Комментарий тимлида (1)

Привет! На связи твой тимлид Алексей. По ходу проекта я буду давать комментарии, если они понадобятся и выделять их цветом - зелёный - всё хорошо, оранжевый - допустимо, но можно доработать, красный - нужно переделать, зачесть нельзя. В скобках я буду ставить номер итерации для данного комментария.

Пожалуйста, сопровождай мои желтые и красные комментарии своими комментариями. Можешь писать

Комментарий джуниора ({iteration_number})

A/B-тестирование

Постановка задачи

Провести оценку результатов A/B-теста. В распоряжении есть датасет с действиями пользователей, техническое задание и несколько вспомогательных датасетов.

- Оценить корректность проведения теста
- Проанализировать результаты 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

- `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

- `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

- `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

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

Загрузка данных, предобработка

In [1]:
```
%matplotlib inline
import pandas as pd
import numpy as np
import datetime
import matplotlib.pyplot as plt
import seaborn as sns
from plotly.subplots import make_subplots
from plotly import graph_objects as go
import scipy.stats as stats
from statsmodels.stats.proportion import proportions_ztest
```

Комментарий тимлида (1)

Видимо ты выполняла проект в локальном окружении - и перед отправкой не проверила - поэтому пути указаны твои локальные, мне пришлось это поправить.

In [2]:
```
ab_project_marketing_events = (
    pd.read_csv('/datasets/ab_project_marketing_events.csv', sep=',', parse_dates = ['start_dt','finish_dt']))
final_ab_new_users = (
    pd.read_csv('/datasets/final_ab_new_users.csv', sep=',', parse_dates = ['first_date']))
final_ab_events = (
    pd.read_csv('/datasets/final_ab_events.csv', sep=',', parse_dates = ['event_dt']))
final_ab_participants = (
    pd.read_csv('/datasets/final_ab_participants.csv', sep=','))
```

In [3]:
```
def show_df(df):
    """
    функция просмотра данных
    """
    display(df.head(),
            df.info(),
            df.isna().mean(),
            df.duplicated().sum()
           )
```

In [4]:
```
# календарь маркетинговых событий на 2020 год
show_df(ab_project_marketing_events)
```

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

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

```
None
name         0.0
regions      0.0
start_dt     0.0
finish_dt    0.0
dtype: float64
0
```

In [5]:

```
# пользователи, зарегистрировавшиеся с 7 по 21 декабря 2020 года
show_df(final_ab_new_users)
```

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

	user_id	first_date	region	device
0	D72A72121175D8BE	2020-12-07	EU	PC

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

```
None
user_id       0.0
first_date    0.0
region        0.0
device        0.0
dtype: float64
0
```

In [6]:
```
# действия новых пользователей в период с 7 декабря 2020 по 4 января 2021 года
show_df(final_ab_events)
```

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

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

```
None
user_id       0.000000
event_dt      0.000000
event_name    0.000000
```

```
details       0.857512
dtype: float64
0
```

In [7]:

```
# таблица участников тестов
show_df(final_ab_participants)
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 18268 entries, 0 to 18267
Data columns (total 3 columns):
user_id    18268 non-null object
group      18268 non-null object
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

```
None
user_id    0.0
group      0.0
ab_test    0.0
dtype: float64
0
```

Анализ данных

In [8]:

```
# посмотрим сколько тестов проводилось в период с 2020-12-07 по 2021-01-04
final_ab_participants['ab_test'].value_counts()
```

Out[8]:

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

Тест проводился одновременно с другим, проверим не искажает ли он наш тест

In [9]:
```
# количество пользователей по группам в каждом тесте
total_users = final_ab_participants.groupby(['ab_test','group']).agg({'user_id':'nunique'}).reset_index()
total_users
```

Out[9]:

	ab_test	group	user_id
0	interface_eu_test	A	5831
1	interface_eu_test	B	5736
2	recommender_system_test	A	3824
3	recommender_system_test	B	2877

In [10]:
```
# проверим есть ли пересекающиеся пользователи
Users_two_test = final_ab_participants.groupby('user_id').agg({'ab_test':'nunique'}).query('ab_test > 1').index
print(f"Количество пользователей, которые попали в оба теста : {Users_two_test.shape[0]}")
```

