Описание проекта

Контекст

Я — аналитик крупного интернет-магазина. Вместе с отделом маркетинга мы подготовили список гипотез для увеличения выручки. Приоритизируем гипотезы, запустим A/B-тест и проанализируем результаты.

Часть 1. Приоритизация гипотез.

В файле /datasets/hypothesis.csv 9 гипотез по увеличению выручки интернет-магазина с указанными параметрами Reach, Impact, Confidence, Effort.
План работы:

- Применим фреймворк ICE для приоритизации гипотез. Отсортируем их по убыванию приоритета.
- Применим фреймворк RICE для приоритизации гипотез. Отсортируем их по убыванию приоритета.
- Укажем, как изменилась приоритизация гипотез при применении RICE вместо ICE. Поймем, почему так произошло.

In [171…
```
import datetime
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import scipy.stats as stats
import datetime as dt
```

In [172…
```
try:
    hyp = pd.read_csv('/datasets/hypothesis.csv')
except:
    hyp = pd.read_csv('hypothesis.csv')
```

In [173…
```
hyp.head()
```

Out[173]:

	Hypothesis	Reach	Impact	Confidence	Efforts
0	Добавить два новых канала привлечения трафика, что позволит привлекать на 30% больше пользователей	3	10	8	6
1	Запустить собственную службу доставки, что сократит срок доставки заказов	2	5	4	10
2	Добавить блоки рекомендаций товаров на сайт интернет магазина, чтобы повысить конверсию и средний чек заказа	8	3	7	3
3	Изменить структура категорий, что увеличит конверсию, т.к. пользователи быстрее найдут нужный товар	8	3	3	8
4	Изменить цвет фона главной страницы, чтобы увеличить вовлеченность пользователей	3	1	1	1

In [174…
```
hyp.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 [175…
```
hyp['ICE_score'] = round(hyp['Impact'] * hyp['Confidence'] / hyp['Efforts
hyp
```

Out[175]:

	Hypothesis	Reach	Impact	Confidence	Efforts	ICE_score
0	Добавить два новых канала привлечения трафика, что позволит привлекать на 30% больше пользователей	3	10	8	6	13.333
1	Запустить собственную службу доставки, что сократит срок доставки заказов	2	5	4	10	2.000
2	Добавить блоки рекомендаций товаров на сайт интернет магазина, чтобы повысить конверсию и средний чек заказа	8	3	7	3	7.000
3	Изменить структура категорий, что увеличит конверсию, т.к. пользователи быстрее найдут нужный товар	8	3	3	8	1.125
4	Изменить цвет фона главной страницы, чтобы увеличить вовлеченность пользователей	3	1	1	1	1.000
5	Добавить страницу отзывов клиентов о магазине, что позволит увеличить количество заказов	3	2	2	3	1.333
6	Показать на главной странице баннеры с актуальными акциями и распродажами, чтобы увеличить конверсию	5	3	8	3	8.000
7	Добавить форму подписки на все основные страницы, чтобы собрать базу клиентов для email-рассылок	10	7	8	5	11.200
8	Запустить акцию, дающую скидку на товар в день рождения	1	9	9	5	16.200

In [176…

```
hyp['RICE_score'] = round(hyp['Reach'] * hyp['Impact'] * hyp['Confidence'
hyp
```

Out[176]:

	Hypothesis	Reach	Impact	Confidence	Efforts	ICE_score	RICE_score
0	Добавить два новых канала привлечения трафика, что позволит привлекать на 30% больше пользователей	3	10	8	6	13.333	40.0
1	Запустить собственную службу доставки, что сократит срок доставки заказов	2	5	4	10	2.000	4.0

2	Добавить блоки рекомендаций товаров на сайт интернет магазина, чтобы повысить конверсию и средний чек заказа	8	3	7	3	7.000	56.0
3	Изменить структура категорий, что увеличит конверсию, т.к. пользователи быстрее найдут нужный товар	8	3	3	8	1.125	9.0
4	Изменить цвет фона главной страницы, чтобы увеличить вовлеченность пользователей	3	1	1	1	1.000	3.0
5	Добавить страницу отзывов клиентов о магазине, что позволит увеличить количество заказов	3	2	2	3	1.333	4.0
6	Показать на главной странице баннеры с актуальными акциями и распродажами, чтобы увеличить конверсию	5	3	8	3	8.000	40.0
7	Добавить форму подписки на все основные страницы, чтобы собрать базу клиентов для email-рассылок	10	7	8	5	11.200	112.0
8	Запустить акцию, дающую скидку на товар в день рождения	1	9	9	5	16.200	16.2

In [177…
```
hyp.sort_values(by='ICE_score', ascending=False)
```

Out[177]:

	Hypothesis	Reach	Impact	Confidence	Efforts	ICE_score	RICE_score
8	Запустить акцию, дающую скидку на товар в день рождения	1	9	9	5	16.200	16.2
	Добавить два новых канала привлечения трафика, что						

0	позволит привлекать на 30% больше пользователей	3	10	8	6	13.333	40.0
7	Добавить форму подписки на все основные страницы, чтобы собрать базу клиентов для email-рассылок	10	7	8	5	11.200	112.0
6	Показать на главной странице баннеры с актуальными акциями и распродажами, чтобы увеличить конверсию	5	3	8	3	8.000	40.0
2	Добавить блоки рекомендаций товаров на сайт интернет магазина, чтобы повысить конверсию и средний чек заказа	8	3	7	3	7.000	56.0
1	Запустить собственную службу доставки, что сократит срок доставки заказов	2	5	4	10	2.000	4.0
5	Добавить страницу отзывов клиентов о магазине, что позволит увеличить количество заказов	3	2	2	3	1.333	4.0
3	Изменить структура категорий, что увеличит конверсию, т.к. пользователи быстрее найдут нужный товар	8	3	3	8	1.125	9.0
4	Изменить цвет фона главной страницы, чтобы увеличить вовлеченность пользователей	3	1	1	1	1.000	3.0

In [178…
```
pd.options.display.max_colwidth = 120
hyp.sort_values(by='RICE_score', ascending=False)
```

Out[178]:

	Hypothesis	Reach	Impact	Confidence	Efforts	ICE_score	RICE_score
	Добавить форму подписки на все						

7	основные страницы, чтобы собрать базу клиентов для email-рассылок	10	7	8	5	11.200	112.0
2	Добавить блоки рекомендаций товаров на сайт интернет магазина, чтобы повысить конверсию и средний чек заказа	8	3	7	3	7.000	56.0
0	Добавить два новых канала привлечения трафика, что позволит привлекать на 30% больше пользователей	3	10	8	6	13.333	40.0
6	Показать на главной странице баннеры с актуальными акциями и распродажами, чтобы увеличить конверсию	5	3	8	3	8.000	40.0
8	Запустить акцию, дающую скидку на товар в день рождения	1	9	9	5	16.200	16.2
3	Изменить структура категорий, что увеличит конверсию, т.к. пользователи быстрее найдут нужный товар	8	3	3	8	1.125	9.0
1	Запустить собственную службу доставки, что сократит срок доставки заказов	2	5	4	10	2.000	4.0
5	Добавить страницу отзывов клиентов о магазине, что позволит увеличить количество заказов	3	2	2	3	1.333	4.0
4	Изменить цвет фона главной страницы, чтобы увеличить вовлеченность пользователей	3	1	1	1	1.000	3.0

Заметим, что приоретизация гипотез меняется в зависимости от того, учитываем ли мы параметр Reach при определении приоритетов. Например, гипотеза 7 влияет на большинство клиентов. Гипотеза 2 также имеет большой охват.

Часть 2. Анализ A/B-теста

Проведем A/B-тест и получим результаты, которые описаны в файлах /datasets/orders.csv и /datasets/visitors.csv

План работы: Проанализируйте A/B-тест:

- [x] 1) Построим график кумулятивной выручки по группам. Сделаем выводы и предположения.
- [x] 2) Построим график кумулятивного среднего чека по группам. Сделаем выводы и предположения.
- [x] 3) Построим график относительного изменения кумулятивного среднего чека группы B к группе A. Сделаем выводы и предположения.
- [x] 4) Построим график кумулятивного среднего количества заказов на посетителя по группам. Сделаем выводы и предположения.
- [x] 5) Построим график относительного изменения кумулятивного среднего количества заказов на посетителя группы B к группе A. Сделаем выводы и предположения.
- [x] 6) Построим точечный график количества заказов по пользователям. Сделаем выводы и предположения.
- [x] 7) Посчитаем 95-й и 99-й перцентили количества заказов на пользователя. Выберем границу для определения аномальных пользователей.
- [x] 8) Построим точечный график стоимостей заказов. Сделаем выводы и предположения.
- [x] 9) Построим 95-й и 99-й перцентили стоимости заказов. Выберем границу для определения аномальных заказов.
- [x] 10) Построим статистическую значимость различий в среднем количестве заказов на посетителя между группами по «сырым» данным. Сделаем выводы и предположения.
- [x] 11) Посчитаем статистическую значимость различий в среднем чеке заказа между группами по «сырым» данным. Сделаем выводы и предположения.
- [x] 12) Посчитаем статистическую значимость различий в среднем количестве заказов на посетителя между группами по «очищенным» данным. Сделаем выводы и предположения.
- [x] 13) Посчитаем статистическую значимость различий в среднем чеке заказа между группами по «очищенным» данным. Сделаем выводы и предположения.
- [x] 14) Примем решение по результатам теста и объясним его. Варианты решений: 1. Остановить тест, зафиксировать победу одной из групп. 2. Остановить тест, зафиксировать отсутствие различий между группами. 3. Продолжить тест.

In [179…
```
orders = pd.read_csv('/datasets/orders.csv', sep=',')
orders['date'] = orders['date'].map(
    lambda x: dt.datetime.strptime(x, '%Y-%m-%d')
)

visitors = pd.read_csv(
    '/datasets/visitors.csv', sep=','
)
visitors['date'] = visitors['date'].map(
    lambda x: dt.datetime.strptime(x, '%Y-%m-%d')
)

display(orders.head())
display(visitors.head())
```

	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
3	3797467345	1196621759	2019-08-15	9759	B
4	2282983706	2322279887	2019-08-15	2308	B

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

In [180…
```
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   datetime64[ns]
 3   revenue        1197 non-null   int64         
 4   group          1197 non-null   object        
dtypes: datetime64[ns](1), int64(3), object(1)
memory usage: 46.9+ KB
```

In [181…
```
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     datetime64[ns]
 1   group     62 non-null     object
 2   visitors  62 non-null     int64
dtypes: datetime64[ns](1), int64(1), object(1)
memory usage: 1.6+ KB
```

In [182…
```
orders.duplicated().sum()
```

Out[182]: 0

In [183…
```
visitors.duplicated().sum()
```

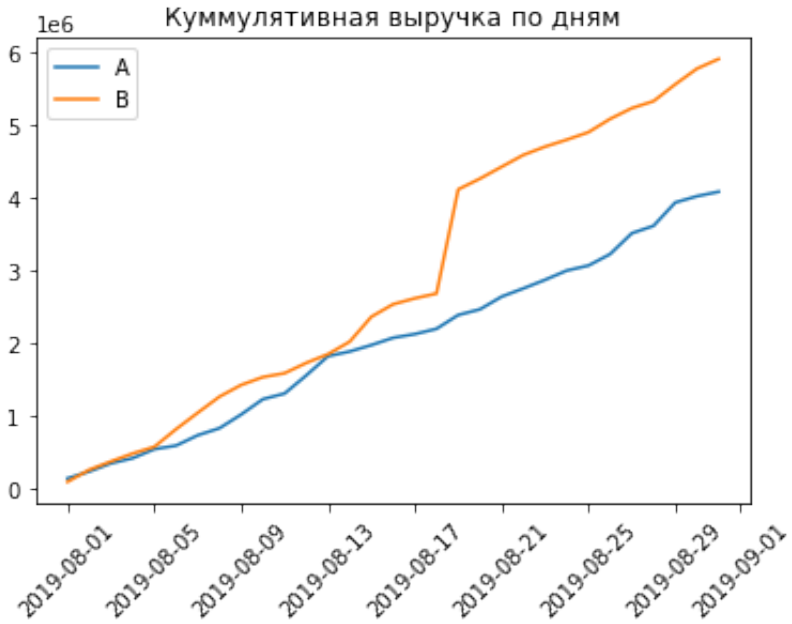
Out[183]: 0

Типы данных соответствуют описанию данных

In [184…
```
# создаем массив уникальных пар значений дат и групп теста
datesGroups = orders[['date','group']].drop_duplicates()

# получаем агрегированные кумулятивные по дням данные о заказах
ordersAggregated = datesGroups.apply(lambda x: orders[np.logical_and(orde

# получаем агрегированные кумулятивные по дням данные о посетителях интер
visitorsAggregated = datesGroups.apply(lambda x: visitors[np.logical_and(

# объединяем кумулятивные данные в одной таблице и присваиваем ее столбца
cumulativeData = ordersAggregated.merge(visitorsAggregated, left_on=['dat
cumulativeData.columns = ['date', 'group', 'orders', 'buyers', 'revenue',

display(cumulativeData.head(5))
```

	date	group	orders	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 [185…
```
# датафрейм с кумулятивным количеством заказов и кумулятивной выручкой по
cumulativeRevenueA = cumulativeData[cumulativeData['group']=='A'][['date'

# датафрейм с кумулятивным количеством заказов и кумулятивной выручкой по
cumulativeRevenueB = cumulativeData[cumulativeData['group']=='B'][['date'

# Строим график выручки группы A
plt.plot(cumulativeRevenueA['date'], cumulativeRevenueA['revenue'], label
plt.xticks(rotation=45)

# Строим график выручки группы в
plt.plot(cumulativeRevenueB['date'], cumulativeRevenueB['revenue'], label

plt.title('Куммулятивная выручка по дням')
plt.legend();
```



Выручка почти равномерно увеличивается в течение всего теста. Однако график выручки группы B испытывает скачкообразный рост в середине периода. Это может сигнализировать о всплесках числа заказов, либо о появлении очень дорогих заказов в выборке.

In [186…
```
plt.plot(cumulativeRevenueA['date'], cumulativeRevenueA['revenue']/cumula
plt.plot(cumulativeRevenueB['date'], cumulativeRevenueB['revenue']/cumula
plt.title('Куммулятивный средний чек по дням')
plt.xticks(rotation=45)
plt.legend();
```

Средний чек для каждой из групп представляет собой ломаную линию, ближе к концу периода средний чек изменяется равномерно. Средний чек для группы B скачкообразно растет в середине периода, затем равномерно снижается.

In [187…

```
# собираем данные в одном датафрейме
mergedCumulativeRevenue = cumulativeRevenueA.merge(cumulativeRevenueB, le

# строим отношение средних чеков
plt.plot(mergedCumulativeRevenue['date'], (mergedCumulativeRevenue['reven

# добавляем ось X
plt.xticks(rotation=45)
plt.title('Отношение средних чеков по дням')
plt.axhline(y=0, color='black', linestyle='--');
```

В нескольких точках график различия между сегментами резко «скачет». Это может свидетельствовать о крупных заказах и выбросах.

Аналогично проанализируем график кумулятивной конверсии.

In [188…
```
orders['date'].min()
```

Out[188]:
```
Timestamp('2019-08-01 00:00:00')
```

In [189…
```
orders['date'].max()
```

Out[189]:
```
Timestamp('2019-08-31 00:00:00')
```

In [190…
```
# считаем куммулятивное среднее число заказов
cumulativeData['conversion'] = cumulativeData['orders']/cumulativeData['v

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

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

# строим графики
plt.plot(cumulativeDataA['date'], cumulativeDataA['conversion'], label='A
plt.plot(cumulativeDataB['date'], cumulativeDataB['conversion'], label='B
plt.xticks(rotation=45)
plt.title('Куммулятивное среднее число заказов по дням')
plt.legend();

# задаем масштаб осей
#plt.axis([datetime.datetime(2019, 4, 1), datetime.datetime(2019, 4, 23),
```

Куммулятивное средниее число заказов сильно колеблется в начале и стабилизируется к концу. Среднее число заказов группы В заметно выше.

In [191…
```
mergedCumulativeData = cumulativeDataA.merge(cumulativeDataB, left_on='da

# строим отношение средних чеков
plt.plot(mergedCumulativeData['date'], (mergedCumulativeData['buyersB']/m

# добавляем ось X
plt.xticks(rotation=45)
plt.title('Отношение куммулятивных средних чисел заказов по дням')
plt.axhline(y=0, color='black', linestyle='--');
```

Заметим, что график сильно колеблется в левой части, затем после резкого скачка вверх убываетю

In [192…
```
mergedCumulativeData.head()
```

Out[192]:

	date	groupA	ordersA	buyersA	revenueA	visitorsA	conversionA	groupB	ordersB
0	2019-08-01	A	24	20	148579	719	0.033380	B	2
1	2019-08-02	A	44	38	242401	1338	0.032885	B	45
2	2019-08-03	A	68	62	354874	1845	0.036856	B	6
3	2019-08-04	A	84	77	425699	2562	0.032787	B	78
4	2019-08-05	A	109	100	549917	3318	0.032851	B	10

Подсчитаем количество заказов по пользователям

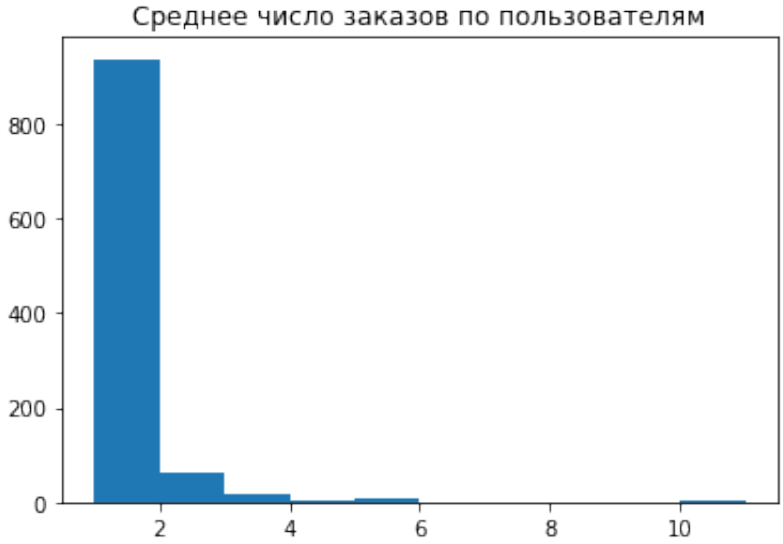
In [193…

```
try:
    data = pd.read_csv('/datasets/orders.csv', sep=',')
except:
    data = pd.read_csv('orders.csv', sep=',')
data['date'] = data['date'].map(lambda x: dt.datetime.strptime(x, '%Y-%m-

ordersByUsers = (
    data.groupby('visitorId', as_index=False)
    .agg({'transactionId': 'nunique'})
)

ordersByUsers.columns = ['visitorId', 'orders']

print(ordersByUsers.sort_values(by='orders', ascending=False).head(10))
```

```
      visitorId  orders
1023  4256040402      11
591   2458001652      11
569   2378935119       9
487   2038680547       8
44     199603092       5
744   3062433592       5
55     237748145       5
917   3803269165       5
299   1230306981       5
897   3717692402       5
```

In [194…

```
plt.hist(ordersByUsers['orders'])
plt.title('Среднее число заказов по пользователям');
```



Построим точечную диаграмму числа заказов на одного пользователя:

In [195…
```
x_values = pd.Series(range(0,len(ordersByUsers)))

plt.scatter(x_values, ordersByUsers['orders'])
plt.title('Точечная диаграмма числа заказов на пользователя');
```

In [196…
```
abnormal_count = ordersByUsers['orders'].quantile(0.99)
print('99 перцентиль числа заказов', round(ordersByUsers['orders'].quanti
print('95 перцентиль числа заказов', round(ordersByUsers['orders'].quanti
```

```
99 перцентиль числа заказов 4.0
95 перцентиль числа заказов 2.0
```

Будем считать пользователя аномальным, если число его заказов больше или равно 99 перцентиля

Оценим стоимость заказов ещё одним способом:

In [197…
```
x_values = pd.Series(range(0,len(data['revenue'])))
plt.scatter(x_values, data['revenue'])
plt.title('График разброса стоимости заказов');
```

In [198…
```
plt.scatter(x_values, data['revenue'])
plt.ylim(0, 120000)
plt.title('График разброса стоимости заказов (с ограничением по y)');
```

Всё, как предполагали: есть немного дорогих заказов. По графику можно выбрать границу аномальных заказов между 20 000 и 40 000 рублей. Однако принято отсеивать от 1% до 5% наблюдений с крайними значениями.

In [199…
```
abnormal_revenue = round(data['revenue'].quantile(0.99), 3)
print('99 перцентиль стоимости заказов', round(data['revenue'].quantile(0
print('95 перцентиль стоимости заказов', round(data['revenue'].quantile(0
```

```
99 перцентиль стоимости заказов 58233.2
95 перцентиль стоимости заказов 28000.0
```

In [200…
```
orders = pd.read_csv('/datasets/orders.csv', sep=',')
```

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

visitors = pd.read_csv(
    '/datasets/visitors.csv', sep=','
)
visitors['date'] = visitors['date'].map(
    lambda x: dt.datetime.strptime(x, '%Y-%m-%d')
)

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

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

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

visitorsBCummulative = visitorsBDaily.apply(
    lambda x: visitorsBDaily[visitorsBDaily['date'] <= x['date']].agg(
        {'date': 'max', 'visitorsPerDateB': 'sum'}
    ),
    axis=1,
)
visitorsBCummulative.columns = ['date', 'visitorsCummulativeB']

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

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',
]

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

```
ordersBDaily.columns = ['date', 'ordersPerDateB', 'revenuePerDateB']

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',
]

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='le
    .merge(visitorsBCummulative, left_on='date', right_on='date', how='le
)

data.head(5)
```

Out[200]:

	date	ordersPerDateA	revenuePerDateA	ordersPerDateB	revenuePerDateB	ordersC
0	2019-08-01	24	148579	21	101217	
1	2019-08-02	20	93822	24	165531	
2	2019-08-03	24	112473	16	114248	
3	2019-08-04	16	70825	17	108571	
4	2019-08-05	25	124218	23	92428	

H0: Среднее число заказов в группах A и B совпадает

H1: Среднее число заказов в группах A и B не совпадает

In [201…

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

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

sampleA = pd.concat(
    [
        ordersByUsersA['orders'],
        pd.Series(
            0,
            index=np.arange(
                data['visitorsPerDateA'].sum() - len(ordersByUsersA['orde
            ),
            name='orders',
        ),
    ],
    axis=0,
)

sampleB = pd.concat(
    [
        ordersByUsersB['orders'],
        pd.Series(
            0,
            index=np.arange(
                data['visitorsPerDateB'].sum() - len(ordersByUsersB['orde
            ),
            name='orders',
        ),
    ],
    axis=0,
)

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

print("difference% = {0:.3f}".format(sampleB.mean() / sampleA.mean() - 1)
```

```
p-value = 0.017
difference% = 0.138
```

Вывод: По "сырым" данным получаем статистически значимое различие между группами A и B (p-value = 0.017 < 0.05). Значит, отвергаем нулевую гипотезу о равенстве среднего числа заказов. Относительный проигрыш группы B составляет 13.8%

Теперь проверим статистическую значимость различий в среднем чеке между сегментами. Нулевая гипотеза: различий в среднем чеке между группами нет. Альтернативная гипотеза: различия в среднем чеке между группами есть.

H0: Средний чек в группах A и B совпадает
H1: Средний чек в группах A и B не совпадает

In [202…
```
print('p-value = {0:.3f}'.format(stats.mannwhitneyu(orders[orders['group'
print('difference% = {0:.3f}'.format(orders[orders['group']=='B']['revenu
```

```
p-value = 0.729
difference% = 0.259
```

P-value значительно больше 0.05. Значит, причин отвергать нулевую гипотезу и считать, что в среднем чеке есть различия, нет. Впрочем, средний чек группы B значительно ниже среднего чека группы A.

Примем за аномальных пользователей тех, кто совершил от 4 заказов или совершил заказ дороже 58233.2 рублей. Так мы уберём 1% пользователей с наибольшим числом заказов и от 1% до 5% пользователей с дорогими заказами. Сделаем срезы пользователей с числом заказов больше или равно 4 — usersWithManyOrders и пользователей, совершивших заказы дороже 58233.2 — usersWithExpensiveOrders. Объединим их в таблице abnormalUsers. Узнаем, сколько всего аномальных пользователей атрибутом shape.

In [203…
```
ordersByUsersA['visitiorId']
```

Out[203]:
```
0         8300375
1        11685486
2        54447517
3        66685450
4        78758296
          ...
498    4243832526
499    4256040402
500    4259830713
501    4266935830
502    4278982564
Name: visitiorId, Length: 503, dtype: int64
```

In [204…
```
ordersByUsersB.head()
```

Out[204]:

	visitorId	orders
0	5114589	1
1	6958315	1
2	8300375	1
3	39475350	1
4	47206413	1

In [205…

```
usersWithManyOrders = pd.concat(
    [
        ordersByUsersA[ordersByUsersA['orders'] >= abnormal_count]['visit
        ordersByUsersB[ordersByUsersB['orders'] >= abnormal_count]['visit
    ],
    axis=0,
)
usersWithExpensiveOrders = orders[orders['revenue'] >= abnormal_revenue][
abnormalUsers = (
    pd.concat([usersWithManyOrders, usersWithExpensiveOrders], axis=0)
    .drop_duplicates()
    .sort_values()
)
print(abnormalUsers.head(5))
print(abnormalUsers.shape[0])
```

```
1099    148427295
18      199603092
23      237748145
949     887908475
744     888512513
dtype: int64
20
```

Всего 20 аномальных пользователей. 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 [206…
```
sampleAFiltered = pd.concat(
    [
        ordersByUsersA[
            np.logical_not(ordersByUsersA['visitiorId'].isin(abnormalUser
        ]['orders'],
        pd.Series(
            0,
            index=np.arange(
                data['visitorsPerDateA'].sum() - len(ordersByUsersA['orde
            ),
            name='orders',
        ),
    ],
    axis=0,
)

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

Применим статистический критерий Манна-Уитни к полученным выборкам:

H0: Среднее число заказов в группах A и B совпадает

H1: Среднее число заказов в группах A и B не совпадает

In [207…
```
print('p-value = {0:.3f}'.format(stats.mannwhitneyu(sampleAFiltered, samp
print('difference% = {0:.3f}'.format(sampleBFiltered.mean()/sampleAFilter
```

```
p-value = 0.014
difference% = 0.151
```

Результаты по среднему количеству заказов практически не изменились.

Произошло ли что-нибудь с результатами по среднему чеку?

H0: Средний чек в группах A и B совпадает

H1: Средний чек в группах A и B не совпадает

In [208…

```
print(
    'p-value = {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(
    "difference% = {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
    )
)
```

```
p-value = 0.959
difference% = -0.014
```

P-value увеличился, но и разница между сегментами сократилась с 25.9% до -1.4%. Изменение составило 27.3%
Хотя общие выводы по результатам теста не изменились, такой пример хорошо показывает, как сильно аномалии могут влиять на результаты A/B-теста!
Какие выводы по тесту можем сделать?
Имеющиеся факты:

- Есть статистически значимое различие по среднему количеству заказов между группами как по «сырым», так и по данным после фильтрации аномалий;
- Нет статистически значимого различия по среднему чеку между группами ни по «сырым», ни по данным после фильтрации аномалий;
- График различия конверсии между группами сообщает, что результаты группы B лучше группы A и нет значительной тенденции к ухудшению:

Часть 3. Вывод

Исходя из обнаруженных фактов, тест следует остановить и признать его успешным. Получено статистически значимое различие в количестве заказов между группами A и B. Исходя из чего делаем вывод, что группа B лучше группы A.