

# Аналитика гипотез интернет-магазина для увеличения выручки

У нас есть список гипотез. Цель: приоритизация гипотез, запуск А/В-теста и анализ результатов.

План работы

1. Изучение общей информации
2. Предобработка данных
3. Часть 1. Приоритизация гипотез
4. Часть 2. Анализ А/В-теста
  - График кумулятивной выручки по группам
  - График кумулятивного среднего чека по группам
  - График относительного изменения кумулятивного среднего чека группы В к группе А
  - График кумулятивной конверсии по группам
  - График относительного изменения кумулятивной конверсии группы В к группе А
  - Точечный график количества заказов по пользователям
  - 95-й и 99-й перцентили количества заказов на пользователя
  - Точечный график стоимостей заказов
  - 95-й и 99-й перцентили стоимости заказов
  - Статистическая значимость различий в конверсии между группами по «сырым» данным
  - Статистическая значимость различий в среднем чеке заказа между группами по «сырым» данным
  - Очистка данных от аномалий
  - Статистическая значимость различий в конверсии между группами по «очищенным» данным
  - Статистическая значимость различий в среднем чеке заказа между группами по «очищенным» данным
5. Вывод

## Изучение общей информации

```
In [1]: import pandas as pd
import datetime as dt
import numpy as np
import matplotlib.pyplot as plt
import scipy.stats as stats
```

```
In [2]: hypothesis = pd.read_csv('datasets/hypothesis.csv')
```

```
In [3]: hypothesis
```

	Hypothesis	Reach	Impact	Confidence	Efforts
0	Добавить два новых канала привлечения трафика,...	3	10	8	6
1	Запустить собственную службу доставки, что сок...	2	5	4	10

		Hypothesis	Reach	Impact	Confidence	Efforts
<b>2</b>	Добавить блоки рекомендаций товаров на сайт ин...	8	3	7	3	
<b>3</b>	Изменить структура категорий, что увеличит кон...	8	3	3	8	
<b>4</b>	Изменить цвет фона главной страницы, чтобы уве...	3	1	1	1	
<b>5</b>	Добавить страницу отзывов клиентов о магазине,...	3	2	2	3	
<b>6</b>	Показать на главной странице баннеры с актуаль...	5	3	8	3	
<b>7</b>	Добавить форму подписки на все основные страни...	10	7	8	5	
<b>8</b>	Запустить акцию, дающую скидку на товар в день...	1	9	9	5	

In [4]: hypothesis.info()

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 9 entries, 0 to 8
Data columns (total 5 columns):
 #   Column      Non-Null Count  Dtype  
--- 
 0   Hypothesis   9 non-null     object 
 1   Reach        9 non-null     int64  
 2   Impact       9 non-null     int64  
 3   Confidence   9 non-null     int64  
 4   Efforts      9 non-null     int64  
dtypes: int64(4), object(1)
memory usage: 488.0+ bytes
```

In [5]: orders = pd.read\_csv('datasets/orders\_ab.csv')

In [7]: visitors = pd.read\_csv('datasets/visitors.csv')

In [8]: orders.head()

Out[8]:

	transactionId	visitorId	date	revenue	group
<b>0</b>	3667963787	3312258926	2019-08-15	1650	B
<b>1</b>	2804400009	3642806036	2019-08-15	730	B
<b>2</b>	2961555356	4069496402	2019-08-15	400	A
<b>3</b>	3797467345	1196621759	2019-08-15	9759	B
<b>4</b>	2282983706	2322279887	2019-08-15	2308	B

In [9]: visitors.head()

Out[9]:

	date	group	visitors
<b>0</b>	2019-08-01	A	719
<b>1</b>	2019-08-02	A	619
<b>2</b>	2019-08-03	A	507
<b>3</b>	2019-08-04	A	717
<b>4</b>	2019-08-05	A	756

In [10]: orders.info()

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1197 entries, 0 to 1196
Data columns (total 5 columns):
 #   Column      Non-Null Count  Dtype  
--- 
 0   transactionId  1197 non-null   int64  
 1   visitorId     1197 non-null   int64  
 2   date          1197 non-null   object  
 3   revenue        1197 non-null   int64  
 4   group          1197 non-null   object  
dtypes: int64(3), object(2)
memory usage: 46.9+ KB
```