```
Количество пользователей, которые попали в оба теста : 1602
```

In [11]:
```
# распределение пересекающихся пользователей по тестам
Users_two_test_df = final_ab_participants[final_ab_participants['user_id'].isin(Users_two_test)]
Users_two_test_df.groupby(['ab_test','group']).agg({'user_id':'nunique'}).reset_index().\
assign(user_ratio = lambda x: x['user_id'] / total_users['user_id'])
```

Out[11]:

	ab_test	group	user_id	user_ratio
0	interface_eu_test	A	819	0.140456
1	interface_eu_test	B	783	0.136506
2	recommender_system_test	A	921	0.240847
3	recommender_system_test	B	681	0.236705

Так как соотношение пользователей по группам двух тестов одинаково, оставляем участников тестов, не теряя аудиторию.

Комментарий тимлида (1)

OK, аргументированно. Но я бы лучше убрал таких пользователей в ущерб кол-ву наблюдений - это ошибка ТЗ, а не проблема аналитика - это нужно выносить в рекомендации к формированию выборки.

In [12]:
```
# возможные акции в период тестрования
ab_project_marketing_events[ab_project_marketing_events['start_dt'] > '2020-12-01']
```

Out[12]:

	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

Тест проводился во время акции Christmas&New Year Promo, перед новым годом. Поведение пользователей в такие периоды отличное от стандартного. Оставим события, с учетом того, что выводы будут верны для акционных периодов.

In [13]:
```
# Количество пользователей
participants_test = final_ab_participants[final_ab_participants['ab_test']== 'recommender_system_test']
display(f"Количество участников в тесте: {participants_test.shape[0]}")
```

```
'Количество участников в тесте: 6701'
```

In [14]:
```
# добавим к участникам теста их события и дату регистрации
participants = participants_test.merge(final_ab_events).merge(final_ab_new_users)
```

In [15]:
```
participants.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 24698 entries, 0 to 24697
Data columns (total 9 columns):
user_id       24698 non-null object
group         24698 non-null object
ab_test       24698 non-null object
event_dt      24698 non-null datetime64[ns]
event_name    24698 non-null object
details       3331 non-null float64
first_date    24698 non-null datetime64[ns]
region        24698 non-null object
device        24698 non-null object
```

```
dtypes: datetime64[ns](2), float64(1), object(6)
memory usage: 1.9+ MB
```

In [16]:
```
display(f"Количество пользователей после объединения таблиц: {participants['user_id'].nunique()}")
```

```
'Количество пользователей после объединения таблиц: 3675'
```

Количество пользователей уменьшилось вдвое. Участники теста зарегистрировались, но за время проведения теста действий не совершали.

In [17]:
```
display(f"Количество событий: {participants.shape[0]}")
```

```
'Количество событий: 24698'
```

In [18]:
```
# фильтруем данные по региону EU
participants = participants[participants['region'] == 'EU']
participants.shape[0]
```

Out[18]:
```
23420
```

In [19]:
```
# добавим столбец с датой +14 дней после регистрации пользователя
participants['date_later_reg'] = participants['first_date'] + datetime.timedelta(days=14)
```

In [20]:
```
# фильтруем события по дате окончания теста
events = participants[participants['date_later_reg'] <= '2021-01-04']
```

In [21]:
```
display(f"Количество событий в логе: {events.shape[0]}")
```

```
'Количество событий в логе: 23420'
```

In [22]:
```
display(f"Количество пользователей в логе: {events['user_id'].nunique()}")
```

```
'Количество пользователей в логе: 3481'
```

In [23]:
```
print('Доля отсеянных от общей выборки  = {:.2%}'.format(1 - events['user_id'].nunique()/participants_test.shape[0]))
```

