Сборный проект - 2

Описание данных

Каждая запись в логе — это действие пользователя, или событие.

- `EventName` — название события;
- `DeviceIDHash` — уникальный идентификатор пользователя;
- `EventTimestamp` — время события;
- `ExpId` — номер эксперимента: 246 и 247 — контрольные группы, а 248 — экспериментальная.

In [1]:
```
import pandas as pd, matplotlib.pyplot as plt, numpy as np, math as mth, seaborn as sns, datetime as dt
from scipy import stats as st
from IPython.display import display, HTML
pd.set_option('display.max_columns', None)
pd.set_option('display.max_colwidth', 500)
```

Шаг 1. Загрузка данных

In [2]:
```
df=pd.read_csv(r'/datasets/logs_exp.csv', sep="\t")
```

In [3]:
```
df.head()
```

Out[3]:

	EventName	DeviceIDHash	EventTimestamp	ExpId
0	MainScreenAppear	4575588528974610257	1564029816	246
1	MainScreenAppear	7416695313311560658	1564053102	246
2	PaymentScreenSuccessful	3518123091307005509	1564054127	248
3	CartScreenAppear	3518123091307005509	1564054127	248
4	PaymentScreenSuccessful	6217807653094995999	1564055322	248

Шаг 2. Подготовка данных

In [4]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 244126 entries, 0 to 244125
Data columns (total 4 columns):
EventName         244126 non-null object
DeviceIDHash      244126 non-null int64
EventTimestamp    244126 non-null int64
ExpId             244126 non-null int64
dtypes: int64(3), object(1)
memory usage: 7.5+ MB
```

In [5]:
```
df.duplicated().sum()
```

Out[5]:
```
413
```

Пропуски отсутствуют, однако имеются дубликаты. Устраним их и переименуем столбцы.

In [6]:
```
df = df.drop_duplicates().reset_index(drop=True)
df.columns=['event', 'userid', 'date_time', 'group']
```

Поменяем тип данных в столбце date_time и добавим отдельный столбец с датами

In [7]:
```
df['date_time'] = pd.to_datetime(df['date_time'], unit = 's')
df.head()
```

Out[7]:

	event	userid	date_time	group
0	MainScreenAppear	4575588528974610257	2019.07.25 04:43:36	246
1	MainScreenAppear	7416695313311560658	2019.07.25 11:11:42	246
2	PaymentScreenSuccessful	3518123091307005509	2019.07.25 11:28:47	248
3	CartScreenAppear	3518123091307005509	2019.07.25 11:28:47	248
4	PaymentScreenSuccessful	6217807653094995999	2019.07.25 11:48:42	248

In [8]:
```
df['date'] = df['date_time'].map(lambda x: x.date())
df.head()
```

Out[8]:

	event	userid	date_time	group	date
0	MainScreenAppear	4575588528974610257	2019.07.25 04:43:36	246	2019.07.25
1	MainScreenAppear	7416695313311560658	2019.07.25 11:11:42	246	2019.07.25
2	PaymentScreenSuccessful	3518123091307005509	2019.07.25 11:28:47	248	2019.07.25
3	CartScreenAppear	3518123091307005509	2019.07.25 11:28:47	248	2019.07.25
4	PaymentScreenSuccessful	6217807653094995999	2019.07.25 11:48:42	248	2019.07.25

Шаг 3. Изучение и проверка данных

In [9]:
```
pd.DataFrame(data=[len(df), df['userid'].nunique(), np.round(len(df)/df['userid'].nunique()),
                   df['date'].min(), df['date'].max()],
                   index=['Всего событий', 'Число пользователей', 'Число событий на пользователя', 'Период с...', '...по
'],
                   columns=['Значение'])
```

Out[9]:

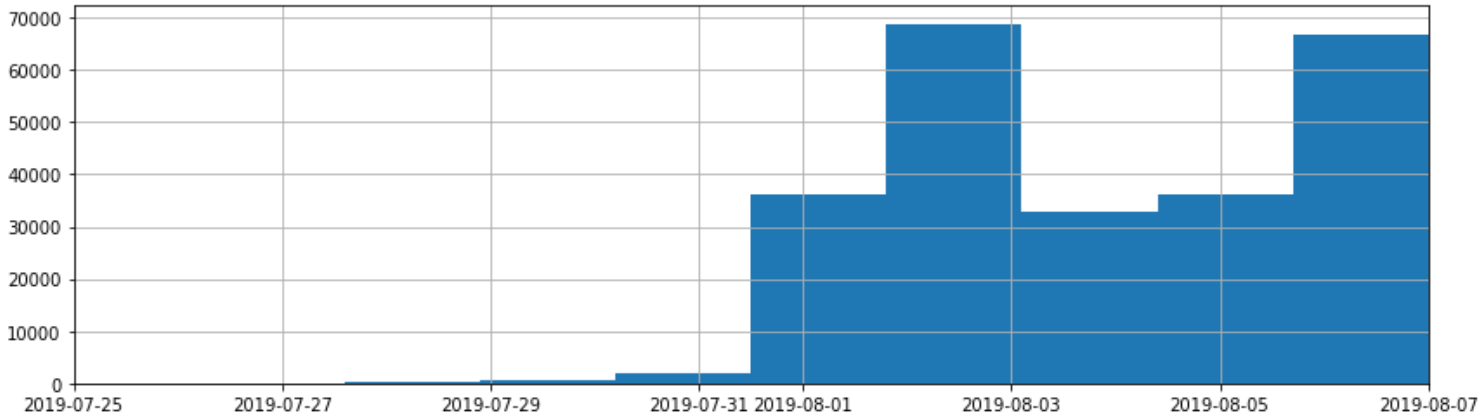
	Значение
Всего событий	243713
Число пользователей	7551
Число событий на пользователя	32
Период с...	2019.07.25
...по	2019.08.07

In [10]:
```
plt.figure(figsize=(14,4))
plt.xlim(df['date'].min(), df['date'].max(), dt.date)
df['date'].hist()
```

