b6b3eab6-8986-473f-ba11-762036a6147d

October 5, 2024

Hola

Soy **Hesus Garcia** como "Jesús" pero con H. Sé que puede ser confuso al principio, pero una vez que lo recuerdes, ¡nunca lo olvidarás! . Como revisor de código de Triple-Ten, estoy emocionado de examinar tus proyectos y ayudarte a mejorar tus habilidades en programación. si has cometido algún error, no te preocupes, pues ¡estoy aquí para ayudarte a corregirlo y hacer que tu código brille! . Si encuentro algún detalle en tu código, te lo señalaré para que lo corrijas, ya que mi objetivo es ayudarte a prepararte para un ambiente de trabajo real, donde el líder de tu equipo actuaría de la misma manera. Si no puedes solucionar el problema, te proporcionaré más información en la próxima oportunidad. Cuando encuentres un comentario, **por favor, no los muevas, no los modifiques ni los borres**.

Revisaré cuidadosamente todas las implementaciones que has realizado para cumplir con los requisitos y te proporcionaré mis comentarios de la siguiente manera:

Comentario del revisor Si todo está perfecto.

Comentario del revisor Si tu código está bien pero se puede mejorar o hay algún detalle que le hace falta.

Comentario del revisor Si de pronto hace falta algo o existe algún problema con tu código o conclusiones.

Puedes responderme de esta forma:

Respuesta del estudiante

¡Empecemos!

1 PROYECTO 9

1.1 INTRODUCCIÓN

Contexto

Como analista en una gran tienda en línea, yjunto al departamento de marketing, se han recopilado una lista de hipótesis que pueden ayudar a aumentar los ingresos, así como las tablas de resultados de un test A/B.

Objetivos

Depurar los datos.

Priorizar listas de hipótesis.

Analizar los resultados del test A/B.

Comentario del revisor Considera la posibilidad de incluir una tabla de contenidos al inicio de tu proyecto. Esto no solo mejorará la estructura y presentación de tu análisis, sino que también facilitará la navegación y comprensión de los distintos temas tratados, especialmente para proyectos extensos con múltiples secciones y subsecciones. Una tabla de contenidos bien organizada puede mejorar significativamente la experiencia del lector y destacar la profesionalidad de tu trabajo.

1.2 PREPROCESAMIENTO DE DATOS

1.2.1 Importación de librerías

[1]:
```
import pandas as pd
import numpy as np
from matplotlib import pyplot as plt
from scipy import stats as st
```

1.2.2 Importación de archivos

[2]:
```
df_hypo = pd.read_csv('/datasets/hypotheses_us.csv')
df_orders = pd.read_csv('/datasets/orders_us.csv')
df_visits = pd.read_csv('/datasets/visits_us.csv')
```

1.2.3 Revisión y optimización de datos

Tabla *hypotheses*

[3]:
```
# Vistazo a primeras filas de la tabla
df_hypo.head()
```

[3]:
```
            Hypothesis;Reach;Impact;Confidence;Effort
0  Add two new channels for attracting traffic. T…
1  Launch your own delivery service. This will sh…
2  Add product recommendation blocks to the store…
3  Change the category structure. This will incre…
4  Change the background color on the main page. …
```

[4]:
```
# Cambiar el delimitador de la tabla
df_hypo = pd.read_csv('/datasets/hypotheses_us.csv', sep=';')
```

[5]:
```
# Revisar información de la tabla
df_hypo.info(memory_usage='deep')
print(df_hypo)
```

```
<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   Effort      9 non-null      int64
dtypes: int64(4), object(1)
memory usage: 1.6 KB
                                          Hypothesis  Reach  Impact  \
0  Add two new channels for attracting traffic. T…      3      10
1  Launch your own delivery service. This will sh…      2       5
2  Add product recommendation blocks to the store…      8       3
3  Change the category structure. This will incre…      8       3
4  Change the background color on the main page. …      3       1
5  Add a customer review page. This will increase…      3       2
6  Show banners with current offers and sales on …      5       3
7  Add a subscription form to all the main pages…      10       7
8  Launch a promotion that gives users discounts …      1       9

   Confidence  Effort
0           8       6
1           4      10
2           7       3
3           3       8
4           1       1
5           2       3
6           8       3
7           8       5
8           9       5
```

[6]:
```
# Cambiar nombre de columnas a minúsculas
df_hypo.columns = df_hypo.columns.str.lower()
df_hypo.head()
```

[6]:
```
                                          hypothesis  reach  impact  \
0  Add two new channels for attracting traffic. T…      3      10
1  Launch your own delivery service. This will sh…      2       5
2  Add product recommendation blocks to the store…      8       3
3  Change the category structure. This will incre…      8       3
4  Change the background color on the main page. …      3       1

   confidence  effort
0           8       6
1           4      10
2           7       3
3           3       8
4           1       1
```

[7]:
```
# Adecuación del índice de la tabla
df_hypo.index = list(range(1,10))
print(df_hypo)
```

```
                                          hypothesis  reach  impact  \
1  Add two new channels for attracting traffic. T…      3      10
2  Launch your own delivery service. This will sh…      2       5
3  Add product recommendation blocks to the store…      8       3
4  Change the category structure. This will incre…      8       3
5  Change the background color on the main page. …      3       1
6  Add a customer review page. This will increase…      3       2
7  Show banners with current offers and sales on …      5       3
8  Add a subscription form to all the main pages…      10       7
9  Launch a promotion that gives users discounts …      1       9

   confidence  effort
1           8       6
2           4      10
3           7       3
4           3       8
5           1       1
6           2       3
7           8       3
8           8       5
9           9       5
```

Tabla *orders*

[8]:
```
# Vistazo a primeras filas de la tabla
df_orders.head()
```

[8]:

	transactionId	visitorId	date	revenue	group
0	3667963787	3312258926	2019-08-15	30.4	B
1	2804400009	3642806036	2019-08-15	15.2	B
2	2961555356	4069496402	2019-08-15	10.2	A
3	3797467345	1196621759	2019-08-15	155.1	B
4	2282983706	2322279887	2019-08-15	40.5	B

[9]:
```
# Revisar información de la tabla
df_orders.info(memory_usage = 'deep')
```

```
<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   float64
 4   group          1197 non-null   object
dtypes: float64(1), int64(2), object(2)
memory usage: 174.3 KB
```

[10]:
```
# Cambiar el tipo de datos de las columnas de fechas
df_orders['date'] = pd.to_datetime(df_orders['date'], format = "%Y-%m-%d")
```

[11]:
```
# Revisar entradas duplicadas
print(df_orders[df_orders.duplicated()])
```

```
Empty DataFrame
Columns: [transactionId, visitorId, date, revenue, group]
Index: []
```

