Project description

Context You are an analyst for a large online store. Together with the marketing department, you have prepared a list of hypotheses for increasing revenue. Prioritize hypotheses, run an A/B test, and analyze the results.

Part 1. Prioritization of hypotheses.

In the /datasets/hypothesis file.csv there are 9 hypotheses for increasing the revenue of an online store with the specified parameters Reach, Impact, Confidence, Effort.

The /datasets/hypothesis.csv file.

• Hypothesis – a brief description of the hypothesis; • Reach — reach users on a 10-point scale; • Impact — impact on users on a 10-point scale; • Confidence — confidence in the hypothesis on a 10-point scale; • Efforts — the cost of resources to test the hypothesis on a 10-point scale. The higher the value of Efforts, the more expensive the hypothesis test.

In [58]:
```
import pandas as pd
import scipy.stats as stats
import datetime as dt
```

In [59]:
```
pd.set_option('display.max_colwidth', False)
```

In [60]:
```
hip= pd.read_csv('hypothesis.csv')

hip
```

Out[60]:

	Hypothesis	Reach	Impact	Confidence	Efforts
0	Add two new channels to attract traffic, which will attract 30% more users	3	10	8	6
1	Launch your own delivery service, which will reduce the delivery time of orders	2	5	4	10
2	Add product recommendation blocks to the website of the online store to increase conversion and the average order check	8	3	7	3
3	Change the structure of categories, which will increase conversion, because users will quickly find the right product	8	3	3	8
4	Change the background color of the homepage to increase user engagement	3	1	1	1
5	Add a page of customer reviews about the store, which will increase the number of orders	3	2	2	3
6	Show banners with current promotions and sales on the main page to increase conversions	5	3	8	3
7	Add a subscription form to all major pages to build a customer base for email newsletters	10	7	8	5
8	Launch a promotion that gives a discount on a product on your birthday	1	9	9	5

In [61]:
```
### Примените фреймворк ICE для приоритизации гипотез. Отсортируйте их по убыванию при

hip['ICE']=round(((hip['Impact']*hip['Confidence'])/hip['Efforts']), 2)

hip[['Hypothesis','ICE']].sort_values(by='ICE', ascending=False)
```

Out[61]:

	Hypothesis	ICE
8	Launch a promotion that gives a discount on a product on your birthday	16.20
0	Add two new channels to attract traffic, which will attract 30% more users	13.33
7	Add a subscription form to all major pages to build a customer base for email newsletters	11.20
6	Show banners with current promotions and sales on the main page to increase conversions	8.00
2	Add product recommendation blocks to the website of the online store to increase conversion and the average order check	7.00
1	Launch your own delivery service, which will reduce the delivery time of orders	2.00
5	Add a page of customer reviews about the store, which will increase the number of orders	1.33
3	Change the structure of categories, which will increase conversion, because users will quickly find the right product	1.12
4	Change the background color of the homepage to increase user engagement	1.00

In [62]:
```
### Примените фреймворк RICE для приоритизации гипотез. Отсортируйте их по убыванию пр
hip['RICE']= (hip['Reach']*hip['Impact']*hip['Confidence'])/hip['Efforts']
print(hip[['Hypothesis','RICE']].sort_values(by='RICE', ascending=False))
```

```
Hypothesis  \
7  Добавить форму подписки на все основные страницы, чтобы собрать базу клиентов для email-рассылок
2  Добавить блоки рекомендаций товаров на сайт интернет магазина, чтобы повысить конверсию и средний чек заказа
0  Добавить два новых канала привлечения трафика, что позволит привлекать на 30% больше пользователей
6  Показать на главной странице баннеры с актуальными акциями и распродажами, чтобы увеличить конверсию
8  Запустить акцию, дающую скидку на товар в день рождения
3  Изменить структура категорий, что увеличит конверсию, т.к. пользователи быстрее найдут нужный товар
1  Запустить собственную службу доставки, что сократит срок доставки заказов
5  Добавить страницу отзывов клиентов о магазине, что позволит увеличить количество заказов
4  Изменить цвет фона главной страницы, чтобы увеличить вовлеченность пользователей

    RICE
7  112.0
2  56.0
0  40.0
6  40.0
8  16.2
3  9.0
1  4.0
5  4.0
4  3.0
```

Indicate how hypothesis prioritization has changed when using RICE instead of ICE. Explain why this happened.

- ICE (from the English impact, confidence, effort / ease "impact, confidence, effort / simplicity")
- RICE (R from reach, "reach")

Reach — how many users will be affected by the change you want to make; Impact — how much the change will affect users, their experience and satisfaction with the product; Confidence — how confident you are that this change will affect users in this way; Efforts — how much it costs to test this hypothesis.

The RICE indicator differs from ICE only in an additional component - Reach. This is in points the reach of the audience. Therefore, in the RICE parameter, hypotheses with 10 points in the Reach-audience coverage column came to the fore. Moreover, RICH is the opposite of Impact. The greater the reach of the audience (Reach) - the smaller the impact (Impact)

That is, in the ICE parameter, the first place belongs to hypotheses with a small audience coverage, but a strong impact on customers: 1st place: Launch a promotion that gives a discount on goods per day - few people will see this announcement per day, but those who see and buy - this will dramatically increase income in the short term.

In the RICE parameter, they take the coverage of the audience - albeit with a small impact: 1st place: Add a subscription form to all major pages - a large audience reach - but this will have

little effect on purchasing power. This is for the long term.

Часть 2. Анализ A/B-теста

Вы провели A/B-тест и получили результаты, которые описаны в файлах /datasets/orders.csv и /datasets/visitors.csv.

Проанализируйте A/B-тест:

Данные для второй части:

Файл /datasets/orders.csv.

• transactionId — идентификатор заказа; • visitorId — идентификатор пользователя, совершившего заказ; • date — дата, когда был совершён заказ; • revenue — выручка заказа; • group — группа A/B-теста, в которую попал заказ.

Файл /datasets/visitors.csv.

• date — дата; • group — группа A/B-теста; • visitors — количество пользователей в указанную дату в указанной группе A/B-теста

In [64]:
```
orders = pd.read_csv('orders.csv')
orders.head(3)
```

Out[64]:

	transactionId	visitorId	date	revenue	group
0	3667963787	3312258926	2019-08-15	1650	B
1	2804400009	3642806036	2019-08-15	730	B
2	2961555356	4069496402	2019-08-15	400	A

In [65]:
```
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 [66]:
```
orders['date'] = pd.to_datetime(orders['date'])
```

In [67]:
```
visitors = pd.read_csv('visitors.csv')
visitors.head(3)
```

Out[67]:

	date	group	visitors
0	2019-08-01	A	719
1	2019-08-02	A	619
2	2019-08-03	A	507

In [68]:
```
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
```

In [69]:
```
visitors['date'] = pd.to_datetime(visitors['date'])
```

In [70]:
```
##Смотрим наличие дубликатов
orders.duplicated().sum()
```

Out[70]:
```
0
```

In [72]:
```
visitors.duplicated().sum()
```

Out[72]:
```
0
```

In [73]:
```
##Смотрим наличие пропусков
visitors.isna().sum()
```

Out[73]:
```
date        0
group       0
visitors    0
dtype: int64
```

In [74]:
```
orders.isna().sum()
```

Out[74]:
```
transactionId    0
visitorId        0
date             0
revenue          0
group            0
dtype: int64
```

In [75]:
```
## сгруппировали данные по группе и посчитали количества в каждом столбце.