```
/opt/conda/lib/python3.7/site-packages/pandas/plotting/_matplotlib/converter.py:103: FutureWarning: Using an implici
tly registered datetime converter for a matplotlib plotting method. The converter was registered by pandas on impor
t. 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()
  warnings.warn(msg, FutureWarning)
```

Out[10]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7fa01d2a5290>
```



Всплеск активности наблюдается с 1 августа 2019 г. и продолжается по 7 августа включительно. Следовательно именно этот период необходимо проанализировать.

In [11]:
```
df_original=df.copy()
```

In [12]:
```
df=df[df['date'] >= dt.date(2019, 8, 1)]
```

In [13]:
```
pd.DataFrame(data=[len(df), df['userid'].nunique(), np.round(len(df)/df['userid'].nunique()),
             df['date'].min(), df['date'].max()],
             index=['Всего событий', 'Число пользователей', 'Число событий на пользователя', 'Период с...', '...по
'],
             columns=['Значение'])
```

Out[13]:

	Значение
Всего событий	240887
Число пользователей	7534
Число событий на пользователя	32
Период с...	2019.08.01
...по	2019.08.07

In [14]:
```
before=pd.DataFrame(data=[len(df_original), df_original['userid'].nunique(), np.round(len(df_original)/df_original
['userid'].nunique()),
             df_original['date'].min(), df_original['date'].max()],
             index=['Всего событий', 'Число пользователей', 'Число событий на пользователя', 'Период с...', '...по
'],
             columns=['Значение начальное'])
after=pd.DataFrame(data=[len(df), df['userid'].nunique(), np.round(len(df)/df['userid'].nunique()),
             df['date'].min(), df['date'].max()],
             index=['Всего событий', 'Число пользователей', 'Число событий на пользователя', 'Период с...', '...по
'],
             columns=['Значение после отбора'])
decrease=pd.DataFrame(before['Значение начальное'] - after['Значение после отбора'], columns=['Убыль'])
```