```
Доля отсеянных от общей выборки  = 48.05%
```

Воронка событий

In [24]:
```
# Посмотрим количество событий пользователя
events['user_id'].value_counts()
```

Out[24]:
```
1198061F6AF34B7B    28
115EBC1CA027854A    27
1BFEE479308EFF44    24
CED71698585A2E46    24
B8EF6F0325A9979F    21
                    ..
3ED90BE0DC2A3FD3     1
87B112DF92E5A3BA     1
921FFC6F0F506A82     1
FE9B25977A8537C2     1
5B4EFE916AD19741     1
Name: user_id, Length: 3481, dtype: int64
```

Самое большое количество событий, совершенных одним пользователем - 28, это немного, выбросов нет.

In [25]:
```
# Посмотрим какие события есть в логах
events_type = events['event_name'].value_counts().reset_index()
events_type
```

Out[25]:

	index	event_name
0	login	10595
1	product_page	6554
2	purchase	3196
3	product_cart	3075

10595 событий входа в сервис, из них 6554 просмотров карточек товаров, 3075 событий просмотра корзины и 3196 событий об оплате.

In [26]:
```
# Посчитаем сколько пользователей совершали каждое из этих событий
users_event = events.groupby(['event_name']).agg({'user_id':'nunique'}).sort_values(by='user_id', ascending=False)
users_event
```

Out[26]:

event_name	user_id
login	3481
product_page	2178
purchase	1082
product_cart	1026

In [27]:

```
# Посчитаем долю пользователей, которые хоть раз совершали событие
initial_users_count = events['user_id'].nunique()
round(users_event/initial_users_count*100,2)
```

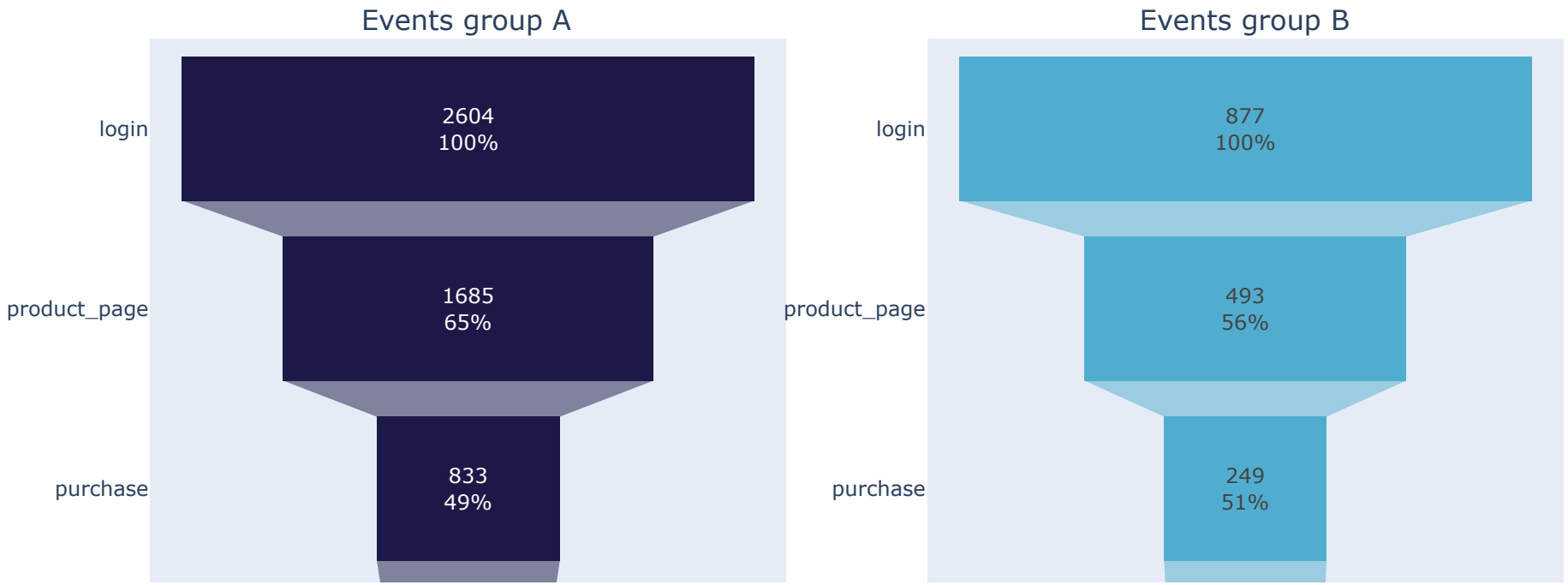
Out[27]:

event_name	user_id
login	100.00
product_page	62.57
purchase	31.08
product_cart	29.47

In [28]:

```
# посмотрим количество и долю пользователей, которые проходят на следующий шаг воронки (от числа пользователей на предыдущем)
test_ab = events.pivot_table(index='event_name',columns='group',values='user_id',aggfunc='nunique').\
                    sort_values(by='A', ascending=False).reset_index()

fig = make_subplots(rows=1, cols=2,subplot_titles=("Events group A","Events group B"))


fig.add_trace(
    go.Funnel(
    y = test_ab['event_name'],
    x = test_ab['A'],
    textposition = "inside",
    textinfo = "value+percent previous",
    marker = {"color": "#1c1847"}
```

```
),
    row=1, col=1
)

fig.add_trace(
    go.Funnel(
    y = test_ab['event_name'],
    x = test_ab['B'],
    textposition = "inside",
    textinfo = "value+percent previous",
        marker = {"color": "#51adcf"}
),
    row=1, col=2
)

fig.update_layout(showlegend=False,height=600, width=1000)
fig.show()
```



В группе A: после входа 65% пользователей переходят к карточке товара и половина из них в корзину и к оплате (конверсия в покупку 16%).

В группе B: после входа 56% пользователей переходят к карточке товара и половина из них в корзину и к оплате (конверсия в покупку 5%).

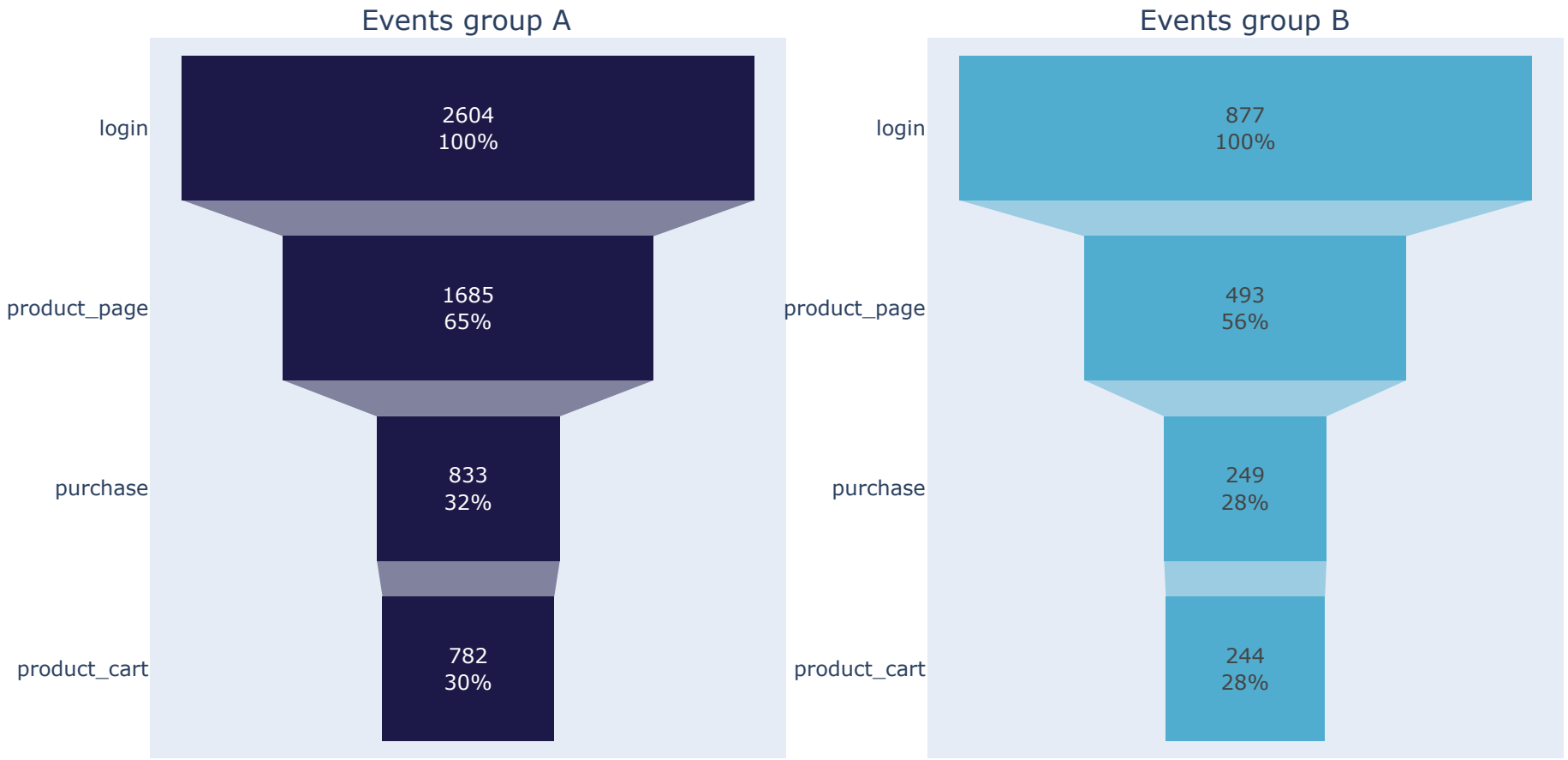
В группе B доля пользователей, переходящих к просмотру карточки товара, ниже на 9%.

Обе группы теряют половину пользователей после просмотра карточки товара.

In [29]:

```
# Количество и доля пользователей, проходящих от первого события до оплаты
fig = make_subplots(rows=1, cols=2,subplot_titles=("Events group A","Events group B"))


fig.add_trace(
    go.Funnel(
    y = test_ab['event_name'],
    x = test_ab['A'],
    textposition = "inside",
    textinfo = "value+percent initial",
    marker = {"color": "#1c1847"}
),
    row=1, col=1
)

fig.add_trace(
    go.Funnel(
    y = test_ab['event_name'],
    x = test_ab['B'],
    textposition = "inside",
    textinfo = "value+percent initial",
        marker = {"color": "#51adcf"}
),
```

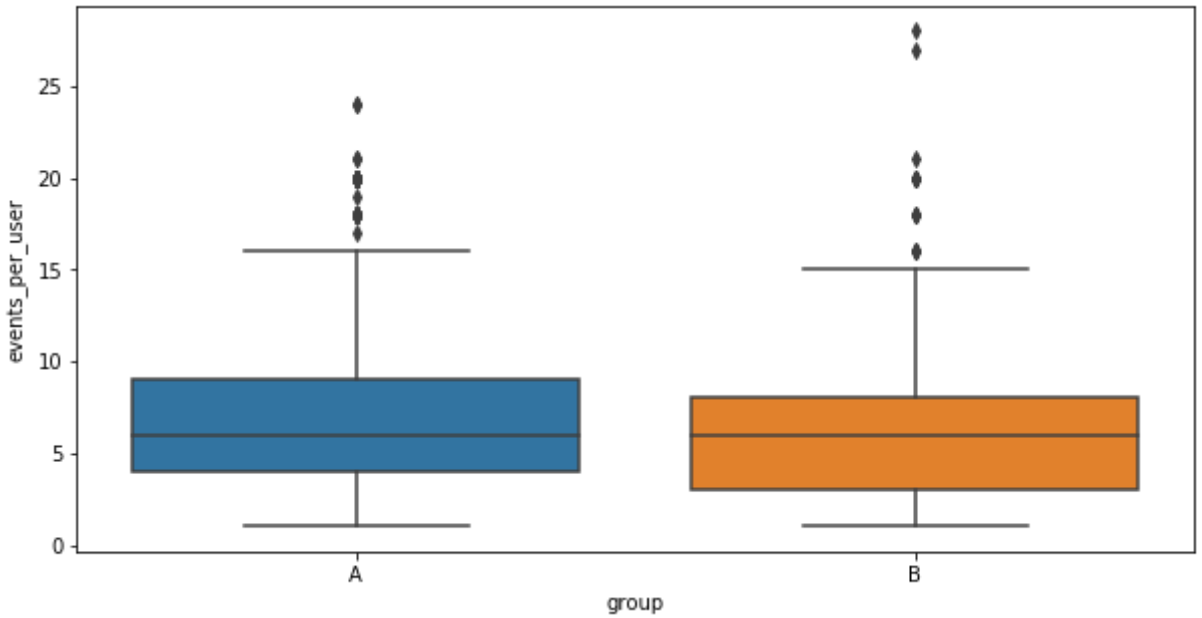
```
    row=1, col=2
)

fig.update_layout(showlegend=False,height=600, width=1000)
fig.show()
```



В грунппе А 30% пользователей доходят до оплаты после входа в сервис, в группе В - 28%.

In [30]:
```
# Распределение событий на пользователя по выбркам
events_per_user=events.groupby(['group','user_id']).agg(events_per_user=('user_id','count')).reset_index()
```