orders.groupby('group').count()
```

Out[75]:

group	transactionId	visitorId	date	revenue
A	557	557	557	557
B	640	640	640	640

In [79]:
```
## метод nunique() - кол-во уникальных данных показывает что есть покупатели которые и

orders.groupby('group').nunique()
```

Out[79]:

group	transactionId	visitorId	date	revenue
A	557	503	31	419
B	640	586	31	450

In [80]:
```
visitors.groupby('group').count()
```

Out[80]:

group	date	visitors
A	31	31
B	31	31

In [81]:
```
visitors.groupby('group').nunique()
```

Out[81]:

group	date	visitors
A	31	31
B	31	30

1. Постройте график кумулятивной выручки по группам. Сделайте выводы и предположения.

Чтобы построить графики, нужно собрать кумулятивные данные. Объявим датафрейм cumulativeData со столбцами:

date — дата; group — группа A/B-теста (A или B);

transaction_cum — кумулятивное количество заказов на указанную дату в указанной группе; (*transactionId)

visitorId_cum — кумулятивное количество пользователей, совершивших хотя бы один заказ, на указанную дату в указанной группе; (*visitorId)

revenue_cum — кумулятивная выручка на указанную дату в указанной группе (средний чек);

visitors_cum — кумулятивное количество посетителей интернет-магазина на указанную дату в определённой группе. (*visitors)

In [82]:
```
# 1. создаем массив уникальных пар значений дат и групп теста
datesGroups = orders[['date','group']].drop_duplicates()
datesGroups.head(3)
```

Out[82]:

	date	group
0	2019-08-15	B
2	2019-08-15	A
45	2019-08-16	A

In [83]:
```
# 2.Получим строки таблицы orders, дата которых меньше или равна дате элемента из date
# а группа теста равна группе из datesGroups. Это и есть агрегир.кумулятивные ДАННЫЕ о

import numpy as np

ordersAggregated = datesGroups.apply(lambda x: orders[np.logical_and(orders['date'] <=
.agg({'date' : 'max', 'group' : 'max', 'transactionId' : 'nunique', 'visitorId' : 'nun
.sort_values(by=['date','group'])

### выводим на печать избранные строки таблицы orders
ordersAggregated.head(5)
```

Out[83]:

	date	group	transactionId	visitorId	revenue
55	2019-08-01	A	24	20	148579
66	2019-08-01	B	21	20	101217
175	2019-08-02	A	44	38	242401
173	2019-08-02	B	45	43	266748
291	2019-08-03	A	68	62	354874

In [85]:
```
# 3. Аналогично получим агрегированные кумулятивные по дням данные о посетителях интер
# Это и есть агрегир.кумулятивные ДАННЫЕ о ПОСЕТИТЕЛЯХ интернет магазина.


visitorsAggregated = datesGroups.apply(lambda x: visitors[np.logical_and(visitors['dat
.sort_values(by=['date','group'])

visitorsAggregated.head(5)
```

Out[85]:

	date	group	visitors
55	2019-08-01	A	719
66	2019-08-01	B	713
175	2019-08-02	A	1338
173	2019-08-02	B	1294
291	2019-08-03	A	1845

In [86]:
```
# 4. Объединим обе таблицы в одну - cumulativeData:

cumulativeData = ordersAggregated.merge(visitorsAggregated, left_on=['date', 'group'],
cumulativeData.columns = ['date', 'group','transaction_cum', 'visitorId_cum','revenue_

cumulativeData.head(5)
```

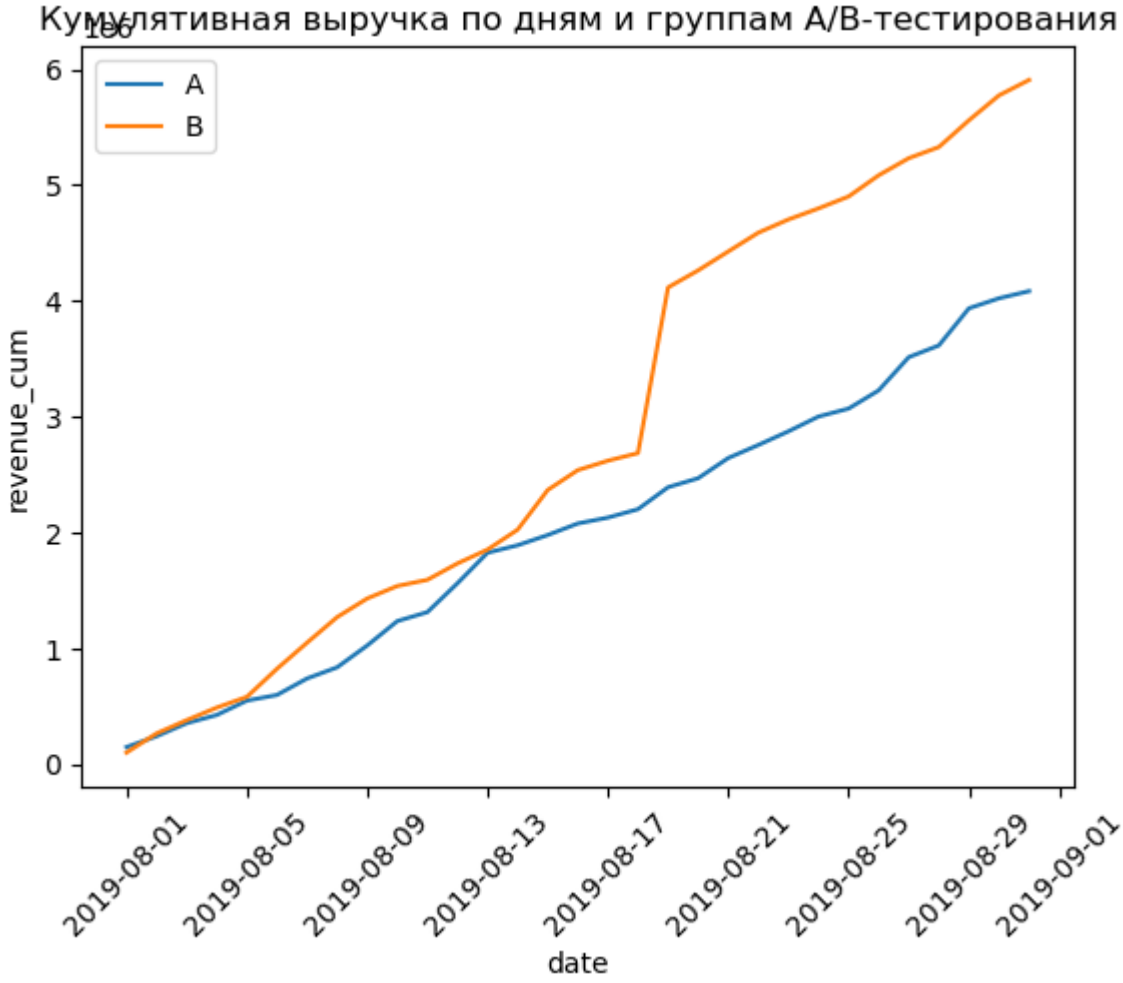
Out[86]:

	date	group	transaction_cum	visitorId_cum	revenue_cum	visitors_cum
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

СТРОИМ ГРАФИКИ.

In [87]:
```
# Построим графики кумулятивной выручки по дням и группам A/B-тестирования:

import matplotlib.pyplot as plt
import numpy as np

# датафрейм с кумулятивным количеством заказов и кумулятивной выручкой по дням в групп
cumulativeRevenueA = cumulativeData[cumulativeData['group']=='A'][['date','revenue_cum

