)
        p_value = (1 - distr.cdf(abs(z_value))) * 2
        print('{} p-значение: {}'.format(test['event_name'][i], p_value))
        if (p_value < alpha):
            print("Отвергаем нулевую гипотезу: между долями есть значимая разница")
        else:
            print("Не получилось отвергнуть нулевую гипотезу, нет оснований считать доли разными")
        print('')
```

Несколько сравнений, проводимых на одних и тех же данных — это множественный тест. Его важная особенность в том, что с каждой новой проверкой гипотезы растёт вероятность ошибки первого рода. Поэтому используем поправку по методу Бонферрони: делим уровень значимости α на число сравнений групп = 3.

Ввод [84]:

```
zval_test('gr_a', 'gr_b', 0.05/3)
```

```
1128 376 2604 876
product_page p-значение: 0.8379975708523837
Не получилось отвергнуть нулевую гипотезу, нет оснований считать доли разными

545 194 2604 876
product_cart p-значение: 0.44622518636781816
Не получилось отвергнуть нулевую гипотезу, нет оснований считать доли разными

577 190 2604 876
purchase p-значение: 0.7721962898027184
Не получилось отвергнуть нулевую гипотезу, нет оснований считать доли разными
```

Значимой разницы в долях выборок не обнаружено.

5 Выводы

На этапе проверки и предобработки данных убедились, что в массивах нет существенных пропусков и строк-дубликатов, названия всех столбцов корректны. Типы данных по столбцам с датами и временем изменили для удобства работы с ними.

Проверка данных на соответствие техническому заданию показала:

- даты ab_project_marketing_events — календаря маркетинговых событий на 2020 год - переходят на 2021 год. Это совпадает с окончанием нашего теста;
- есть данные по 2 тестам: interface_eu_test и recommender_system_test. 1602 пользователя участвуют в них одновременно. Доля второго теста в группе А recommender_system_test: 0.33. Доля второго теста в В recommender_system_test: 0.39. Доли участников второго теста в группах нашего теста не одинаковы, решено не исключать этих участников, чтобы не уменьшить значительно количество пользователей в тесте;
- всего 6351 пользователь из региона EU относится к тесту recommender_system_test, это 14% новых пользователей из региона EU. По ТЗ ожидалось 6000 и 15 %. Если убрать пользователей, участвующих в другом тесте, это количество существенно сократится;

- после того, как мы убрали события пережившие 14 дней с момента регистрации (по условиям ТЗ: "за 14 дней с момента регистрации пользователи покажут улучшение каждой метрики не менее, чем на 10%"), количество пользователей в группах существенно сократилось, особенно в группе B. Увеличилась разница в количестве участников групп. Пользователи распределены неравномерно, проверим это на этапе исследовательского анализа;
- судя по данным созданной воронки - метрики новых пользователей за 14 дней с момента регистрации не только не улучшились на 10 %, но даже стали хуже на всех шагах после логина. Конверсия группы B на каждом шаге меньше, чем конверсия контрольной группы.

Аж 2 промо компании попадают на даты нашего теста, что может существенно повлиять на активность пользователей в эти даты и измерении абсолютных значений.
Кроме того - предпразничные и праздничные даты - не лучшее время для тестирования улучшения рекомендательной системы, т.к. нельзя будет определить в случае улучшений - связаны они с изменениями в системе или с праздничным повышением спроса.

До очистки данных и приведения их в соответствие ТЗ пользователей из Европы в группе A было практически в 4 раза больше, чем в группе B. После всех изменений на этапе логина сохранилось примерно такое же соотношение пользователей групп.
Также обнаружена некоторая неравномерность распределения устройств по группам, разная плотность, максимумы. Последовательность популярных устройств при этом сохранена.
События распределены также, как и устройства. Это связано с самим первоначальным делением пользователей на группы.

Типы событий по датам в обеих группах распределены пропорционально их количеству. А вот количество событий в группах по датам весьма неравномерно. В группе A похоже были какие-то проблемы с привлечением, доступом пользователей в систему до 14.12, после чего наблюдается резкий всплеск активности пользователей. В группе Б активность в первые дни 7-10 декабря идет волнами, но достаточна высока. Спад приходится на 11-13.12. После 14.12 активность повышается опять. Возможно, были технические проблемы в районе 10-13.12 в обеих группах. В последние дни тестирования, начиная с 22.12 - активность пользователей снижается в обеих группах.

События в группе A 13.12 практически сошли на нет. И зарегистрировано всего 15 пользователей. Набор пользователей идет неравномерно, больше всего пользователей группы A набирается во второй половине временного интервала теста, когда пользователи уже не успеют прожить 14 дней в системе.

При проверке данных на соответствие ТЗ мы обнаружили, что внедрение улучшенной рекомендательной системы не привело к улучшению каждой метрики не менее, чем на 10% за 14 дней с момента регистрации пользователей:

- конверсия в product_page группы A 64 %, группы B 57 %;
- конверсия в product_cart группы A 31 %, группы B 29 %;
- конверсия в purchase группы A 33 %, группы B 29 %.

Проведена проверка гипотезы о равенстве долей пользователей, переходящих на следующий шаг воронки, при помощи Z-теста для группы A и B. Проверено, находят ли статистические критерии разницу между этими группами. Нулевая гипотеза (h0): между долями групп A и B нет значимой разницы, альтернативная: между долями есть значимая разница.
Статистически значимой разницы между долями выборок не обнаружено.

Рекомендации:

- набрать больше пользователей с признаками нашей целевой аудитории;
- соблюсти равномерность распределения пользователей с разными признаками по группам теста;
- провести тест не одновременно с другими тестами и проверить пересечение пользователей в контрольной и тестовой группе;
- исключить наложение тестовых дат на праздничные даты, а также на даты промо-акций и др. маркетинговых активностей;
- провести тест заново, так как доверять результатам проведенного теста нельзя из-за вышеперечисленных недостатков.