In [11]: `visitors.info()`

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 62 entries, 0 to 61
Data columns (total 3 columns):
 #   Column      Non-Null Count  Dtype  
--- 
 0   date         62 non-null    object  
 1   group        62 non-null    object  
 2   visitors     62 non-null    int64  
dtypes: int64(1), object(2)
memory usage: 1.6+ KB
```

Пропущенных значений нет

Нужно привести названия столбцов к нижнему регистру и перевести столбцы с датами в нужный формат

## Предобработка данных

In [12]: `hypothesis.columns = hypothesis.columns.str.lower()`

In [13]: `orders.columns = ['transaction_id', 'visitor_id', 'date', 'revenue', 'group']`

In [14]: `orders['date'] = orders['date'].map(lambda x: dt.datetime.strptime(x, '%Y-%m-%d'))`

In [15]: `visitors['date'] = visitors['date'].map(lambda x: dt.datetime.strptime(x, '%Y-%m-%d'))`

Проверим таблицы на наличие дубликатов

In [16]: `orders.duplicated().sum()`

Out[16]: 0

In [17]: `visitors.duplicated().sum()`

Out[17]: 0

In [18]: `hypothesis = hypothesis.reset_index()`

In [19]: `hypothesis['hyp_id'] = hypothesis['index']  
hypothesis = hypothesis.set_index('index')`

Перейдем к основной части

## Часть 1. Приоритизация гипотез

Применим фреймворк ICE для приоритизации гипотез

$$ICE = \frac{Impact * Confidence}{Efforts}$$

In [20]: `hypothesis['ice'] = hypothesis['impact'] * hypothesis['confidence'] / hypothesis['efforts']`

In [21]: `hypothesis.sort_values('ice', ascending = False)[['hyp_id', 'ice']]`

Out[21]:

index	hyp_id	ice
8	8	16.200000
0	0	13.333333
7	7	11.200000
6	6	8.000000
2	2	7.000000
1	1	2.000000
5	5	1.333333
3	3	1.125000
4	4	1.000000

Теперь применим фреймворк RICE

$$RICE = \frac{Reach * Impact * Confidence}{Efforts}$$

In [22]: `hypothesis['rice'] = hypothesis['reach'] * hypothesis['impact'] * hypothesis['confidence'] / hypothesis['efforts']`

In [23]: `hypothesis.sort_values('rice', ascending = False)[['hyp_id', 'rice']]`

Out[23]:

index	hyp_id	rice
7	7	112.0
2	2	56.0
0	0	40.0
6	6	40.0
8	8	16.2
3	3	9.0
1	1	4.0
5	5	4.0
4	4	3.0

Гипотезы 7, 0 и 6 в топе во обоих фреймворках. Разница в первенстве из-за охвата.

Так как охват пользователей стоит учитывать, то предлагаю воспользоваться приоритизацией фреймворком RICE

$$ICE = \frac{Impact * Confidence}{Efforts}$$

$$RICE = \frac{Research * Impact * Confidence}{Efforts}$$

## Часть 2. Анализ А/В-теста

Мы провели А/В-тест и получили результаты, которые описаны в файлах /datasets/orders.csv и /datasets/visitors.csv

Перейдем к его анализу

### График кумулятивной выручки по группам

```
In [24]: dates_group = orders[['date', 'group']].drop_duplicates()
```

```
In [25]: orders_aggregated = dates_group.apply(lambda x: orders[np.logical_and(orders['date'] == x['date'], 'group' == x['group'])].groupby(['date', 'group']).agg({'transaction_id': pd.Series.nunique, 'visitor_id': pd.Series.nunique, 'revenue': 'sum'}), axis=1).sort_values(by=['date', 'group'])
```

```
In [26]: visitors_aggregated = dates_group.apply(lambda x: visitors[np.logical_and(visitors['date'] == x['date'], 'group' == x['group'])].groupby(['date', 'group']).agg({'visitors': 'sum'}), axis=1).sort_values(by=['date', 'group'])
```

```
In [27]: cumulative_data = orders_aggregated.merge(visitors_aggregated, left_on = ['date', 'group'], right_on = ['date', 'group'], how='left').drop(['visitors'], axis=1)
```

```
In [28]: cumulative_data.head()
```

```
Out[28]:
```

	date	group	transactions	buyers	revenue	visitors
0	2019-08-01	A	24	20	148579	719
1	2019-08-01	B	21	20	101217	713
2	2019-08-02	A	44	38	242401	1338
3	2019-08-02	B	45	43	266748	1294
4	2019-08-03	A	68	62	354874	1845

```
In [29]: cumulative_revenue_a = cumulative_data[cumulative_data['group'] == 'A'][['date', 'revenue']]
```

```
In [30]: cumulative_revenue_b = cumulative_data[cumulative_data['group'] == 'B'][['date', 'revenue']]
```