# датафрейм с кумулятивным количеством заказов и кумулятивной выручкой по дням в групп
cumulativeRevenueB = cumulativeData[cumulativeData['group']=='B'][['date','revenue_cum


# Строим график выручки группы A
plt.plot(cumulativeRevenueA['date'], cumulativeRevenueA['revenue_cum'], label='A')

# Строим график выручки группы B
plt.plot(cumulativeRevenueB['date'], cumulativeRevenueB['revenue_cum'], label='B')


## название графика
plt.title('Кумулятивная выручка по дням и группам A/B-тестирования')

plt.legend(loc='upper left')
plt.xlabel('date')
plt.ylabel('revenue_cum')
```

```
plt.xticks(rotation=45)# повернули дату под углом 45 градусов
plt.show()
```



Выручка почти равномерно увеличивается в течение всего теста. Хороший знак. Однако графики выручки обеих групп в нескольких точках резко растут. Это может сигнализировать о всплесках числа заказов, либо о появлении очень дорогих заказов в выборке.

2.Постройте график кумулятивного среднего чека по группам. Сделайте выводы и предположения.

In [88]:
```
### Построим графики среднего чека по группам — разделим кумулятивную выручку на кумул

plt.plot(cumulativeRevenueA['date'], cumulativeRevenueA['revenue_cum']/cumulativeRever

plt.plot(cumulativeRevenueB['date'], cumulativeRevenueB['revenue_cum']/cumulativeRever

## название графика
plt.title('Средний чек по группам  группам A/B-тестирования')

plt.legend(loc='upper left')
plt.xlabel('date')
plt.ylabel('Middle receipt')
```

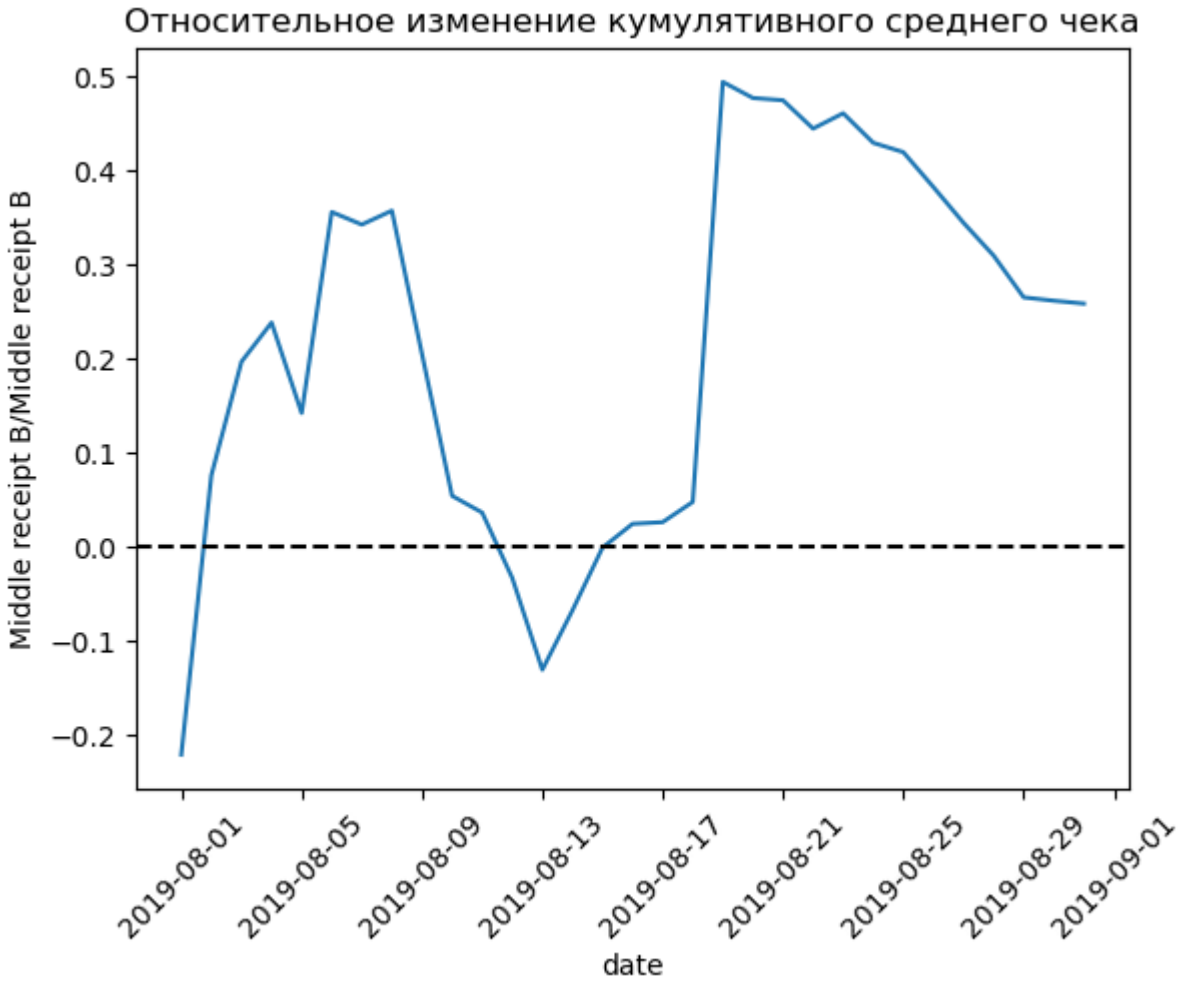
```
plt.xticks(rotation=45)# повернули дату под углом 45 градусов
plt.show()
```

Средний чек тоже становится равномерным ближе к концу теста: установился для группы B и продолжает падать для группы A. Возможно, в группу A в первой половине теста попали крупные заказы (резкий всплеск на графике). Тогда ей нужно больше данных, чтобы прийти к реальному среднему чеку и установиться на его уровне.

3. Постройте график относительного изменения кумулятивного среднего чека группы B к группе A. Сделайте выводы и предположения.

In [89]:
```
### Добавим горизонтальную ось методом axhline() (от англ. «горизонтальная линия попер
# собираем данные в одном датафрейме

mergedCumulativeRevenue = cumulativeRevenueA.merge(cumulativeRevenueB, left_on='date',

# строим отношение средних чеков
plt.plot(mergedCumulativeRevenue['date'], (mergedCumulativeRevenue['revenue_cumB']/mer

# добавляем ось X
plt.axhline(y=0, color='black', linestyle='--')

## название графика
```

```
plt.title('Относительное изменение кумулятивного среднего чека')


plt.ylabel('Middle receipt B/Middle receipt B')
plt.xlabel('date')

plt.xticks(rotation=45)# повернули дату под углом 45 градусов
plt.show()
```



В нескольких точках график различия между сегментами резко «скачет». Где-то уж точно спрятались крупные заказы и выбросы!

4. Постройте график кумулятивного среднего количества заказов на посетителя по группам. Сделайте выводы и предположения.

In [90]:
```
# считаем количество заказов на 1 посетителя
cumulativeData['conversion'] = cumulativeData['transaction_cum']/cumulativeData['visit