In [15]:
```
before.join(after).join(decrease)
```

Out[15]:

	Значение начальное	Значение после отбора	Убыль
Всего событий	243713	240887	2826
Число пользователей	7551	7534	17
Число событий на пользователя	32	32	0
Период с...	2019.07.25	2019.08.01	.7 days, 0:00:00
...по	2019.08.07	2019.08.07	0:00:00

In [16]:
```
df['group'].value_counts()
```

Out[16]:
```
248    84563
246    79302
247    77022
Name: group, dtype: int64
```

Пользователи есть из трёх экспериментальных групп.

Шаг 4. Изучение воронки событий

In [17]:
```
events=pd.DataFrame(data=df['event'].value_counts().values,
             index=[df['event'].value_counts().index], columns=['Количество']).reset_index()
events.columns=['event','количество']
events['событие']=['появление главного экрана', 'появление экрана с предложением товара',
                   'появление экрана с корзиной', 'экран успешной оплаты', 'руководство по использованию']
events=events[['event', 'событие', 'количество']]
```

In [18]:
```
events=events.merge(df.groupby('event')['userid'].agg('nunique').reset_index(), on='event')
events['доля польз-лей. %']=np.round(100* events['userid']/events['userid'][0], 2)
events=events.rename(columns = {'userid':'число польз-лей'})
events
```

Out[18]:

	event	событие	количество	число польз-лей	доля польз-лей. %
0	MainScreenAppear	появление главного экрана	117328	7419	100.00
1	OffersScreenAppear	появление экрана с предложением товара	46333	4593	61.91
2	CartScreenAppear	появление экрана с корзиной	42303	3734	50.33
3	PaymentScreenSuccessful	экран успешной оплаты	33918	3539	47.70
4	Tutorial	руководство по использованию	1005	840	11.32

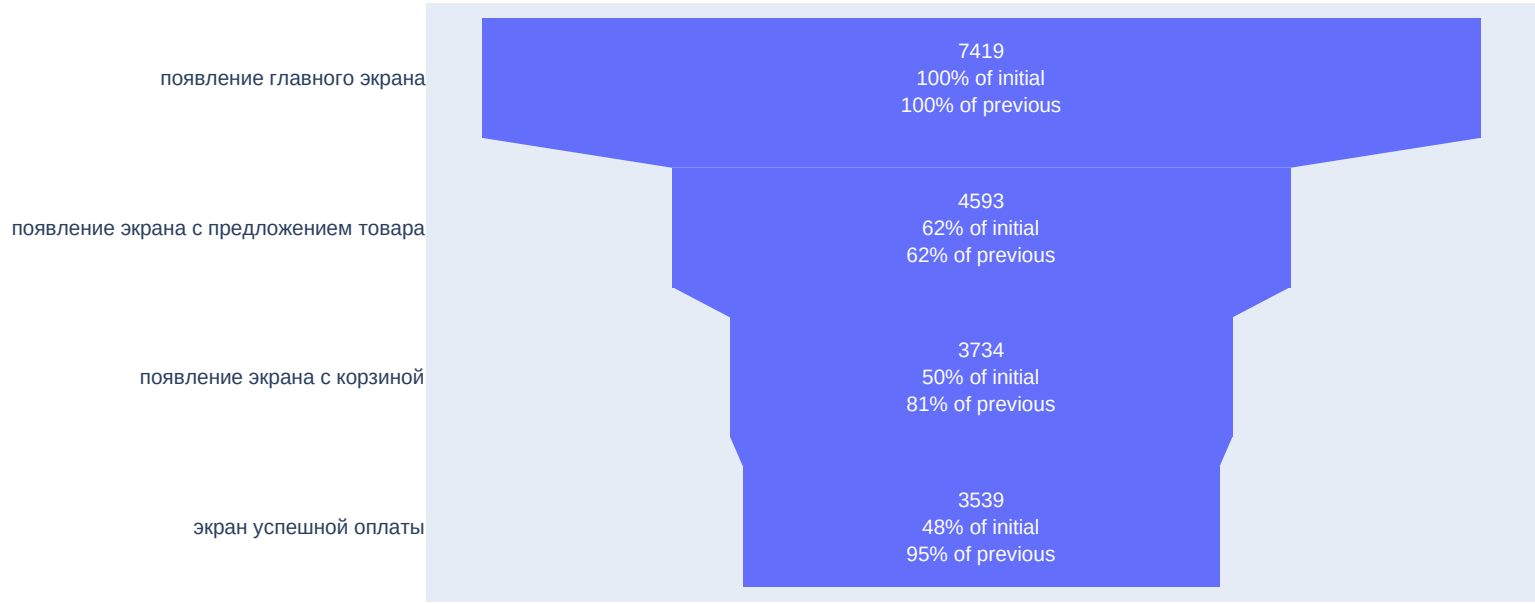
MainScreenAppear-->OffersScreenAppear-->CartScreenAppear-->PaymentScreenSuccessful--> Tutorial