```
In [31]: plt.figure(figsize=(13,9))
plt.xlabel('Дата')
plt.ylabel('Выруча')
plt.title('Кумулятивная выручка по группам', color="k", fontsize=18, fontstyle="italic")
plt.plot(cumulative_revenue_a['date'], cumulative_revenue_a['revenue'], label = 'A',
         color="red", marker='o', linestyle='solid')
plt.plot(cumulative_revenue_b['date'], cumulative_revenue_b['revenue'], label = 'B',
         color="blue", marker='x', linestyle='dashed')
plt.grid()
plt.legend()
```

Out[31]: &lt;matplotlib.legend.Legend at 0x224a6c5fd30&gt;

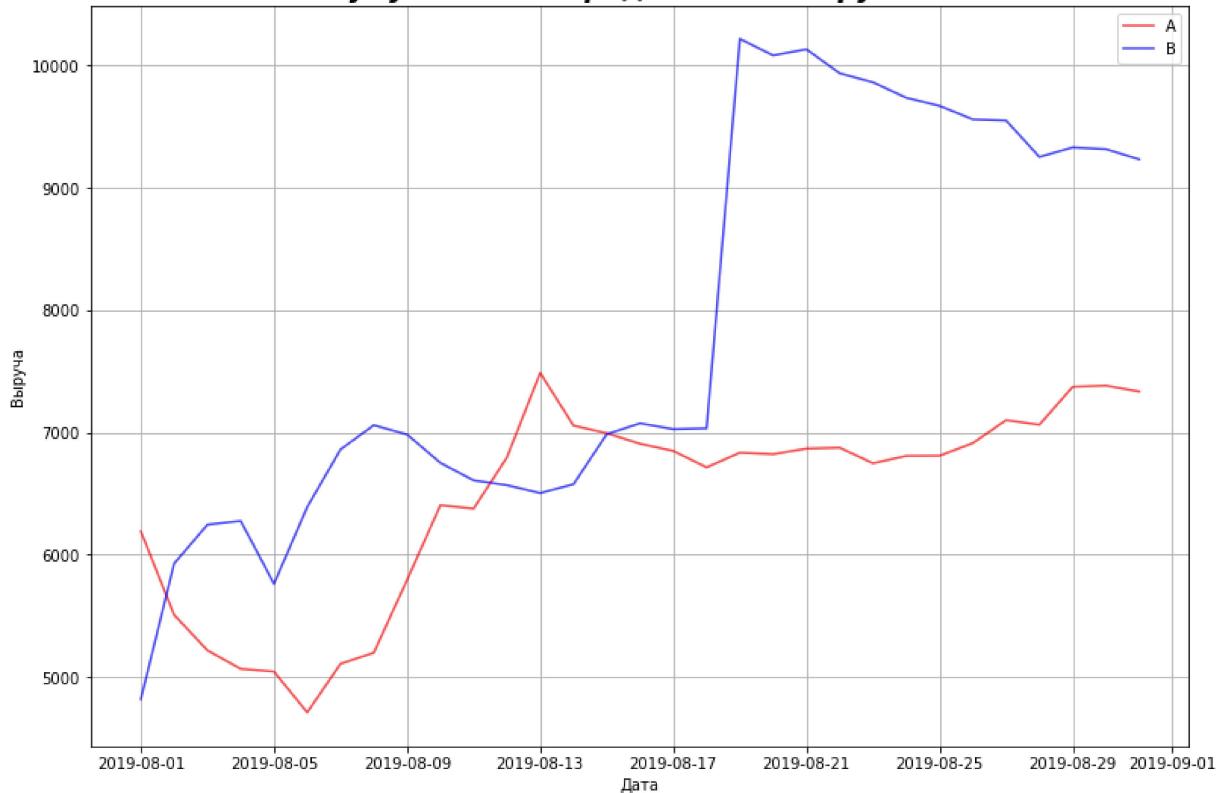


В начале теста выручки были примерно одинаковыми, затем группа В вырвалась в лидеры до конца теста

Наблюдаются резкие скачки, возможно, из-за аномально крупных заказов, что может повлиять на исследование

## График кумулятивного среднего чека по группам