# отделяем данные по группе A
cumulativeDataA = cumulativeData[cumulativeData['group']=='A']

# отделяем данные по группе B
cumulativeDataB = cumulativeData[cumulativeData['group']=='B']
```

```
## название графика
plt.title('Кумулятивное количество заказов на посетителя групп A и B')

## x и y
plt.xlabel('date')
plt.ylabel('Заказ на посетителя')


# строим графики
plt.plot(cumulativeDataA['date'], cumulativeDataA['conversion'], label='A')
plt.plot(cumulativeDataB['date'], cumulativeDataB['conversion'], label='B')


plt.xticks(rotation=45)# повернули дату под углом 45 градусов

plt.legend(loc='upper right')
plt.show()
```

Симметричный график получился! Группы колебались около одного значения, но затем количество заказов группы B вырвалось вперёд и зафиксировалось, а количество заказов группы A просело и также зафиксировалось.

5.Постройте график относительного изменения кумулятивного среднего количества заказов на

посетителя группы B к группе A. Сделайте выводы и предположения.

In [91]:
```
mergedCumulativeConversions = cumulativeDataA[['date','conversion']]\
.merge(cumulativeDataB[['date','conversion']], left_on='date', right_on='date', how='l

plt.plot(mergedCumulativeConversions['date'], mergedCumulativeConversions['conversionB



plt.axhline(y=0, color='black', linestyle='--')
plt.axhline(y=-0.1, color='grey', linestyle='--')

## название графика
plt.title('Относительного изменение кумулятивного количество заказов на посетителя гру

## x и y
plt.xlabel('date')
plt.ylabel('Заказ на посетителя B к A')

plt.xticks(rotation=45)# повернули дату под углом 45 градусов


plt.show()
```

В начале теста группа A значительно выигрывала группе B, затем группа B вырвалась вперёд. Потом количество сново падала, но теперь постепенно растёт. В целом отношение количества группы B к A ещё не установилось, и сейчас делать какие-либо выводы по тесту нельзя. Впрочем, сперва стоит проанализировать аномалии, возможно, они изменят картину.

6. Постройте точечный график количества заказов по пользователям. Сделайте выводы и

предположения.

Файл /datasets/orders.csv.

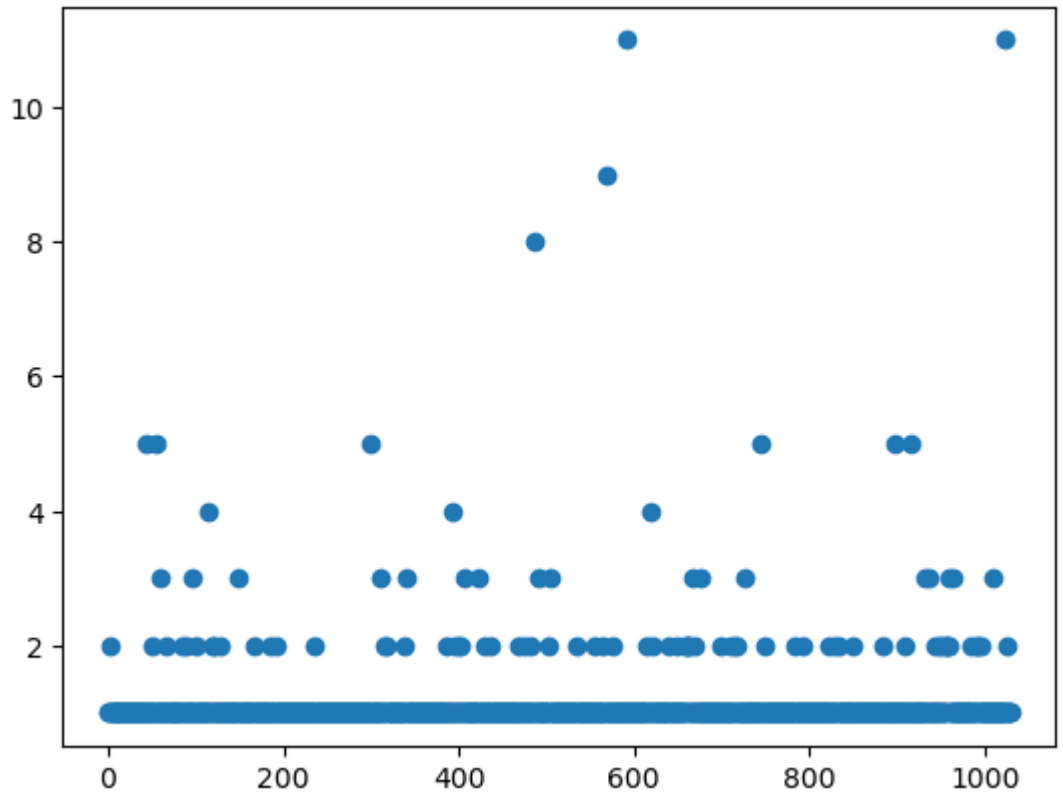
•transactionId — идентификатор заказа; •visitorId — идентификатор пользователя, совершившего заказ;

In [92]:
```
## введём переменную, чтобы выделить уникальных пользователей и
## добавим количество уникальных товаров - 'nunique':

orders_2 = (
    orders.groupby('visitorId', as_index=False)
    .agg({'transactionId': 'nunique'})
)
orders_2.columns = ['userId', 'orders']

print(orders_2.sort_values(by='orders', ascending=False).head(5))
```

```
          userId  orders
1023  4256040402  11
591   2458001652  11
569   2378935119  9
487   2038680547  8
44    199603092   5
```

In [93]:
```
## Построим диаграмму методом scatter(),ей передают значения по осям X и Y.

x_values = pd.Series(range(0, len(orders_2)))
plt.scatter(x_values, orders_2['orders'])
```

Out[93]:
```
<matplotlib.collections.PathCollection at 0x235a2e82c40>
```



Пользователей заказавших больше 2-х раз совсем мало. Они вполне могут быть аномальными. 2 заказа - это нормально или много даст ответ подсчёт выборочных перцентлей.

7. Посчитайте 95-й и 99-й перцентили количества заказов на пользователя. Выберите границу для определения аномальных пользователей.

Посчитаем 95-й и 99-й выборочные перцентили количества заказов по пользователям методом np.percentile()

In [94]:
```
## Метод np.percentile('column', [percentile1, percentile2, percentile3]) находит пер

