Содержание

1.Основа проекта

Этот проект основан на данных, предоставленных на https://www.kaggle.com/datasets/faviovaz/marketing-ab-testing?datasetId=1660669. Это простая маркетинговая кампания с экспериментальной и контрольной группами для A/B-тестирования.

****1.1 О dataset-е****

Создание A/B тестирования для dataset-а Маркетинговые компании стремятся проводить успешные кампании, но рынок сложен, и существует несколько вариантов, которые могут сработать. Поэтому обычно они проводят A/B-тестирование, которое представляет собой процесс случайного эксперимента, при котором две или более версии переменной (веб-страница, элемент страницы, баннер и т. д.) одновременно показываются разным сегментам людей для определения, какая версия оказывает максимальное воздействие и способствует достижению бизнес-метрик.

Компании интересуются ответом на два вопроса:

1. Будет ли кампания успешной?
2. Если кампания будет успешной, насколько этот успех можно связать с рекламой?

С учетом второго вопроса мы обычно проводим A/B-тестирование. Большинство людей будут видеть рекламу (экспериментальная группа), а небольшая часть людей (контрольная группа) увидит общественное информирование или ничего в точно таком же размере и месте, где обычно размещается реклама.

Идея этого набора данных заключается в анализе групп, определении успешности рекламы, оценке потенциальной прибыли от рекламы и выяснении, является ли разница между группами статистически значимой.

1.2 Словарь:

- **Index**: Индекс строки
- **user id**: Идентификатор пользователя (уникальный)

- **test group**: Если "ad", то человек видел рекламу, если "psa", то он видел общественное информационное объявление
- **converted**: Если человек купил продукт, то значение True, иначе False
- **total ads**: Количество просмотренных рекламных объявлений человеком
- **most ads day**: День, в который человек увидел наибольшее количество рекламы
- **most ads hour**: Час дня, когда человек увидел наибольшее количество рекламы

```
import numpy as np  # линейная алгебра
import pandas as pd  # обработка данных, чтение и запись файлов CSV

# Входные данные находятся в только для чтения директории "../input/"
# Например, запуск этого кода (нажатие кнопки "Run" или нажатие
Shift+Enter) покажет все файлы в директории input

import os
for dirname, _, filenames in os.walk('/kaggle/input'):
    for filename in filenames:
        print(os.path.join(dirname, filename))

# Можно записывать до 20 ГБ в текущую директорию (/kaggle/working/),
которая сохраняется в виде выходных данных при создании версии через
"Save & Run All"
# Также можно создавать временные файлы в /kaggle/temp/, но они не
будут сохранены после завершения текущей сессии
```

```
/kaggle/input/marketing-ab-testing/marketing_AB.csv
```

2. Подготовка данных

2.1 Загрузка данных

```
# импортировать библиотеки и данные
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np

df =
pd.read_csv('/kaggle/input/marketing-ab-testing/marketing_AB.csv')  #
загрузить данные из CSV файла
df.head()  # вывод первых нескольких строк данных для ознакомления
```

```
   Unnamed: 0  user id test group  converted  total ads most ads
day  \
0           0  1069124         ad      False        130       Monday

1           1  1119715         ad      False         93      Tuesday

2           2  1144181         ad      False         21      Tuesday
```

```
3           3  1435133         ad      False        355      Tuesday

4           4  1015700         ad      False        276       Friday


   most ads hour
0             20
1             22
2             18
3             10
4             14
```

2.2 Чистка данных

После просмотра первых пяти строк таблицы данных мы обнаружили **один лишний столбец** с названием: Unnamed:0. Этот столбец нужно удалить.

1.Основа проекта

```
df.drop('Unnamed: 0', axis=1, inplace=True)
df.head()

   user id test group  converted  total ads most ads day  most ads
hour
0  1069124         ad      False        130       Monday
20
1  1119715         ad      False         93      Tuesday
22
2  1144181         ad      False         21      Tuesday
18
3  1435133         ad      False        355      Tuesday
10
4  1015700         ad      False        276       Friday
14
```

После удаления столбца "Unnamed:0", текущие названия столбцов **содержат пробелы между словами**, что может вызвать проблемы в дальнейшем. Чтобы избежать этих проблем, лучше переименовать столбец 'user id' в формат 'user_id'. Поскольку требуется изменить большинство названий столбцов, я буду использовать лямбда-функцию.

```
df.rename(columns=lambda x: x.replace(' ', '_'), inplace=True)
df.head()

   user_id test_group  converted  total_ads most_ads_day
most_ads_hour
0  1069124         ad      False        130       Monday
20
1  1119715         ad      False         93      Tuesday
22
2  1144181         ad      False         21      Tuesday
```

```
18
3  1435133          ad      False         355      Tuesday
10
4  1015700          ad      False         276       Friday
14
```

2.3 Data Exploration

Теперь датафрейм выглядит хорошо, поэтому можно проверить, есть ли пропущенные значения.

```
df.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 588101 entries, 0 to 588100
Data columns (total 6 columns):
 #   Column          Non-Null Count   Dtype 
---  ------          --------------   ----- 
 0   user_id         588101 non-null  int64 
 1   test_group      588101 non-null  object
 2   converted       588101 non-null  bool  
 3   total_ads       588101 non-null  int64 
 4   most_ads_day    588101 non-null  object
 5   most_ads_hour   588101 non-null  int64 
dtypes: bool(1), int64(3), object(2)
memory usage: 23.0+ MB
```

К счастью, ни в одном из наших столбцов нет пропущенных значений. Вернемся к цели анализа этого проекта: мы хотим узнать, значительно ли запуск рекламы улучшает конверсию. Первое, с чего следует начать, это взглянуть на **размер выборки экспериментальной группы (ad) и контрольной группы (psa).**

```
# count the numbers of different groups of the categorical column
df['test_group'].value_counts()

ad     564577
psa     23524
Name: test_group, dtype: int64

# count the True or False of buying products by grouping test_group.
df.groupby('test_group')['converted'].value_counts()

test_group  converted
ad          False        550154
            True          14423
psa         False         23104
            True            420
Name: converted, dtype: int64
```

Clearly, the sample size of ad is **so much greater than** the sample size of psa. When the sample sizes of the experimental group and control group are **imbalanced**, it can potentially introduce **certain issues** or considerations in statistical analysis, such as affecting the statistical power of the analysis and the precision of estimates. For example, the precision of estimates, such as means or proportions, can be influenced by imbalanced sample sizes. The group with a larger sample size will generally have more precise estimates compared to the group with a smaller sample size. Therefore, in this case, we consider strategies such as matching, stratification, or using appropriate statistical techniques that can account for imbalanced sample sizes, such as weighted analyses or **resampling methods like bootstrapping.**

Before doing bootstrapping, we can first **subset the original dataframe** into experimental group dataframe and control group dataframe.

```
#subset the original dataframe
ad_experimental=df[df['test_group']=='ad']
psa_control=df[df['test_group']=='psa']

alpha_90 = 0.1
alpha_99 = 0.01

#find the average converted rate of each group
ad_converted=np.percentile(ad_experimental['converted'],100 * alpha_90
/ 2)
psa_converted=np.percentile(psa_control['converted'],100* alpha_90/2)

print(ad_converted,psa_converted)
```

```
0.025546559636683747 0.01785410644448223
```

Данный интервал является результатом наивного подхода. Однако, далее, при помощи метода бутстрэп, я буду стремиться улучшить его точность.

На основе имеющихся выборочных данных, средний уровень конверсии рекламы (2.55%) на 0.76% выше, чем уровень конверсии объявления PSA (1.79%). Кажется, что запуск рекламы успешно улучшает уровень конверсии. Однако, является ли это результатом значимым для более широкой аудитории? Могут ли эти числа быть повлияны большим размером выборки рекламы? Чтобы ответить на эти вопросы, нам необходимо провести статистические тесты на значимость (z-тест или t-тест) на основе данных с сбалансированными размерами выборок (bootstrap).

3. Бутстреп

Метод бутстреп — это **метод повторной выборки**, используемый в статистическом анализе. По сути, он включает в себя повторную выборку наблюдений из набора данных с заменой для создания нескольких повторных выборок. Этот процесс повторной выборки позволяет **оценить выборочное распределение статистики или сделать выводы о параметрах совокупности**. В этом проекте я собираюсь выполнить повторную выборку исходного файла данных, чтобы создать **кадры данных с повторной выборкой из 1000 выборочных средних** как для экспериментальной группы, так и для контрольной группы.

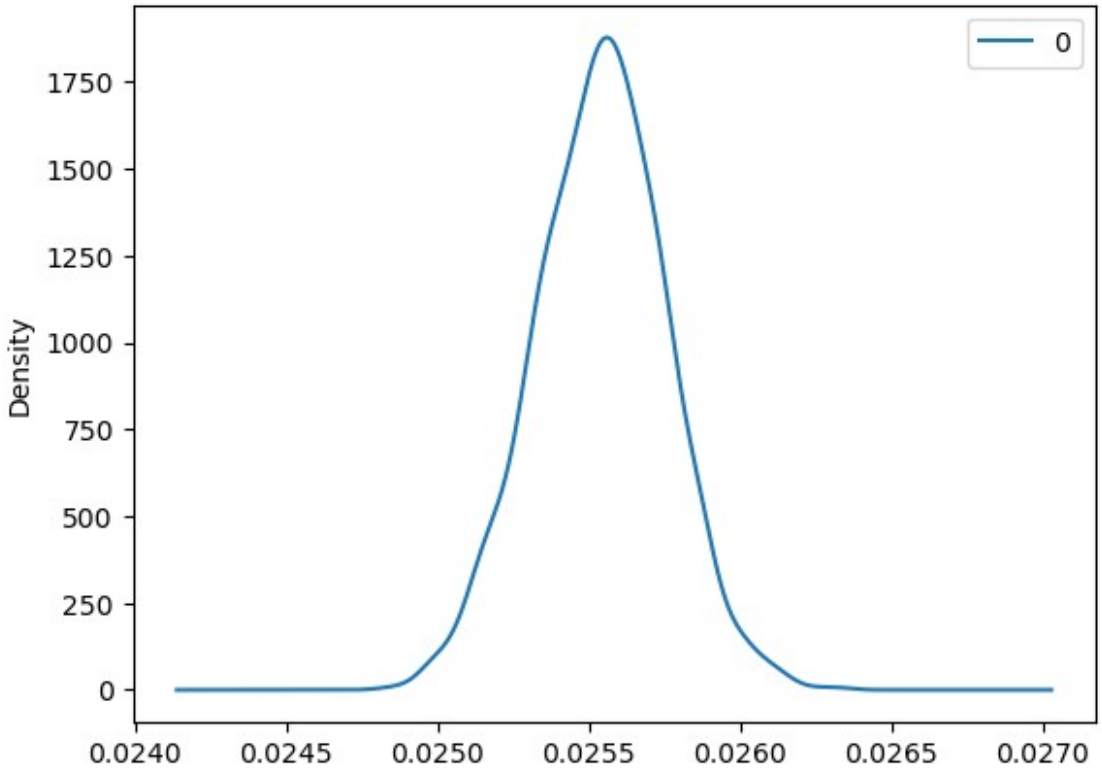
```
#создаем пустой список для хранения загрузочных средств
boot_ad=[]

#Настройте цикл, который будет повторяться 1000 раз. На каждой
итерации будет генерироваться новый загрузочный образец.
for i in range (1000):
    boot_mean=ad_experimental.sample(frac=1,replace=True)
['converted'].mean()
    boot_ad.append(boot_mean)


boot_ad=pd.DataFrame(boot_ad)

#Создаем график плотности загрузочных средств
boot_ad.plot(kind='density')


<Axes: ylabel='Density'>
```



На графике плотности мы ясно видим, что выборочное распределение среднего значения (полученное в процессе бутстреп-перевыборки) является **приблизительно нормальным**. Это позволяет применять статистические тесты и доверительные интервалы, основанные на предположении о нормальности. Например, мы можем выполнить проверку гипотез или построить доверительные интервалы, используя такие методы, как **t-тест или z-тест, основанные на предположении о нормальности.**

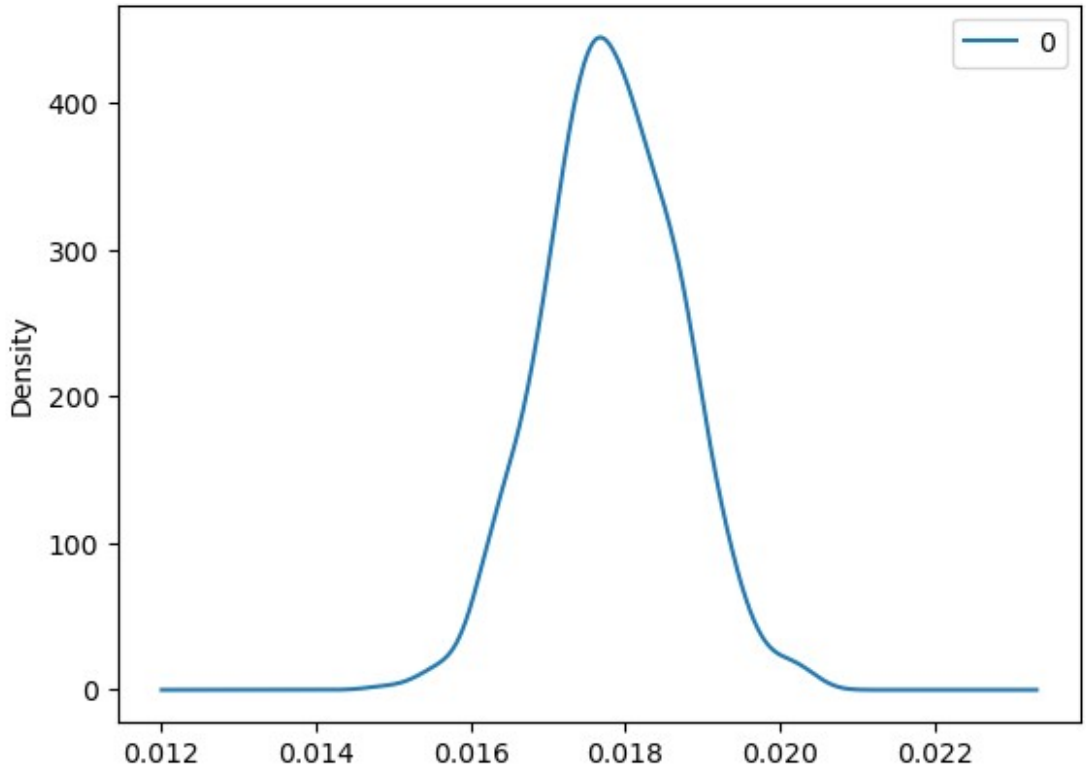
Мы **повторим процесс бутстреп** для группы psa.

```
boot_psa=[]

for i in range(1000):
    boot_mean=psa_control.sample(frac=1,replace=True)
['converted'].mean()
    boot_psa.append(boot_mean)

boot_psa=pd.DataFrame(boot_psa)
boot_psa.plot(kind='density')
```

```
<Axes: ylabel='Density'>
```



Среднее значение начальной загрузки группы psa также **следует нормальному распределению**. Теперь мы **уверены, что сможем провести проверку гипотезы**, запустив z-тест или t-тест на основе только что полученных данных начальной загрузки.

Прежде чем приступить к дальнейшему анализу, нам нужно сначала объединить бутстреп данные группы ad и psa.

```
#Name the column
boot_ad.columns = ['ad_converted']
boot_psa.columns=['psa_converted']
boot_psa.head()
```

```
   psa_converted
0       0.017557
1       0.019257
2       0.018407
3       0.017684
4       0.017259
```

```
#concat two bootstrap dataframes into boot_strap
boot_strap=pd.concat([boot_ad,boot_psa],axis=1)
boot_strap.head(10)
boot_strap.info()
```

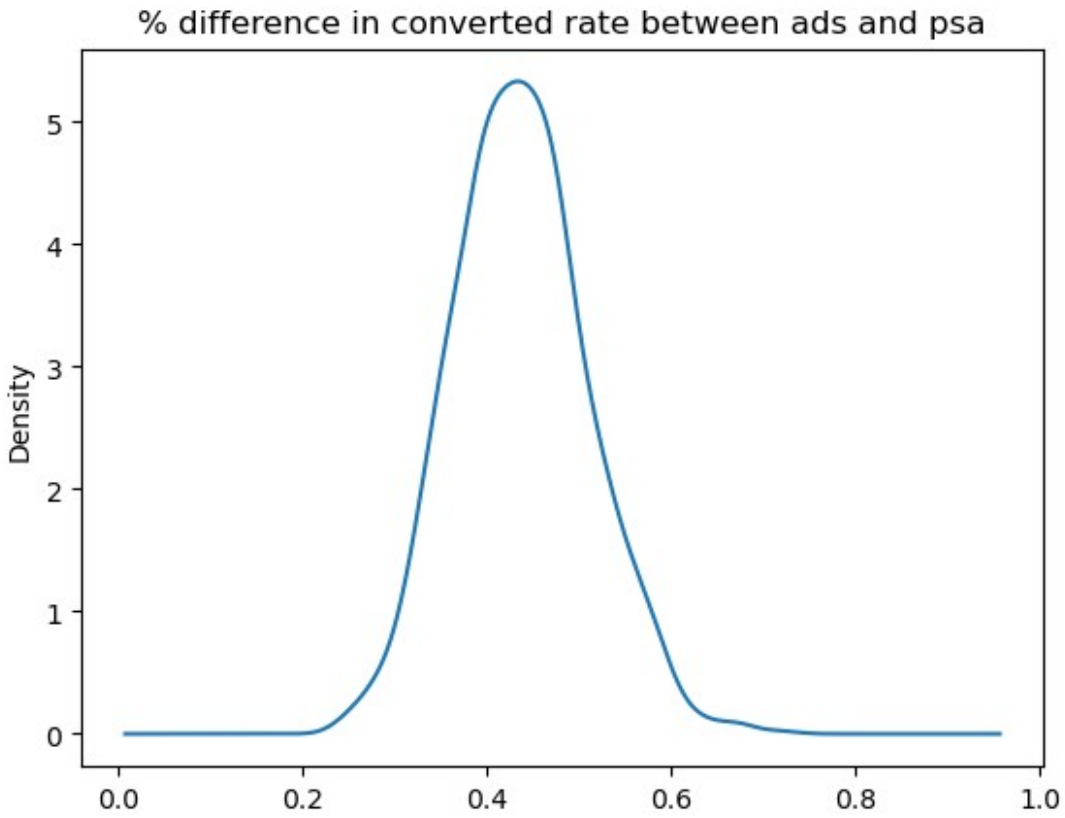
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1000 entries, 0 to 999
Data columns (total 2 columns):
 #   Column         Non-Null Count  Dtype  
---  ------         --------------  -----  
 0   ad_converted   1000 non-null   float64
 1   psa_converted  1000 non-null   float64
dtypes: float64(2)
memory usage: 15.8 KB
```

```
#создаем столбец различий, вычисляя разницу между ad_converted и
psa_converted
boot_strap['diff']=(boot_strap['ad_converted']-
boot_strap['psa_converted'])/boot_strap['psa_converted']
boot_strap.head(10)
```

```
   ad_converted  psa_converted      diff
0      0.025338       0.017557  0.443198
1      0.025989       0.019257  0.349611
2      0.025465       0.018407  0.383466
3      0.025701       0.017684  0.453323
4      0.025603       0.017259  0.483474
5      0.025499       0.018747  0.360164
6      0.025678       0.017216  0.491458
7      0.025616       0.017599  0.455513
8      0.026016       0.016409  0.585489
9      0.025382       0.016536  0.534916
```

```
#постройте разницу и проверьте распределение
ax=boot_strap['diff'].plot(kind='density')
ax.set_title('% difference in converted rate between ads and psa')
```

```
Text(0.5, 1.0, '% difference in converted rate between ads and psa')
```




```
#Найдите вероятность того, что коэффициент конверсии рекламы превысит PSA.
(boot_strap['diff']>0).mean()
```

```
1.0
```

Выражение (boot_strap['diff'] > 0).mean() вычисляет долю выборок бутстрепа, в которых разница в конвертированном курсе между рекламой и PSA больше нуля. В данном случае я получил значение 1,0, что означает, что **во всех выборках бутсреп коэффициент конвертации рекламы был выше, чем у psa.**

Однако, несмотря на то, что доля образцов бутстрепа с положительной разницей (1,0) предполагает устойчивую картину группы объявлений с более высоким коэффициентом конверсии, **это не обеспечивает формальную проверку гипотезы или меру статистической значимости.** Чтобы установить статистическую значимость и количественно оценить уровень достоверности, мы можем приступить к использованию z-теста или t-теста.

Ниже можно продолжить анализ и протестировать гипотизы. Но в курсвой нас волнуют задачи поиска доверительных интевалов. Решение такой задачи для средних значений и тестрование гипотиз можно найти по адрессу (https://www.kaggle.com/code/kouyuyang/a-b-testing-bootstrap-hypothesis-test)