```
In [32]: plt.figure(figsize=(13,9))
plt.xlabel('Дата')
plt.ylabel('Выручка')
plt.title('Кумулятивный средний чек по группам', color="k", fontsize=18, fontstyle="")
plt.plot(cumulative_revenue_a['date'], cumulative_revenue_a['revenue'] / cumulative_
plt.plot(cumulative_revenue_b['date'], cumulative_revenue_b['revenue'] / cumulative_
plt.legend()
plt.grid()
```

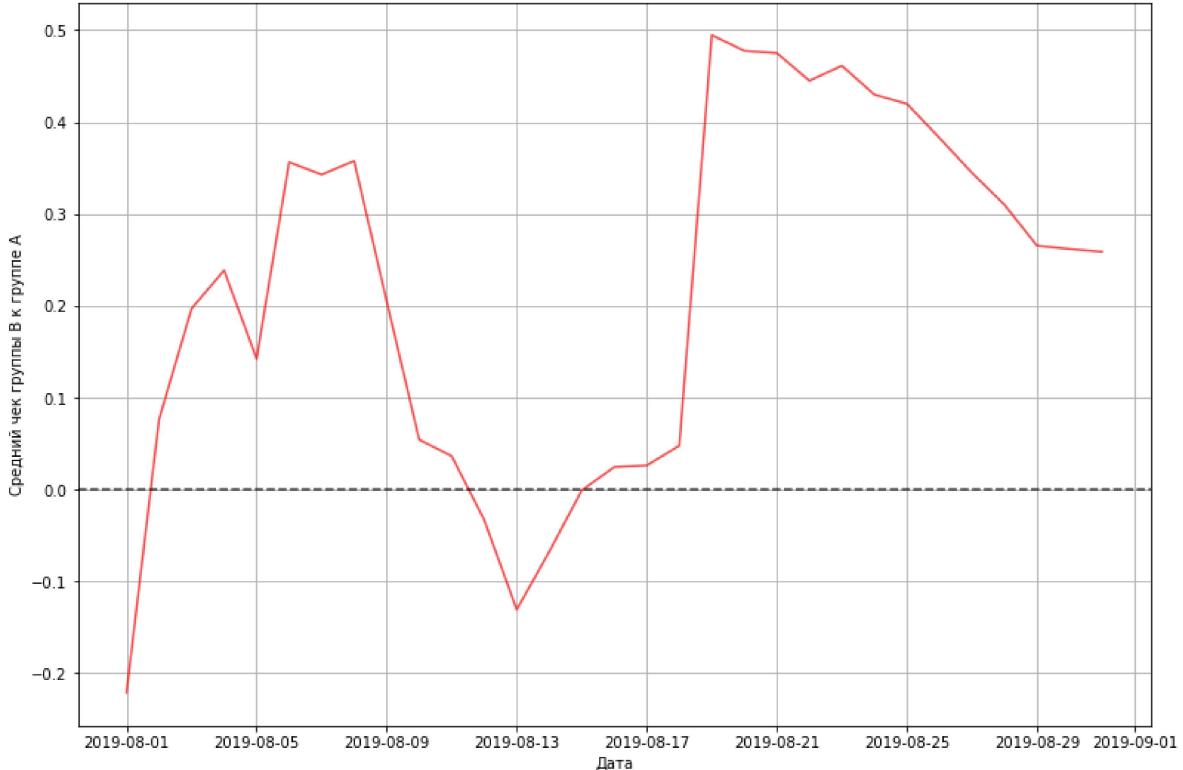
**Кумулятивный средний чек по группам**

Кумулятивные значения средних чеков продолжают скакать тоже из-за аномально крупных заказов

**График относительного изменения кумулятивного среднего чека группы В к группе А**

```
In [33]: merged_cumulative_revenue = cumulative_revenue_a.merge(cumulative_revenue_b, left_on='date', right_on='date')

plt.figure(figsize=(13,9))
plt.xlabel('Дата')
plt.ylabel('Средний чек группы В к группе А')
plt.title('Относительное изменение кумулятивного среднего чека группы В к группе А')
plt.plot(merged_cumulative_revenue['date'], (merged_cumulative_revenue['revenueB']/merged_cumulative_revenue['revenueA']) - 1)
plt.axhline(y=0, color='black', alpha = 0.7, linestyle='--')
plt.grid()
```

**Относительное изменение кумулятивного среднего чека группы B к группе A**

Значения резко меняются в некоторых точках. Наверное, в эти даты и были совершены аномально крупные и мелкие заказы

**График кумулятивной конверсии по группам**

```
In [34]: cumulative_data['conversion'] = cumulative_data['transactions'] / cumulative_data['visitors']
```

```
In [35]: cumulative_data_a = cumulative_data[cumulative_data['group']=='A']
cumulative_data_b = cumulative_data[cumulative_data['group']=='B']
```

Обозначим границы графика, чтобы лучше видеть колебания и стабилизацию