import numpy as np
print(np.percentile(orders_2['orders'], [95, 99]))
```

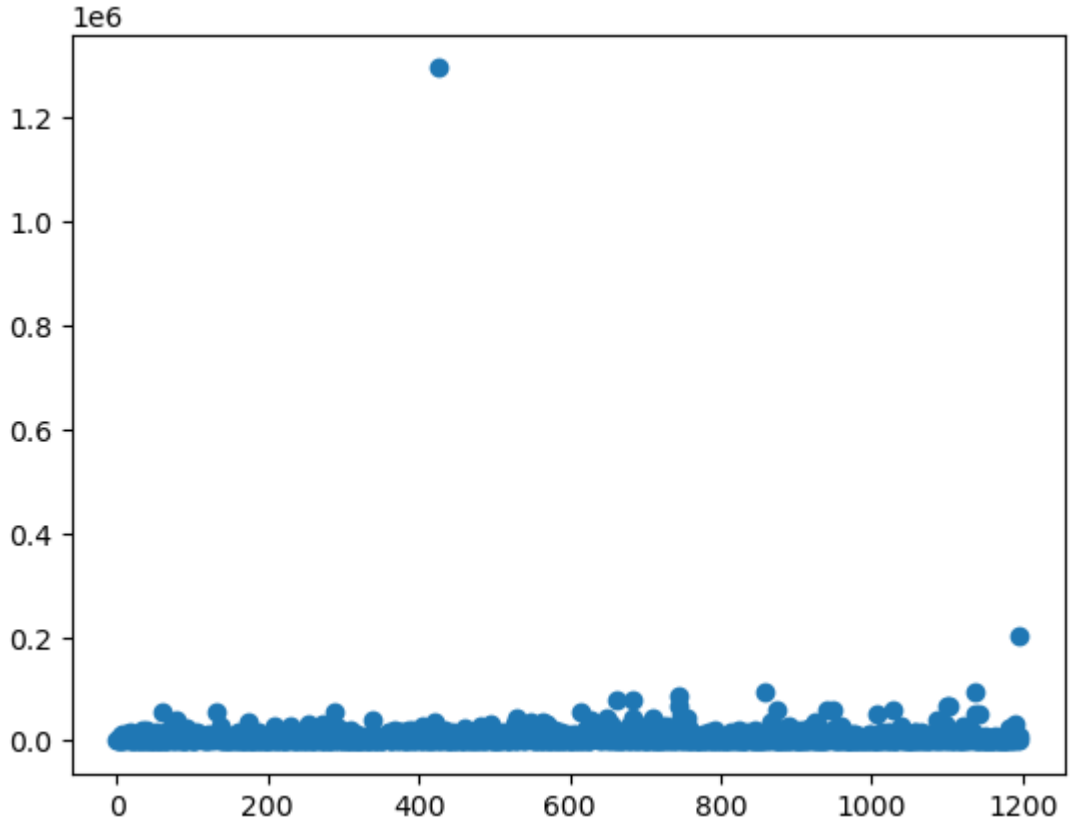
```
[2. 4.]
```

Не более 5% пользователей совершали больше 2 заказов. И не более 1% пользователей - больше 4 заказов. Граница аномального количества заказов от 2 заказов.

8. Постройте точечный график стоимостей заказов. Сделайте выводы и предположения.

In [95]:
```
##  Построим диаграмму методом scatter(). Напомним, что ему передают значения по осям
## Значения по горизонтальной оси есть в прекоде, в переменной x_values – сгенерирован
## Значения для вертикальной оси возьмём из столбца 'revenue' датафрейма orders.

x_values = pd.Series(range(0,len(orders['revenue'])))
plt.scatter(x_values, orders['revenue'])
```

Out[95]:
```
<matplotlib.collections.PathCollection at 0x235a2e1c670>
```



Есть вброс в районе 1.2 млн рублей и в районе 200 тыс.рублей. Заказ на 1.2 млн. рублей выглядит аномально.

9. Посчитайте 95-й и 99-й перцентили стоимости заказов. Выберите границу для определения аномальных заказов.

revenue — выручка заказа

In [96]:
```
# Передадим методу np.percentile() столбец 'revenue' датафрейма orders
# и список перцентилей — [95, 99]
##import numpy as np
print(np.percentile(orders['revenue'], [95, 99]))
```

```
[28000.  58233.2]
```

Не более 5% заказов чек дороже 28000 рублей. И не больше, чем у 1 % заказов-дороже 58233 рублей. Граница аномальной стоимости заказов - всё что выше 28000 рублей - аномалия.

10. Посчитайте статистическую значимость различий в среднем КОЛИЧЕСТВЕ заказов на посетителя между группами по «сырым» данным.

СНАЧАЛА соберем таблицу - data

Файл /datasets/orders.csv.

• transactionId — идентификатор заказа; • visitorId — идентификатор пользователя, совершившего заказ; • date — дата, когда был совершён заказ; • revenue — выручка заказа; • group — группа A/B-теста, в которую попал заказ.

Начинаем с таблицы visitors

In [98]:
```
## 1. visitorsA,B – количество пользователей в выбранную дату в группе A,B:

visitorsADaily = visitors[visitors['group'] == 'A'][['date', 'visitors']]
visitorsADaily.columns = ['date', 'visitorsPerDateA']

visitorsBDaily = visitors[visitors['group'] == 'B'][['date', 'visitors']]
visitorsBDaily.columns = ['date', 'visitorsPerDateB']

visitorsADaily.head(5)
```

Out[98]:

	date	visitorsPerDateA
0	2019-08-01	719
1	2019-08-02	619
2	2019-08-03	507
3	2019-08-04	717
4	2019-08-05	756

In [99]:
```
visitorsBDaily.head(5)
```

Out[99]:

	date	visitorsPerDateB
31	2019-08-01	713
32	2019-08-02	581
33	2019-08-03	509
34	2019-08-04	770
35	2019-08-05	707

In [100…
```
## 2. пункт

visitorsACummulative = visitorsADaily.apply(
    lambda x: visitorsADaily[visitorsADaily['date'] <= x['date']].agg(
        {'date': 'max', 'visitorsPerDateA': 'sum'}
    ),
    axis=1,
)
visitorsACummulative.columns = ['date', 'visitorsCummulativeA']

visitorsBCummulative = visitorsBDaily.apply(
    lambda x: visitorsBDaily[visitorsBDaily['date'] <= x['date']].agg(
        {'date': 'max', 'visitorsPerDateB': 'sum'}
```

```
    ),
    axis=1,
)
visitorsBCummulative.columns = ['date', 'visitorsCummulativeB']

visitorsACummulative.head(3)
```

Out[100]:

	date	visitorsCummulativeA
0	2019-08-01	719
1	2019-08-02	1338
2	2019-08-03	1845

Теперь приступаем к таблице orders

In [101…

```
## 1. пункт. Делаем 2 датафрейма, где в каждом данные сгруппированны по дате и примене
## функции.

ordersADaily = (
    orders[orders['group'] == 'A'][['date', 'transactionId', 'visitorId', 'revenue']]
    .groupby('date', as_index=False)
    .agg({'transactionId': pd.Series.nunique, 'revenue': 'sum'})
)
ordersADaily.columns = ['date', 'ordersPerDateA', 'revenuePerDateA']


ordersBDaily = (
    orders[orders['group'] == 'B'][['date', 'transactionId', 'visitorId', 'revenue']]
    .groupby('date', as_index=False)
    .agg({'transactionId': pd.Series.nunique, 'revenue': 'sum'})
)
ordersBDaily.columns = ['date', 'ordersPerDateB', 'revenuePerDateB']


ordersADaily.head()
```

Out[101]:

	date	ordersPerDateA	revenuePerDateA
0	2019-08-01	24	148579
1	2019-08-02	20	93822
2	2019-08-03	24	112473
3	2019-08-04	16	70825
4	2019-08-05	25	124218

In [102…

```
ordersADaily.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 31 entries, 0 to 30
Data columns (total 3 columns):
 #   Column           Non-Null Count  Dtype
---  ------           --------------  -----
 0   date             31 non-null     datetime64[ns]
 1   ordersPerDateA   31 non-null     int64
 2   revenuePerDateA  31 non-null     int64
dtypes: datetime64[ns](1), int64(2)
memory usage: 872.0 bytes
```

In [103…