Последовательность событий выглядит логичной за исключением события Tutorial

In [19]:
```
from plotly import graph_objects as go
```

In [20]:
```
fig = go.Figure(go.Funnel(
    y = ['появление главного экрана', 'появление экрана с предложением товара',
                    'появление экрана с корзиной', 'экран успешной оплаты'],
    x = events['число польз-лей'][:len(events)-1], textposition = "inside",
    textinfo = "value+percent previous+percent initial"
    ))

fig.show()
```



При переходе от главного экрана на страницу с предложением товара теряется самое большое количество пользователей! От первого события до оплаты доходит 48 % пользователей.

Шаг 5. Изучение результатов эксперимента

Посчитаем количество пользователей в каждой экспериментальной группе.

In [21]:
```
count_by_group=df.groupby('group')['userid'].agg('nunique').reset_index()
count_by_group
```

Out[21]:

	group	userid
0	246	2484
1	247	2513
2	248	2537

Посчитаем количество пользователей по каждому событию

In [22]:
```
df_pivot=df.pivot_table(values='userid', columns='event', index='group', aggfunc='nunique')
df_pivot['total_users']=count_by_group['userid'].values
df_pivot=df_pivot.reset_index()
df_pivot=df_pivot[['group', 'total_users', 'MainScreenAppear', 'OffersScreenAppear','CartScreenAppear',
       'PaymentScreenSuccessful']]
df_pivot=df_pivot.T.reset_index()
df_pivot.columns = df_pivot.iloc[0]
df_pivot=df_pivot.drop(df_pivot.index[0])
df_pivot.rename(columns={'group':'event'}, inplace=True)
df_pivot=df_pivot.reset_index(drop=True)
df_pivot
```

Out[22]:

	event	246	247	248
0	total_users	2484	2513	2537
1	MainScreenAppear	2450	2476	2493
2	OffersScreenAppear	1542	1520	1531
3	CartScreenAppear	1266	1238	1230
4	PaymentScreenSuccessful	1200	1158	1181

Проверим, контрольные выборки 246 и 247 на предмет статистической значимости различий. Примем следующую нулевую гипотезу . "Между выборками 246 и 247 нет значимой разницы".

In [23]:
```
def z_test(group_list1, group_list2, alpha_level):
    alpha = alpha_level
    result_list=[None]
    for i in range(1,len(group_list1)):
        successes = np.array([group_list1[i], group_list2[i]])
        trials = np.array([group_list1[i-1], group_list2[i-1]])
        p1 = successes[0]/trials[0]
        p2 = successes[1]/trials[1]
        p_combined = (successes[0] + successes[1]) / (trials[0] + trials[1])
        difference = p1 - p2
        z_value = difference / mth.sqrt(p_combined * (1 - p_combined) * (1/trials[0] + 1/trials[1]))
        distr = st.norm(0, 1)
        p_value = (1 - distr.cdf(abs(z_value))) * 2
        if (p_value < alpha):
            result = " Отвергаем нулевую гипотезу, между долями есть значимая разница"
        else:
            result = " Не отвергаем нулевую гипотезу, доли не разные"
        result_list.append('p-value:' + str(np.round(p_value, 3)) + result)
    return result_list
```