```
In [36]: plt.figure(figsize=(13,9))
plt.title('Кумулятивная конверсия по группам', color="k", fontsize=18, fontstyle="italic")
plt.xlabel('Дата')
plt.ylabel('Конверсия')
plt.plot(cumulative_data_a['date'], cumulative_data_a['conversion'], label = 'A', color='red')
plt.plot(cumulative_data_b['date'], cumulative_data_b['conversion'], label = 'B', color='blue')
plt.legend()
plt.axis(["2019-08-01", '2019-09-01', 0.02, 0.05])
plt.grid()
```

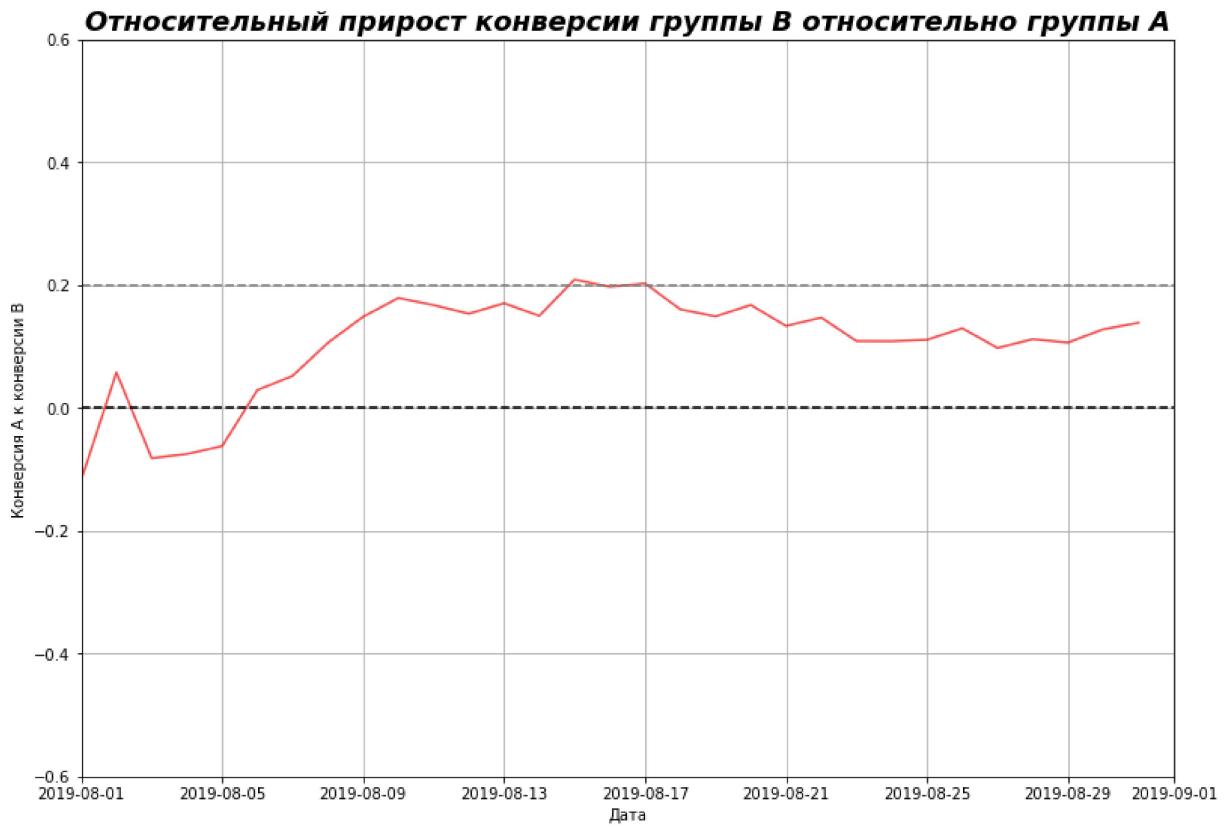


В начале теста конверсии групп колебались, затем группа В вырвалась вперед. Во второй части теста конверсии зафиксировались

## График относительного изменения кумулятивной конверсии группы В к группе А

```
In [37]: merged_cumulative_conversions = cumulative_data_a[['date', 'conversion']].merge(cumu
```