```
ordersADaily['date'] = pd.to_datetime(ordersADaily['date'])
ordersBDaily['date'] = pd.to_datetime(ordersBDaily['date'])
```

In [104…

```
## 2 пункт - КУМУЛЯТИВНЫЕ ДАННЫЕ-нарастающие данные. Применена функция lambda.

ordersACummulative = ordersADaily.apply(
    lambda x: ordersADaily[ordersADaily['date'] <= x['date']].agg(
        {'date': 'max', 'ordersPerDateA': 'sum', 'revenuePerDateA': 'sum'}
    ),
    axis=1,
).sort_values(by=['date'])

ordersACummulative.columns = [
    'date',
    'ordersCummulativeA',
    'revenueCummulativeA',
]


ordersBCummulative = ordersBDaily.apply(
    lambda x: ordersBDaily[ordersBDaily['date'] <= x['date']].agg(
        {'date': 'max', 'ordersPerDateB': 'sum', 'revenuePerDateB': 'sum'}
    ),
    axis=1,
).sort_values(by=['date'])
ordersBCummulative.columns = [
    'date',
    'ordersCummulativeB',
    'revenueCummulativeB',
]

ordersACummulative.head()
```

Out[104]:

	date	ordersCummulativeA	revenueCummulativeA
0	2019-08-01	24	148579
1	2019-08-02	44	242401
2	2019-08-03	68	354874
3	2019-08-04	84	425699
4	2019-08-05	109	549917

Теперь приступаем к созданию таблицы data. Соединяем с помощью .merge()

In [105…

```
data = (
    ordersADaily.merge(
        ordersBDaily, left_on='date', right_on='date', how='left'
    )
    .merge(ordersACummulative, left_on='date', right_on='date', how='left')
    .merge(ordersBCummulative, left_on='date', right_on='date', how='left')
    .merge(visitorsADaily, left_on='date', right_on='date', how='left')
    .merge(visitorsBDaily, left_on='date', right_on='date', how='left')
    .merge(visitorsACummulative, left_on='date', right_on='date', how='left')
    .merge(visitorsBCummulative, left_on='date', right_on='date', how='left')
)

print(data.head(5))
```

```
        date  ordersPerDateA  revenuePerDateA  ordersPerDateB  \
0 2019-08-01  24              148579           21
1 2019-08-02  20              93822            24
2 2019-08-03  24              112473           16
3 2019-08-04  16              70825            17
4 2019-08-05  25              124218           23

   revenuePerDateB  ordersCummulativeA  revenueCummulativeA  \
0  101217           24                  148579
1  165531           44                  242401
2  114248           68                  354874
3  108571           84                  425699
4  92428            109                 549917

   ordersCummulativeB  revenueCummulativeB  visitorsPerDateA  \
0  21                  101217               719
1  45                  266748               619
2  61                  380996               507
3  78                  489567               717
4  101                 581995               756

   visitorsPerDateB  visitorsCummulativeA  visitorsCummulativeB
0  713               719                   713
1  581               1338                  1294
2  509               1845                  1803
3  770               2562                  2573
4  707               3318                  3280
```

Напомним названия столбцов данных:

date — дата;

ordersPerDateA — количество заказов в выбранную дату в группе A;

revenuePerDateA — суммарная выручка в выбранную дату в группе A;

ordersPerDateB — количество заказов в выбранную дату в группе B;

revenuePerDateB — суммарная выручка в выбранную дату в группе B;

ordersCummulativeA — суммарное число заказов до выбранной даты включительно в группе A;

revenueCummulativeA — суммарная выручка до выбранной даты включительно в группе A;

ordersCummulativeB — суммарное количество заказов до выбранной даты включительно в группе B;

revenueCummulativeB — суммарная выручка до выбранной даты включительно в группе B;

visitorsPerDateA — количество пользователей в выбранную дату в группе A;

visitorsPerDateB — количество пользователей в выбранную дату в группе B;

visitorsCummulativeA — количество пользователей до выбранной даты включительно в группе A;

visitorsCummulativeB — количество пользователей до выбранной даты включительно в группе B.

Посчитаем статистическую значимость различия в среднем количестве заказов между группами.

In [106…

```
## 1. Создадим переменные ordersByUsersA и ordersByUsersB со столбцами ['userId', 'orc
## В них для пользователей, которые заказывали хотя бы 1 раз, укажем число совершённых

ordersByUsersA = (
    orders[orders['group'] == 'A']
    .groupby('visitorId', as_index=False)
    .agg({'transactionId': pd.Series.nunique})
)

ordersByUsersA.columns = ['userId', 'orders']

ordersByUsersB = (
    orders[orders['group'] == 'B']
    .groupby('visitorId', as_index=False)
    .agg({'transactionId': pd.Series.nunique})
)

ordersByUsersB.columns = ['userId', 'orders']

ordersByUsersA.head(5)
```

Out[106]:

	userId	orders
0	8300375	1
1	11685486	1
2	54447517	1
3	66685450	1
4	78758296	1

2. Объявим переменные sampleA и sampleB, в которых пользователям из разных групп будет

соответствовать количество заказов.

Тем, кто ничего не заказал, будут соответствовать нули. Это нужно, чтобы подготовить выборки к проверке критерием Манна-Уитни.

Переменная sampleA должна состоять из двух частей:

1. Список с количеством заказов для каждого из пользователей: ordersByUsersA['orders'].
2. Нули для пользователей, которые ничего не заказывали. Их количество равно разнице между суммой посетителей и количеством записей о заказах:

data['visitorsPerDateA'].sum() - len(ordersByUsersA['orders']). Создадим объект pd.Series нужной длины:

In [107…
```
##  Список индексов создали функцией np.arange().
## Она работает так же, как функция range(), только создаёт массив индексов в формате

pd.Series(0, index=np.arange(data['visitorsPerDateA'].sum() - len(ordersByUsersA['orde
```

Out[107]:
```
0        0
1        0
2        0
3        0
4        0
        ..
18228    0
18229    0
18230    0
18231    0
18232    0
Name: orders, Length: 18233, dtype: int64
```

In [108…
```
## В Pandas последовательности объединяют функцией pd.concat() (от англ. concatenate,
## Сперва ей передают то, что объединяют — в нашем случае список из первой и второй чс
## Дальше передадим аргумент, сообщающий, что объекты Series нужно объединить по строк
## (то есть записать подряд): pd.concat([...], axis=0). То же делаем для sampleB и пол

sampleA = pd.concat([ordersByUsersA['orders'],pd.Series(0, index=np.arange(data['visit

sampleB = pd.concat([ordersByUsersB['orders'],pd.Series(0, index=np.arange(data['visit

sampleA.head(10)
```

Out[108]:
```
0    1
1    1
2    1
3    1
4    1
5    1
6    1
7    1
8    1
9    1
Name: orders, dtype: int64
```

Применим критерий и отформатируем p-value, округлив его до трёх знаков после запятой.

Напомним, что в sampleA сохранили выборку, где каждый элемент — число заказов определённого пользователя, в том числе ноль. Значит, число элементов sampleA — это количество пользователей, сумма всех элементов — количество заказов. Чтобы получить среднее число заказов на пользователя, поделим сумму заказов на число пользователей — найдём среднее в выборке sampleA методом mean(). Аналогично найдём среднее группы B: SampleB.mean().

Выведем относительный прирост среднего числа заказов группы B: среднее число заказов группы B / среднее число заказов группы A - 1. Округлим до трёх знаков после запятой.

Нулевая гипотеза: статистически значимых различий в среднем числе заказов между группами нет. Альтернативная гипотеза: различия в среднем числе заказов между группами есть.

In [109…