```
dims = (10, 5)
fig, ax = plt.subplots(figsize=dims)

sns.boxplot(data=events_per_user, x='group', y='events_per_user')
plt.xlabel('group')
plt.ylabel('events_per_user')
plt.show()
```



In [31]:
```
events_per_user.groupby('group')['events_per_user'].describe()
```

Out[31]:

group	count	mean	std	min	25%	50%	75%	max
A	2604.0	7.031106	3.872263	1.0	4.0	6.0	9.0	24.0
B	877.0	5.827822	3.492164	1.0	3.0	6.0	8.0	28.0

Распределение событий, совершаемых одним пользователем, по двум группам почти одинаково. В среднем пользователь совершает по 6 событий.

In [32]:
```
# проверим есть ли пересекающиеся пользователи в группах
events.groupby('user_id').agg({'group':'nunique'}).query('group > 1')
```

Out[32]:

user_id	group

Пересекающихся пользователей в группах нет

In [33]:
```
# Построим гистограмму распределения событий по дате и времени
plt.figure(figsize=(17,5))
plt.hist(events['event_dt'], bins=20, label="Количество событий по дате")
plt.title('График распределения событий')
plt.legend()
plt.xlabel('Дата')
plt.xticks(np.arange(events['event_dt'].min(),
                     events['event_dt'].max() + pd.to_timedelta('1 day'),
                     pd.to_timedelta('1 day')), rotation = 45)
plt.show()
```

```
/opt/conda/lib/python3.7/site-packages/pandas/plotting/_matplotlib/converter.py:103: FutureWarning:

Using an implicitly registered datetime converter for a matplotlib plotting method. The converter was registered by pandas on impo
rt. Future versions of pandas will require you to explicitly register matplotlib converters.

To register the converters:
        >>> from pandas.plotting import register_matplotlib_converters
        >>> register_matplotlib_converters()
```

Наблюдается скачок активности с 14 декабря и постепенный спад за неделю до нового года. Для проверки гипотез оставим весь период, в него входят пользователи, зарегистрированные за две недели до окончания теста.
Явных аномалий в данных не найдено, кроме того, что тест проводился перед новым годом и половина участников за небольшой период не совершили никаких действий.

Результаты эксперимента

In [34]:
```
# Количество и доля пользователей в каждой группе
initial_users_groups=events.groupby(['group']).agg(count=('user_id','nunique')).\
assign(proportion = lambda x: x['count'] / events['user_id'].nunique())
initial_users_groups
```

Out[34]:

group	count	proportion
A	2604	0.748061
B	877	0.251939

Тестовая группа B в 3 раза меньше контрольной, это допустимо для ожидания увеличения в конверсии событий.

In [35]:
```
# Количество пользователей и доля, совершивших события в каждой из групп
test_ab.\
assign(funnel_A = lambda x: x['A'].pct_change()).\
assign(funnel_B = lambda x: x['B'].pct_change())
```

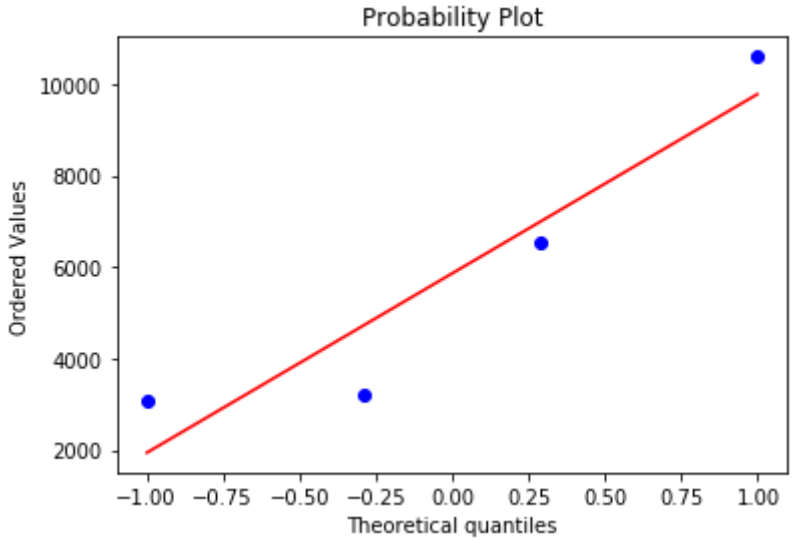
Out[35]:

	group	event_name	A	B	funnel_A	funnel_B
0		login	2604	877	NaN	NaN
1		product_page	1685	493	-0.352919	-0.437856
2		purchase	833	249	-0.505638	-0.494929
3		product_cart	782	244	-0.061224	-0.020080

После входа в сервис не переходят на карточки товаров 44% пользователей в группе B, что значительно хуже группы A(35%), из них покупки совершает половина пользователей в двух группах.

In [36]:
```
# Проверка нормальности распределения данных по критерию Шапиро-Уилка и по диаграмме QQ-plot
print('p-значение: ', stats.shapiro(events_type['event_name'])[1])
stats.probplot(events_type['event_name'], dist='norm', plot=plt)
plt.show()
```