```
In [38]: plt.figure(figsize=(13,9))
plt.title('Относительный прирост конверсии группы В относительно группы А', color="k
plt.xlabel('дата')
plt.ylabel('Конверсия А к конверсии В')
plt.plot(merged_cumulative_conversions['date'], merged_cumulative_conversions['conve
plt.axhline(y=0, color='black', linestyle='--')
plt.axhline(y=0.2, color='grey', linestyle='--')
plt.axis(["2019-08-01", '2019-09-01', -0.6, 0.6])
plt.grid()
```

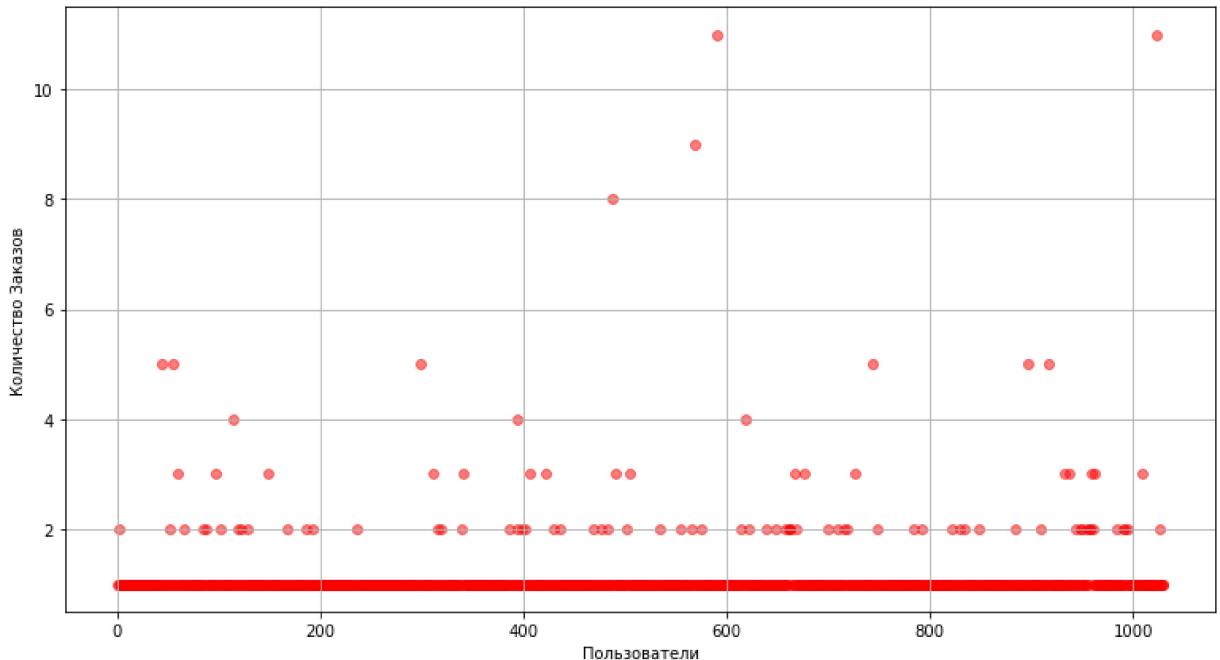


Группа В почти сразу вырвалась вперед, набрала преимущество, немного упала и зафиксировалась в районе прироста 10%

## Точечный график количества заказов по пользователям

```
In [39]: orders_users = orders.drop(['group', 'revenue', 'date'], axis = 1).groupby('visitor_id').count()
orders_users.columns = ['visitor_id', 'n_orders']
```

```
In [40]: plt.figure(figsize=(13,7))
plt.xlabel('Пользователи')
plt.ylabel('Количество Заказов')
plt.title('Количество заказов по пользователям', color="k", fontsize=18, fontstyle="italic")
x_values = pd.Series(range(0, len(orders_users)))
plt.scatter(x_values, orders_users['n_orders'], color = 'r', alpha = 0.5)
plt.grid()
```

**Количество заказов по пользователям**

Пользователи чаще совершают по одному заказу, крайне редко больше двух

**95-й и 99-й перцентили количества заказов на пользователя**

```
In [41]: print(np.percentile(orders_users['n_orders'], [95, 99]))
```

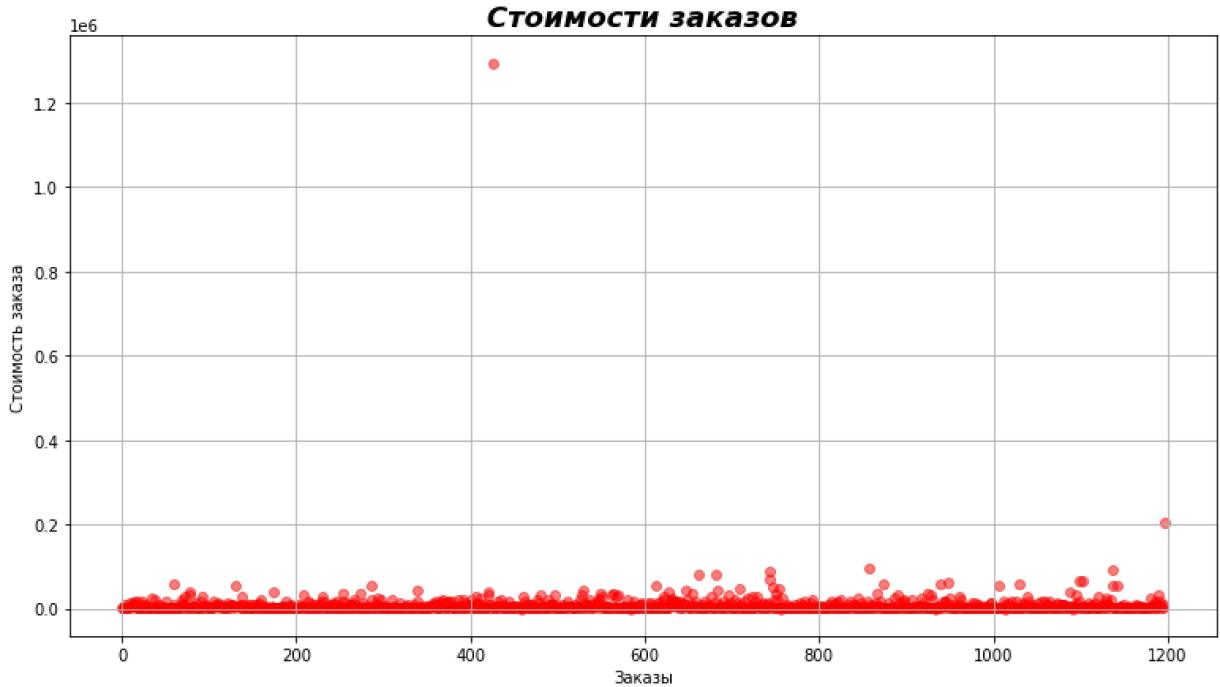
```
[2. 4.]
```

Не более 5% пользователей совершили более 2x заказов и не более 1% - более 4x

Значит, пользователей, совершивших более 2x заказов будем считать аномальными

**Точечный график стоимостей заказов**