```
## p-value- статистическая значимость:
print("{0:.3f}".format(stats.mannwhitneyu(sampleA, sampleB)[1]))

##  относительный прирост среднего числа заказов группы B
print("{0:.3f}".format(sampleB.mean() / sampleA.mean() - 1))
```

```
0.017
0.138
```

Вывод: По «сырым» данным РАЗЛИЧИЯ в среднем ЧИСЛЕ ЗАКАЗОВ групп A и B есть.

Первое число — p-value = 0.017 < 0.05. Значит, нулевую гипотезу о том, что статистически значимых различий в среднем числе заказов между группами нет - отвергаем. Однако относительный выигрыш группы B равен 13.8% — второе число в выводе. Сегмент B значительно лучше сегмента A

11. Посчитайте статистическую значимость различий в среднем чеке заказа между группами по «сырым» данным.

Сделайте выводы и предположения.

Нулевая гипотеза: различий в среднем чеке между группами нет. Альтернативная гипотеза: различия в среднем чеке между группами есть.

Чтобы рассчитать статистическую значимость различий в среднем чеке, передадим критерию mannwhitneyu() данные о выручке с заказов. А ещё найдём относительные различия в среднем чеке между группами:

In [110…
```
## статистическая значимость различий в среднем чеке - P-value
print('{0:.3f}'.format(stats.mannwhitneyu(orders[orders['group']=='A']['revenue'], orc

## относительные различия в среднем чеке между группами
print('{0:.3f}'.format(orders[orders['group']=='B']['revenue'].mean()/orders[orders['g
```

```
0.729
0.259
```

ВЫВОД: P-value(0.729) > 0.05. - различий в среднем чеке между группами нет.

Значит, нулевая гипотеза - не отвергается. Кроме того,относительное различие среднего чека между сегментами В к А всего 25,9 %

Вспомним, что найденные в прошлых уроках 95-й и 99-й перцентили средних чеков равны 28000 и 58233 рублям. А 95-й и 99-й перцентили количества заказов на одного пользователя равны 2 и 4 заказа на пользователя.

Примем за аномальных пользователей тех, кто совершил от 2 заказов или совершил заказ дороже 28000 000 рублей. Так мы уберём 5% пользователей с наибольшим числом заказов и от 1% до 5% пользователей с дорогими заказами.

Сделаем срезы пользователей с числом заказов больше 2 — usersWithManyOrders и пользователей, совершивших заказы дороже 28 000 — usersWithExpensiveOrders.

Объединим их в таблице abnormalUsers. Узнаем, сколько всего аномальных пользователей атрибутом shape.

In [111…
```
usersWithManyOrders = pd.concat(
    [
        ordersByUsersA[ordersByUsersA['orders'] > 2]['userId'],
        ordersByUsersB[ordersByUsersB['orders'] > 2]['userId'],
    ],
    axis=0,
)

usersWithExpensiveOrders = orders[orders['revenue'] > 28000]['visitorId']

abnormalUsers = (
    pd.concat([usersWithManyOrders, usersWithExpensiveOrders], axis=0)
    .drop_duplicates()
    .sort_values()
)

print(abnormalUsers.head(5))
print(abnormalUsers.shape[0])
```

```
1099    148427295
18      199603092
928     204675465
23      237748145
37      249864742
dtype: int64
74
```

ВЫВОД: Всего 74 аномальных пользователей.

12. Посчитайте статистическую значимость различий в среднем количестве заказов на посетителя между группами по «очищенным» данным.

Нулевая гипотеза

Узнаем, как их действия повлияли на результаты теста. Посчитаем статистическую значимость различий в среднем количестве заказов между группами теста по очищенным данным. Сначала подготовим выборки количества заказов по пользователям по группам теста:

In [112…
```
sampleAFiltered = pd.concat(
    [
        ordersByUsersA[
            np.logical_not(ordersByUsersA['userId'].isin(abnormalUsers))
        ]['orders'],
        pd.Series(
            0,
            index=np.arange(
                data['visitorsPerDateA'].sum() - len(ordersByUsersA['orders'])
            ),
            name='orders',
        ),
    ],
    axis=0,
)

sampleBFiltered = pd.concat(
    [
        ordersByUsersB[
            np.logical_not(ordersByUsersB['userId'].isin(abnormalUsers))
        ]['orders'],
        pd.Series(
            0,
            index=np.arange(
                data['visitorsPerDateB'].sum() - len(ordersByUsersB['orders'])
            ),
            name='orders',
        ),
    ],
    axis=0,
)
```

Нулевая гипотеза: статистически значимых различий в среднем числе заказов между группами нет. Альтернативная гипотеза: различия в среднем числе заказов между группами есть.

In [113…
```
## Применим статистический критерий Манна-Уитни к полученным выборкам:
print('{0:.3f}'.format(stats.mannwhitneyu(sampleAFiltered, sampleBFiltered)[1]))
```

```
print('{0:.3f}'.format(sampleBFiltered.mean()/sampleAFiltered.mean()-1))
```

```
0.013
0.173
```

ВЫВОД: Нулевая гипотеза отвергается - различия в среднем числе заказов между группами есть.

Результаты по среднему количеству заказов изменились. Было:0.017 0.138

13. Посчитайте статистическую значимость различий в среднем чеке заказа между группами по «очищенным» данным.

Нулевая гипотеза: различий в среднем чеке между группами нет. Альтернативная гипотеза: различия в среднем чеке между группами есть.

In [114…

```
print(
    '{0:.3f}'.format(
        stats.mannwhitneyu(
            orders[
                np.logical_and(
                    orders['group'] == 'A',
                    np.logical_not(orders['visitorId'].isin(abnormalUsers)),
                )
            ]['revenue'],
            orders[
                np.logical_and(
                    orders['group'] == 'B',
                    np.logical_not(orders['visitorId'].isin(abnormalUsers)),
                )
            ]['revenue'],
        )[1]
    )
)

print(
    "{0:.3f}".format(
        orders[
            np.logical_and(
                orders['group'] == 'B',
                np.logical_not(orders['visitorId'].isin(abnormalUsers)),
            )
        ]['revenue'].mean()
        / orders[
            np.logical_and(
                orders['group'] == 'A',
                np.logical_not(orders['visitorId'].isin(abnormalUsers)),
            )
        ]['revenue'].mean()
        - 1
    )
)
```