[12]:
```
# Revisar visitantes de un grupo presentes en otro
orders_AB = df_orders[df_orders.duplicated(subset = ['visitorId'], keep=False)]␣
 ↪# Registros y sus duplicados de visitantes sin discriminar
orders_B = orders_AB.query('group == "B"') # Los registros con duplicados del␣
 ↪grupo B
orders_A = orders_AB.query('group == "A"') # Los registros con duplicados del␣
 ↪grupo A

# Visitantes de grupo A encontrados en grupo B
orders_Bu = orders_B.drop_duplicates(subset = 'visitorId') # Descarto␣
 ↪duplicados dentro del mismo grupo
orders_Au = orders_A.drop_duplicates(subset = 'visitorId') # Descarto␣
 ↪duplicados dentro del mismo grupo
order_merge = orders_Au.merge(orders_Bu, how = 'inner', on = 'visitorId') #␣
 ↪Intersecar ambas tablas en función de las ID de visitantes que se encuentran␣
 ↪en ambos grupos
print(order_merge)
```

```
    transactionId_x   visitorId     date_x  revenue_x group_x  \
0        2961555356  4069496402 2019-08-15       10.2       A
1        2223239646   199603092 2019-08-15       55.7       A
2        2220299125  3803269165 2019-08-15       15.8       A
3         486237576  2378935119 2019-08-15       30.4       A
4        2594268419   237748145 2019-08-15       20.8       A
5        1120327437  4256040402 2019-08-01       90.2       A
6         722060263  2038680547 2019-08-22       15.7       A
7        1665445278  1738359350 2019-08-22       50.2       A
8        3060563671  2458001652 2019-08-22       80.8       A
9        1170827001  3891541246 2019-08-22       20.2       A
10       2632798290  2716752286 2019-08-02       35.2       A
11       2066718132  3656415546 2019-08-23       45.8       A
```

```
12       2139320439  2686716486 2019-08-23      170.6       A
13       2931845376  2954449915 2019-08-17      170.0       A
14       3124204494  2927087541 2019-08-23      425.8       A
15       1360774578  3234906277 2019-08-18      120.2       A
16       2370268995   457167155 2019-08-23       25.9       A
17       3743515850  2579882178 2019-08-18       30.6       A
18       1067267410  3957174400 2019-08-18       40.2       A
19       1101659272  1648269707 2019-08-18       15.2       A
20        609915801  2780786433 2019-08-28       20.5       A
21       4276811111   818047933 2019-08-28       55.2       A
22        857639553  2044997962 2019-08-24       40.1       A
23       1863281703  1959144690 2019-08-25      255.5       A
24       1168756094  3202540741 2019-08-24       50.5       A
25       1750160666  1333886533 2019-08-06       25.5       A
26       3075639014   351125977 2019-08-06       70.0       A
27        246848596  3951559397 2019-08-29       40.3       A
28       3031564664   393266494 2019-08-19      335.5       A
29       3925488023  3984495233 2019-08-04      315.1       A
30       4212256267  4120364173 2019-08-29      615.3       A
31       1811671147  4266935830 2019-08-29     1220.2       A
32       3667885894  1230306981 2019-08-09      105.7       A
33       2348589867  1294878855 2019-08-09      425.4       A
34       2663041816  1614305549 2019-08-29       25.5       A
35       4293855558     8300375 2019-08-07       30.5       A
36       3335803766   477780734 2019-08-30      115.3       A
37       2339954598  1668030113 2019-08-27       40.0       A
38       3362484972  3717692402 2019-08-21       40.9       A
39       2514024187  3766097110 2019-08-30       20.1       A
40       2076434956  3941795274 2019-08-21       20.2       A
41       3734714128   471551937 2019-08-05      355.1       A
42        192721366  1316129916 2019-08-27     1450.2       A
43       1197739160  1801183820 2019-08-05       10.0       A
44       2792059099  1602967004 2019-08-08       65.2       A
45       3478707774  2587333274 2019-08-08       60.4       A
46       2316868256  2600415354 2019-08-31       45.9       A
47         83566152   232979603 2019-08-31        5.9       A
48       2898835960  3972127743 2019-08-09      130.8       A
49       1814628689  1404934699 2019-08-11      135.9       A
50       3894437543   276558944 2019-08-12       80.7       A
51       1092419081  3062433592 2019-08-12      265.3       A
52       1254962016  2654030115 2019-08-14       35.5       A
53       1277417350  3963646447 2019-08-14      100.2       A
54       3757656646  4186807279 2019-08-14       80.8       A
55       3612788481  2712142231 2019-08-14       40.3       A
56       4052155355  2949041841 2019-08-14        5.9       A
57       1458356232   963407295 2019-08-14       80.0       A

    transactionId_y     date_y  revenue_y group_y
```

0	1473132782	2019-08-12	10.8	B
1	437656952	2019-08-02	55.7	B
2	473864496	2019-08-23	40.3	B
3	2213813903	2019-08-25	330.5	B
4	1630050528	2019-08-15	120.2	B
5	1421016313	2019-08-16	875.5	B
6	3666913472	2019-08-15	30.8	B
7	3006440800	2019-08-28	15.7	B
8	1177690313	2019-08-25	40.7	B
9	4007826947	2019-08-04	10.4	B
10	4141167864	2019-08-01	125.9	B
11	265631116	2019-08-11	5.8	B
12	1545495643	2019-08-23	530.3	B
13	2781850870	2019-08-06	50.2	B
14	1651227034	2019-08-24	35.7	B
15	1162046357	2019-08-15	120.2	B
16	252633006	2019-08-22	95.8	B
17	131747281	2019-08-18	305.6	B
18	3241914033	2019-08-26	40.2	B
19	4189935502	2019-08-17	5.5	B
20	847315305	2019-08-27	10.3	B
21	2357685128	2019-08-14	10.2	B
22	3754751399	2019-08-26	95.7	B
23	189332332	2019-08-27	255.5	B
24	1061451265	2019-08-24	85.8	B
25	511953429	2019-08-29	50.4	B
26	4252514150	2019-08-15	235.6	B
27	2825038272	2019-08-29	480.8	B
28	936917445	2019-08-08	35.7	B
29	3532704780	2019-08-03	315.1	B
30	969750843	2019-08-31	240.5	B
31	1216533772	2019-08-29	1220.2	B
32	1748608673	2019-08-09	100.5	B
33	371848868	2019-08-28	50.4	B
34	2972137054	2019-08-29	150.9	B
35	3679129301	2019-08-01	165.7	B
36	132561921	2019-08-30	60.2	B
37	3909269888	2019-08-28	80.0	B
38	615966907	2019-08-28	15.0	B
39	2470658885	2019-08-30	10.5	B
40	698171827	2019-08-22	25.3	B
41	2676541142	2019-08-14	55.2	B
42	3922986948	2019-08-27	15.8	B
43	2726113349	2019-08-05	5.5	B
44	2726404029	2019-08-27	75.9	B
45	1825231501	2019-08-06	140.5	B
46	1954636284	2019-08-06	25.5	B
47	2670069237	2019-08-31	45.6	B

```
48       3061324106 2019-08-04      130.2       B
49       1441855393 2019-08-03      420.1       B
50       1701653566 2019-08-22      190.0       B
51       2736731761 2019-08-01      225.9       B
52       3620682463 2019-08-13       35.5       B
53       2289555915 2019-08-14      120.2       B
54       1759418862 2019-08-21      290.5       B
55       1251767592 2019-08-16       15.2       B
56       1329499668 2019-08-14      160.7       B
57       2904772834 2019-08-15        5.6       B
```

Se encontraron 57 ID de visitantes de un grupo coincidentes con el de otro. Obviamente las entradas de la tabla que contengan a esos visitantes no pueden considerarse para el análisis.

[13]:
```
# Una nueva tabla filtrando las ID de los visitantes de un grupo que se␣
 ↪encuentran en el otro.
df_orders_us = df_orders[~df_orders.visitorId.isin(order_merge['visitorId'])]
df_orders_us
```

[13]:

	transactionId	visitorId	date	revenue	group
0	3667963787	3312258926	2019-08-15	30.4	B
1	2804400009	3642806036	2019-08-15	15.2	B
3	3797467345	1196621759	2019-08-15	155.1	B
4	2282983706	2322279887	2019-08-15	40.5	B
5	182168103	935554773	2019-08-15	35.0	B
…	…	…	…	…	…
1191	3592955527	608641596	2019-08-14	255.7	B
1192	2662137336	3733762160	2019-08-14	100.8	B
1193	2203539145	370388673	2019-08-14	50.1	A
1194	1807773912	573423106	2019-08-14	165.3	A
1196	3936777065	2108080724	2019-08-15	3120.1	B

[1016 rows x 5 columns]

Tabla *visits*

[14]:
```
# Vistazo a primeras filas de la tabla
df_visits.head()
```

[14]:

	date	group	visits
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

[15]:
```
# Revisar información de la tabla
df_visits.info(memory_usage = 'deep')
```

```
df_visits.head(10)
```

```
<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   visits  62 non-null     int64 
dtypes: int64(1), object(2)
memory usage: 8.2 KB
```

[15]:

	date	group	visits
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
5	2019-08-06	A	667
6	2019-08-07	A	644
7	2019-08-08	A	610
8	2019-08-09	A	617
9	2019-08-10	A	406

[16]:
```
# Cambiar el tipo de datos de las columnas de fechas
df_visits['date'] = pd.to_datetime(df_visits['date'], format="%Y-%m-%d")
```

[17]:
```
# Revisar entradas duplicadas
print(df_visits[df_visits.duplicated()])
```

```
Empty DataFrame
Columns: [date, group, visits]
Index: []
```

Comentario del revisor

Buen trabajo en la depuración y optimización de los datos. La revisión de duplicados, la eliminación de valores no válidos y el ajuste de los tipos de datos son pasos cruciales que has implementado correctamente. También has hecho un análisis claro de las ID de los visitantes que aparecen en ambos grupos, filtrando adecuadamente estos casos para asegurar la integridad de los datos en el test A/B.