```
In [42]: plt.figure(figsize=(13,7))
plt.ylabel('Стоимость заказа')
plt.xlabel('Заказы')
plt.title('Стоимости заказов', color="k", fontsize=18, fontstyle="italic", fontweight="bold")
x_values = pd.Series(range(0, len(orders['revenue'])))
plt.scatter(x_values, orders['revenue'], color = 'r', alpha = 0.5)
plt.grid()
```



Поменяем масштаб оси Y

```
In [43]: plt.figure(figsize=(13,7))
plt.ylabel('Стоимость заказа')
plt.xlabel('Заказы')
plt.title('Стоимости заказов', color="k", fontsize=18, fontstyle="italic", fontweight="bold")
x_values = pd.Series(range(0,len(orders['revenue'])))
plt.scatter(x_values, orders['revenue'], color = 'r', alpha = 0.5)
plt.axis([0, 1200, -0, 125000])
plt.grid()
```



Большинство заказов не привышают 20000 рублей, есть выбросы в районе 80К, 100К и даже 1200К

## 95-й и 99-й перцентили стоимости заказов

```
In [44]: print(np.percentile(orders['revenue'], [95, 99]))
```

```
[28000.  58233.2]
```

5% пользователей совершили заказы на сумму более 28000 рублей и не более 1% пользователей - на сумму более 58233 рублей

Все заказы на сумму более 28000 рублей будем считать аномальными

## Статистическая значимость различий в конверсии между группами по «сырым» данным

Сформулируем нулевую гипотезу - между конверсиями группы А и конверсиями группы В разницы нет

```
In [45]: alpha = .05
```

```
In [46]: orders_users_a = orders[orders['group'] == 'A'].groupby('visitor_id', as_index = False)
orders_users_a.columns = ['visitor_id', 'n_orders']
```

```
In [47]: orders_users_b = orders[orders['group'] == 'B'].groupby('visitor_id', as_index = False)
orders_users_b.columns = ['visitor_id', 'n_orders']
```

```
In [48]: sample_a = pd.concat([orders_users_a['n_orders'], pd.Series(0, index=np.arange(visit
```

```
In [49]: sample_b = pd.concat([orders_users_b['n_orders'], pd.Series(0, index=np.arange(visit
```

```
In [50]: orders_per_date = orders.groupby(['date', 'group'], as_index = False).agg({'transaction_id': 'count'})
orders_per_date
```

```
Out[50]:
```

	date	group	transaction_id
0	2019-08-01	A	24
1	2019-08-01	B	21
2	2019-08-02	A	20
3	2019-08-02	B	24
4	2019-08-03	A	24
...	...	...	...
57	2019-08-29	B	20
58	2019-08-30	A	11
59	2019-08-30	B	24
60	2019-08-31	A	12
61	2019-08-31	B	20

62 rows × 3 columns

```
In [51]: orders_per_date = orders_per_date.merge(visitors, on = ['date', 'group'])
```

```
In [52]: orders_per_date['conversion'] = orders_per_date['transaction_id'] / orders_per_date[
```

```
In [53]: print('p-value:', '{0:.5f}'.format(stats.mannwhitneyu(sample_a, sample_b)[1]))
if (stats.mannwhitneyu(sample_a, sample_b)[1] < alpha):
    print("Отвергаем нулевую гипотезу: между конверсиями есть значимая разница")
else:
    print("Не получилось отвергнуть нулевую гипотезу, нет оснований считать конверси
```

```
print('Относительный прирост конверсии группы В к конверсии группы А:', '{:.0%}'.for
p-value: 0.00840
Отвергаем нулевую гипотезу: между конверсиями есть значимая разница
Относительный прирост конверсии группы В к конверсии группы А: 15%
```

## Статистическая значимость различий в среднем чеке заказа между группами по «сырым» данным

Нулевая гипотеза - различий в среднем чеке заказов группы А и группы В нет