```
0.738
-0.020
```

P-value немного увеличился, на 0.9 пункта.

ВЫВОД: P-value(0.738) > 0.05. Различий в среднем чеке между группами нет.

Разница между сегментами увеличилась с 25.9 % до - 2 %. Было много вбросов в группе А Общие выводы по результатам теста изменились, такой пример хорошо показывает, как сильно аномалии могут влиять на результаты A/B-теста!

Было: 0.729 и 0.259

14. Примите решение по результатам теста и объясните его. Варианты решений: 1. Остановить тест, зафиксировать победу одной из групп. 2. Остановить тест, зафиксировать отсутствие различий между группами. 3. Продолжить тест.

1. Посчитайте статистическую значимость различий в среднем КОЛИЧЕСТВЕ заказов на посетителя между группами по «сырым» данным.

Нулевая гипотеза: статистически значимых различий в среднем числе заказов между группами нет. Альтернативная гипотеза: различия в среднем числе заказов между группами есть. 0.017 < 0.05 –нулевая гипотеза отвергается

0.017 0.138

Вывод: По «сырым» данным РАЗЛИЧИЯ в среднем ЧИСЛЕ ЗАКАЗОВ групп А и В есть.

1. Посчитайте статистическую значимость различий в среднем чеке заказа между группами по «сырым» данным.

Нулевая гипотеза: различий в среднем чеке между группами нет. Альтернативная гипотеза: различия в среднем чеке между группами есть.

0.729 0.259

ВЫВОД: P-value(0.729) > 0.05. Различий в среднем чеке между группами нет.

Значит, нулевая гипотеза - не отвергается. Кроме того,относительное различие среднего чека между сегментами В к А всего 25,9 %

1. Посчитайте статистическую значимость различий в среднем количестве заказов на посетителя между группами по «очищенным» данным. Нулевая гипотеза: статистически значимых различий в среднем числе заказов между группами нет. Альтернативная гипотеза: различия в среднем числе заказов между группами есть.Узнаем, как их действия повлияли на результаты теста.

0.013 0.173

ВЫВОД: P-value(0.013) < 0.05. Различия в среднем чеке между группами есть.

Было: 0.017 0.138

1. Посчитайте статистическую значимость различий в среднем чеке заказа между группами по «очищенным» данным.

Нулевая гипотеза: различий в среднем чеке между группами нет. Альтернативная гипотеза: различия в среднем чеке между группами есть.

0.738 -0.020

ВЫВОД: P-value (0.738) > 0.05. Различий в среднем чеке между группами нет.

Какие выводы по тесту можем сделать?

Имеющиеся факты:

1. По «сырым» и по "очищенным" данным - статистически значимых различий в среднем ЧИСЛЕ заказов между группами A и B - ЕСТЬ.
2. По «сырым» и по "очищенным" данным - статистически значимых различий в среднем ЧЕКЕ заказа между группами A и B - НЕТ.

График различия среднего количества заказов между группами сообщает, что результаты группы B лучше группы A и нет значительной тенденции к улучшению:

In [115…

```
## График различия среднего количества заказов между группами сообщает, что результаты

mergedCumulativeConversions = cumulativeDataA[['date','conversion']].merge(cumulativeD

plt.plot(mergedCumulativeConversions['date'], mergedCumulativeConversions['conversionB

plt.legend()

plt.axhline(y=0, color='black', linestyle='--')
plt.axhline(y=-0.1, color='grey', linestyle='--')
```

Out[115]:

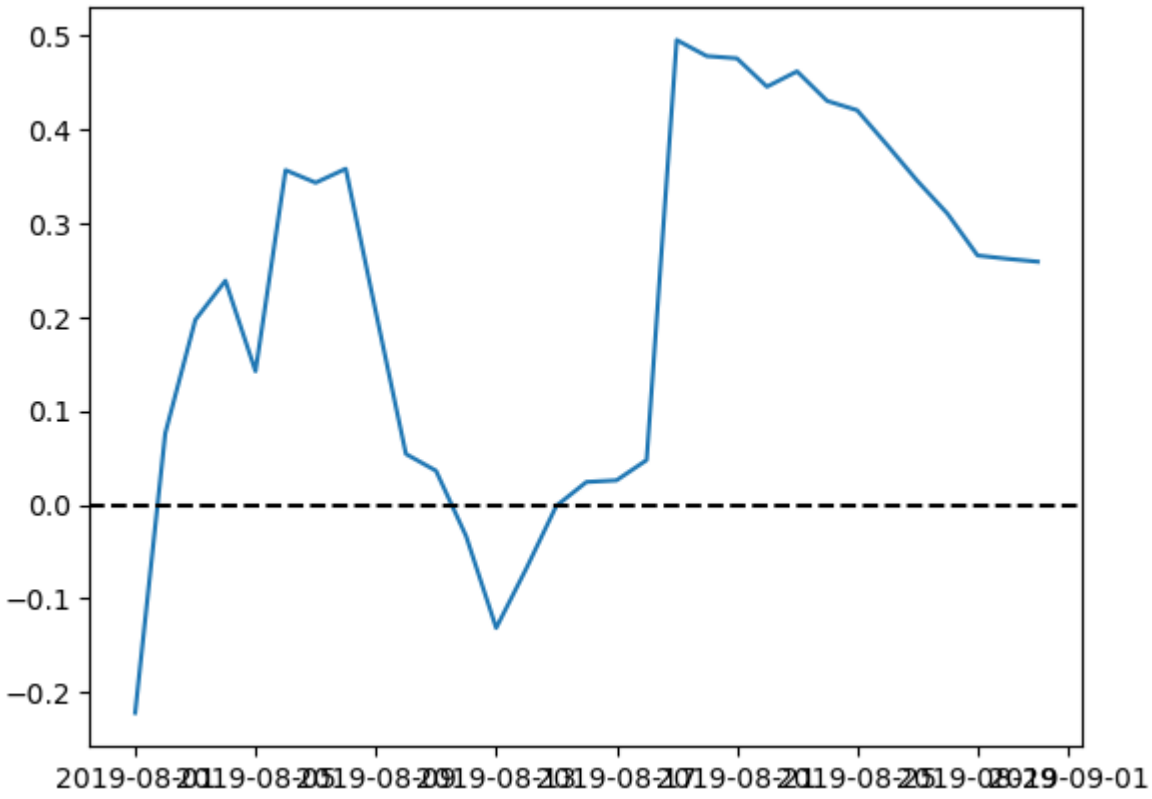
```
<matplotlib.lines.Line2D at 0x235a26ad3d0>
```

In [116…

```
## График различия среднего чека говорит о том, что результаты группы B
## улудшаются день ото дня:

mergedCumulativeRevenue = cumulativeRevenueA.merge(cumulativeRevenueB, left_on='date',

# строим отношение средних чеков
plt.plot(mergedCumulativeRevenue['date'], (mergedCumulativeRevenue['revenue_cumB']/mer

# добавляем ось X
plt.axhline(y=0, color='black', linestyle='--')
```

Out[116]:

```
<matplotlib.lines.Line2D at 0x235a1526190>
```



Исходя из обнаруженных фактов, тест следует ОСТАНОВИТЬ и признать его успешным и перейти к проверке следующей гипотезы

Общий вывод ревьюера

Проделана хорошая работа. Есть несколько замечаний, исправь их и проект будет принят. - по ссылкам ниже интересные материалы по A/B тестам https://habr.com/ru/company/avito/blog/571094/ https://habr.com/ru/company/yandex/blog/476826/ https://academy.yandex.ru/journal/kak-provesti-a-b-testirovanie-6-prostykh-shagov https://habr.com/ru/company/yandex/blog/342704/ https://r-analytics.blogspot.com/2013/10/blog-post_13.html

- добавь, пожалуйста, проверку на наличие дубликатов и пропусков; - добавь, пожалуйста, заголовки и подписи осей для всех графиков в проекте; - скорректируй, пожалуйста, определение границы аномальной стоимости заказов; - определи, пожалуйста, загрузку всех датасетов проекта только один раз в начале проекта; - сформулируй, пожалуйста, гипотезы; - скорректируй, пожалуйста, промежуточный вывод к анализу

статистической значимости различий в среднем чеке заказа между группами по "сырым" данным; - скорректируй, пожалуйста, общий вывод.