```
p-значение:  0.29245197772979736
```



p-значение больше 0.05, распределение нормальное - для оценки статистической разницы долей используем z-критерий

Сформулируем нулевую и альтернативную гипотезы:

H_0 - нет различий в конверсиях групп A и B

H_1 - есть различия в конверсиях групп A и B

In [37]:

```
# функция проверки статистических отличий между группами для всех событий
def test(value_alpha):
    alpha = value_alpha
    groups = ['A','B']
    for j in range(1,4):
            count = np.array(test_ab.loc[(j,groups)])
            nobs = np.array(initial_users_groups.loc[(groups,'count')])
            stat, pval = proportions_ztest(count, nobs)
            print('')
            print('Событие:', j)
            print('Количество пользователей, совершивших событие:', count)
            print('Общее количество пользователей в группе:', nobs)
            print('p-значение: ', '{0:0.3f}'.format(pval))
            if (pval < alpha):
                print("Отвергаем нулевую гипотезу: между долями есть значимая разница")
            else:
                print("Не получилось отвергнуть нулевую гипотезу, нет оснований считать доли разными")
```

In [38]:
```
# проверка отличий с принятым уровнем значимости 0.05
test(0.05)
```

```
Событие: 1
Количество пользователей, совершивших событие: [1685 493]
Общее количество пользователей в группе: [2604  877]
p-значение:  0.000
Отвергаем нулевую гипотезу: между долями есть значимая разница

Событие: 2
Количество пользователей, совершивших событие: [833 249]
Общее количество пользователей в группе: [2604  877]
p-значение:  0.047
Отвергаем нулевую гипотезу: между долями есть значимая разница

Событие: 3
Количество пользователей, совершивших событие: [782 244]
Общее количество пользователей в группе: [2604  877]
p-значение:  0.215
Не получилось отвергнуть нулевую гипотезу, нет оснований считать доли разными
```

Статистически значимая разница есть между долями групп по событиям просмотра карточек и покупки товаров.

Вывод:

Тестирование изменений, связанных с внедрением улучшенной рекомендательной системы, не показали улучшения метрик.

Просмотр карточек товаров (событие product_page) - уменьшение конверсии на 10% (в группе A конверсия в просмотр карточек товаров от предыдущего шага 35%, а в группе B - 44%);

просмотры корзины (product_cart) - нет статистически значимых различий;

покупки (purchase) - уменьшение конверсии на 10% (в группе A конверсия в покупку от предыдущего шага 15%, а в группе B - 5%).

Тест проводился в декабре перед новым годом, поэтому сделанные выводы мы не можем применить относительно обычных периодов.

Количество участников теста удовлетворяет техническому заданию, но после регистрации половина участников никаких действий за период проведения теста не совершили. Также большой процент пользователей после авторизации на сервисе не просматривают товары.

Необходимо проверить юзабилити сервиса.

Комментарий тимлида (1)

Не к чему придраться - проект выполнен отлично, за исключением того, что ты не убрала пользователей из пересекающегося теста. Однако в работе слишком много плюсов, чтобы их игнорировать, например проверка гипотез вне всяческих похвал. Поэтому, учитывая близящийся

дедлайн, принимаю работу.

Принято.