In [24]:
```
df_pivot['test_result']=z_test(df_pivot[246], df_pivot[247], 0.05/16)
df_pivot[['event', 246, 247,'test_result']]
```

Out[24]:

	event	246	247	test_result
0	total_users	2484	2513	None
1	MainScreenAppear	2450	2476	p.value:0.757 Не отвергаем нулевую гипотезу, доли не разные
2	OffersScreenAppear	1542	1520	p.value:0.262 Не отвергаем нулевую гипотезу, доли не разные
3	CartScreenAppear	1266	1238	p.value:0.639 Не отвергаем нулевую гипотезу, доли не разные
4	PaymentScreenSuccessful	1200	1158	p.value:0.182 Не отвергаем нулевую гипотезу, доли не разные

Конверсии между контрольными группами на каждом этапе не отличаются. Разбиение на группы работает корректно.

Сравним результаты с каждой из контрольных групп в отдельности по каждому событию.

In [25]:
```
df_pivot['test_result']=z_test(df_pivot[246], df_pivot[248], 0.05/16)
df_pivot[['event', 246, 248,'test_result']]
```

Out[25]:

	event	246	248	test_result
0	total_users	2484	2537	None
1	MainScreenAppear	2450	2493	p.value:0.295 Не отвергаем нулевую гипотезу, доли не разные
2	OffersScreenAppear	1542	1531	p.value:0.268 Не отвергаем нулевую гипотезу, доли не разные
3	CartScreenAppear	1266	1230	p.value:0.211 Не отвергаем нулевую гипотезу, доли не разные
4	PaymentScreenSuccessful	1200	1181	p.value:0.143 Не отвергаем нулевую гипотезу, доли не разные

In [26]:
```
df_pivot['test_result']=z_test(df_pivot[247], df_pivot[248], 0.05/16)
df_pivot[['event', 247, 248,'test_result']]
```

Out[26]:

	event	247	248	test_result
0	total_users	2513	2537	None
1	MainScreenAppear	2476	2493	p.value:0.459 Не отвергаем нулевую гипотезу, доли не разные
2	OffersScreenAppear	1520	1531	p.value:0.987 Не отвергаем нулевую гипотезу, доли не разные
3	CartScreenAppear	1238	1230	p.value:0.436 Не отвергаем нулевую гипотезу, доли не разные
4	PaymentScreenSuccessful	1158	1181	p.value:0.006 Не отвергаем нулевую гипотезу, доли не разные

In [27]:
```
df_pivot['246_247_combined'] = df_pivot[246] + df_pivot[247]
df_pivot['test_result']=z_test(df_pivot['246_247_combined'], df_pivot[248], 0.05/16)
df_pivot[['event', '246_247_combined', 248,'test_result']]
```

Out[27]:

	event	246_247_combined	248	test_result
0	total_users	4997	2537	None
1	MainScreenAppear	4926	2493	p.value:0.294 Не отвергаем нулевую гипотезу, доли не разные
2	OffersScreenAppear	3062	1531	p.value:0.531 Не отвергаем нулевую гипотезу, доли не разные
3	CartScreenAppear	2504	1230	p.value:0.239 Не отвергаем нулевую гипотезу, доли не разные
4	PaymentScreenSuccessful	2358	1181	p.value:0.017 Не отвергаем нулевую гипотезу, доли не разные

Так как мы имеем дело со множественным тестом, проводимым на одном и том же наборе данных, необходимо введение поправки уровня значимости. Руководствуясь, методом Бонферрони изначальное значение уровня значимости равное 0,05 было снижено в 16 раз до 0,003125, так как общее число сравнений . 16.

Вывод:

Статистически значимых различий не наблюдается, следовательно изменение шрифтов не повлияло на конверсию.

In [ ]: