

ИССЛЕДОВАНИЕ:

Мобильное приложение "Ненужные вещи". Выделение групп пользователей на основе их поведения

Заказчик исследования: "НЕНУЖНЫЕ ВЕЩИ".

Анатик исследования: Анися Сухамбердиева.

Дата исследования: 16.03.2025 год

Настоящее исследование посвящено анализу поведения пользователей мобильного приложения "Ненужные вещи" — платформы для купли-продажи б/у товаров.

Цель исследования: Выявить ключевые факторы, влияющие на удержание пользователей, их вовлеченность и конверсию в целевые действия, такие как просмотр контактных данных контактов продавцов. Анализ позволит сформировать сегменты пользователей на основе их поведения и разработать стратегии для улучшения ключевых метрик приложения.

Ожидаемые результаты:

- Определение наиболее эффективных каналов привлечения пользователей.
- Выявление паттернов поведения, способствующих удержанию и совершению целевых действий.
- Формулирование гипотез и рекомендаций для оптимизации работы приложения и повышения лояльности пользователей.

Результаты исследования будут использованы для принятия решений, направленных на улучшение пользовательского опыта, увеличение удержания пользователей и повышение эффективности маркетинговых кампаний.

Декомпозиция исследования:

1. Подготовка данных (Предобработка):

- 1.1 Загрузка и обзор данных: типы, пропуски, дубликаты.
- 1.2 Обработка данных: форматирование, обработка пропусков, удаление дубликатов.
- 1.3 Добавление новых полей, объединение таблиц.
- 1.4 Выводы по предобработке.

2. Анализ удержания (Retention Rate):

- 2.1 Определение когорт пользователей.
- 2.2 Расчет Retention Rate (1/7/14 дней).
- 2.3 Анализ Retention Rate по источникам привлечения.
- 2.4 Выводы по удержанию.

3. Анализ сессий:

- 3.1 Определение таймаута сессии.
- 3.2 Формирование сессий для каждого пользователя.
- 3.3 Расчет количества и длительности сессий.
- 3.4 Анализ распределения длительности сессий (среднее, медиана).
- 3.5 Выводы по сессиям.

4. Анализ активности пользователей:

- 4.1 Расчет количества действий для каждого пользователя.
- 4.2 Анализ распределения количества действий.
- 4.3 Выводы по активности пользователей.

5. Анализ конверсии в целевое действие "contacts_show":

- 5.1 Расчет конверсии в просмотр контактов "contacts_show" для всех пользователей и по источникам.
- 5.2 Анализ количества сессий, необходимых для совершения целевого действия.
- 5.3 Выводы по конверсии пользователей.

6. Сегментация пользователей:

- 6.1 Выбор критериев сегментации.
- 6.2 Формирование сегментов пользователей.
- 6.3 Описание сегментов.
- 6.4 Расчёт Retention rate для каждой группы отдельно.
- 6.5 Расчёт конверсии для каждой группы отдельно.
- 6.4 Выводы и рекомендации по сегментации.

7. Проверка гипотез:

- 7.1 Гипотеза 1:
 - H0: Конверсия в просмотр контактов не зависит от источника привлечения (Google vs. Yandex).
 - H1: Пользователи из Yandex имеют более высокую конверсию в просмотр контактов, чем пользователи из Google.
- 7.2 Гипотеза 2:
 - H0: Средняя длительность сессий одинакова для обоих источников трафика.
 - H1: Средняя длительность сессий отличается от источника трафика, для Yandex выше чем для Google

Общие выводы и рекомендации по проведенному исследовательскому анализу

Подготовка данных (Предобработка)

In [1]: *# Импортируем необходимые для анализа данных библиотеки:*

```
import pandas as pd
import numpy as np
import math as mth
import statsmodels.stats.proportion as smp
from scipy import stats as st
from matplotlib import pyplot as plt
import seaborn as sns
from datetime import timedelta, date, datetime
import warnings
warnings.simplefilter('ignore')
```

Объявление необходимых функций для исследовательского анализа.

In [2]: *# Определение сессий, установление их продолжительности и присвоение порядкового*

```
def calculate_session(df, session_break=3600):
    """
    Вычисляет длительность сессии для каждого пользователя,
    где первая сессия пользователя имеет длительность 0.

    Args:
        Функция на вход принимает DataFrame с колонками 'user_id' и 'full_datetime'
        и значение session_break с типом int (дефолтное значение 3600).

    Returns:
        Измененный DataFrame с добавленной колонкой 'gap_requests' и номер сеанса
    """
    # 1. Создаём пустое поле с NaN
    df['gap_requests'] = np.nan
    df['gap_requests'] = df.groupby('user_id')['full_datetime'].diff().dt.total_seconds()

    # 2. Определение начала новой сессии с установкой флага (True/False)
    df['new_session'] = (df['gap_requests'] > session_break).astype(int)

    # 3. Присвоение идентификаторов сессиям
    df['session_num'] = df.groupby('user_id')['new_session'].cumsum() + 1 # нумерация

    # Удаляем вспомогательный столбец
    df = df.drop('new_session', axis=1)
    return df
```

In [3]: **def** get_retention(profiles, observation_date, horizon_days, dimensions=[], ignore_horizon=False):

```
# исключаем пользователей, не «доживших» до горизонта анализа
last_suitable_acquisition_date = observation_date
if not ignore_horizon:
    last_suitable_acquisition_date = observation_date - timedelta(
        days=horizon_days - 1)

result_raw = profiles.query('first_session <= @last_suitable_acquisition_date')
result_raw['lifetime'] = (result_raw['date_event'] - result_raw['first_session_date']).dt.days

result_grouped = result_raw.pivot_table(
    index=dimensions, columns='lifetime', values='user_id', aggfunc='nunique')

cohort_sizes = (
    result_raw.groupby(dimensions)
```

```

        .agg({'user_id': 'nunique'})
        .rename(columns={'user_id': 'cohort_size'}))

result_grouped = cohort_sizes.merge(
    result_grouped, on=dimensions, how='left'
).fillna(0)
result_grouped = result_grouped.div(result_grouped['cohort_size'], axis=0)

# исключаем все лайфтаймы, превышающие горизонт анализа
result_grouped = result_grouped[
    ['cohort_size'] + list(range(horizon_days))]

# восстанавливаем столбец с размерами когорт
result_grouped['cohort_size'] = cohort_sizes

if not isinstance(result_grouped.index, pd.MultiIndex):
    result_grouped.index = result_grouped.index.strftime('%Y-%m-%d')

else:
    pass
# возвращаем таблицу удержания и сырые данные
return result_grouped

```

```

In [4]: # Функция построения тепловой карты удержания
def heatmap_of_retention(data):
    """
        Функция для построения тепловой карты удержания.
        Args:
        Функция на вход принимает DataFrame обработанный в функции get_retention

        Returns:
        None, результатом работы становится тепловая карта.
    """

    plt.figure(figsize=(15, 6)) # задаём размер графика
    sns.heatmap(
        data.drop(columns=['cohort_size', 0]), # удаляем размеры когорт
        annot=True, # включаем подписи
        fmt='.2%', # переводим значения в проценты
        cmap="crest",
    )
    plt.title('Тепловая карта удержания') # название графика
    plt.show()

```

```

In [5]: def plot_retention_dynamics(source_list: list, result_grouped):
    """
        Функция для построения динамики удержания пользователей по каналам привл

        Args:
        Функция на вход 2 параметра: source_list список с уникальными каналами п
        DataFrame обработанный в функции get_retention, принимает 2-ую возвращае

        Returns:
        None, результатом работы становится 3 линейных графика динамики удержани
    """

    report = result_grouped.drop(columns=['cohort_size', 0])

```

```

# Определяем количество строк и столбцов для подграфиков.
num_cols = 1
num_rows = len(source_list)

# Создаем фигуру и набор подграфиков
fig, axes = plt.subplots(nrows=num_rows, ncols=num_cols, figsize=(15, 5 * num_rows))

for j, device in enumerate(full_mobile_dataset['source'].unique()):

    ax = axes[j] # Выбираем текущий подграфик
    ax.set_ylim(0, 0.30)
    # Строим график на текущем подграфике
    report.query('source == @device').droplevel(['source']).plot(grid=True,

    # Настраиваем подграфик
    ax.set_xlabel('Дата привлечения')
    ax.set_title(f'Удержание для {device}')
plt.tight_layout()
plt.show()

```

```

In [6]: def consider_zero_sessions(y_axis, data):
        """ Функция для графического (столбчатая диаграмма)
        изображения данных о нулевых сессиях

        Args:
            Функция на вход 2 параметра: y_axis параметр для оси Y,
            data с отфильтрованными данными по длительности сессий и со сформированным

        Returns:
            None, результатом работы становится барплот по количеству того или иного
            """"

        plt.figure(figsize=(17, 8))
        sns.barplot(x=data['event_name'],
                    y=data[y_axis],
                    data=data, palette="ch:start=.2,rot=-.3")
        plt.xlabel('Действие ')
        plt.ylabel('Количество')
        plt.title('График количества действий с нулевой длительностью')
        plt.xticks(rotation=45, ha='right')
        plt.show()

```

Переменные объявленные и используемые в процессе анализа.

mobile_sources - Исходный датасет с данными источника установки приложения

mobile_dataset - Исходных датасет с данными действий пользователей

full_mobile_dataset - Объединенный датасет с данными пользователей и

источником установки приложения в процессе работы наполняется лайтамами,

номерами сессии, днём недели каждой сессии. **cohort_size** - когорты с размерами

result_grouped - общая переменная для расчёта удержания **user_sessions_data** -

пользователи с количеством сессий и с номером последней сессии **zero_sessions** -

все сессии с нулевой длительностью

```

In [7]: sns.set_style('whitegrid')

```

```
In [8]: # Сохраняем представленные данные в библиотеку
mobile_sources = pd.read_csv('mobile_sources.csv')
mobile_dataset = pd.read_csv('mobile_dataset.csv')
```

```
In [9]: # Изучим общую информацию о представленных данных
print(mobile_dataset.info())
mobile_dataset.head(2)
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 74197 entries, 0 to 74196
Data columns (total 3 columns):
#   Column      Non-Null Count  Dtype
---  -
0   event.time   74197 non-null  object
1   event.name   74197 non-null  object
2   user.id      74197 non-null  object
dtypes: object(3)
memory usage: 1.7+ MB
None
```

```
Out[9]:
```

	event.time	event.name	user.id
0	2019-10-07 00:00:00.431357	advert_open	020292ab-89bc-4156-9acf-68bc2783f894
1	2019-10-07 00:00:01.236320	tips_show	020292ab-89bc-4156-9acf-68bc2783f894

```
In [10]: print(mobile_sources.info())
mobile_sources.head(2)
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4293 entries, 0 to 4292
Data columns (total 2 columns):
#   Column  Non-Null Count  Dtype
---  -
0   userId  4293 non-null   object
1   source  4293 non-null   object
dtypes: object(2)
memory usage: 67.2+ KB
None
```

```
Out[10]:
```

	userId	source
0	020292ab-89bc-4156-9acf-68bc2783f894	other
1	cf7eda61-9349-469f-ac27-e5b6f5ec475c	yandex

Предварительно можно увидеть отсутствие пропусков, так как количество строк соответствует длине полей. Обнаружено несоответствие типа данных для поля "event.time", содержащего дату и время. Считаю необходимым для дальнейшего анализа дополнительно разделить дату и время события на разные поля. Кроме того, необходимо переименовать поля и привести их к стандартному способу наименования. В поле с датой также видно, что данные представлены с точностью до миллисекунд, которые не несут ценности для анализа, поэтому достаточно ограничиться секундами. Так же для дальнейшего удобства считаю необходимым слить представленные таблицы в одну.

```
In [11]: # Проверим на наличие пропусков
print(f'Пропуски в таблице "mobile_dataset":\n{mobile_dataset.isna().sum()} \n')
print(f'Пропуски в таблице "mobile_sources":\n{mobile_sources.isna().sum()} \n')
```

```
Пропуски в таблице "mobile_dataset":
event.time      0
event.name      0
user.id         0
dtype: int64
```

```
Пропуски в таблице "mobile_sources":
userId         0
source         0
dtype: int64
```

Явная проверка на наличие пропусков лишь подтверждает ранее высказанное предположение об их отсутствии.

```
In [12]: # Наличие явных дубликатов
print(f'Пропуски в таблице "mobile_dataset":\n{mobile_dataset.duplicated().sum()} \n')
print(f'Пропуски в таблице "mobile_sources":\n{mobile_sources.duplicated().sum()} \n')
```

```
Пропуски в таблице "mobile_dataset":
0
```

```
Пропуски в таблице "mobile_sources":
0
```

Дубликаты отсутствуют. Считаю важным проверить сразу после удаления миллисекунд в дате.

```
In [13]: # Преобразую столбец, содержащий дату и время, в формат datetime64, отсекая миллисекунды
mobile_dataset['event.time'] = pd.to_datetime(mobile_dataset['event.time'], format='%Y-%m-%d %H:%M:%S', errors='coerce')
mobile_dataset.head(2)
```

```
Out[13]:
```

	event.time	event.name	user.id
0	2019-10-07 00:00:00	advert_open	020292ab-89bc-4156-9acf-68bc2783f894
1	2019-10-07 00:00:01	tips_show	020292ab-89bc-4156-9acf-68bc2783f894

```
In [14]: # Повторная проверка явных дубликатов
print(f'Пропуски в таблице "mobile_dataset":\n{mobile_dataset.duplicated().sum()} \n')
print(f'Пропуски в таблице "mobile_sources":\n{mobile_sources.duplicated().sum()} \n')
```

```
Пропуски в таблице "mobile_dataset":
1143
```

```
Пропуски в таблице "mobile_sources":
0
```

По итог общего знакомства данных можно сделать несколько заключений:

- Неподходящие наименования столбцов;
- Несоответствие типов полей;

- Необходимо объединить таблицы для дальнейшего удобства анализа;
- Необходимо обработать пропуски, возникшие при упрощении временных данных;
- Разделение даты и времени в разные поля таблицы.

Обработка данных: форматирование, обработка пропусков, удаление дубликатов.

```
In [15]: # Убираю точку из наименования полей и меняю ее на нижнее подчеркивание в табли
for ind, elem in enumerate(mobile_dataset.columns):
    mobile_dataset.rename(columns={elem: '_' + elem.split('.')}, inplace=True)

# Переименовываю поле в таблице mobile_sources и приводим к "змеиному типу":
mobile_sources.rename(columns={'userId': 'user_id'}, inplace=True)
```

```
In [16]: # Проверяю изменились ли наименования полей
print(mobile_dataset.columns)
mobile_sources.columns
```

Index(['event_time', 'event_name', 'user_id'], dtype='object')

Out[16]: Index(['user_id', 'source'], dtype='object')

```
In [17]: # Удаляем обнаруженные пропуски
mobile_dataset = mobile_dataset.drop_duplicates().reset_index(drop=True)
mobile_dataset
```

Out[17]:

	event_time	event_name	user_id
0	2019-10-07 00:00:00	advert_open	020292ab-89bc-4156-9acf-68bc2783f894
1	2019-10-07 00:00:01	tips_show	020292ab-89bc-4156-9acf-68bc2783f894
2	2019-10-07 00:00:02	tips_show	cf7eda61-9349-469f-ac27-e5b6f5ec475c
3	2019-10-07 00:00:07	tips_show	020292ab-89bc-4156-9acf-68bc2783f894
4	2019-10-07 00:00:56	advert_open	cf7eda61-9349-469f-ac27-e5b6f5ec475c
...
73049	2019-11-03 23:53:29	tips_show	28fccdf4-7b9e-42f5-bc73-439a265f20e9
73050	2019-11-03 23:54:00	tips_show	28fccdf4-7b9e-42f5-bc73-439a265f20e9
73051	2019-11-03 23:56:57	search_1	20850c8f-4135-4059-b13b-198d3ac59902
73052	2019-11-03 23:57:06	tips_show	28fccdf4-7b9e-42f5-bc73-439a265f20e9
73053	2019-11-03 23:58:12	tips_show	28fccdf4-7b9e-42f5-bc73-439a265f20e9

73054 rows × 3 columns

```
In [18]: mobile_dataset.info()
```



```

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 73054 entries, 0 to 73053
Data columns (total 3 columns):
#   Column      Non-Null Count  Dtype
---  -
0   event_time  73054 non-null  datetime64[ns]
1   event_name  73054 non-null  object
2   user_id     73054 non-null  object
dtypes: datetime64[ns](1), object(2)
memory usage: 1.7+ MB

```

```

In [19]: # Перименую события show_contacts и contacts_show к единому имени события 'cont
# Для более наглядного представления информации приведем все события search к ед

mobile_dataset.loc[mobile_dataset['event_name'].str.startswith('show_'), 'event_
mobile_dataset.loc[mobile_dataset['event_name'].str.startswith('search_'), 'even

```

Пропуски удалили, поле с датой соответствует требуемому типу данных, наименование полей приведено к стандартному способу.