```
In [54]: orders_per_date = orders_per_date.merge(orders.groupby(['date', 'group']), as_index=False)
In [55]: orders_per_date['avg_check'] = orders_per_date['revenue'] / orders_per_date['transactions']
In [56]: print('p-value:', '{0:.5f}'.format(stats.mannwhitneyu(orders_per_date[orders_per_date['group']=='A']['avg_check'],
if (stats.mannwhitneyu(orders_per_date[orders_per_date['group']=='A']['avg_check'],
    print("Отвергаем нулевую гипотезу: между средними чеками есть значимая разница")
else:
    print("Не получилось отвергнуть нулевую гипотезу, нет оснований считать средние чеки разными")
print('Относительное различие между средними чеками групп:', "{:.1%}".format(orders[orders['group']=='B']['avg_check'].mean() / orders[orders['group']=='A']['avg_check'].mean() - 1)))
p-value: 0.20309
Не получилось отвергнуть нулевую гипотезу, нет оснований считать средние чеки разными
И
Относительное различие между средними чеками групп: 25.9%
```

## Очистка данных от аномалий

```
In [57]: expensive_orders = orders[orders['revenue'] > 28000]['visitor_id']
In [58]: many_orders = orders_users[orders_users['n_orders'] > 2]['visitor_id']
In [59]: abnormal_visitors = pd.concat([many_orders, expensive_orders], axis=0).drop_duplicates()
In [60]: abnormal_visitors.shape
Out[60]: (83,)
```

83 аномальных пользователя

## Статистическая значимость различий в конверсии между группами по «очищенным» данным

Сформулируем нулевую гипотезу - между конверсиями группы А и конверсиями группы В разницы нет

```
In [61]: sample_a_filt = pd.concat([orders_users_a[np.logical_not(orders_users_a['visitor_id'].isin(abnormal_visitors))], orders_users_b[np.logical_not(orders_users_b['visitor_id'].isin(abnormal_visitors))]])
In [62]: print('p-value', '{0:.5f}'.format(stats.mannwhitneyu(sample_a_filt, sample_b_filt)[1]))
if (stats.mannwhitneyu(sample_a_filt, sample_b_filt)[1] < alpha):
    print("Отвергаем нулевую гипотезу: между конверсиями есть значимая разница")
else:
    print("Не получилось отвергнуть нулевую гипотезу, нет оснований считать конверсии одинаковыми")
print('Разница конверсий между группами:', '{0:.1%}'.format(sample_b_filt.mean() / sample_a_filt.mean() - 1))
p-value 0.00608
Отвергаем нулевую гипотезу: между конверсиями есть значимая разница
```

Разница конверсий между группами: 17.6%

Чистые данные показали то же, что и сырье. Статистическая значимость достигнута.

Конверсии группы В на 17,6% выше конверсий группы А

## Статистическая значимость различий в среднем чеке заказа между группами по «очищенным» данным

Нулевая гипотеза - различий в среднем чеке заказов группы А и группы В нет

```
In [63]: print('p-value:', '{0:.3f}'.format(stats.mannwhitneyu(orders[np.logical_and(orders['group'] == 'A', np.logical_not(orders['group'] == 'B'))], orders[np.logical_and(orders['group'] == 'B', np.logical_not(orders['group'] == 'A'))])) if (stats.mannwhitneyu(orders[np.logical_and(orders['group'] == 'A', np.logical_not(orders['group'] == 'B'))], orders[np.logical_and(orders['group'] == 'B', np.logical_not(orders['group'] == 'A'))]).pvalue < 0.05): print("Отвергаем нулевую гипотезу: между средними чеками есть значимая разница") else: print("Не получилось отвергнуть нулевую гипотезу, нет оснований считать средние чеки разными") print('Разница между чеками', '{0:.1%}'.format(orders[np.logical_and(orders['group'] == 'B', np.logical_not(orders['group'] == 'A'))].mean() - orders[np.logical_and(orders['group'] == 'A', np.logical_not(orders['group'] == 'B'))].mean()))
```

p-value: 0.376  
 Не получилось отвергнуть нулевую гипотезу, нет оснований считать средние чеки разными  
 Разница между чеками -2.2%

После удаления аномалий p-value увеличился до 0.376, статистических различий в среднем чеке между группами так и нет. Хотя теперь группа В отстает на 2.2% от группы А

## Вывод

В результате проделанной работы выяснили:

- по конверсии группа В лучше группы А (и по сырым, и по очищенным данным)
- нет статистически значимого различия по среднему чеку между группами ни по «сырым», ни по данным после фильтрации аномалий
- график различия конверсии между группами говорит о том, что результаты группы В лучше
- график различия среднего чека колеблется, поэтому нужно было очистить данные от аномалий

Так как цель увеличить выручку интернет-магазина, конверсии в продажи помогут повысить выручку. Останавливаем тест, группа В победила