1.3 PRIORIZACIÓN DE HIPÓTESIS

[18]:
```
# Despliegue de hipótesis
with pd.option_context('display.max_colwidth', None):
    print(df_hypo['hypothesis'])
```

```
1                                                 Add two new channels for
attracting traffic. This will bring 30% more users
2                                                       Launch your own
delivery service. This will shorten delivery time
3            Add product recommendation blocks to the store's site. This will
increase conversion and average purchase size
4    Change the category structure. This will increase conversion since users
will find the products they want more quickly
5                                          Change the background color on the
main page. This will increase user engagement
6                                                    Add a customer review
page. This will increase the number of orders
7                                   Show banners with current offers and sales
on the main page. This will boost conversion
8                                  Add a subscription form to all the main
pages. This will help you compile a mailing list
9                                                     Launch a promotion
that gives users discounts on their birthdays
Name: hypothesis, dtype: object
```

1.3.1 Framework ICE para priorizar hipótesis

[19]:
```
# Cálculo del ICE para las hipótesis
df_hypo['ICE'] = (df_hypo['impact'] * df_hypo['confidence'] /␣
 ↪df_hypo['effort']).round(3)
print (df_hypo[['hypothesis','ICE']].sort_values(by='ICE', ascending = False))
```

```
                                        hypothesis     ICE
9  Launch a promotion that gives users discounts …  16.200
1  Add two new channels for attracting traffic. T…  13.333
8  Add a subscription form to all the main pages…  11.200
7  Show banners with current offers and sales on …   8.000
3  Add product recommendation blocks to the store…   7.000
2  Launch your own delivery service. This will sh…   2.000
6  Add a customer review page. This will increase…   1.333
4  Change the category structure. This will incre…   1.125
5  Change the background color on the main page. …   1.000
```

El ICE presenta la hipótesis 9 como la más prometedora, seguida de la 1, 8 y (ya con un sólo dígito en puntaje) 7.

1.3.2 Framework RICE para priorizar hipótesis

[20]:
```
# Cálculo del RICE para las hipótesis
df_hypo['RICE'] = (df_hypo['reach'] * df_hypo['impact'] * df_hypo['confidence']␣
 ↪/ df_hypo['effort']).round(3)
print (df_hypo[['hypothesis','RICE']].sort_values(by='RICE', ascending = False))
```

```
                                        hypothesis   RICE
8  Add a subscription form to all the main pages…  112.0
3  Add product recommendation blocks to the store…   56.0
1  Add two new channels for attracting traffic. T…   40.0
7  Show banners with current offers and sales on …   40.0
9  Launch a promotion that gives users discounts …   16.2
4  Change the category structure. This will incre…    9.0
2  Launch your own delivery service. This will sh…    4.0
6  Add a customer review page. This will increase…    4.0
5  Change the background color on the main page. …    3.0
```

El RICE en cambio presenta, y por mucho, la hipótesis 8 como la más prometedora. Le sigue, con la mitad del puntaje, la 3; luego la 1 y la 7; la hipótesis 9, primera con el método anterior, se encuentra en quinto lugar aquí.

El criterio de priorización cambia con el método RICE porque, aparte de considerar lo fuerte que es la hipótesis, este método considera a cuántos usuarios afectará. En esa línea, la hipótesis 8 tiene un puntaje de 10/10 en alcance, mientras que la hipótesis 9 un 1/10. Si bien esta última supera en dos puntos el impacto al usuario, la efectividad de su implementación se empañaría si su alcance es muy bajo. Aparte de esta hipótesis podría considerarse además la 1, que se posiciona relativamente bien con ambos métodos.

Comentario del revisor

El uso de los frameworks ICE y RICE está muy bien fundamentado. Has realizado un análisis detallado de la diferencia en los resultados de ambas métricas y cómo la hipótesis con mejor puntaje puede variar dependiendo del alcance (reach). Esto muestra una buena comprensión del impacto de cada método de priorización y cómo puede influir en la toma de decisiones.

Comentario del revisor

Podrías expandir un poco más sobre las diferencias cualitativas entre las hipótesis destacadas por ICE y RICE. Además, sería útil realizar una breve discusión sobre cómo la combinación de ambos métodos puede ayudar a tomar decisiones más equilibradas.

1.4 ANÁLISIS DEL TEST A/B

[21]:
```
# Preparación de tablas acumuladas de pedidos y de visitas
dates_groups = df_orders_us[['date','group']].drop_duplicates()
orders_agg = dates_groups.apply(lambda x: df_orders_us[np.
 ↪logical_and(df_orders_us['date'] <= x['date'],
           df_orders_us['group'] == x['group'])].agg({'date' : 'max', 'group' :
 ↪ 'max', 'transactionId' : 'nunique',
                                                    'visitorId' : 'nunique',␣
 ↪'revenue' : 'sum'}),
                                axis=1).sort_values(by=['date','group'])
print(orders_agg)
visitors_agg = dates_groups.apply(lambda x: df_visits[np.
 ↪logical_and(df_visits['date'] <= x['date'],
```

```
            df_visits['group'] == x['group'])].agg({'date' : 'max', 'group' :␣
 ↪'max', 'visits' : 'sum'}),
                                    axis=1).sort_values(by=['date','group'])
print(visitors_agg)
```

```
          date group  transactionId  visitorId  revenue
55  2019-08-01     A             23         19   2266.6
66  2019-08-01     B             17         17    967.2
175 2019-08-02     A             42         36   3734.9
173 2019-08-02     B             40         39   3535.3
291 2019-08-03     A             66         60   5550.1
..         …   …             …        …       …
533 2019-08-29     B            510        490  74576.7
757 2019-08-30     A            460        437  52363.7
690 2019-08-30     B            531        511  77863.5
958 2019-08-31     A            468        445  53212.0
930 2019-08-31     B            548        528  79651.2

[62 rows x 5 columns]
          date group  visits
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
..         …   …      …
533 2019-08-29     B   17708
757 2019-08-30     A   18037
690 2019-08-30     B   18198
958 2019-08-31     A   18736
930 2019-08-31     B   18916

[62 rows x 3 columns]
```

[22]:
```
# Tabla acumulada
cum_data = orders_agg.merge(visitors_agg, on=['date', 'group'], how='inner')
cum_data.rename(columns = {'transactionId' : 'orders', 'visitorId' : 'buyers'},␣
 ↪inplace = True)
cum_data['conversion'] = cum_data['orders'] / cum_data['visits']
print(cum_data.head())
```

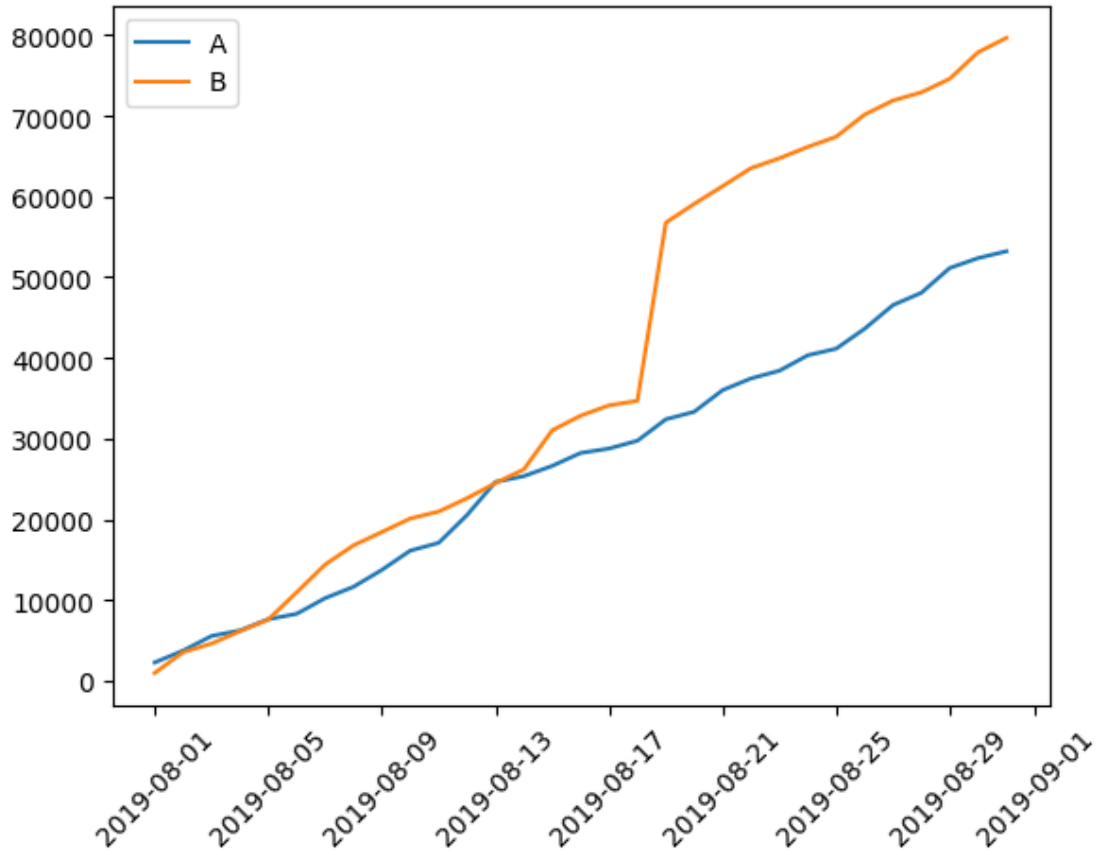
```
        date group  orders  buyers  revenue  visits  conversion
0 2019-08-01     A      23      19   2266.6     719    0.031989
1 2019-08-01     B      17      17    967.2     713    0.023843
2 2019-08-02     A      42      36   3734.9    1338    0.031390
3 2019-08-02     B      40      39   3535.3    1294    0.030912
4 2019-08-03     A      66      60   5550.1    1845    0.035772
```

1.4.1 Ingreso acumulado por grupo

[23]:
```
cum_data_A = cum_data[cum_data['group'] == 'A'][['date','revenue', 'orders',␣
 ↪'conversion']] # Acumulado del grupo A
cum_data_B = cum_data[cum_data['group'] == 'B'][['date','revenue', 'orders',␣
 ↪'conversion']] # Acumulado del grupo B

plt.plot(cum_data_A['date'], cum_data_A['revenue'], label = 'A')
plt.plot(cum_data_B['date'], cum_data_B['revenue'], label = 'B')
plt.xticks(rotation=45)
plt.legend()
```

[23]:
```
<matplotlib.legend.Legend at 0x7fa2abdf3130>
```

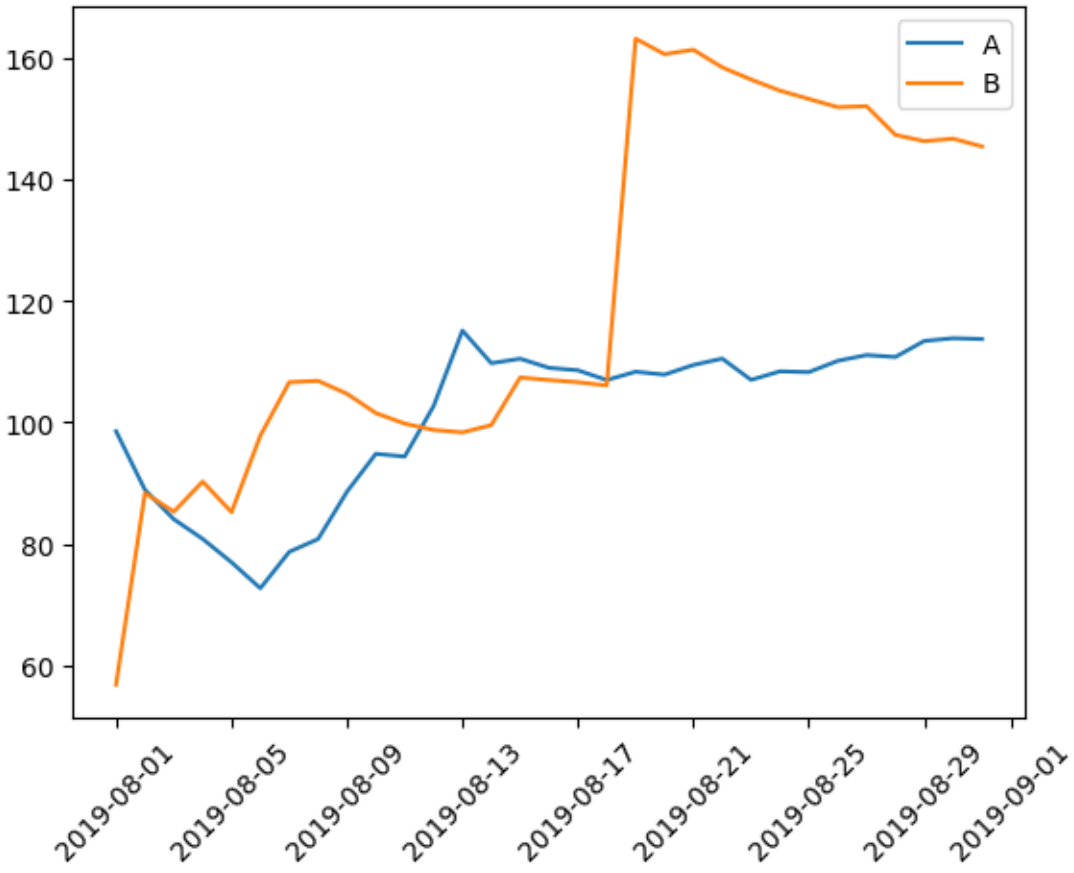


El segmento B empezó a sobrepasar al A desde del quinto día de la prueba; pero a partir de la mitad la brecha se agrandó de forma abrupta, manteniéndose la diferencia hasta el final. Estos cambios sugieren la presencia de pedidos anormalmente grandes.

1.4.2 Tamaño de pedido promedio acumulado por grupo

[24]:
```
plt.plot(cum_data_A['date'], cum_data_A['revenue'] / cum_data_A['orders'],␣
 ↪label='A')
plt.plot(cum_data_B['date'], cum_data_B['revenue'] / cum_data_B['orders'],␣
 ↪label='B')
plt.xticks(rotation=45)
plt.legend()
```

[24]:
```
<matplotlib.legend.Legend at 0x7fa2abdb3580>
```

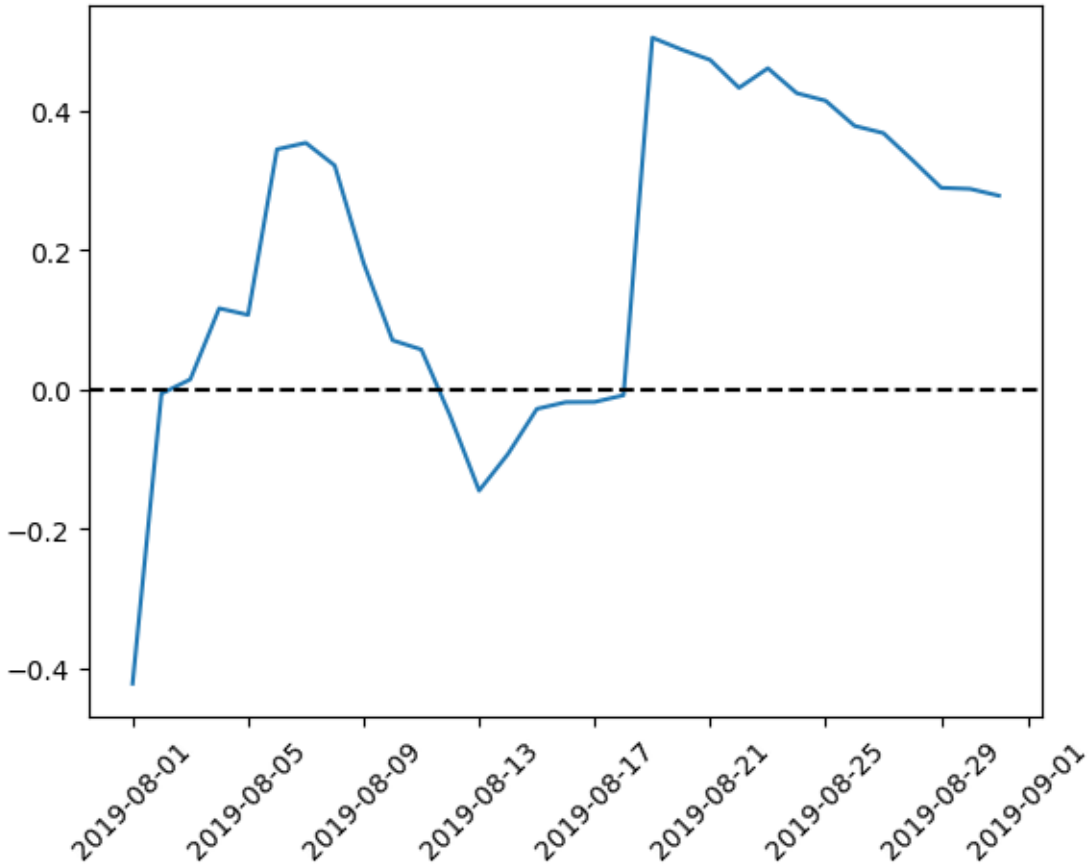


El valor acumulado de los tamaños promedio de compra de los segmentos sigue un patrón similar, pero con fluctuaciones más acentuadas. Habría entonces que analizar los valores atípicos y su influencia en los resultados.

1.4.3 Diferencia relativa en el tamaño de pedido promedio acumulado para el grupo B en comparación con el grupo A

[25]:
```
merged_cum = cum_data_A.merge(cum_data_B, on='date', suffixes=['A', 'B'])
plt.plot(merged_cum['date'], (merged_cum['revenueB'] / merged_cum['ordersB']) /
         (merged_cum['revenueA']/merged_cum['ordersA'])-1)
plt.xticks(rotation=45)
plt.axhline(y=0, color='black', linestyle='--')
```

[25]:
```
<matplotlib.lines.Line2D at 0x7fa2adebefd0>
```



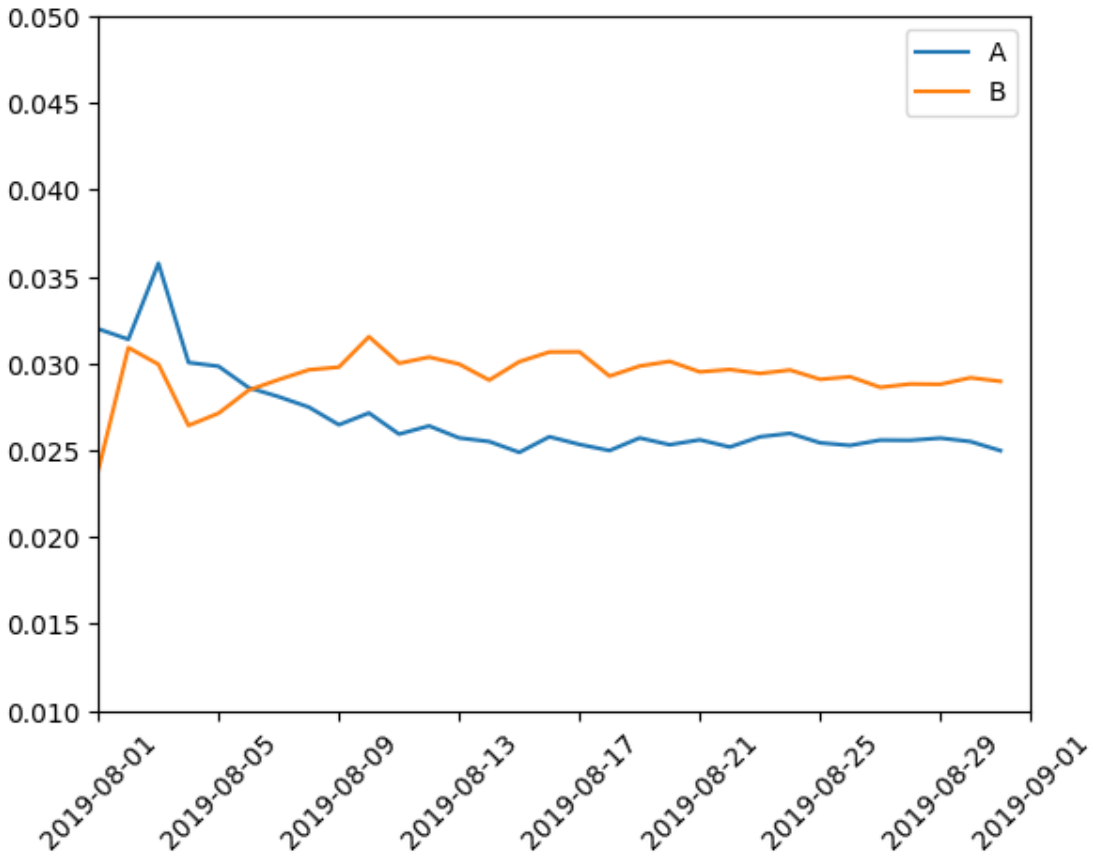
Los cambios repentinos y marcados que se observan pueden estar relacionados con compras anormalmente voluminosas.

1.4.4 Tasa de conversión de cada grupo

[26]:
```
plt.plot(cum_data_A['date'], cum_data_A['conversion'], label='A')
plt.plot(cum_data_B['date'], cum_data_B['conversion'], label='B')
plt.xticks(rotation=45)
```

```
plt.axis([pd.to_datetime('2019-08-01'), pd.to_datetime('2019-09-01'), 0.01, 0.
↪05])
plt.legend()
```

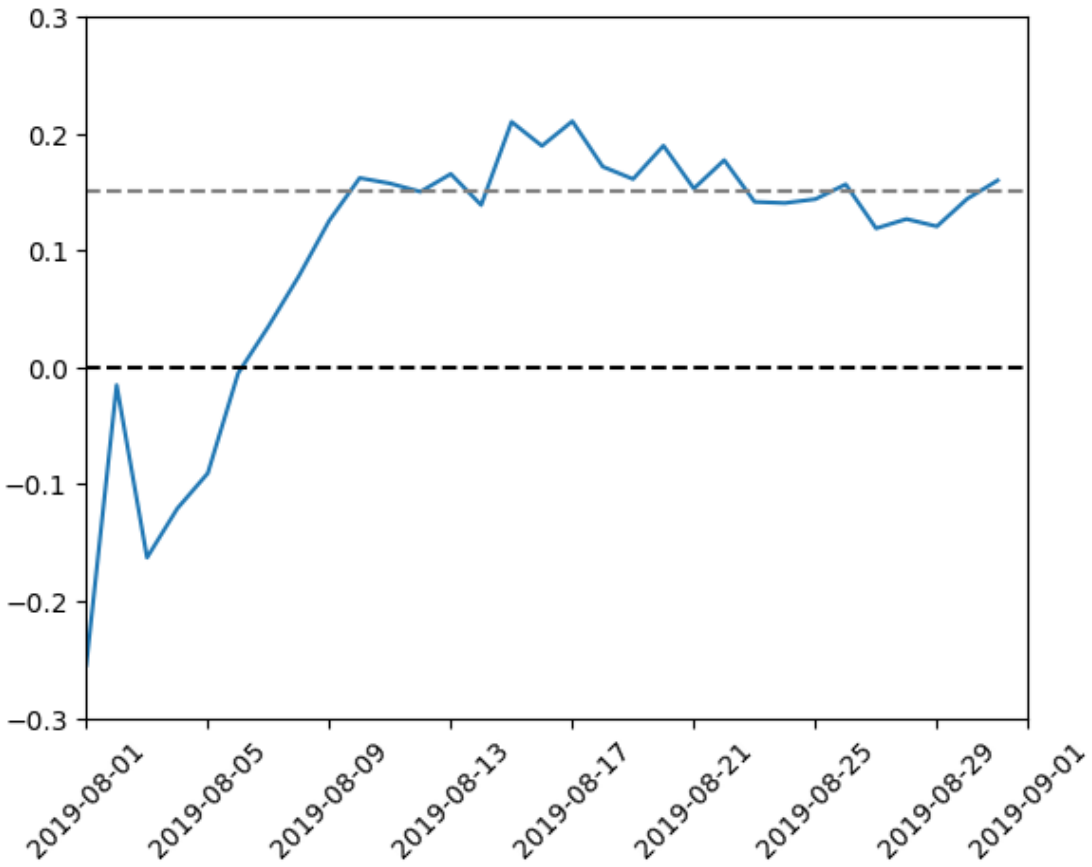
[26]: <matplotlib.legend.Legend at 0x7fa2abc3e880>



Al inicio de la prueba el segmento A tenía una tasa de conversión más alta, pero después de una semana el segmento B tomó la delantera y luego se estabilizó en un valor casi constante y en paralelo con el segmenteo A.

[27]:
```
plt.plot(merged_cum['date'], merged_cum['conversionB']/
↪merged_cum['conversionA']-1)
plt.xticks(rotation=45)
plt.axhline(y=0, color='black', linestyle='--')
plt.axhline(y=0.15, color='grey', linestyle='--')
plt.axis([pd.to_datetime('2019-08-01'), pd.to_datetime('2019-09-01'), -0.3, 0.
↪3])
```

[27]: (18109.0, 18140.0, -0.3, 0.3)



Al ver la diferencia relativa en las tasas de conversión, el grupo B llevó la cabeza desde la primera semana estabilizándose en aproximadamente 15% más que el grupo A.

Comentario del revisor

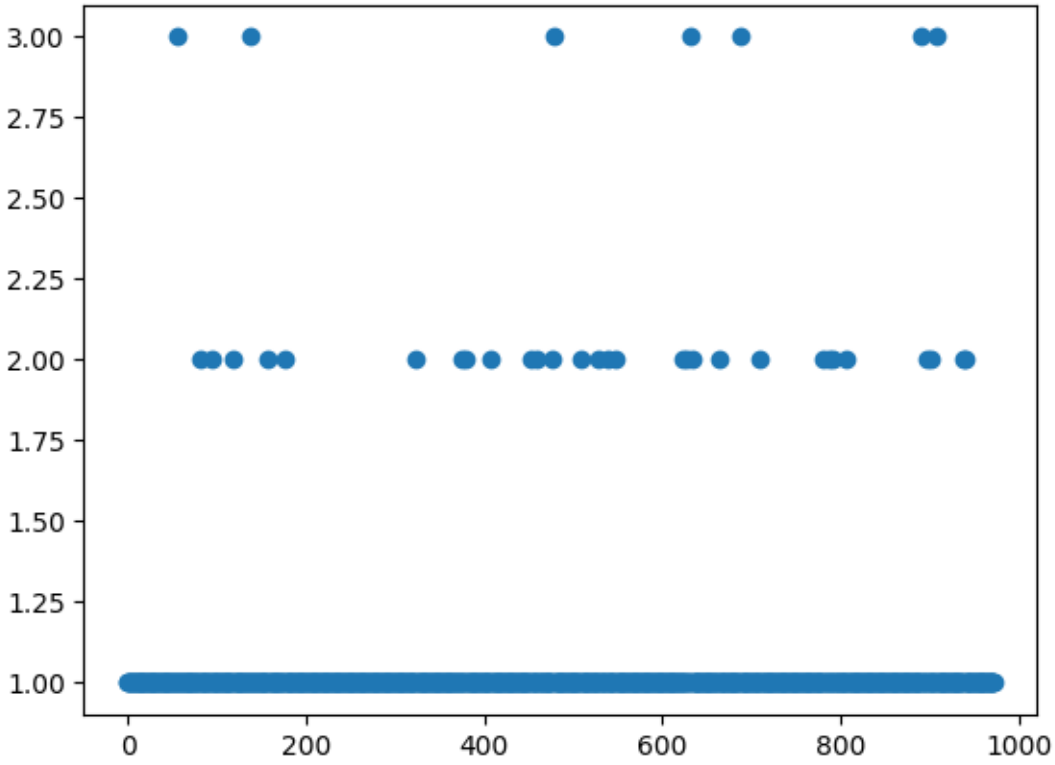
La creación de tablas acumuladas y la visualización de los ingresos y tamaños de pedido promedio son claros y efectivos. Las gráficas muestran las diferencias entre los grupos de manera comprensible y revelan la influencia de los valores atípicos. Tu análisis de la significancia estadística es sólido, usando pruebas de Mann-Whitney para validar las diferencias en la conversión y el tamaño de pedido entre los grupos.

1.4.5 Dispersión del número de pedidos

[28]:
```
# Tabla de pedidos por usuario
orders_users = df_orders_us.groupby('visitorId', as_index=False).
 ↪agg({'transactionId': pd.Series.nunique})
orders_users.columns = ['visitorId', 'orders']

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

[28]: <matplotlib.collections.PathCollection at 0x7fa2abb6c340>



La gran mayoría de usuarios solo hicieron un pedido y muy pocos con dos o más pedidos (tres pedidos es lo máximo registrado, pero se aprecia que es bastante anómalo).

1.4.6 Estudio de anomalías en el número de pedidos

[29]:
```
# Cálculo de los percentiles 95 y 99 para el número de pedidos por usuario
print(np.percentile(orders_users['orders'],[95,99]))
```

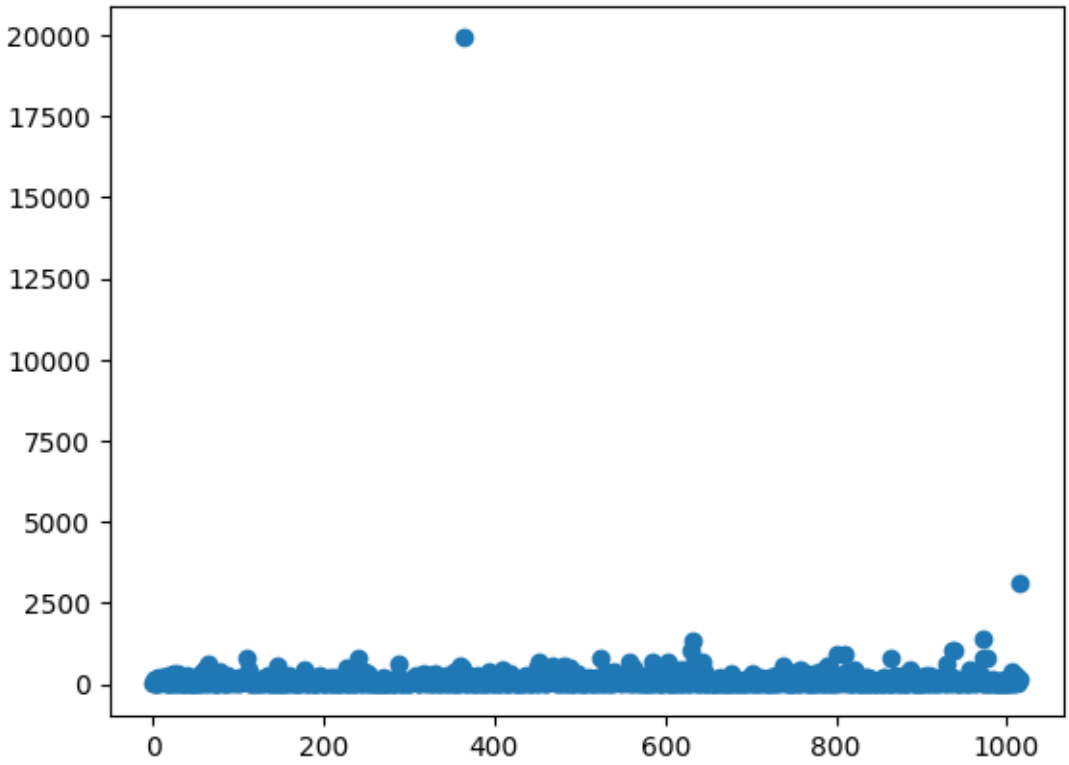
```
[1. 2.]
```

No más del 5% de los usuarios realizaron más de un pedido y menos del 1% realizaron tres.

1.4.7 Dispersión del precio de los pedidos

[30]:
```
x_revenue = pd.Series(range(0, len(df_orders_us['revenue'])))
plt.scatter(x_revenue, df_orders_us['revenue'])
```

[30]: <matplotlib.collections.PathCollection at 0x7fa2abad4d90>

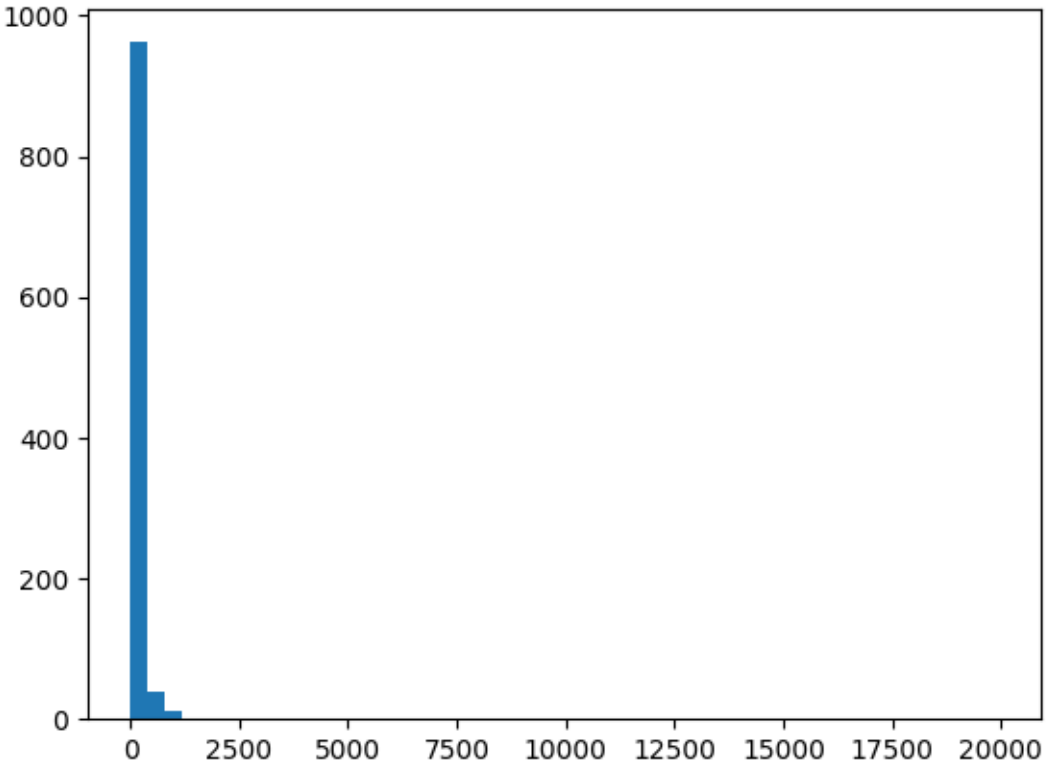


La gran mayoría de los pedidos no superan los 2 000. Hay un par de valores atípicos relativamente extremos que superan los 2 500 (uno de ellos alcanzando los 20 000, un número claramente distorsionador). Sin embargo, este gráfico muestra que hay mucha coalescencia de valores típicos en las cifras más bajas.

[31]:
```
# Histograma de distribución de ingresos de pedidos
plt.hist(df_orders_us['revenue'], bins=50)
```

[31]:
```
(array([962.,  39.,  11.,   2.,   0.,   0.,   0.,   1.,   0.,   0.,   0.,
          0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,
          0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,
          0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,
          0.,   0.,   0.,   0.,   0.,   1.]),
 array([5.0000000e+00, 4.0330800e+02, 8.0161600e+02, 1.1999240e+03,
        1.5982320e+03, 1.9965400e+03, 2.3948480e+03, 2.7931560e+03,
        3.1914640e+03, 3.5897720e+03, 3.9880800e+03, 4.3863880e+03,
        4.7846960e+03, 5.1830040e+03, 5.5813120e+03, 5.9796200e+03,
        6.3779280e+03, 6.7762360e+03, 7.1745440e+03, 7.5728520e+03,
        7.9711600e+03, 8.3694680e+03, 8.7677760e+03, 9.1660840e+03,
        9.5643920e+03, 9.9627000e+03, 1.0361008e+04, 1.0759316e+04,
        1.1157624e+04, 1.1555932e+04, 1.1954240e+04, 1.2352548e+04,
        1.2750856e+04, 1.3149164e+04, 1.3547472e+04, 1.3945780e+04,
```

```
        1.4344088e+04, 1.4742396e+04, 1.5140704e+04, 1.5539012e+04,
        1.5937320e+04, 1.6335628e+04, 1.6733936e+04, 1.7132244e+04,
        1.7530552e+04, 1.7928860e+04, 1.8327168e+04, 1.8725476e+04,
        1.9123784e+04, 1.9522092e+04, 1.9920400e+04]),
<BarContainer object of 50 artists>)
```



Con el histograma se puede constatar la prevalencia de pedidos baratos en la mayoría de los casos.

1.4.8 Estudio de anomalías en el precio de los pedidos

[32]:
```
# Cálculo de los percentiles 95 y 99 para el precio de los pedidos
print(np.percentile(df_orders_us['revenue'],[95,99]))
```

```
[414.275 830.3  ]
```

Según el cálculo de percentiles menos del 5% de los usuarios realizaron pedidos que costaron más de 414 y no más del 1% realizaron pedidos que superen los 830. Aun así estas cifras son bastante más bajas a los valores atípicos visualizados en los gráficos anteriores.

Comentario del revisor

El filtrado de valores atípicos y el análisis de su impacto en los resultados de la prueba A/B son puntos fuertes de tu proyecto. Has manejado bien el análisis de los percentiles 95 y 99 para

identificar los usuarios y pedidos atípicos, y los resultados son interesantes al mostrar cómo el filtrado puede cambiar la dirección de la ganancia relativa entre los grupos.

1.4.9 Significancia estadística de la diferencia en la conversión entre los grupos utilizando los datos en bruto

[33]:
```
# Tablas de muestras de cada grupo
orders_usersA = df_orders_us[df_orders_us['group'] == 'A'].groupby('visitorId',␣
 ↪as_index=False).agg({'transactionId' : 'nunique'})
orders_usersA.columns = ['visitorId', 'orders']
sampleA = pd.concat([orders_usersA['orders'], pd.Series(0, index=np.
 ↪arange(df_visits[df_visits['group'] == 'A']['visits'].sum()
                                                                    -␣
 ↪len(orders_usersA['orders'])), name = 'orders')], axis = 0)
orders_usersB = df_orders_us[df_orders_us['group'] == 'B'].groupby('visitorId',␣
 ↪as_index=False).agg({'transactionId' : 'nunique'})
orders_usersB.columns = ['visitorId', 'orders']
sampleB = pd.concat([orders_usersB['orders'], pd.Series(0, index=np.
 ↪arange(df_visits[df_visits['group'] == 'B']['visits'].sum()
                                                                    -␣
 ↪len(orders_usersB['orders'])), name = 'orders')], axis = 0)

# Prueba de Mann-Whitney para calcular la significancia estadística en la␣
 ↪conversión entre los dos grupos
whitneyAB = st.mannwhitneyu(sampleA, sampleB)[1]
print('Valor p:','{:.5f}'.format(whitneyAB))
if whitneyAB < 0.05:
    print('Rechazamos la hipótesis nula')
else:
    print('No podemos rechazar la hipótesis nula')
```

```
Valor p: 0.01102
Rechazamos la hipótesis nula
```

El análisis estadístico de los datos en bruto muestra que las tasas de conversión de los grupos tienen diferencias estadísticamente significativas. Sin embargo, en estos datos no ajustados los valores atípicos en los pedidos pueden darnos una perspectiva distorisonada.

[34]:
```
# Diferencia relativa entre las tasas de conversión de ambos grupos
print("{:.3f}".format(sampleB.mean()/sampleA.mean()-1))
```

```
0.160
```

En los datos en bruto la ganancia de conversión relativa del grupo B en comparación con el grupo A es del 16%.

1.4.10 Significancia estadística de la diferencia en el tamaño promedio de pedido entre los grupos utilizando los datos en bruto

[35]:
```
# Prueba de Mann-Whitney
print("Valor p:",'{0:.5f}'.format(st.
 ↪mannwhitneyu(df_orders_us[df_orders_us['group'] == 'A']['revenue'],
                                          df_orders_us[df_orders_us['group'] ==␣
 ↪'B']['revenue'])[1]))
```

```
Valor p: 0.86223
```

[36]:
```
# Diferencia relativa entre el tamaño de pedido de ambos grupos
print('{0:.3f}'.format(df_orders_us[df_orders_us['group'] == 'B']['revenue'].
 ↪mean() /
                       df_orders_us[df_orders_us['group'] == 'A']['revenue'].
 ↪mean()-1))
```

```
0.278
```

El valor p es muy superior a 0.05, ergo se puede inferir que no existen diferencias estadísticamente significativas entre los tamaños de pedido promedio de los grupos. Sin embargo, la diferencia relativa entre ambos grupos es del 27.8%. Esto sólo se puede explicar por valores atípicos sesgando fuertemente los resultados.

1.4.11 Significancia estadística de la diferencia en la conversión entre los grupos utilizando los datos filtrados

[37]:
```
# Filtrar usuarios con más de 2 pedidos
users_many_orders = pd.concat([orders_usersA[orders_usersA['orders'] >␣
 ↪2]['visitorId'],
                               orders_usersB[orders_usersB['orders'] >␣
 ↪2]['visitorId']], axis = 0)

# Filtrar usuarios con pedidos mayores a 830
users_expensive_orders = df_orders_us.query("revenue > 830")['visitorId']

abnormal_users = pd.concat([users_many_orders, users_expensive_orders], axis =␣
 ↪0).drop_duplicates().sort_values()
print(abnormal_users)
```

```
1099     148427295
33       249864742
58       611059232
949      887908475
744      888512513
1103    1164614297
1136    1307669133
425     1920142716
```

```
1196    2108080724
211     2108163459
287     2254456485
131     2254586615
347     2742574263
310     2988190573
409     3908431265
613     3931967268
416     3967698036
940     4003628586
743     4133034833
Name: visitorId, dtype: int64
```

Se clasificaron como usuarios con valores atípicos a los usuarios con 3 pedidos (<1%) y los pedidos mayores a 830 (1%) .

[38]:
```
# Eliminación de valores atípicos en las muestras
sample_filterA = pd.concat([orders_usersA[~(orders_usersA['visitorId'].
 ↪isin(abnormal_users))]['orders'],
                            pd.Series(0, index=np.
 ↪arange(df_visits[df_visits['group'] == 'A']['visits'].sum()
                          - len(orders_usersA['orders'])),name='orders')], axis =␣
 ↪0)

sample_filterB = pd.concat([orders_usersB[~(orders_usersB['visitorId'].
 ↪isin(abnormal_users))]['orders'],
                            pd.Series(0, index=np.
 ↪arange(df_visits[df_visits['group'] == 'B']['visits'].sum()
                          - len(orders_usersB['orders'])),name='orders')], axis =␣
 ↪0)

# Prueba de Mann-Whitney para calcular la significancia estadística en la␣
 ↪conversión entre los dos grupos filtrados
whitneyAB_fil = st.mannwhitneyu(sample_filterA, sample_filterB)[1]
print('Valor p:','{:.5f}'.format(whitneyAB_fil))
if whitneyAB_fil < 0.05:
    print('Rechazamos la hipótesis nula')
else:
    print('No podemos rechazar la hipótesis nula')
```

```
Valor p: 0.00629
Rechazamos la hipótesis nula
```

Igual que con los datos sin filtrar se ratifica que la diferencia en la conversión entre los grupos son estadísticamente significativos.

[39]:
```
# Diferencia relativa entre las tasas de conversión de ambos grupos utilizando␣
 ↪datos filtrados
```

```
print("{:.3f}".format(sample_filterB.mean()/sample_filterA.mean()-1))
```

```
0.192
```

En los datos procesados la ganancia de conversión relativa del grupo B en comparación con el grupo A es del 19.2%, aun más alto que en los datos no filtrados.

[52]:
```
print('Promedio de la muestra del grupo A:', sample_filterA.mean())
print('Promedio de la muestra del grupo B:', sample_filterB.mean())
```

```
Promedio de la muestra del grupo A: 0.023819696646015808
Promedio de la muestra del grupo B: 0.028399174996033633
```

Sin embargo, si se observan en términos de valores absolutos, la diferencia en la tasa de conversión entre ambos grupos es poco menos de 0.5%.

1.4.12 Significancia estadística de la diferencia en el tamaño promedio de pedido entre los grupos utilizando los datos filtrados

[40]:
```
# Prueba de Mann-Whitney para el tamaño promedio de pedido entre grupos␣
 ↪filtrados
print('Valor p:','{0:.3f}'.format(st.mannwhitneyu(df_orders_us[np.
 ↪logical_and(df_orders_us['group'] == 'A',
                                          np.
 ↪logical_not(df_orders_us['visitorId'].isin(abnormal_users)))]['revenue'],
                                      df_orders_us[np.
 ↪logical_and(df_orders_us['group'] == 'B',
                                          np.
 ↪logical_not(df_orders_us['visitorId'].
 ↪isin(abnormal_users)))]['revenue'])[1]))
```

```
Valor p: 0.877
```

[41]:
```
# Ganancia relativa del grupo B con respecto al grupo A
print('{0:.3f}'.format(df_orders_us[np.logical_and(df_orders_us['group'] == 'B',
                                    np.logical_not(df_orders_us['visitorId'].
 ↪isin(abnormal_users)))]['revenue'].mean() /
                        df_orders_us[np.logical_and(df_orders_us['group']=='A',␣
 ↪np.logical_not(df_orders_us['visitorId'].isin(abnormal_users)))]['revenue'].
 ↪mean() - 1))
```

```
-0.014
```

Con la eliminación de los valores atípicos no se obtuvo un valor p demasiado distinto al anterior. Sin embargo, la ganancia relativa se invierte y es el grupo A el que resulta con un más alto que el B, un 1.4% específicamente. Este valor difiere por mucho con el resultado de los datos no procesados. Podríamos probar un filtro más amplio para las ganancias (descartando el 5% de los datos) y comprobar.

[42]:
```
# Filtrar usuarios con pedidos mayores a 414
users_expensive_orders_2 = df_orders_us.query("revenue > 414")['visitorId']

abnormal_users_2 = pd.concat([users_many_orders, users_expensive_orders_2],␣
 ↪axis = 0).drop_duplicates().sort_values()
```

[43]:
```
# Prueba de Mann-Whitney para el tamaño promedio de pedido entre grupos␣
 ↪filtrados
print('Valor p:','{0:.3f}'.format(st.mannwhitneyu(df_orders_us[np.
 ↪logical_and(df_orders_us['group'] == 'A',
                                            np.
 ↪logical_not(df_orders_us['visitorId'].isin(abnormal_users_2)))]['revenue'],
                                       df_orders_us[np.
 ↪logical_and(df_orders_us['group'] == 'B',
                                            np.
 ↪logical_not(df_orders_us['visitorId'].
 ↪isin(abnormal_users_2)))]['revenue'])[1]))

# Ganancia relativa del grupo B con respecto al grupo A usando un filtro␣
 ↪alternativo
print('{0:.3f}'.format(df_orders_us[np.logical_and(df_orders_us['group'] == 'B',
                                    np.logical_not(df_orders_us['visitorId'].
 ↪isin(abnormal_users_2)))]['revenue'].mean() /
                        df_orders_us[np.logical_and(df_orders_us['group']=='A',␣
 ↪np.logical_not(df_orders_us['visitorId'].
 ↪isin(abnormal_users_2)))]['revenue'].mean() - 1))
```

```
Valor p: 0.680
-0.047
```

Aun eliminando el 5% de pedidos más caros la diferencia estadística sigue sin ser significativa y además los resultados se invierten todavía más en favor del grupo A.

1.4.13 Decisión basada en los resultados de la prueba

Si bien hay una diferencia estadísticamente significativa en la conversión entre los grupos, que favorece al grupo B, y según los datos en bruto y filtrados, a efectos prácticos no tiene importancia. En ganancias se conluye que no se encuentran diferencias significativas entre los grupos. Las diferencias en tamaños de pedido por grupo se explican por la presencia de valores atípicos.

1.4.14 Comentario Final del Revisor

Comentario del revisor

¡Felicidades! Has realizado un análisis sólido del test A/B, con un enfoque claro en la depuración de los datos y la priorización de hipótesis. El proyecto **está aprobado**, pero te dejo algunas sugerencias para profundizar en tu análisis:

Puntos Positivos:

- **Depuración de datos:** Has implementado adecuadamente un proceso de limpieza y filtrado, asegurando la validez de los datos antes de realizar el análisis del test A/B.
- **Priorización de hipótesis:** El uso de los frameworks ICE y RICE está bien fundamentado, y has explicado claramente las diferencias entre ambos métodos.
- **Análisis detallado del test A/B:** Las tablas acumuladas, visualizaciones y pruebas de significancia estadística están bien implementadas, mostrando un buen entendimiento del análisis de resultados.

Áreas para Seguir Investigando:

- **Explicaciones adicionales:** Sería interesante profundizar en la justificación de los umbrales usados para el filtrado de valores atípicos y en cómo estos umbrales pueden afectar las conclusiones.
- **Impacto en decisiones comerciales:** Expande más sobre cómo los resultados de la prueba A/B podrían influir en las decisiones comerciales y cómo la empresa podría aplicar estos resultados para mejorar sus ingresos.

¡Sigue así, tu trabajo es excelente!