Добавление новых полей, объединение таблиц

```

In [20]: # Для удобства изучения и обработки данных добавлю дополнительные поля содержащи
mobile_dataset['date_event'] = mobile_dataset['event_time'].dt.normalize()
mobile_dataset['time_event'] = mobile_dataset['event_time'].dt.time

```

```

In [21]: # Объединяем таблицу и сохраняем в новую переменную "full_mobile_dataset"
full_mobile_dataset = mobile_dataset.merge(mobile_sources[['user_id', 'source']])

```

```

In [22]: # Добавим новое поле для каждого пользователя с датой и временем первой сессии
full_mobile_dataset = full_mobile_dataset.merge(mobile_dataset.groupby('user_id')

```

```

In [23]: full_mobile_dataset

```

Out[23]:

	event_time	event_name	user_id	date_event	time_event	source	session
0	2019-10-07 00:00:00	advert_open	020292ab- 89bc-4156- 9acf- 68bc2783f894	2019-10- 07	00:00:00	other	2019-
1	2019-10-07 00:00:01	tips_show	020292ab- 89bc-4156- 9acf- 68bc2783f894	2019-10- 07	00:00:01	other	2019-
2	2019-10-07 00:00:02	tips_show	cf7eda61- 9349-469f- ac27- e5b6f5ec475c	2019-10- 07	00:00:02	yandex	2019-
3	2019-10-07 00:00:07	tips_show	020292ab- 89bc-4156- 9acf- 68bc2783f894	2019-10- 07	00:00:07	other	2019-
4	2019-10-07 00:00:56	advert_open	cf7eda61- 9349-469f- ac27- e5b6f5ec475c	2019-10- 07	00:00:56	yandex	2019-
...
73049	2019-11-03 23:53:29	tips_show	28fccdf4- 7b9e-42f5- bc73- 439a265f20e9	2019-11- 03	23:53:29	google	2019-
73050	2019-11-03 23:54:00	tips_show	28fccdf4- 7b9e-42f5- bc73- 439a265f20e9	2019-11- 03	23:54:00	google	2019-
73051	2019-11-03 23:56:57	search	20850c8f- 4135-4059- b13b- 198d3ac59902	2019-11- 03	23:56:57	google	2019-
73052	2019-11-03 23:57:06	tips_show	28fccdf4- 7b9e-42f5- bc73- 439a265f20e9	2019-11- 03	23:57:06	google	2019-
73053	2019-11-03 23:58:12	tips_show	28fccdf4- 7b9e-42f5- bc73- 439a265f20e9	2019-11- 03	23:58:12	google	2019-

73054 rows × 7 columns



In [24]:

```
# Меняю порядок полей в датафрейме, для более удобного восприятия таблицы
full_mobile_dataset = full_mobile_dataset[['user_id', 'event_name', 'source', 'e
full_mobile_dataset
```

Out[24]:

	user_id	event_name	source	event_time	session_start	date_event	time_
0	020292ab-89bc-4156-9acf-68bc2783f894	advert_open	other	2019-10-07 00:00:00	2019-10-07	2019-10-07	00
1	020292ab-89bc-4156-9acf-68bc2783f894	tips_show	other	2019-10-07 00:00:01	2019-10-07	2019-10-07	00
2	cf7eda61-9349-469f-ac27-e5b6f5ec475c	tips_show	yandex	2019-10-07 00:00:02	2019-10-07	2019-10-07	00
3	020292ab-89bc-4156-9acf-68bc2783f894	tips_show	other	2019-10-07 00:00:07	2019-10-07	2019-10-07	00
4	cf7eda61-9349-469f-ac27-e5b6f5ec475c	advert_open	yandex	2019-10-07 00:00:56	2019-10-07	2019-10-07	00
...
73049	28fccdf4-7b9e-42f5-bc73-439a265f20e9	tips_show	google	2019-11-03 23:53:29	2019-10-16	2019-11-03	23
73050	28fccdf4-7b9e-42f5-bc73-439a265f20e9	tips_show	google	2019-11-03 23:54:00	2019-10-16	2019-11-03	23
73051	20850c8f-4135-4059-b13b-198d3ac59902	search	google	2019-11-03 23:56:57	2019-10-27	2019-11-03	23
73052	28fccdf4-7b9e-42f5-bc73-439a265f20e9	tips_show	google	2019-11-03 23:57:06	2019-10-16	2019-11-03	23
73053	28fccdf4-7b9e-42f5-bc73-439a265f20e9	tips_show	google	2019-11-03 23:58:12	2019-10-16	2019-11-03	23

73054 rows × 7 columns



Выводы по предобработке.

Успешно изучили и подготовили данные для дальнейшего исследовательского анализа. По итогам предобработки удалось сохранить 99.9 % итоговой информации.

Анализ удержания (Retention Rate)

```
In [25]: # Сортируем данные по пользователям и датам их сессий
full_mobile_dataset = full_mobile_dataset.sort_values(['user_id', 'event_time'])
full_mobile_dataset
```

Out[25]:

	user_id	event_name	source	event_time	session_start	date_event	time
0	0001b1d5-b74a-4cbf-aeb0-7df5947bf349	tips_show	other	2019-10-07 13:39:45	2019-10-07	2019-10-07	1
1	0001b1d5-b74a-4cbf-aeb0-7df5947bf349	tips_show	other	2019-10-07 13:40:31	2019-10-07	2019-10-07	1
2	0001b1d5-b74a-4cbf-aeb0-7df5947bf349	tips_show	other	2019-10-07 13:41:05	2019-10-07	2019-10-07	1
3	0001b1d5-b74a-4cbf-aeb0-7df5947bf349	tips_show	other	2019-10-07 13:43:20	2019-10-07	2019-10-07	1
4	0001b1d5-b74a-4cbf-aeb0-7df5947bf349	tips_show	other	2019-10-07 13:45:30	2019-10-07	2019-10-07	1
...
73049	fffb9e79-b927-4dbb-9b48-7fd09b23a62b	tips_show	google	2019-11-03 15:51:23	2019-10-12	2019-11-03	1
73050	fffb9e79-b927-4dbb-9b48-7fd09b23a62b	contacts_show	google	2019-11-03 15:51:57	2019-10-12	2019-11-03	1
73051	fffb9e79-b927-4dbb-9b48-7fd09b23a62b	tips_show	google	2019-11-03 16:07:40	2019-10-12	2019-11-03	1
73052	fffb9e79-b927-4dbb-9b48-7fd09b23a62b	tips_show	google	2019-11-03 16:08:18	2019-10-12	2019-11-03	1
73053	fffb9e79-b927-4dbb-9b48-7fd09b23a62b	tips_show	google	2019-11-03 16:08:25	2019-10-12	2019-11-03	1

73054 rows × 7 columns



In [26]:

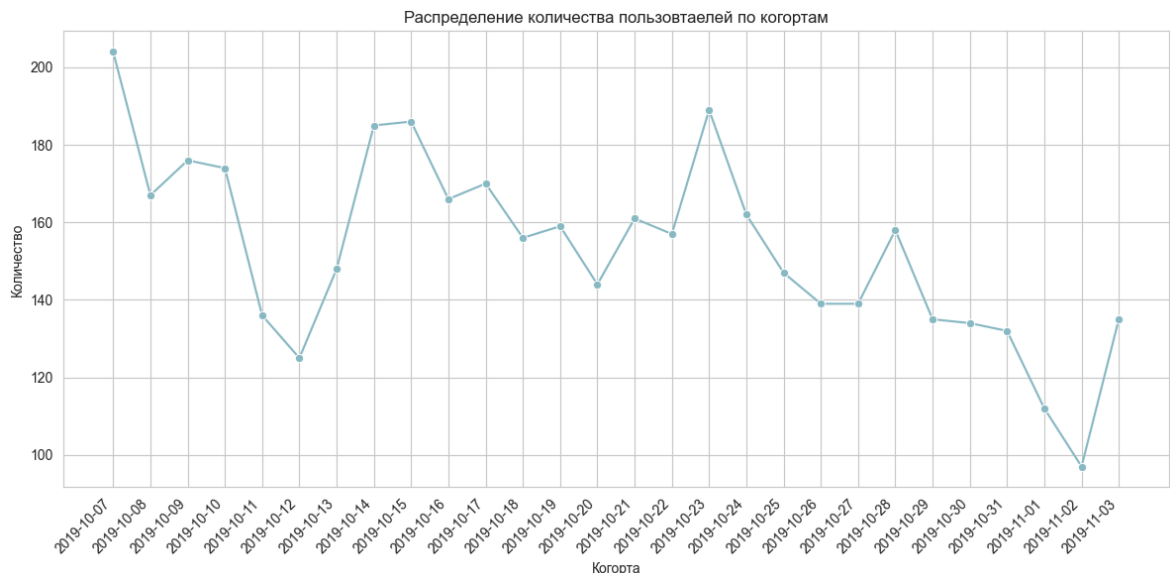
```
# Для избежания путаницы форматирую наименования столбцов, для первой сессии 'se
# Для события с полным временным представлением 'event_time' => 'full_datetime'
full_mobile_dataset.rename(columns={'session_start': 'first_session', 'event_time'
```

Определение когорт пользователей.

В процессе предварительной обработки данных для каждого пользователя был создан столбец "session_start", содержащий дату его первой активности в приложении. Эта дата принята в качестве определяющего критерия для отнесения пользователя к конкретной когорте.

```
In [27]: # Посмотрим размер этих когорт
cohort_size = full_mobile_dataset.groupby('first_session', as_index=False)['user_id'].count().rename(columns={'user_id': 'cohort_size'}, inplace=True)
```

```
In [28]: plt.figure(figsize=(12, 6)) # Задаем размер графика
sns.lineplot(x="first_session", y="cohort_size", data=cohort_size, marker='o', color='teal')
plt.title('Распределение количества пользователей по когортам')
plt.xlabel('Когорта')
plt.ylabel('Количество')
plt.xticks(rotation=45, ha='right')
plt.tight_layout()
plt.show()
```



На представленном графике выше видим распределение пользователей по когортам, сформированным по дате регистрации в период с 7 октября по 3 ноября 2019 года.

Можем увидеть значительные колебания численности пользователей в разных когортах, с самым высоким пиком приходящимся на день начала анализа 7 октября, так же в середине месяца 15 и 16 октября, а также пик 23 октября. Далее можем наблюдать нисходящий тренд к концу периода с небольшим подъемом в последний день анализа. Самые же провальные дни приходятся на первую и последнюю неделю анализа 12 октября и 2 ноября соответственно.

Расчет Retention Rate (1/7/14 дней).

```
In [29]: # Задаём максимальную дату анализа
max_date = full_mobile_dataset['date_event'].max()
```

```
In [30]: # Рассмотрим динамику удержания пользователей, проанализировав данные за 1-дневн
result_grouped = get_retention(profiles=full_mobile_dataset, observation_date=ma
                                horizon_days=2, dimensions=['first_session'],
```

```
In [31]: full_mobile_dataset
```

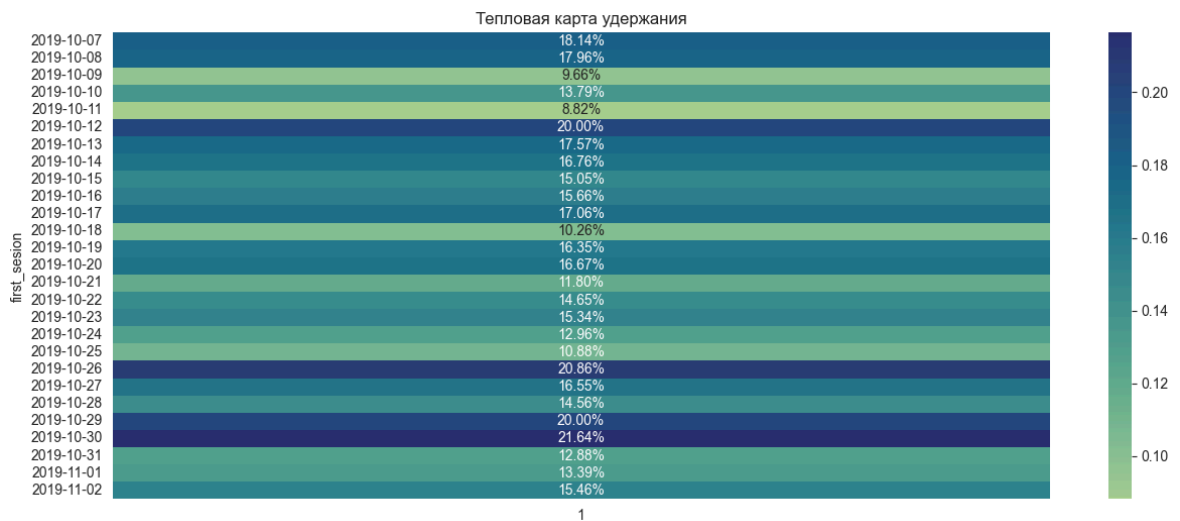
Out[31]:

	user_id	event_name	source	full_datetime	first_session	date_event	tim
0	0001b1d5-b74a-4cbf-aeb0-7df5947bf349	tips_show	other	2019-10-07 13:39:45	2019-10-07	2019-10-07	
1	0001b1d5-b74a-4cbf-aeb0-7df5947bf349	tips_show	other	2019-10-07 13:40:31	2019-10-07	2019-10-07	
2	0001b1d5-b74a-4cbf-aeb0-7df5947bf349	tips_show	other	2019-10-07 13:41:05	2019-10-07	2019-10-07	
3	0001b1d5-b74a-4cbf-aeb0-7df5947bf349	tips_show	other	2019-10-07 13:43:20	2019-10-07	2019-10-07	
4	0001b1d5-b74a-4cbf-aeb0-7df5947bf349	tips_show	other	2019-10-07 13:45:30	2019-10-07	2019-10-07	
...
73049	fffb9e79-b927-4dbb-9b48-7fd09b23a62b	tips_show	google	2019-11-03 15:51:23	2019-10-12	2019-11-03	
73050	fffb9e79-b927-4dbb-9b48-7fd09b23a62b	contacts_show	google	2019-11-03 15:51:57	2019-10-12	2019-11-03	
73051	fffb9e79-b927-4dbb-9b48-7fd09b23a62b	tips_show	google	2019-11-03 16:07:40	2019-10-12	2019-11-03	
73052	fffb9e79-b927-4dbb-9b48-7fd09b23a62b	tips_show	google	2019-11-03 16:08:18	2019-10-12	2019-11-03	
73053	fffb9e79-b927-4dbb-9b48-7fd09b23a62b	tips_show	google	2019-11-03 16:08:25	2019-10-12	2019-11-03	

73054 rows × 7 columns



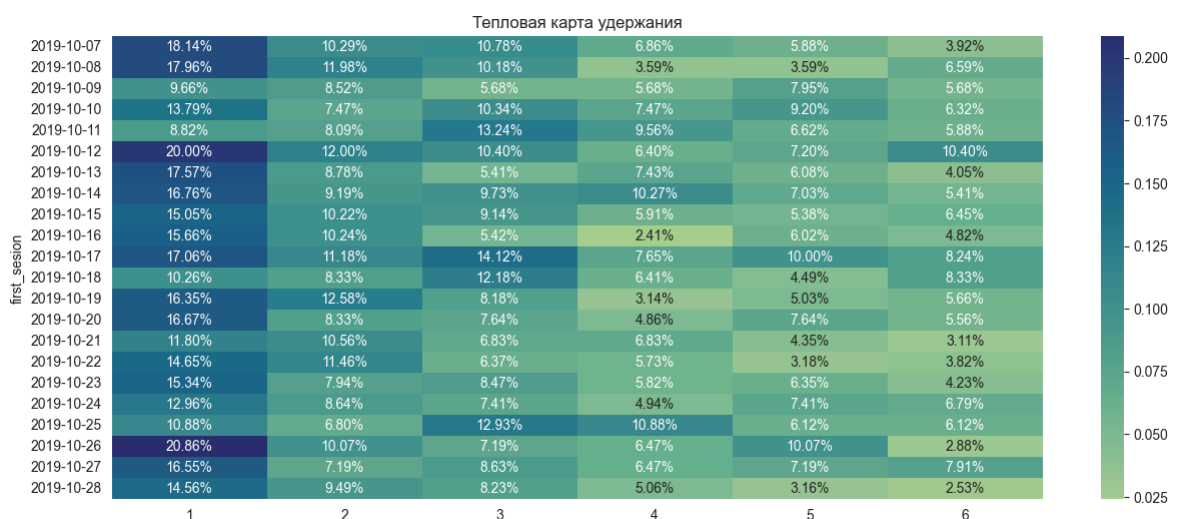
In [32]: heatmap_of_retention(result_grouped)



- Высокие значения: В начале периода и ближе к концу можем увидеть дни с более высоким удержанием ближе 20 %.
- Низкие значения: Есть заметные провалы в удержании, например, 2019-10-09, 2019-10-11 и 2019-10-25. В эти дни удержание падало ниже 10%.
- Общий тренд: Не видно явного восходящего или нисходящего тренда, что говорит о том, что удержание, в основном, стабильно, но с заметными колебаниями от 9 до 20 % за весь период

In [33]: `# Рассмотрим динамику удержания пользователей, проанализировав данные за 7-дневн`
`result_grouped = get_retention(profiles=full_mobile_dataset, observation_date=ma`
`horizon_days=7, dimensions=['first_session'],`

In [34]: `heatmap_of_retention(result_grouped)`

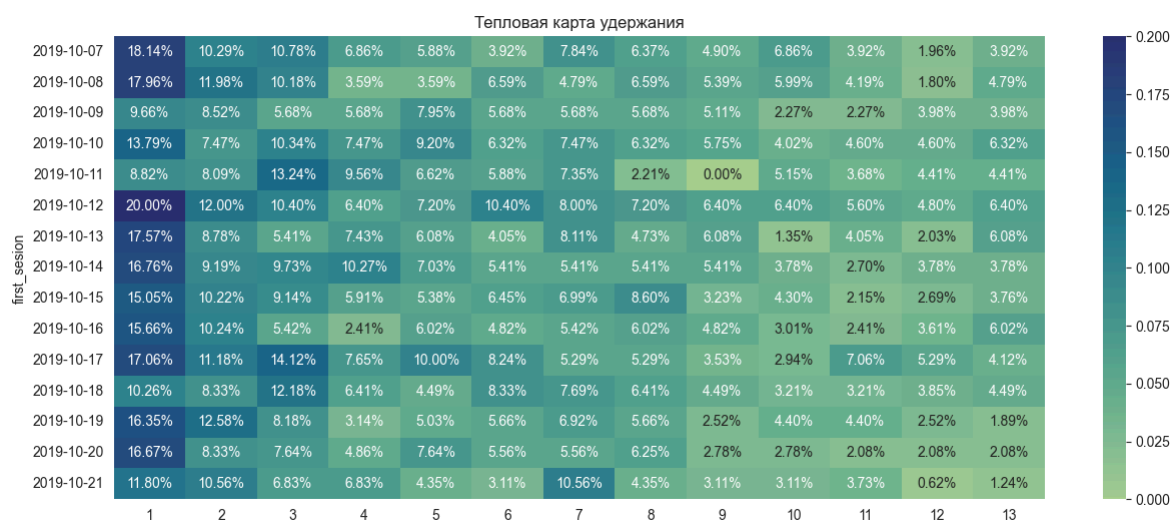


- На 7 день удержания мы видим быстрый спад, удержание значительно снижается уже на второй день лайфтайма. Затем снижение замедляется. Это может указывать, что многие пользователи уходят на 2-3 день после импользования приложения.
- К 7 дню удержание падает до очень низких значений ниже 10% в отдельных случаях ниже 5%.

- Для дней повышенного удержания для лайфтайма 1 не видим, повышенного удержания в последующие дни для этих дат, что говорит о том, что причины высокого удержания в первый день не оказывают долгосрочного эффекта.
- Разные когорты показывают немного отличающееся удержание, но общая тенденция (быстрый спад) сохраняется для всех. Нет закономерности для удержания в первый и последний день, например сильнее когорты 12 октября и 26 октября на 7 день удержания показывают диаметрально разное удержание 10.4% и 2.88 соответственно.

```
In [35]: # Рассмотрим динамику удержания пользователей, проанализировав данные за 14-днев
result_grouped = get_retention(profiles=full_mobile_dataset, observation_date=ma
                                horizon_days=14, dimensions=['first_session'],
```

```
In [36]: heatmap_of_retention(result_grouped)
```



- Общий уровень удержания очень низкий. Большинство ячеек после 7 дня и далее, имеют очень бледный цвет, что говорит о том, что лишь малый процент пользователей остаётся активным спустя неделю после первого использования. К 13 дню (последняя колонка) удержание в большинстве случаев составляет единицы процентов.
- Не видно чётких закономерностей, например, какого-то дня недели, когда удержание стабильно выше или ниже.

ОБЩИЙ ВЫВОД

Удержание пользователей в приложении резко падает с течением времени, демонстрируя нестабильность и отсутствие долгосрочной лояльности. Хотя в отдельные дни наблюдаются пики привлечения, они не приводят к улучшению удержания в последующие дни.

- Удержание 1-го дня: Колеблется от 9% до 20%, без выраженного тренда. Пики удержания не обеспечивают долгосрочного эффекта.
- Удержание 7-го дня: Сильный спад в второй и последующие дни. К концу недели удержание падает до крайне низких значений. Отсутствует корреляция между

успешным привлечением и удержанием на 7-й день.

- Удержание 14-го дня: К концу второй недели лишь единицы процентов пользователей остаются активными. Какие-либо закономерности в долгосрочном удержании отсутствуют.

Анализ Retention Rate по источникам привлечения.

```
In [37]: # Рассмотрим динамику удержания пользователей, проанализировав данные за 7-дневн
result_grouped = get_retention(profiles=full_mobile_dataset, observation_date=ma
                                horizon_days=7, dimensions=['first_session', ']
```

```
In [38]: plot_retention_dynamics(full_mobile_dataset['source'].unique(), result_grouped)
```



Графики показывают динамику 7-дневного удержания пользователей, привлеченных из разных каналов (Other, Yandex, Google)

Основные моменты:

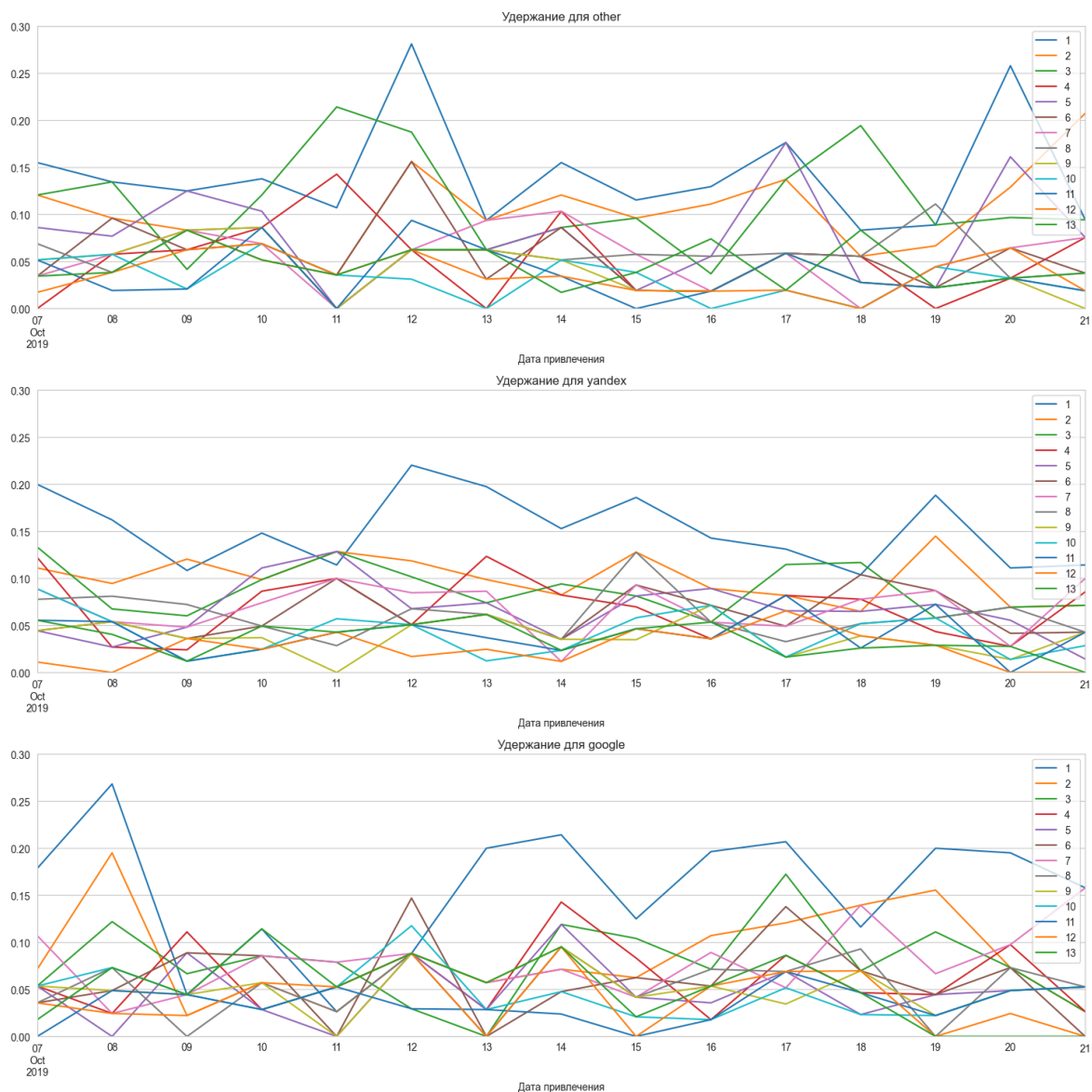
Все графики характеризуются высокой изменчивостью. Удержание меняется день ото дня, что указывает на зависимость от внешних факторов, событий или

конкретных кампаний.

- Уровень удержания в основном не превышает 20%, а часто опускается ниже 10%.
- Нельзя выделить канал с однозначно лучшим удержанием. Динамика удержания для Yandex и Google в целом схожа, но в отдельные дни какой-то из каналов может показывать лучшие результаты.
- Канал Yandex показывает более низкие проценты удержания, чуть превышают 20%, для Google и Other видим пики превышающие 25% порога.
- В данном анализе нельзя выделить канал с однозначно лучшим удержанием. Динамика удержания в целом схожа, чуть хуже для Yandex, но в отдельные дни какой-то из каналов может показывать лучшие результаты.
- На всех каналах отчётливо прослеживается выраженная линия динамики привлечения для первой когорты. Остальные когорты демонстрируют схожие показатели с редкими, кратковременными всплесками активности, не формирующими устойчивой тенденции.

```
In [39]: # Рассмотрим динамику удержания пользователей, проанализировав данные за 14-дней
result_grouped = get_retention(profiles=full_mobile_dataset, observation_date=ma
                                horizon_days=14, dimensions=['first_session',
```

```
In [40]: plot_retention_dynamics(full_mobile_dataset['source'].unique(), result_grouped)
```



- Удержание на 14-й день для всех каналов в основном не превышает 15% и часто находится ниже 10%.
- Динамика удержания крайне нестабильна, с резкими колебаниями для всех каналов в зависимости от дня привлечения.
- Нет явной корреляции между датой привлечения и уровнем удержания.

- Other: Характеризуется наибольшей изменчивостью и наименее предсказуемым поведением когорт. Встречаются резкие пики и провалы.
- Yandex: В целом демонстрирует более стабильное удержание по сравнению с Other, но также подвержен колебаниям. Максимальные значения удержания несколько ниже, чем у Google.
- Google: Показывает более плавную динамику по сравнению с Other

Ключевые моменты:

- Для канала Other наблюдается самый большой разброс значений, что указывает на неоднородный трафик.
- Канал Yandex имеет больше пиков удержания, но общая динамика схожа с Google.

В целом, удержание пользователей плохое и нестабильное для всех рассматриваемых каналов.

Выводы по удержанию.

Удержание пользователей в приложении характеризуется очень низким уровнем и высокой нестабильностью. Большинство пользователей покидают приложение в течение первой недели, и к концу второй недели активными остаются лишь единицы процентов. Динамика удержания крайне изменчива, с резкими колебаниями в зависимости от дня привлечения. Пики удержания в отдельные дни не приводят к долгосрочному улучшению. Каналы привлечения демонстрируют разную степень нестабильности, при этом канал Other характеризуется наибольшей изменчивостью, а Yandex и Google - более плавными изменениями, однако все они показывают в целом низкое удержание на 14-й день, не превышающее в большинстве случаев 15%.

РЕКОМЕНДАЦИИ Необходимо выявить причины низкого удержания и разработать стратегию для повышения вовлеченности пользователей, особенно в первые дни после первого использования. Необходимо провести анализ эффективности каналов привлечения, чтобы оптимизировать расходы и улучшить качество привлекаемого трафика, в частности, тщательно изучить состав канала Other.

Анализ сессий

Определение таймаута сессии и формирование сессий для каждого пользователя

```
In [41]: # Добавляем в датасет длительность перерыва сессий и её номер
full_mobile_dataset = calculate_session(full_mobile_dataset, session_break=3600)
full_mobile_dataset
```

Out[41]:

	user_id	event_name	source	full_datetime	first_session	date_event	tim
0	0001b1d5-b74a-4cbf-aeb0-7df5947bf349	tips_show	other	2019-10-07 13:39:45	2019-10-07	2019-10-07	
1	0001b1d5-b74a-4cbf-aeb0-7df5947bf349	tips_show	other	2019-10-07 13:40:31	2019-10-07	2019-10-07	
2	0001b1d5-b74a-4cbf-aeb0-7df5947bf349	tips_show	other	2019-10-07 13:41:05	2019-10-07	2019-10-07	
3	0001b1d5-b74a-4cbf-aeb0-7df5947bf349	tips_show	other	2019-10-07 13:43:20	2019-10-07	2019-10-07	
4	0001b1d5-b74a-4cbf-aeb0-7df5947bf349	tips_show	other	2019-10-07 13:45:30	2019-10-07	2019-10-07	
...
73049	fffb9e79-b927-4dbb-9b48-7fd09b23a62b	tips_show	google	2019-11-03 15:51:23	2019-10-12	2019-11-03	
73050	fffb9e79-b927-4dbb-9b48-7fd09b23a62b	contacts_show	google	2019-11-03 15:51:57	2019-10-12	2019-11-03	
73051	fffb9e79-b927-4dbb-9b48-7fd09b23a62b	tips_show	google	2019-11-03 16:07:40	2019-10-12	2019-11-03	
73052	fffb9e79-b927-4dbb-9b48-7fd09b23a62b	tips_show	google	2019-11-03 16:08:18	2019-10-12	2019-11-03	
73053	fffb9e79-b927-4dbb-9b48-7fd09b23a62b	tips_show	google	2019-11-03 16:08:25	2019-10-12	2019-11-03	

73054 rows × 9 columns



In [42]: `full_mobile_dataset['event_name'].value_counts()`

```
Out[42]: event_name
tips_show      39907
photos_show    9352
search         6765
advert_open    6146
contacts_show  4376
map            3749
favorites_add   1414
tips_click     811
contacts_call   534
Name: count, dtype: int64
```

Добавили в таблицу интервалы между стартом сессии (секунды) и так же выделили номер сессии с учётом таймаута сессии 60 мин (3600 сек). На выходе получаем таблицу с новыми полями gap_requests (секунды в перерывах начала сессии) session_num (номер сессии)

Расчет количества и длительности сессий.

```
In [43]: # Группируем сессии по пользователям и её номеру с сохранением старта сессии пер
# Добавляем длительность сессии складывая секунды для каждой группы сессий польз
# игнорируя первое значение, так как оно показывает интервал между разными сессии

session_durations = full_mobile_dataset.groupby(['user_id', 'session_num'])['gap

user_sessions_data = pd.merge(full_mobile_dataset, session_durations, on=['user_
user_sessions_data
```


Out[43]:

	user_id	event_name	source	full_datetime	first_session	date_event	tim
0	0001b1d5-b74a-4cbf-aeb0-7df5947bf349	tips_show	other	2019-10-07 13:39:45	2019-10-07	2019-10-07	
1	0001b1d5-b74a-4cbf-aeb0-7df5947bf349	tips_show	other	2019-10-07 13:40:31	2019-10-07	2019-10-07	
2	0001b1d5-b74a-4cbf-aeb0-7df5947bf349	tips_show	other	2019-10-07 13:41:05	2019-10-07	2019-10-07	
3	0001b1d5-b74a-4cbf-aeb0-7df5947bf349	tips_show	other	2019-10-07 13:43:20	2019-10-07	2019-10-07	
4	0001b1d5-b74a-4cbf-aeb0-7df5947bf349	tips_show	other	2019-10-07 13:45:30	2019-10-07	2019-10-07	
...	
73049	fffb9e79-b927-4dbb-9b48-7fd09b23a62b	tips_show	google	2019-11-03 15:51:23	2019-10-12	2019-11-03	
73050	fffb9e79-b927-4dbb-9b48-7fd09b23a62b	contacts_show	google	2019-11-03 15:51:57	2019-10-12	2019-11-03	
73051	fffb9e79-b927-4dbb-9b48-7fd09b23a62b	tips_show	google	2019-11-03 16:07:40	2019-10-12	2019-11-03	
73052	fffb9e79-b927-4dbb-9b48-7fd09b23a62b	tips_show	google	2019-11-03 16:08:18	2019-10-12	2019-11-03	
73053	fffb9e79-b927-4dbb-9b48-7fd09b23a62b	tips_show	google	2019-11-03 16:08:25	2019-10-12	2019-11-03	

73054 rows × 10 columns



In [44]:

```
user_sessions_data
```

Out[44]:

	user_id	event_name	source	full_datetime	first_session	date_event	tim
0	0001b1d5-b74a-4cbf-aeb0-7df5947bf349	tips_show	other	2019-10-07 13:39:45	2019-10-07	2019-10-07	
1	0001b1d5-b74a-4cbf-aeb0-7df5947bf349	tips_show	other	2019-10-07 13:40:31	2019-10-07	2019-10-07	
2	0001b1d5-b74a-4cbf-aeb0-7df5947bf349	tips_show	other	2019-10-07 13:41:05	2019-10-07	2019-10-07	
3	0001b1d5-b74a-4cbf-aeb0-7df5947bf349	tips_show	other	2019-10-07 13:43:20	2019-10-07	2019-10-07	
4	0001b1d5-b74a-4cbf-aeb0-7df5947bf349	tips_show	other	2019-10-07 13:45:30	2019-10-07	2019-10-07	
...
73049	fffb9e79-b927-4dbb-9b48-7fd09b23a62b	tips_show	google	2019-11-03 15:51:23	2019-10-12	2019-11-03	
73050	fffb9e79-b927-4dbb-9b48-7fd09b23a62b	contacts_show	google	2019-11-03 15:51:57	2019-10-12	2019-11-03	
73051	fffb9e79-b927-4dbb-9b48-7fd09b23a62b	tips_show	google	2019-11-03 16:07:40	2019-10-12	2019-11-03	
73052	fffb9e79-b927-4dbb-9b48-7fd09b23a62b	tips_show	google	2019-11-03 16:08:18	2019-10-12	2019-11-03	
73053	fffb9e79-b927-4dbb-9b48-7fd09b23a62b	tips_show	google	2019-11-03 16:08:25	2019-10-12	2019-11-03	

73054 rows × 10 columns



Я целенаправленно использую датасет не с объединенными сессиями, так как в одну сессию попадают разные дейсвия и при объединении не возможно сохранить все совершенные дейсвия пользователем. ДАлее в тераде, где уже наименования

событий неважны, а важна лишь длительность сессий я группирую таблицу по пользователям и по номеру сессии.

```
In [45]: # Проверим наличие нулевых сессий:  
zero_sessions = user_sessions_data[user_sessions_data['session_duration']==0]  
print(f'Количество уникальных пользователей с нулевыми сессиями: {zero_sessions[  
zero_sessions
```

Количество уникальных пользователей с нулевыми сессиями: 988

Всего нулевых сессий: 1782

Out[45]:

	user_id	event_name	source	full_datetime	first_sesion	date_event	tir
105	00157779-810c-4498-9e05-a1e9e3cedf93	contacts_show	yandex	2019-11-03 17:12:09	2019-10-19	2019-11-03	
155	00551e79-152e-4441-9cf7-565d7eb04090	photos_show	yandex	2019-10-29 02:17:12	2019-10-25	2019-10-29	
188	00753c79-ea81-4456-acd0-a47a23ca2fb9	contacts_show	yandex	2019-10-20 14:57:06	2019-10-20	2019-10-20	
312	013bbb57-ca6f-4af3-b586-4a046d3d3dee	map	other	2019-10-11 12:20:48	2019-10-11	2019-10-11	
363	0164902d-7393-47e1-9d5b-0ec4c0171cdc	map	yandex	2019-10-26 18:50:33	2019-10-26	2019-10-26	
...
72816	fffb9e79-b927-4dbb-9b48-7fd09b23a62b	tips_show	google	2019-10-17 11:49:24	2019-10-12	2019-10-17	
73013	fffb9e79-b927-4dbb-9b48-7fd09b23a62b	contacts_show	google	2019-10-30 00:15:43	2019-10-12	2019-10-30	
73014	fffb9e79-b927-4dbb-9b48-7fd09b23a62b	tips_show	google	2019-10-30 00:15:43	2019-10-12	2019-10-30	
73015	fffb9e79-b927-4dbb-9b48-7fd09b23a62b	tips_show	google	2019-10-30 11:31:45	2019-10-12	2019-10-30	
73018	fffb9e79-b927-4dbb-9b48-7fd09b23a62b	tips_show	google	2019-11-02 01:16:48	2019-10-12	2019-11-02	

1782 rows × 10 columns



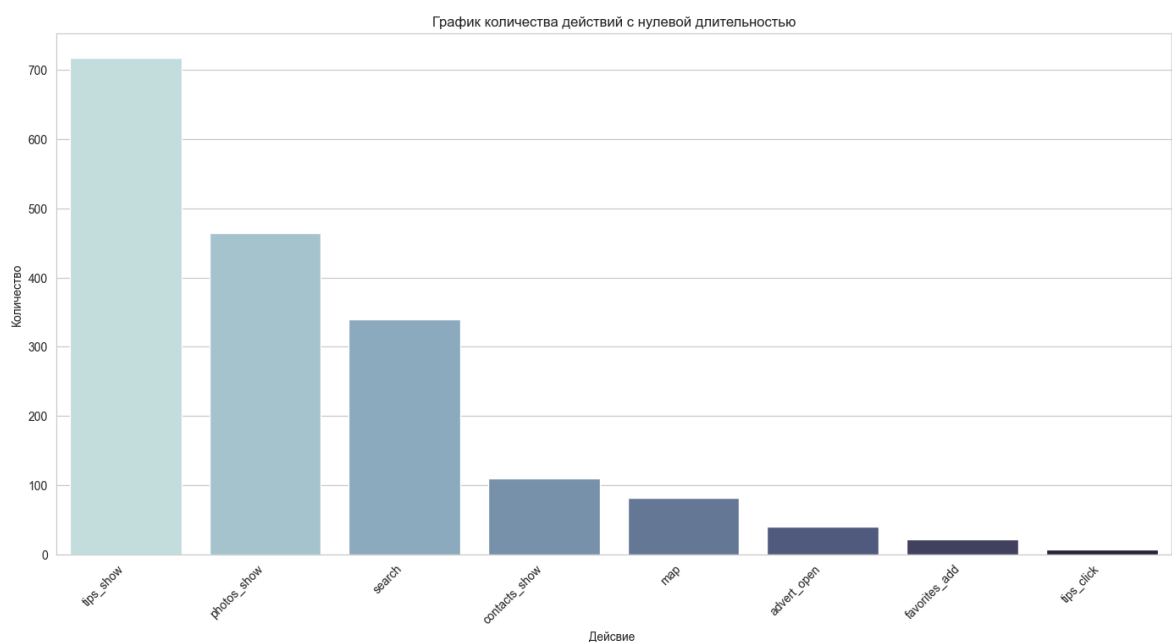
In [46]:

```
grouping_of_zero_sessions = zero_sessions.groupby('event_name', as_index=False).grouping_of_zero_sessions
```

```
Out[46]:
```

	event_name	user_cnt
7	tips_show	717
4	photos_show	464
5	search	340
1	contacts_show	110
3	map	82
0	advert_open	40
2	favorites_add	22
6	tips_click	7

```
In [47]: consider_zero_sessions('user_cnt', grouping_of_zero_sessions)
```



По графику сессий с нулевой длительностью, по моим расчётам, можно увидеть, что самым частым действием для пользователей в этой группе является "tips_show" увидеть рекомендованное. Самым редким "tips_click".

Так же достаточное редкое нулевое по длительности 'favorites_add', но необходимо рассмотреть все действия только если они были последними для пользователя

После поиска "search" может не последовать действий если пользователь не нашёл искомый товар.

Так же для действия 'contacts_show' будет считаться логичным завершением сессий данным действием, так как пользователь нашёл необходимый товар и просмотрел контакт.

"contacts_call" Действие звонок по номеру и вовсе не попало в нулевые.

```
In [48]: # Найдём для каждого пользователя общее число сессий и по соответствию номера,
# общего количества и типа сессии можно будет определить является ли нулевая сессия
```

```
max_values = user_sessions_data.groupby('user_id')['session_num'].max().reset_index()
max_values.rename(columns={'session_num': 'last_num_session'}, inplace=True)

zero_sessions = pd.merge(zero_sessions, max_values, on='user_id', how='left')
```

In [49]: zero_sessions

Out[49]:

	user_id	event_name	source	full_datetime	first_session	date_event	time
0	00157779-810c-4498-9e05-a1e9e3cedf93	contacts_show	yandex	2019-11-03 17:12:09	2019-10-19	2019-11-03	1
1	00551e79-152e-4441-9cf7-565d7eb04090	photos_show	yandex	2019-10-29 02:17:12	2019-10-25	2019-10-29	0
2	00753c79-ea81-4456-acd0-a47a23ca2fb9	contacts_show	yandex	2019-10-20 14:57:06	2019-10-20	2019-10-20	1
3	013bbb57-ca6f-4af3-b586-4a046d3d3dee	map	other	2019-10-11 12:20:48	2019-10-11	2019-10-11	1
4	0164902d-7393-47e1-9d5b-0ec4c0171cdc	map	yandex	2019-10-26 18:50:33	2019-10-26	2019-10-26	1
...
1777	fffb9e79-b927-4dbb-9b48-7fd09b23a62b	tips_show	google	2019-10-17 11:49:24	2019-10-12	2019-10-17	1
1778	fffb9e79-b927-4dbb-9b48-7fd09b23a62b	contacts_show	google	2019-10-30 00:15:43	2019-10-12	2019-10-30	0
1779	fffb9e79-b927-4dbb-9b48-7fd09b23a62b	tips_show	google	2019-10-30 00:15:43	2019-10-12	2019-10-30	0
1780	fffb9e79-b927-4dbb-9b48-7fd09b23a62b	tips_show	google	2019-10-30 11:31:45	2019-10-12	2019-10-30	1
1781	fffb9e79-b927-4dbb-9b48-7fd09b23a62b	tips_show	google	2019-11-02 01:16:48	2019-10-12	2019-11-02	0

1782 rows × 11 columns



In [50]:

```
# Рассмотрим только те сессии которые для пользователя были последними в данном
# После данных событий пользователи не вернулись в приложение.

last_zero_session = zero_sessions[zero_sessions['session_num'] == zero_sessions[
```

```
print(f'Всего с последней нулевой сессией пользователей в приложении: {last_zero}')
print(f'Примерно {round((last_zero_session["user_id"].nunique() / full_mobile_data["user_id"].nunique()) * 100, 2)} % пользователей уходят после одиночной сессии')
```

Всего с последней нулевой сессией пользователей в приложении: 501
Примерно 11.67 % пользователей уходят после одиночной сессии

In [51]: last_zero_session.head()

Out[51]:

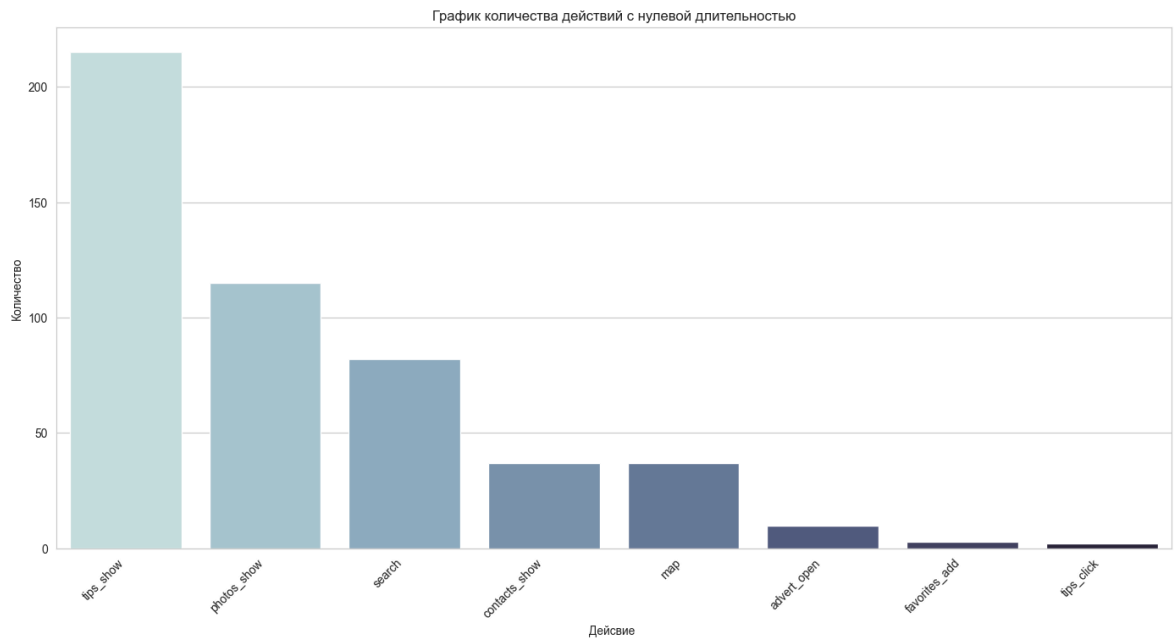
	user_id	event_name	source	full_datetime	first_session	date_event	time_event
0	00157779-810c-4498-9e05-a1e9e3cedf93	contacts_show	yandex	2019-11-03 17:12:09	2019-10-19	2019-11-03	17:12:09
1	00551e79-152e-4441-9cf7-565d7eb04090	photos_show	yandex	2019-10-29 02:17:12	2019-10-25	2019-10-29	02:17:12
8	01d283e1-cb1c-407a-a4e0-9f72f3deecca	photos_show	google	2019-10-20 07:54:01	2019-10-18	2019-10-20	07:54:01
9	02012123-f8ee-40f4-8ccd-c5859f3fbc41	tips_show	google	2019-10-29 09:13:21	2019-10-15	2019-10-29	09:13:21
10	0216e1bd-0984-4ba4-a2b2-c186bf56b6f9	tips_show	yandex	2019-10-27 17:44:36	2019-10-26	2019-10-27	17:44:36

In [52]: last_zero_session = last_zero_session.groupby('event_name', as_index=False).agg({'user_id': 'nunique'})

Out[52]:

	event_name	user_cnt
7	tips_show	215
4	photos_show	115
5	search	82
1	contacts_show	37
3	map	37
0	advert_open	10
2	favorites_add	3
6	tips_click	2

In [53]: consider_zero_sessions('user_cnt', last_zero_session)



В этом графике можем увидеть последнее событие нулевое по длительности, то есть единственное в таймауте сессий.

"advert_open", "favorites_add", "tips_click" крайне редко становятся последними действиями для пользователя в рассматриваемый период так как данные события редки для пользователей в целом.

Так же "contacts_show" редко становится последним

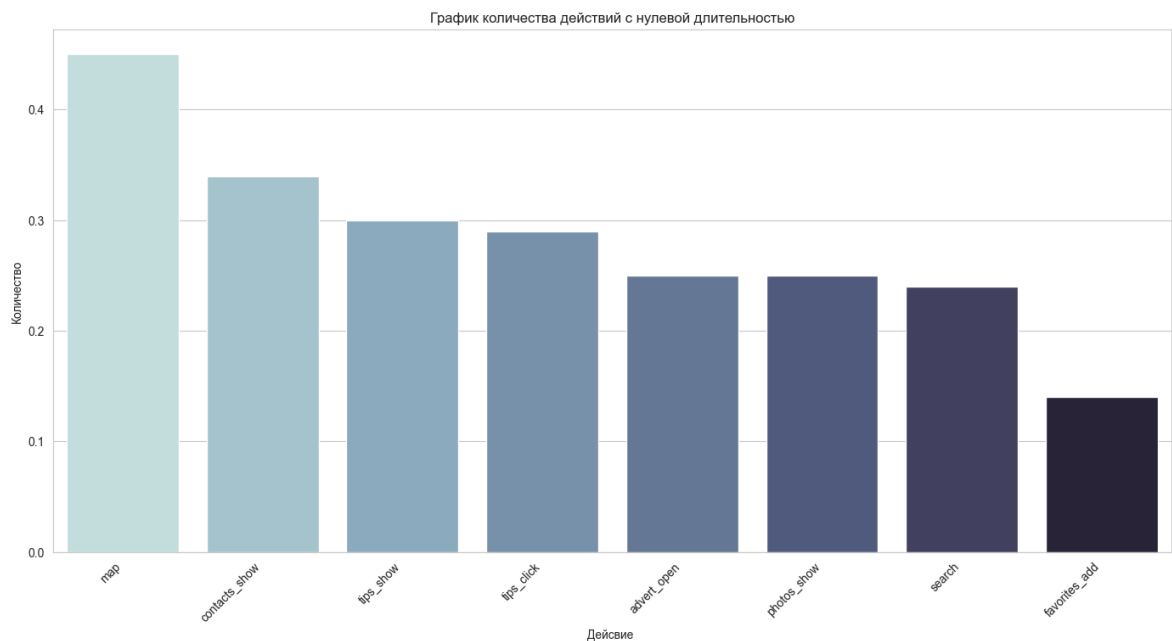
После поиска или просмотра карты пользователи чаще больше не возвращаются к приложению, но самое частое действие "tips_show" увидеть рекомендованное объявление. Совокупность этих факторов может говорить о нерелевантной выдаче рекомендации или результатов поиска в приложении. Необходимо рассмотреть соотношение события было оно последним или не последним для пользователей.

```
In [54]: last_zero_session['all_zero'] = grouping_of_zero_sessions['user_cnt']
last_zero_session['share_recent'] = round(last_zero_session['user_cnt'] / groupi
last_zero_session = last_zero_session.sort_values('share_recent', ascending=False)
last_zero_session
```

Out[54]:

	event_name	user_cnt	all_zero	share_recent
3	map	37	82	0.45
1	contacts_show	37	110	0.34
7	tips_show	215	717	0.30
6	tips_click	2	7	0.29
0	advert_open	10	40	0.25
4	photos_show	115	464	0.25
5	search	82	340	0.24
2	favorites_add	3	22	0.14

```
In [55]: consider_zero_sessions('share_recent', last_zero_session)
```



Опираясь на график и анализ причин нулевых по длительности сессий, можно выделить следующее:

Просмотр карты становится самой частой точкой завершения взаимодействия, где пользователи чаще всего прекращают использование приложения после просмотра. Это может указывать на проблему с отображением местоположения, например, неудобное расположение продавца, выявляемое только на карте. Предложения по улучшению включают добавление фильтра по местоположению или отображение района продавца в карточке объявления. Около 45% пользователей не возвращаются после просмотра карты, что подчеркивает важность решения этой проблемы.

Показ контактов также часто завершает сессии, что может свидетельствовать об успешном поиске. Действительно, в 34% случаев просмотр контактов становится последним действием, вероятно, потому что пользователь нашел подходящий товар и связался с продавцом.

Что касается нулевых по длительности сессий, они могут возникать по разным причинам, включая технические ошибки и особенности поведения пользователя. Из 4293 уникальных пользователей, у 988 встречаются сессии нулевой длительности. Чаще всего это связано с показом рекомендаций и просмотром фото в объявлении, где пользователь может прервать просмотр из-за отсутствия подходящих предложений. При этом, эти действия не часто являются последними в сессии, что говорит о возвращении пользователей после перерыва.

Для дальнейшего анализа длительности сессий имеет смысл исключить нулевые сессии, которые не являются последними для пользователя, поскольку они, вероятно, случайны и после пользователь всё ещё возвращается. Вместо этого, следует сосредоточиться на нулевых сессиях, которые стали последними, так как они несут в себе информацию о факторах, приводящих к окончательному

прекращению использования приложения. Анализ именно этих сессий позволит выявить реальные проблемы, требующие внимания.

```
In [56]: user_sessions_data.drop(zero_sessions[zero_sessions['session_num'] != zero_sess
```

```
In [57]: user_sessions_data = user_sessions_data.reset_index(drop=True)
```

```
In [58]: user_sessions_data
```

Out[58]:

	user_id	event_name	source	full_datetime	first_session	date_event	tim
0	0001b1d5-b74a-4cbf-aeb0-7df5947bf349	tips_show	other	2019-10-07 13:39:45	2019-10-07	2019-10-07	
1	0001b1d5-b74a-4cbf-aeb0-7df5947bf349	tips_show	other	2019-10-07 13:40:31	2019-10-07	2019-10-07	
2	0001b1d5-b74a-4cbf-aeb0-7df5947bf349	tips_show	other	2019-10-07 13:49:41	2019-10-07	2019-10-07	
3	0001b1d5-b74a-4cbf-aeb0-7df5947bf349	map	other	2019-10-09 18:33:55	2019-10-07	2019-10-09	
4	0001b1d5-b74a-4cbf-aeb0-7df5947bf349	map	other	2019-10-09 18:35:28	2019-10-07	2019-10-09	
...	
71768	fffb9e79-b927-4dbb-9b48-7fd09b23a62b	tips_show	google	2019-11-03 15:51:23	2019-10-12	2019-11-03	
71769	fffb9e79-b927-4dbb-9b48-7fd09b23a62b	contacts_show	google	2019-11-03 15:51:57	2019-10-12	2019-11-03	
71770	fffb9e79-b927-4dbb-9b48-7fd09b23a62b	tips_show	google	2019-11-03 16:07:40	2019-10-12	2019-11-03	
71771	fffb9e79-b927-4dbb-9b48-7fd09b23a62b	tips_show	google	2019-11-03 16:08:18	2019-10-12	2019-11-03	
71772	fffb9e79-b927-4dbb-9b48-7fd09b23a62b	tips_show	google	2019-11-03 16:08:25	2019-10-12	2019-11-03	

71773 rows × 10 columns



In [59]:

```
print(f'После удаления сохранили {round((user_sessions_data["user_id"].nunique()
```

После удаления сохранили 99.79% уникальных пользователей в данных

In [60]: user_sessions_data

Out[60]:

	user_id	event_name	source	full_datetime	first_session	date_event	tim
0	0001b1d5-b74a-4cbf-aeb0-7df5947bf349	tips_show	other	2019-10-07 13:39:45	2019-10-07	2019-10-07	
1	0001b1d5-b74a-4cbf-aeb0-7df5947bf349	tips_show	other	2019-10-07 13:40:31	2019-10-07	2019-10-07	
2	0001b1d5-b74a-4cbf-aeb0-7df5947bf349	tips_show	other	2019-10-07 13:49:41	2019-10-07	2019-10-07	
3	0001b1d5-b74a-4cbf-aeb0-7df5947bf349	map	other	2019-10-09 18:33:55	2019-10-07	2019-10-09	
4	0001b1d5-b74a-4cbf-aeb0-7df5947bf349	map	other	2019-10-09 18:35:28	2019-10-07	2019-10-09	
...	
71768	fffb9e79-b927-4dbb-9b48-7fd09b23a62b	tips_show	google	2019-11-03 15:51:23	2019-10-12	2019-11-03	
71769	fffb9e79-b927-4dbb-9b48-7fd09b23a62b	contacts_show	google	2019-11-03 15:51:57	2019-10-12	2019-11-03	
71770	fffb9e79-b927-4dbb-9b48-7fd09b23a62b	tips_show	google	2019-11-03 16:07:40	2019-10-12	2019-11-03	
71771	fffb9e79-b927-4dbb-9b48-7fd09b23a62b	tips_show	google	2019-11-03 16:08:18	2019-10-12	2019-11-03	
71772	fffb9e79-b927-4dbb-9b48-7fd09b23a62b	tips_show	google	2019-11-03 16:08:25	2019-10-12	2019-11-03	

71773 rows × 10 columns



Ниже я группирую данные для сессий для более точного определения количества и длительностей сессий пользователей.

```
In [61]: user_sessions_group = user_sessions_data.groupby(['user_id', 'session_num'], as_
```

```
In [62]: user_sessions_group
```

Out[62]:

	user_id	start_session	first_session	session_num	session_duration	source
0	0001b1d5-b74a-4cbf-aeb0-7df5947bf349	2019-10-07 13:39:45	2019-10-07	1	596.0	other
1	0001b1d5-b74a-4cbf-aeb0-7df5947bf349	2019-10-09 18:33:55	2019-10-07	2	507.0	other
2	0001b1d5-b74a-4cbf-aeb0-7df5947bf349	2019-10-21 19:53:17	2019-10-07	3	900.0	other
3	0001b1d5-b74a-4cbf-aeb0-7df5947bf349	2019-10-22 11:20:12	2019-10-07	4	758.0	other
4	00157779-810c-4498-9e05-a1e9e3cedf93	2019-10-19 21:46:52	2019-10-19	1	1521.0	yandex
...
9510	fffb9e79-b927-4dbb-9b48-7fd09b23a62b	2019-11-01 00:24:31	2019-10-12	21	22.0	google
9511	fffb9e79-b927-4dbb-9b48-7fd09b23a62b	2019-11-02 01:16:48	2019-10-12	22	0.0	google
9512	fffb9e79-b927-4dbb-9b48-7fd09b23a62b	2019-11-02 18:01:27	2019-10-12	23	974.0	google
9513	fffb9e79-b927-4dbb-9b48-7fd09b23a62b	2019-11-02 19:25:53	2019-10-12	24	297.0	google
9514	fffb9e79-b927-4dbb-9b48-7fd09b23a62b	2019-11-03 14:32:55	2019-10-12	25	5730.0	google

9515 rows × 6 columns

Анализ распределения длительности сессий (среднее, медиана).

```
In [63]: user_sessions_group['session_duration'].describe()
```

```
Out[63]: count      9515.000000  
mean       1055.433736  
std        1716.146694  
min         0.000000  
25%         45.000000  
50%        412.000000  
75%       1310.000000  
max       26100.000000  
Name: session_duration, dtype: float64
```

```
In [64]: print(f'95 % всех значений меньше данной длительности: {user_sessions_group["ses  
user_sessions_group[user_sessions_group['session_duration'] > 3862]
```

95 % всех значений меньше данной длительности: 4212.199999999997 секунды

Out[64]:

	user_id	start_session	first_session	session_num	session_duration	source
5	00157779-810c-4498-9e05-a1e9e3cedf93	2019-10-20 19:17:24	2019-10-19	2	4572.0	yandex
31	01147bf8-cd48-49c0-a5af-3f6eb45f8262	2019-11-01 21:37:52	2019-11-01	1	4148.0	google
33	013bbb57-ca6f-4af3-b586-4a046d3d3dee	2019-10-11 15:56:15	2019-10-11	2	5042.0	other
54	0203893c-b45b-4807-8cca-2b10ee92321b	2019-10-08 14:08:39	2019-10-08	1	4992.0	other
61	02c90994-8de8-49e8-a384-415fb1602ac5	2019-10-15 19:28:33	2019-10-14	3	7382.0	other
...
9491	fffb9e79-b927-4dbb-9b48-7fd09b23a62b	2019-10-16 11:45:21	2019-10-12	2	7207.0	google
9495	fffb9e79-b927-4dbb-9b48-7fd09b23a62b	2019-10-17 14:17:17	2019-10-12	6	6228.0	google
9497	fffb9e79-b927-4dbb-9b48-7fd09b23a62b	2019-10-18 12:23:20	2019-10-12	8	6174.0	google
9507	fffb9e79-b927-4dbb-9b48-7fd09b23a62b	2019-10-29 13:58:47	2019-10-12	18	8053.0	google
9514	fffb9e79-b927-4dbb-9b48-7fd09b23a62b	2019-11-03 14:32:55	2019-10-12	25	5730.0	google

567 rows × 6 columns

In [65]: `user_sessions_group['session_num'].describe()`

```
Out[65]: count    9515.000000
         mean      3.625013
         std       5.653156
         min       1.000000
         25%       1.000000
         50%       2.000000
         75%       4.000000
         max       72.000000
         Name: session_num, dtype: float64
```

Выводы по сессиям.

- Данные о длительности сессий имеют большое разброс.
- Средняя длительность сессии сильно смещена вправо из-за наличия длинных сессий. Медиана более репрезентативна для типичной продолжительности.
- Минимальная длительность сессии равна 0 секунд. Это подтверждает наличие сессий с нулевой продолжительностью, о которых я говорила ранее и аргументировала решение сохранить некоторые из них.
- Первый квартиль равен 45 секундам. Это означает, что четверть сессий меньше минуты.
- Медиана равна 412 секундам примерно 7 минут. Это значение более устойчиво к выбросам, чем среднее.
- Третий квартиль равен 1310 секундам приблизительно 21 минут. Это означает, что три четверти сессий длятся 21 минут и меньше.
- Максимальная длительность сессии составляет 26100 секунд (более 7 часов). Это говорит о том, что есть очень длинные сессии, которые значительно влияют на среднее значение.

Анализ активности пользователей

Расчет количества действий для каждого пользователя.

```
In [66]: full_mobile_dataset.groupby('user_id', as_index=False)['event_name'].count().sor
```

Out[66]:

	user_id	event_name
3397	cb36854f-570a-41f4-baa8-36680b396370	470
3794	e13f9f32-7ae3-4204-8d60-898db040bcfc	463
2629	9ce63488-758a-481a-bcb5-a02b467e1d84	406
615	21230dd9-2f7f-4b77-a436-43d4d10388e0	398
3162	be1449f6-ca45-4f94-93a7-ea4b079b8f0f	396
...
2447	90aec49c-47f7-41ba-bc28-416855b51d6b	1
3911	e89a6681-7227-457e-a11c-753385fe082f	1
3906	e85710d2-a93c-4d0c-b1a2-efcbafad853d	1
2189	830a7974-f65c-41e2-a99d-60c52fe4f527	1
4277	fe92fa6c-7eef-484f-b31b-fa0db4e4d895	1

4293 rows × 2 columns

```
In [67]: full_mobile_dataset.groupby('user_id', as_index=False)['event_name'].count().sort
```

Out[67]:

	event_name
count	4293.000000
mean	17.017004
std	28.597431
min	1.000000
25%	5.000000
50%	9.000000
75%	17.000000
max	470.000000

```
In [68]: full_mobile_dataset.groupby(['user_id', 'event_name']).agg(count_event=('event_n
```

Out[68]:

	user_id	event_name	count_event
0	0001b1d5-b74a-4cbf-aeb0-7df5947bf349	map	6
1	0001b1d5-b74a-4cbf-aeb0-7df5947bf349	tips_show	29
2	00157779-810c-4498-9e05-a1e9e3cedf93	advert_open	2
3	00157779-810c-4498-9e05-a1e9e3cedf93	contacts_call	5
4	00157779-810c-4498-9e05-a1e9e3cedf93	contacts_show	11
...
9631	ffe68f10-e48e-470e-be9b-eeb93128ff1a	photos_show	7
9632	ffe68f10-e48e-470e-be9b-eeb93128ff1a	search	5
9633	fffb9e79-b927-4dbb-9b48-7fd09b23a62b	contacts_show	68
9634	fffb9e79-b927-4dbb-9b48-7fd09b23a62b	map	2
9635	fffb9e79-b927-4dbb-9b48-7fd09b23a62b	tips_show	232

9636 rows × 3 columns

In [69]:

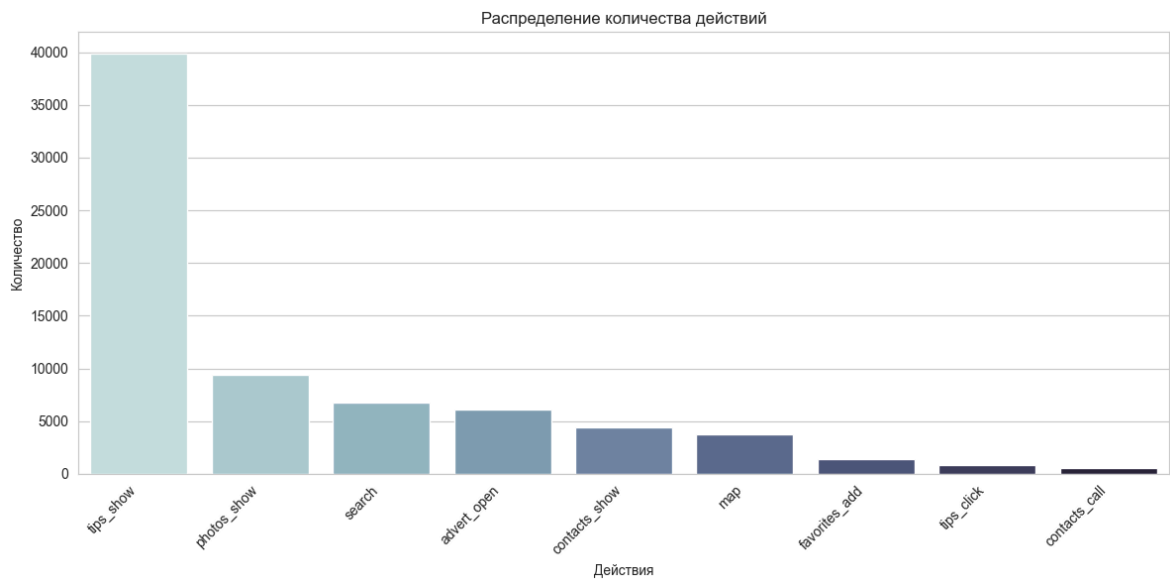
```
all_event = full_mobile_dataset.groupby('event_name', as_index=False).agg(count_
                                     'count_event')
all_event
```

Out[69]:

	event_name	count_event
8	tips_show	39907
5	photos_show	9352
6	search	6765
0	advert_open	6146
2	contacts_show	4376
4	map	3749
3	favorites_add	1414
7	tips_click	811
1	contacts_call	534

In [70]:

```
plt.figure(figsize=(12, 6)) # Задаем размер графика
sns.barplot(x='event_name', y='count_event', data=all_event, palette="ch:start=
plt.title('Распределение количества действий')
plt.xlabel('Действия')
plt.ylabel('Количество')
plt.xticks(rotation=45, ha='right')
plt.tight_layout()
plt.show()
```



- Действие "tips_show" является самым распространенным, значительно превосходя все остальные по количеству. Это означает, что рекомендации очень часто показываются пользователям. Количество просмотров "tips_show" приближается к 40,000.

Действие "photos_show" занимает второе место по распространенности, но значительно уступает "tips_show") почти в 4 раза.

Действия "search", "advert_open" и "contacts_show" также относительно часто встречаются (количество примерно от 4,000 до 7,000).

Действия "map" и "favorites_add" выполняются реже (в районе 2,000 - 4,000), а "tips_click" и "contacts_call" встречаются крайне редко.

Анализ распределения количества действий по дням недели.

```
In [71]: full_mobile_dataset['day_of_week'] = full_mobile_dataset['full_datetime'].dt.weekday
event_by_day_of_week = full_mobile_dataset.groupby(['day_of_week', 'event_name'])
```

```
In [72]: event_by_day_of_week
```

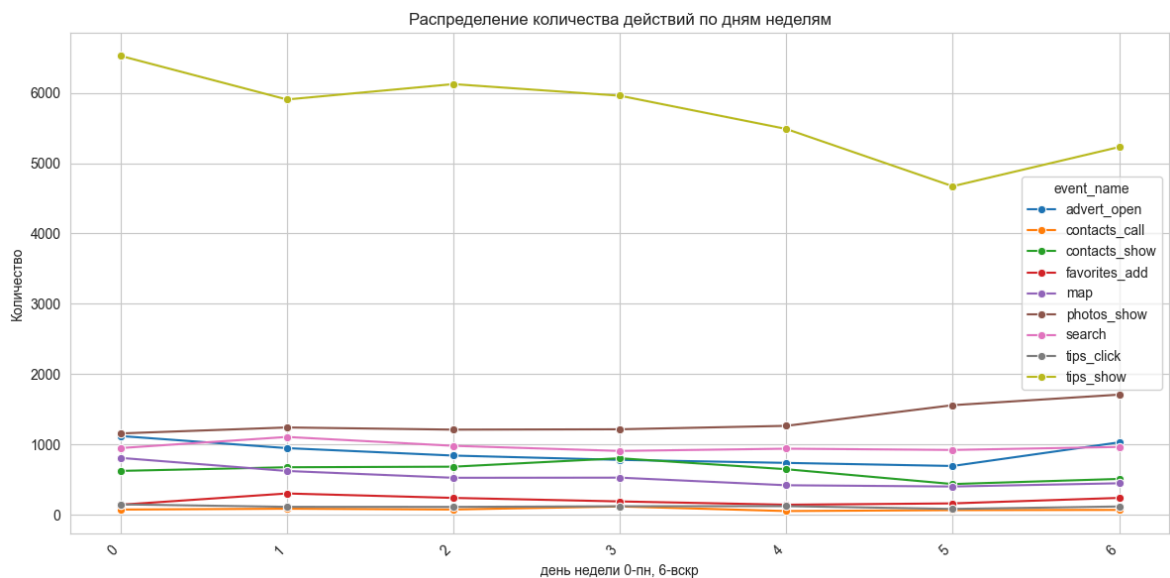
Out[72]:

	day_of_week	event_name	user_id
0	0	advert_open	1119
1	0	contacts_call	73
2	0	contacts_show	623
3	0	favorites_add	142
4	0	map	808
...
58	6	map	447
59	6	photos_show	1708
60	6	search	964
61	6	tips_click	117
62	6	tips_show	5230

63 rows × 3 columns

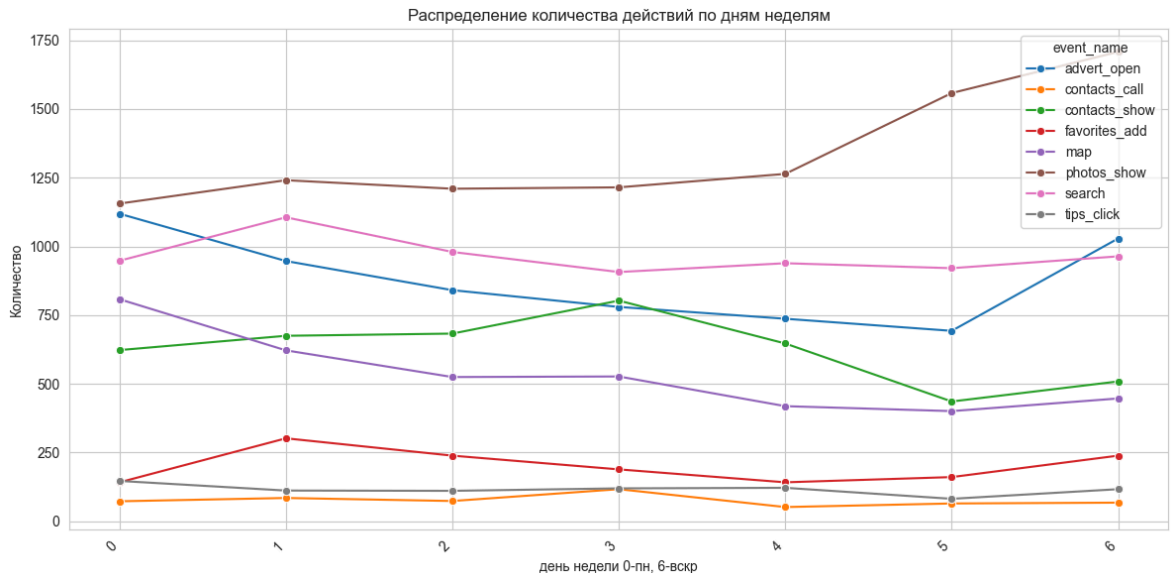
In [73]:

```
plt.figure(figsize=(12, 6)) # Задаем размер графика
sns.lineplot(x='day_of_week', y='user_id', data=event_by_day_of_week, hue='event_name')
plt.title('Распределение количества действий по дням неделям')
plt.xlabel('день недели 0-пн, 6-вскр')
plt.ylabel('Количество')
plt.xticks(rotation=45, ha='right')
plt.tight_layout()
plt.show()
```



Можем увидеть как к концу недели частота показов рекомендаций снижается, пользователи к воскресенью видят меньше рекомендаций. Из-за большого числа действий "tips_show" сложно рассмотреть частоту других событий. Считаю, что необходимо рассмотреть иные действия без "tips_show", чтобы было лучше видно их частоту.

```
In [74]: plt.figure(figsize=(12, 6)) # Задаем размер графика
sns.lineplot(x='day_of_week', y='user_id', data=event_by_day_of_week[event_by_da
plt.title('Распределение количества действий по дням неделям')
plt.xlabel('день недели 0-пн, 6-вскр')
plt.ylabel('Количество')
plt.xticks(rotation=45, ha='right')
plt.tight_layout()
plt.show()
```



В целом по остальным действиям можно заметить снижение к концу недели с небольшим подъемом в воскресенье, за исключением просмотра фотографий. и просмотр карточки товара в выходные они возрастают сильнее остальных. Просматривают контакты пользователи чаще к середине недели в четверг, это и логично, что чаще звонят по объявлению к концу рабочей недели, чтобы обсудить приглянувшуюся вещь и если всё устроит, то договориться о встрече на выходных.

Выводы по активности пользователей.

- Просмотр рекомендаций - самое частое действие. Пользователи видят много предложений и рекомендаций, но редко кликают по нему примерно в 2 % случаях. Это говорит о нерелевантной подборке рекомендаций.
- Наряду с рекомендациями, "photos_show" (просмотр фото) занимает второе место по популярности. Пользователи в основном смотрят, а не активно взаимодействуют с приложением.
- Снижение активности к концу недели: Общая активность (кроме просмотра фото) падает к концу недели, что может быть связано с изменением поведения пользователей в выходные дни. В эти дни пользователи отдыхают и реже взаимодействуют с платформой, возможно в данные дни следует ввести рекламные рассылки и напоминания о избранных товарах.
- В выходные пользователи больше смотрят фотографии и карточки товаров ("advert_open"), что говорит о большем времени, уделяемом развлечениям и поиску

предложений.

- Пользователи чаще смотрят контакты ("contacts_show") в четверг, возможно, для организации встреч или звонков в конце рабочей недели.
- Низкая вовлеченность в действия "tips_click" и "contacts_call" используются нечасто.

Рекомендации:

- Улучшить вовлечение в рекомендации: Рассмотреть способы стимулирования пользователей к взаимодействию с рекомендациями, выдавать более релевантные рекомендации.
- Оптимизировать контент для выходных: Учитывая, что в выходные пользователи больше смотрят фотографии и товары, можно оптимизировать контент и предложения, ориентируясь на эти интересы и формировать соответствующие рекомендации.
- Анализировать падение активности в конце недели: Разобраться, почему снижается общая активность к концу недели. Возможно, стоит предлагать пользователям что-то новое в это время, чтобы стимулировать их активность.
- Сбалансировать стратегию подачи рекомендации. Слишком частое появление рекомендации может отвлекать пользователей от основного функционала или даже раздражать их, что, в свою очередь, может влиять на их общую вовлеченность. Рассмотрите возможность предоставления рекомендаций по запросу или на основе анализа поведения пользователей, чтобы рекомендации были максимально релевантными и полезными.

Анализ конверсии в целевое действие "contacts_show"

Расчет конверсии в просмотр контактов "contacts_show" для всех пользователей и по источникам.

```
In [75]: # Рассмотрим целевое действие по всем сессиям
print(f'Конверсия пользователей в целевое действие равняется: ')
round(len(full_mobile_dataset[full_mobile_dataset['event_name'] == 'contacts_sho
```

Конверсия пользователей в целевое действие равняется:

```
Out[75]: 22.85
```

Конверсия в просмотр контактов (contacts_show) составляет 22.85%. Это означает, что из общего числа пользователей, всего примерно 22.85% дошли до действия посмотреть контакт. Это невысокий показатель конверсии. Данный процент может говорить о нескольких проблемах платформы: - неудобный интерфейс; - нерелевантные выдачи поиска; - плохо подобранные рекомендации. Контакты

просматривают в примерно 4 случаях из 10. В целом конверсию в целевое действие можно считать высокой.

Анализ количества сессий, необходимых для совершения целевого действия.

```
In [76]: # user_sessions_data[user_sessions_data['event_name'] == 'contacts_show']['sessi
```

```
In [77]: # Если посмотрим необходимое количество для достижения целевого действия по пер
user_sessions_data[user_sessions_data['event_name'] == 'contacts_show'].groupby(
```

Out[77]:

	first_show
count	973.000000
mean	1.620761
std	1.380739
min	1.000000
25%	1.000000
50%	1.000000
75%	2.000000
max	18.000000

Множество пользователей достигают первый раз целевого действия в рамках первой сессии.

Действие "contacts_show" достигли 973 пользователя. В среднем для достижения целевого действия необходимо 1.6 сессии.

Минимальное количество выполнений "contacts_show" в сессии равно 1.

До половины в первых сессиях пользователи выполняют целевое действие.

В 75% сессий "contacts_show" выполняется выполняется второй.

Максимальное количество выполнений сессий до достижения "contacts_show" равно 18.

```
In [78]: # Рассмотрим целевое действие по сгруппированным сессиям для всех сессий по всем
user_sessions_data[user_sessions_data['event_name'] == 'contacts_show']['session
```

```
Out[78]: count    4320.000000
         mean      4.380324
         std       6.086894
         min       1.000000
         25%       1.000000
         50%       2.000000
         75%       4.000000
         max       68.000000
         Name: session_num, dtype: float64
```

Действие "contacts_show" достигли в данных всего в 4320 раз. В среднем для достижения целевого действия необходимо 4 сессии.

Минимальное количество выполнений "contacts_show" в сессии равно 1.

В 25% сессий "contacts_show" выполняется сразу.

В 50% сессий "contacts_show" выполняется минимум второй .

В 75% сессий "contacts_show" выполняется выполняется четвёртой.

Максимальное количество выполнений "contacts_show" в сессии равно 68.

```
In [79]: # Рассмотрим целевое действие по всем сессиям
         full_mobile_dataset[full_mobile_dataset['event_name'] == 'contacts_show']['sessi
```

```
Out[79]: count    4376.000000
         mean      4.356718
         std       6.056952
         min       1.000000
         25%       1.000000
         50%       2.000000
         75%       4.000000
         max       68.000000
         Name: session_num, dtype: float64
```

Действие "contacts_show" встречается 4376 раз.

В среднем на достигают целевого действия через 4.4 сессии.

Минимальное количество просмотров за сессию 1.

25% пользователей за 1 сессию просматривают контакт.

50% пользователей за 2 сессию просматривают контакт.

75% пользователей за 4 сессию просматривают контакт.

Максимальное количество просмотров 68.

Выводы по конверсии пользователей.

Просмотр контактов ("contacts_show") имеет низкую конверсию - 23 % пользователей доходят до этого действия. При этом пользователи, которые доходят до просмотра контактов, в среднем просматривают 4-5 контактов за сессию.

Однако, большинство (75%) просматривают не более 5 контактов, а половина - всего 1-2.

Несмотря на то, что пользователи, добравшиеся до просмотра контактов, демонстрируют определенную активность, и демонстрируют высокую конверсию и достаточно низкое удержание.

Рекомендации:

Основной приоритет - повысить удержание. Стоит сосредоточиться на:

- Выявить, на каком этапе пользователи "отваливаются" и почему они не доходят до просмотра контактов.
- Убедиться, что интерфейс удобен и интуитивно понятен, чтобы пользователям было легко найти и просмотреть объявления и в целом интуитивно понятный интерфейс приложения.
- Оптимизировать алгоритмы поиска и рекомендации, чтобы предлагать пользователям только релевантные контакты.

Сегментация пользователей

Выбор критериев сегментации.

Для сегментации считаю необходимым выбрать такие параметры как активность пользователя и источника установки приложения. То есть выбрать критерий определяющий активных и неактивных пользователей, а так же разделить по источникам.

АРГУМЕНТАЦИЯ:

1. Активность пользователя:

- Активные и неактивные пользователи ведут себя по-разному в приложении. Сравнивая эти сегменты, можно выявить основные факторы, влияющие на активность. Например, какие функции используют активные пользователи, а неактивные игнорируют.
- Активные пользователи удовлетворены приложением и находят его полезным. Неактивные пользователи, вероятно, столкнулись с проблемами, не нашли ценности т.п. Понимание их разных потребностей позволяет адаптировать стратегии развития платформы.
- Подход к активным и неактивным пользователям должен быть разным. Активных нужно удерживать, мотивировать к дальнейшему использованию и поощрять лояльность. Неактивных нужно "возвращать", предлагая им персонализированные предложения, напоминания о ценности приложения и устраняя причины их ухода.

- Если в приложение вносят изменения, внедряются новые функции, изменение, таком случае, сегментация по активности позволит оценить, как изменения повлияют на поведение разных групп пользователей.

- Зная, какие факторы влияют на активность, можно более эффективно распределять ресурсы на улучшение тех аспектов приложения, которые действительно важны для удержания пользователей.

Определение критерия "активности":

- Частота использования: Количество сессий за период представленных данных. По нашим данным половина проводит 2 сессии и примем данные показатель за активность, если больше (не включительно), то данный пользователь активный если менее 2 (включительно) ,то пользователь считается неактивным.

2. Источник установки приложения:

- Пользователи, пришедшие из разных источников могут иметь разную мотивацию, ожидания и уровень вовлеченности.

- Сегментация по источнику установки позволяет оценить, какие каналы привлекают наиболее ценных пользователей (с точки зрения активности, конверсии и удержания).

- Сегментация по источнику установки помогает определить, какие каналы наиболее эффективно приводят к целевому действию в приложении.

- Можно адаптировать онбординг для пользователей, пришедших из разных источников, чтобы учесть их специфические потребности и ожидания.

Сегментация по активности пользователя и источнику установки приложения - это мощный инструмент для глубокого понимания поведения пользователей, оптимизации продуктовых стратегий.

Формирование сегментов пользователей.

```
In [80]: user_sessions_group['session_num'].describe()
```

```
Out[80]: count    9515.000000
mean         3.625013
std          5.653156
min          1.000000
25%          1.000000
50%          2.000000
75%          4.000000
max          72.000000
Name: session_num, dtype: float64
```

По распределению количества сессий видим, что половина совершает лишь 2 сессии поэтому для формирования активных пользователей мы выберем всех кто

совершил 3 или более (то есть больше 2 не включительно). Соответственно для неактивных 2 или меньше или строго меньше 3.

```
In [81]: # 1. Определяем количество сессий для каждого пользователя
session_counts = user_sessions_data.groupby('user_id', as_index=False).agg(total
                                                                              first
                                                                              source

# 2. Определяем, совершал ли пользователь 'contacts_show' и устанавливаем флаг
contact_show_users = user_sessions_data[user_sessions_data['event_name'] == 'con
session_counts['contact_show'] = session_counts['user_id'].isin(contact_show_use

# 3. Сегментируем пользователей
active_yandex = session_counts[(session_counts['total_sessions'] >= 3)
                               & (session_counts['source'] == 'yandex')][['user

inactive_yandex = session_counts[(session_counts['total_sessions'] < 3)
                                  & (session_counts['source'] == 'yandex')][['us

active_google = session_counts[(session_counts['total_sessions'] >= 3)
                                & (session_counts['source'] == 'google')][['u

inactive_google = session_counts[(session_counts['total_sessions'] < 3)
                                  & (session_counts['source'] == 'google')][[

active_other = session_counts[(session_counts['total_sessions'] >= 3)
                              & (session_counts['source'] == 'other')][['user_

inactive_other = session_counts[(session_counts['total_sessions'] < 3)
                                & (session_counts['source'] == 'other')][['use
```

```
In [82]: session_counts
```

Out[82]:

	user_id	total_sessions	first_session	source	contact_show
0	0001b1d5-b74a-4cbf-aeb0-7df5947bf349	4	2019-10-07	other	False
1	00157779-810c-4498-9e05-a1e9e3cedf93	5	2019-10-19	yandex	True
2	00463033-5717-4bf1-91b4-09183923b9df	1	2019-11-01	yandex	False
3	004690c3-5a84-4bb7-a8af-e0c8f8fca64e	6	2019-10-18	google	False
4	00551e79-152e-4441-9cf7-565d7eb04090	1	2019-10-25	yandex	True
...
4279	ffab8d8a-30bb-424a-a3ab-0b63ebbf7b07	2	2019-10-13	yandex	False
4280	ffc01466-fdb1-4460-ae94-e800f52eb136	1	2019-10-07	yandex	True
4281	ffcf50d9-293c-4254-8243-4890b030b238	1	2019-10-23	google	False
4282	ffe68f10-e48e-470e-be9b-eeb93128ff1a	3	2019-10-21	yandex	True
4283	fffb9e79-b927-4dbb-9b48-7fd09b23a62b	25	2019-10-12	google	True

4284 rows × 5 columns

```
In [83]: active_yandex['activity_and_source'] = 'active_yandex'
inactive_yandex['activity_and_source'] = 'inactive_yandex'
active_google['activity_and_source'] = 'active_google'
inactive_google['activity_and_source'] = 'inactive_google'
active_other['activity_and_source'] = 'active_other'
inactive_other['activity_and_source'] = 'inactive_other'
```

```
In [84]: connection_segments = pd.concat([active_yandex, inactive_yandex, active_google,
```

```
In [85]: # Проверим совпадает ли количество уникальных пользователей полученных таблиц с
# с уникальным количеством пользователей исходного датафрейма.
if (len(inactive_yandex) + len(active_yandex) + len(inactive_google) + len(active_google)) == len(connection_segments):
    print('Уникальное количество пользователей СОВПАДАЕТ')
else:
    print('Уникальное количество пользователей НЕ СОВПАДАЕТ')
```

Уникальное количество пользователей СОВПАДАЕТ

Рассмотрим получившиеся таблицы:

```
In [86]: connection_segments
```

Out[86]:

	user_id	total_sessions	contact_show	first_session	source	activity_and_sour
0	00157779-810c-4498-9e05-a1e9e3cedf93	5	True	2019-10-19	yandex	active_yanc
1	042ebe74-a35b-41d5-abf8-ef6786918951	4	True	2019-10-18	yandex	active_yanc
2	04e06319-c049-465c-9170-362a191a1287	5	True	2019-10-17	yandex	active_yanc
3	06b73062-0e64-4e7a-91e7-2c2773234919	3	False	2019-10-21	yandex	active_yanc
4	06bdb96e-2712-47b3-a0af-d19f297abd6c	12	True	2019-10-12	yandex	active_yanc
...
4279	fe872615-4057-43cb-adaa-b059334aaa70	1	False	2019-10-18	other	inactive_otl
4280	fe9e6bd4-aec3-4f62-9d54-1b3ce6ad4855	1	False	2019-10-22	other	inactive_otl
4281	febe9e0b-b804-4084-a30d-1c75550e7fa3	1	False	2019-10-10	other	inactive_otl
4282	ff1554b5-919e-40b1-90bb-ee1f7f6d5846	2	True	2019-10-21	other	inactive_otl
4283	ff24f3a3-d3fe-4d36-838a-3a29548e6c91	1	False	2019-10-10	other	inactive_otl

4284 rows × 6 columns



In [87]: `connection_segments['source'].value_counts()`

```
Out[87]: source
yandex    1930
other     1227
google    1127
Name: count, dtype: int64
```

Так же проверить на соответствие размерности таблиц друг другу (меньшая не меньше 10 % самой большой)

```
In [88]: len(inactive_yandex), len(active_yandex), len(inactive_google) , len(active_google)
```

```
Out[88]: (1499, 431, 872, 255, 307, 920)
```

```
In [89]: smaller = min(len(inactive_yandex), len(active_yandex), len(inactive_google) , len(active_google))
bigger = max(len(inactive_yandex), len(active_yandex), len(inactive_google) , len(active_google))
```

```
In [90]: print(f'Наименьшая таблица размером {smaller} записей соответствует от {round((smaller/bigger)*100)} % самой большой')
```

Наименьшая таблица размером 255 записей соответствует от 17 % наибольшей таблицы длиной 1499 записей.

Описание сегментов.

Сегменты были сформированы по 2 показателям: активности и источнику установки приложения.


Рассмотрим слабые и сильные стороны каждого канала, выявим возможные причины низкой активности пользователей в зависимости от источника установки приложения.

Расчёт Retention rate для каждой группы отдельно.

```
In [91]: user_sessions_data = user_sessions_data.merge(connection_segments[['user_id', 'app_id']],
on='user_id', how='left')
user_sessions_data.head()
```


Out[91]:

	user_id	event_name	source	full_datetime	first_sesion	date_event	time_event
0	0001b1d5-b74a-4cbf-aeb0-7df5947bf349	tips_show	other	2019-10-07 13:39:45	2019-10-07	2019-10-07	13:39:45
1	0001b1d5-b74a-4cbf-aeb0-7df5947bf349	tips_show	other	2019-10-07 13:40:31	2019-10-07	2019-10-07	13:40:31
2	0001b1d5-b74a-4cbf-aeb0-7df5947bf349	tips_show	other	2019-10-07 13:49:41	2019-10-07	2019-10-07	13:49:41
3	0001b1d5-b74a-4cbf-aeb0-7df5947bf349	map	other	2019-10-09 18:33:55	2019-10-07	2019-10-09	18:33:55
4	0001b1d5-b74a-4cbf-aeb0-7df5947bf349	map	other	2019-10-09 18:35:28	2019-10-07	2019-10-09	18:35:28



```
In [92]: user_sessions_data['activity_and_source'].isna().sum()
```

Out[92]: np.int64(0)

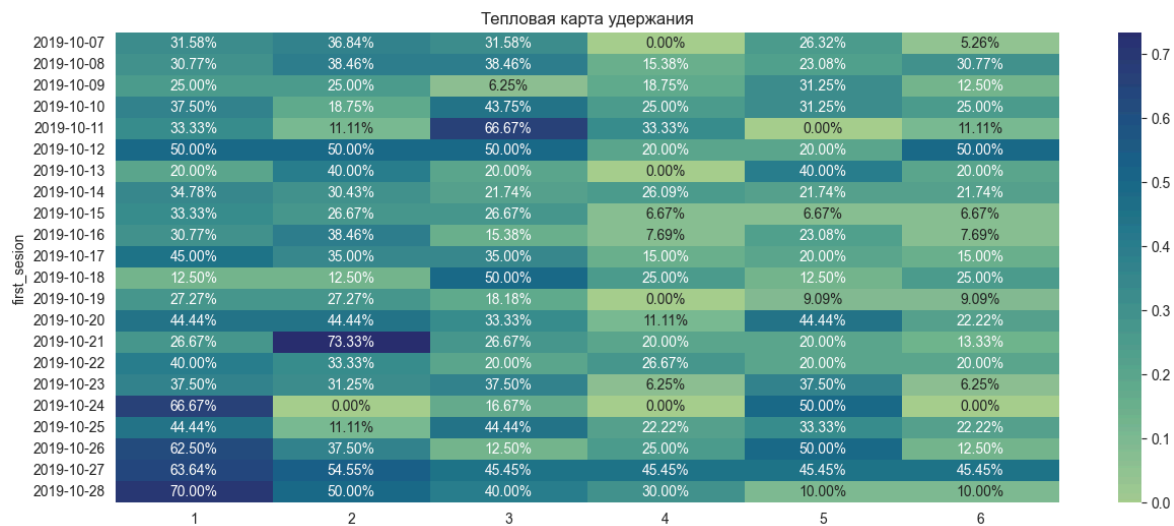
```
In [93]: user_sessions_data['contact_show'].isna().sum()
```

Out[93]: np.int64(0)

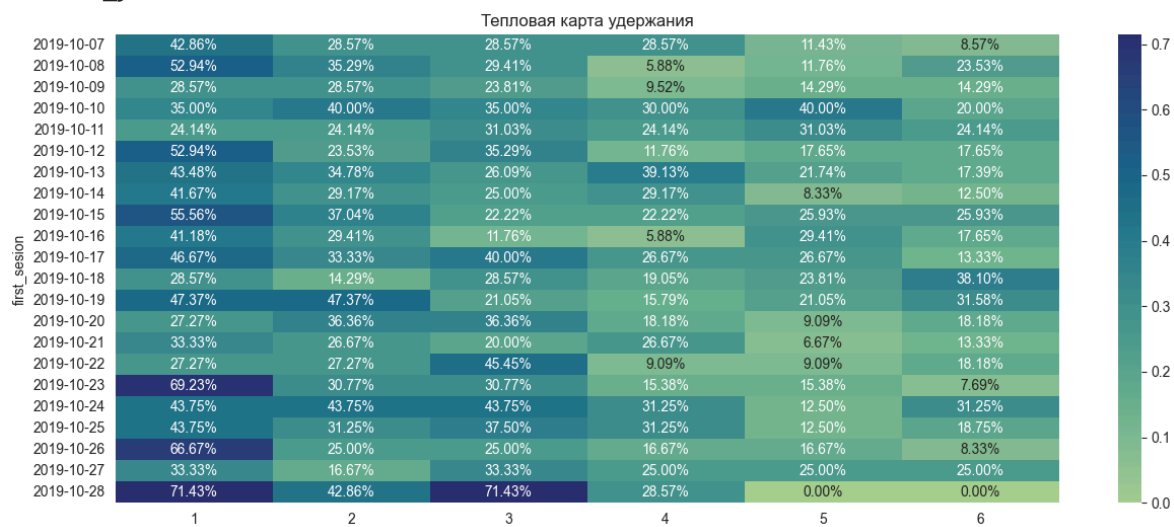
```
In [94]: # Рассмотрим динамику удержания пользователей, проанализировав данные за 1-дневн
for act_sou in user_sessions_data['activity_and_source'].unique():
    print(act_sou)
    result_grouped = get_retention(profiles=user_sessions_data[user_sessions_data
                                                                    horizon_days=7, dimensions=['first_session'],

    heatmap_of_retention(result_grouped)
```

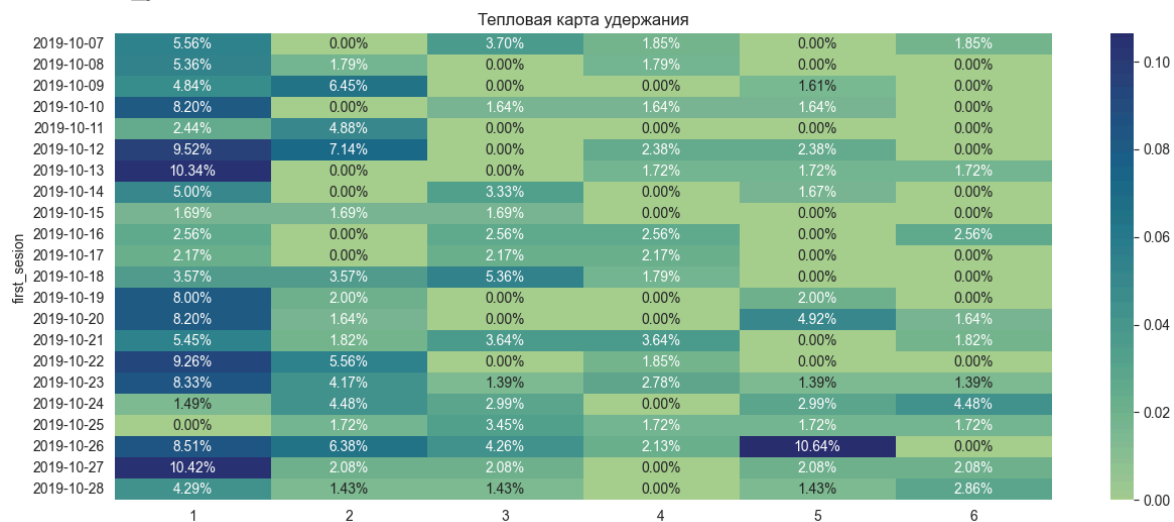
active_other



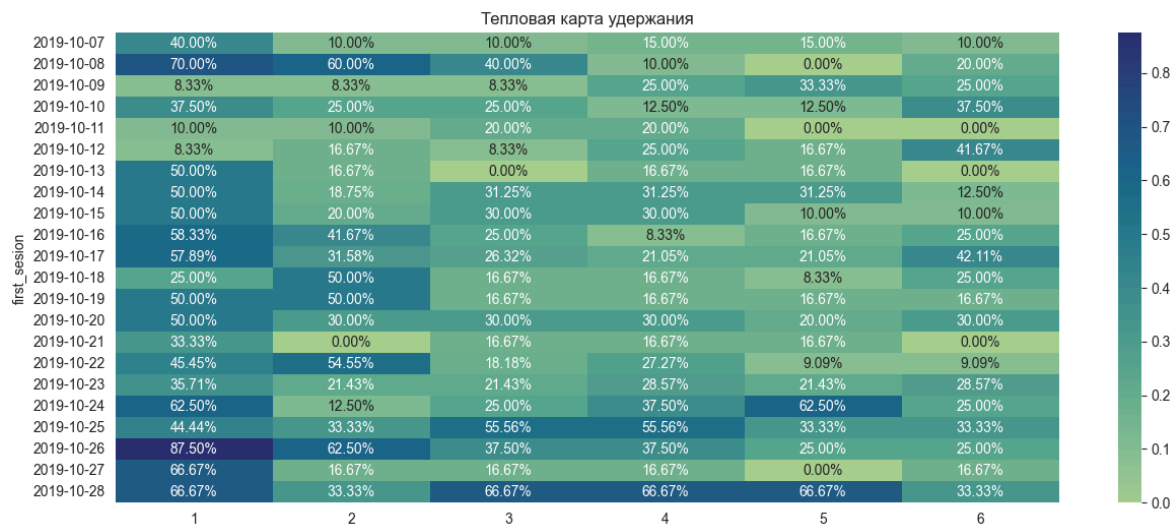
active_yandex



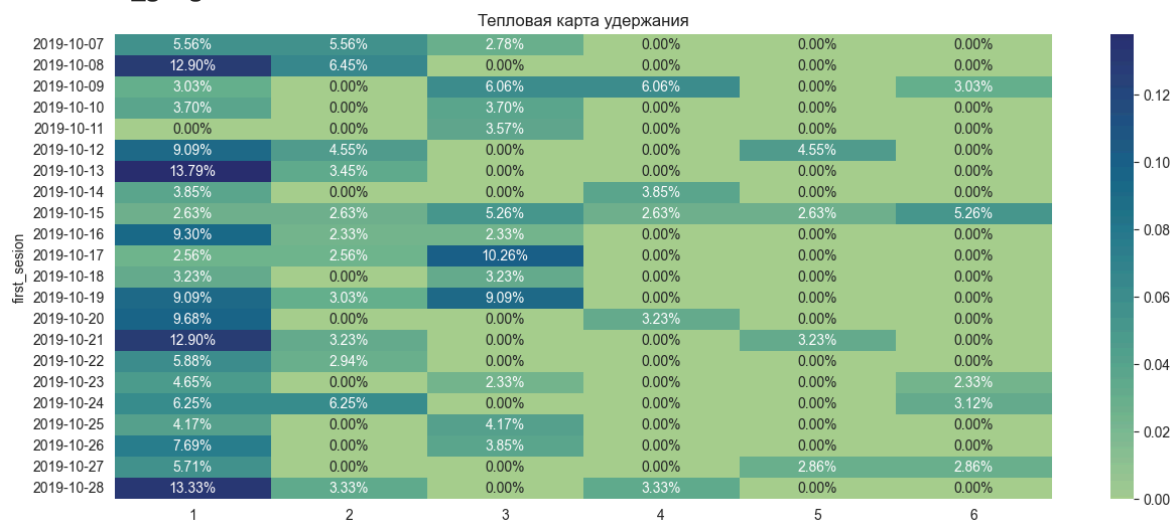
inactive_yandex



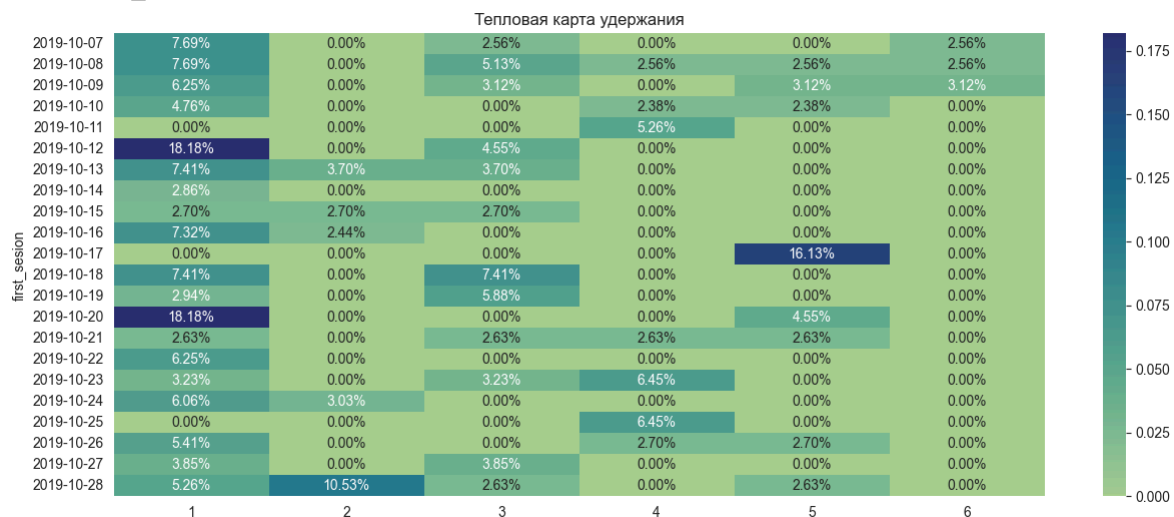
active_google



inactive_google



inactive_other



Тепловые карты удержания демонстрируют лучший показатель удержания для активных и более плохой пости нулевой для неактивных.

Сравнивая тепловые карты удержания только для активных пользователей можно выделить следующие моменты:

Удержание пользователей (retention) снижается с течением времени. Это ожидаемо, так как не все пользователи возвращаются в приложение каждый день.

Удержание пользователей на протяжении 7 дней периодов выше, чем у Yandex и Google. Отсутствуют критические просадки удержания, более темный цвет карты свидетельствует о большем % удержания

Удержание показывает более высокие результаты, чем Google в промежутке с 2019-10-07 по 2019-10-16 (первые 10 дней) в целом. Далее можно наблюдать сопоставимые результаты

Наиболее неоднородная тепловая карта. Есть дни с высоким показателем удержания, но и критические просадки тоже имеются. Видим высокое удержание в последней кагорте.

Ключевые отличия и выводы:

- `active_other`: Самое стабильное удержание пользователей среди активных. Вероятно, это самая лояльная аудитория или самая целевая (её не так просто найти и она более заинтересована). Стоит глубже проанализировать, что это за пользователи, и перенести удачные решения на Yandex и Google.
- `active_yandex` и `active_google`: В целом демонстрируют похожие паттерны, но с некоторым преимуществом в начальный период удержания для Yandex. Стоит посмотреть, какие активности проводились в период с 2019-10-07 по 2019-10-16 для Yandex, и попытаться повторить или усилить их эффект.

Анализ тепловых карт удержания неактивных пользователей выявляет следующие ключевые моменты:

Удержание очень низкое для всех каналов привлечения неактивных пользователей. Преобладают светлые тона на графиках, что указывает на крайне малый процент возвращающихся пользователей.

Удержание `inactive_yandex` (неактивных из яндекса) в течение 7 дней крайне низкое. Удержание пользователей в течении 1-2 периодов несколько выше чем у `inactive_google` и `inactive_other`, однако эти значения близки к нулю и не демонстрируют устойчивого интереса к продукту.

- `inactive_google` схожая с `inactive_yandex` тенденция. Удержание очень низкое.
- `inactive_other` так же демонстрирует низкий уровень удержания, однако на 19.10.2019 можно увидеть значительное увеличение удержания к 5-6 периоду.

Общая проблема для всех неактивных пользователей. Мало кто из них возвращается в продукт, что говорит о необходимости пересмотра стратегий привлечения и удержания.

Низкая эффективность каналов для реактивации: Все каналы (Yandex, Google, Other) показывают низкую эффективность в возвращении неактивных пользователей.

Расчёт конверсии для каждой группы отдельно.

```
In [95]: for conv in [active_yandex, inactive_yandex, active_google, inactive_google, active_other, inactive_other]:
          print(f'Рассмотрим конверсию для {conv["activity_and_source"].unique()}:')
          print(round(sum(conv['contact_show']) / len(conv) * 100, 2), '%\n')
```

Рассмотрим конверсию для ['active_yandex']:
40.6 %

Рассмотрим конверсию для ['inactive_yandex']:
20.08 %

Рассмотрим конверсию для ['active_google']:
41.96 %

Рассмотрим конверсию для ['inactive_google']:
18.81 %

Рассмотрим конверсию для ['active_other']:
38.44 %

Рассмотрим конверсию для ['inactive_other']:
11.74 %

- Активные пользователи (active) демонстрируют более высокую конверсию, чем неактивные (inactive) пользователи для всех источников трафика.
- Наиболее эффективный источник Google (конверсия ~41.96% для активных пользователей).
- Наименее эффективный источник Other (конверсия ~11.74% для неактивных пользователей).
- Яндекс: Конверсия активных пользователей 40.6%
- Вывод: Стратегии привлечения и удержания пользователей нужно оптимизировать, учитывая источник трафика и активность пользователей. Необходимо улучшить показатели конверсии для неактивных пользователей и для трафика из "Other" источников.

Выводы и рекомендации по сегментации.

Рекомендации

Необходимо разработать новые, более эффективные подходы к возвращению неактивных пользователей. Проанализировать причины ухода и предложить релевантные решения (например, персонализированные предложения, опросы о причинах оттока).

- Сегментация неактивных пользователей: Не все неактивные пользователи одинаковы. Сегментировать их по времени неактивности, по поведению до ухода, по демографическим данным и другим параметрам, чтобы предложить более таргетированные и эффективные сообщения.

- Улучшение сбора обратной связи: Настроить сбор обратной связи от уходящих пользователей, чтобы лучше понимать причины оттока и предотвращать его в будущем.
- Фокус на удержании активных: Поскольку удержание неактивных пользователей крайне сложно, следует уделять больше внимания удержанию активных пользователей, чтобы предотвратить их уход.

1. Уделить приоритетное внимание удержанию активных пользователей.
Предотвращение оттока – более эффективная стратегия, чем реактивация неактивных.
2. Разработать новые стратегии реактивации неактивных пользователей. Текущие подходы неэффективны.
3. Проанализировать влияние конкретных событий и акций. Выявить причины колебаний удержания в разные периоды времени.
4. Продолжить сегментацию пользователей. Чем лучше вы понимаете своих пользователей, тем более таргетированные и эффективные стратегии удержания вы сможете разработать.
5. Постоянно измерять и оптимизировать. Удержание – это непрерывный процесс. Необходимо постоянно отслеживать показатели, тестировать новые подходы и адаптировать стратегии на основе полученных данных.

Проверка гипотез

Гипотеза 1:

H0: Конверсия в просмотр контактов не зависит от источника привлечения (Google vs. Yandex).

H1: Пользователи из Yandex имеют более высокую конверсию в просмотр контактов, чем пользователи из Google.

```
In [96]: all_yandex = len(active_yandex['user_id']) + len(inactive_yandex['user_id'].unique())
target_yandex = sum(active_yandex['contact_show']) + sum(inactive_yandex['contact_show'])
all_google = len(active_google) + len(inactive_google) # Общее количество пользо
target_google = sum(active_google['contact_show']) + sum(inactive_google['contact_show'])
```

```
In [97]: alpha = .05 # критический уровень статистической значимости

# successes = np.array([78, 120])
# trials = np.array([830, 909])

# пропорция успехов в первой группе:
p1 = target_yandex/all_yandex

# пропорция успехов во второй группе:
p2 = target_google/all_google
```

```

print(target_yandex,target_google,all_yandex, all_google )

# пропорция успехов в комбинированном датасете:
p_combined = (successes[0] + target_google) / (all_yandex + all_google)

# разница пропорций в датасетах
difference = p1 - p2

# считаем статистику в ст.отклонениях стандартного нормального распределения
z_value = difference / math.sqrt(p_combined * (1 - p_combined) * (1/trials[0] + 1

# 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: ', 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

```

476 271 1930 1127

```

-----
NameError                                Traceback (most recent call last)
Cell In[97], line 14
     11 print(target_yandex,target_google,all_yandex, all_google )
     13 # пропорция успехов в комбинированном датасете:
--> 14 p_combined = (successes[0] + target_google) / (all_yandex + all_google)
     16 # разница пропорций в датасетах
     17 difference = p1 - p2

NameError: name 'successes' is not defined

```

Гипотеза 2:

H0: Средняя длительность сессий одинакова для обоих источников трафика.

H1: Средняя длительность сессий отличается от источника трафика, для Yandex выше чем для Google

```
In [ ]: user_sessions_group.head()
```

```
In [ ]: google_sessions = user_sessions_group[user_sessions_group['source'] == 'google']
yandex_sessions = user_sessions_group[user_sessions_group['source'] == 'yandex']
```

```

# Проведём односторонний t-тест yandex > google
t_statistic_sessions, p_value_sessions = st.ttest_ind(yandex_sessions, google_sessions)

print("\nT-test (односторонний):")
print("T-статистика:", t_statistic_sessions)
print("P-значение:", p_value_sessions)

if p_value_sessions < alpha:
    print("Есть статистически значимые доказательства того, что среднее коли
else:
    print("\nНет статистически значимых доказательств того, что среднее коли

```

Нет статистических оснований считать группы по источнику установки приложения отличными по показателям конверсии в целевое действие, а так же средней длительности сессий. В последующий анализ необходимо добавить дополнительную сегментацию по активности и достижению целевого действия, так как это поможет более глубоко проанализировать пользователей и выявить причины низкого удержания.

Общие выводы и рекомендации по проведенному анализу

Удержание пользователей в приложении характеризуется очень низким уровнем и высокой нестабильностью. Данные о длительности сессий имеют большой разброс.

Средняя длительность сессии сильно смещена вправо из-за наличия длинных сессий. Медиана более репрезентативна для типичной продолжительности и составляет примерно 7 минут.

Просмотр карты становится самой частой точкой завершения взаимодействия, где пользователи чаще всего прекращают использование приложения после просмотра. Множество нулевых сессий примерно у 21 % пользователей, говорит о техническом несовершенстве приложения, требующего доработки.

Действие "tips_show" является самым распространенным, значительно превосходя все остальные по количеству. Это означает, что рекомендации очень часто показываются пользователям. Количество просмотров "tips_show" приближается к 40,000. Действие "photos_show" занимает второе место по распространенности, но значительно уступает "tips_show") почти в 4 раза. К концу недели частота показов рекомендаций снижается. Как и остальные действия снижаются в течении недели и выходным повышается активность.

Конверсия в просмотр контактов (contacts_show) составляет 22.85%. Это означает, что из общего числа пользователей, всего примерно 23% дошли до действия посмотреть контакт. Это высокий показатель конверсии. Но имея в виду очень низкий процент удержания пользователей, то можно говорить о том что пользователи достигая один раз целевого действия редко вновь возвращаются в приложение. Необходимо сфокусироваться на повышении удержания пользователей и мотивировать их возвращение в приложение.

Рекомендации:

1. Улучшение рекомендательной системы:

- Персонализация: Улучшить алгоритмы для более релевантных рекомендаций, учитывая историю просмотров, предпочтения пользователей, истории поиска. Что в свою очередь повысит удержание и пользователи будут возвращаться в

приложение. Так как сейчас имеем весьма лояльную аудиторию, которые часто выполняют (23 %) целевое действие, но не возвращаются.

- Тестирование: A/B тестирование различных форматов и типов рекомендаций. И дополнительно тестировать группы по сегментированные активности и достижению целевого действия. Это поможет выявить особенности группы активные и достигающие ц.д. и по образцу их можно будет работать с группами активные не достигающие ц.д. и неактивные но с достижением ц.д, что потенциально увеличит конверсию и удержание.

2. Повышение вовлеченности:

- Интерактивность: Внедрить больше возможностей фильтровать данные, например район или возможность ограничить радиус поиска по районам города.
- Мотивация: Разработать систему вознаграждений за активные действия. В дни низкой активности (начало недели) дополнительно ввести акции или иные программы для повышения активности в неактивные дни.

3. Оптимизация конверсии в просмотр контактов:

- Улучшение интерфейса: Оптимизировать интерфейс поиска и просмотра контактов для упрощения навигации и повышения удобства. По исследованию многие сессии становятся последними после поиска, просмотра карты или взаимодействия с рекомендациями (увидели или кликнули).
- Улучшение релевантности: Обеспечить более релевантные результаты поиска и рекомендаций контактов.

4. Работа с удержанием пользователей:

- Сегментация: Разделить неактивных пользователей на сегменты и разработать таргетированные стратегии возврата для каждого сегмента.
- Обратная связь: Наладить сбор обратной связи от уходящих пользователей для выявления причин оттока.
- Опросы: Проводить дополнительно среди активных и неактивных пользователей опросы и выявлять причины неактивности, с учётом недельной активности дополнительно сформировать рассылки или push-уведомления с напоминанием о поиске совершенном ранее, с предложениями похожего товара или о товаре в избранном. Это разумнее делать ближе к среде или четвергу, когда активность снижается максимально.
- Удержание активных: Сосредоточиться на удержании активных пользователей, предлагая им ценные бонусы и персонализированный контент.

5. Следить за длительностью сессий:

- Анализ экстремальных значений: Проанализировать причины очень коротких и очень длинных сессий для выявления проблем или возможностей.

- Триггеры: на длинных сессиях предлагать помощь или контент, относящийся к потребностям.

Дополнительные замечания:

- Незначительная разница между источниками (Google vs. Yandex) в конверсии в просмотр контактов, дальнейшее тестирование не требуется.
- Необходим постоянный мониторинг и анализ данных для оценки эффективности принятых мер и корректировки стратегии.

Презентация

[Презентация. ЯНДЕКС-ДИСК.](#)