分析A/B测试结果

目录

简介

对于这个项目，你将要了解的是电子商务网站运行的 A/B 测试的结果。你的目标是通过这个 notebook 来帮助公司弄清楚他们是否应该使用新的页面，保留旧的页面，或者应该将测试时间延长，之后再做出决定。

I - 概率

让我们先导入库，然后开始你的任务吧。

In [2]:

```
import pandas as pd
import numpy as np
import random
import matplotlib.pyplot as plt
%matplotlib inline
#We are setting the seed to assure you get the same answers on quizzes as we set up
random.seed(42)
```

1. 现在，导入 ab_data.csv 数据，并将其存储在 df 中。

a. 导入数据集，并在这里查看前几行：

In [3]:

```
df = pd.read_csv('./ab_data.csv')
df.head()
```

Out[3]:

	user_id	timestamp	group	landing_page	converted
0	851104	2017-01-21 22:11:48.556739	control	old_page	0
1	804228	2017-01-12 08:01:45.159739	control	old_page	0
2	661590	2017-01-11 16:55:06.154213	treatment	new_page	0
3	853541	2017-01-08 18:28:03.143765	treatment	new_page	0
4	864975	2017-01-21 01:52:26.210827	control	old_page	1

b. 使用下面的单元格来查找数据集中的行数。

In [4]:

```
df.shape[0]
```

Out[4]:

294478

c. 数据集中独立用户的数量。

In [5]:

```
df.user_id.nunique()
```

Out[5]:

290584

d. 用户转化的比例。

In [6]:

```
df.converted.mean()
```

Out[6]:

0.11965919355605512

e. new_page 与 treatment 不一致的次数。

In [7]:

```
len(df.query('group == "treatment"').query('landing_page != "new_page"')) + len(df.query('group
 != "treatment"').query('landing_page == "new_page"'))
```

Out[7]:

3893

f. 是否有任何行存在缺失值?

In [8]:

```
df.isnull().any().any()
```

Out[8]:

False

2. 对于 **treatment** 不与 **new_page** 一致的行或 **control** 不与 **old_page** 一致的行，我们不能确定该行是否真正接收到了新的或旧的页面。

a. 将新 dataframe 存储在 **df2** 中。

In [9]:

```
df2 = df.query('group == "treatment"').query('landing_page == "new_page"')
df_t = df.query('group == "control"').query('landing_page == "old_page"')
```

In [10]:

```
df2 = df2.append(df_t)
```

In [11]:

```
# Double Check all of the correct rows were removed - this should be 0
df2[((df2['group'] == 'treatment') == (df2['landing_page'] == 'new_page')) == False].shape[0]
```

Out[11]:

0

3. a. **df2** 中有多少唯一的 **user_id**?

In [12]:

```
df2.user_id.nunique()
```

Out[12]:

290584

b. **df2** 中有一个重复的 **user_id** 。它是什么?

In [13]:

```
df2[df2.user_id.duplicated()].user_id.values[0]
```

Out[13]:

773192

c. 这个重复的 **user_id** 的行信息是什么?

In [14]:

```
df2[df2.user_id.duplicated()]
```

Out[14]:

	user_id	timestamp	group	landing_page	converted
2893	773192	2017-01-14 02:55:59.590927	treatment	new_page	0

d. 删除 **一个** 含有重复的 **user_id** 的行， 但需要确保你的 dataframe 为 **df2**。

In [15]:

```
df2.drop_duplicates(subset='user_id', inplace=True)
df2[df2.user_id.duplicated()]
```

Out[15]:

	user_id	timestamp	group	landing_page	converted

4. 在下面的单元格中，a. 不管它们收到什么页面，单个用户的转化率是多少?

In [16]:

```
df2.converted.mean()
```

Out[16]:

0.11959708724499628

c. 假定一个用户处于 treatment 组中，他的转化率是多少?

In [17]:

```
df2.query('group == "treatment"').converted.mean()
```

Out[17]:

0.11880806551510564

d. 一个用户收到新页面的概率是多少?

In [18]:

```
df.query('landing_page == "new_page"').user_id.nunique() / df.user_id.nunique()
```

Out[18]:

0.5035273793464197

e. 使用这个问题的前两部分的结果，给出你的建议：你是否认为有证据表明一个页面可以带来更多的转化？在下面写出你的答案。

在这里写出你的答案。
用户处于 treatment 组中，他的转化率是11.9%，而 一个用户收到新页面的概率是50%，由此我不认为一个页面可以带来更多的转化。

II - A/B 测试

请注意，由于与每个事件相关的时间戳，你可以在进行每次观察时连续运行假设检验。

然而，问题的难点在于，一个页面被认为比另一页页面的效果好得多的时候你就要停止检验吗？还是需要在一定时间内持续发生？你需要将检验运行多长时间来决定哪个页面比另一个页面更好？

一般情况下，这些问题是A / B测试中最难的部分。如果你对下面提到的一些知识点比较生疏，请先回顾课程中的“描述统计学”部分的内容。

1. 现在，你要考虑的是，你需要根据提供的所有数据做出决定。如果你想假定旧的页面效果更好，除非新的页面在类型I错误率为5%的情况下才能证明效果更好，那么，你的零假设和备择假设是什么？ 你可以根据单词或旧页面与新页面的转化率 p_{old} 与 p_{new} 来陈述你的假设。

在这里给出你的答案。

$$H_0 : P_{old} \geq P_{new}$$

$$H_1 : P_{old} < P_{new}$$

2. 假定在零假设中，不管是新页面还是旧页面， p_{new} and p_{old} 都具有等于 **转化** 成功率的“真”成功率，也就是说， p_{new} 与 p_{old} 是相等的。此外，假设它们都等于**ab_data.csv** 中的 **转化** 率，新旧页面都是如此。

每个页面的样本大小要与 **ab_data.csv** 中的页面大小相同。

执行两次页面之间 **转化** 差异的抽样分布，计算零假设中10000次迭代计算的估计值。

使用下面的单元格提供这个模拟的必要内容。如果现在还没有完整的意义，不要担心，你将通过下面的问题来解决这个问题。你可以通过做课堂中的 **测试 5** 来确认你掌握了这部分内容。

a. 在零假设中，p_{new} 的 **convert rate（转化率）** 是多少？

In [19]:

```
p_new = df2.converted.mean()
p_new
```

Out[19]:

0.11959708724499628

b. 在零假设中， p_{old} 的 **convert rate（转化率）** 是多少？

In [20]:

```
p_old = df2.converted.mean()
p_old
```

Out[20]:

0.11959708724499628

c. n_{new} 是多少?

In [21]:

```
n_new = len(df2.query('landing_page == "new_page"'))
n_new
```

Out[21]:

145310

d. n_{old}?是多少?

In [22]:

```
n_old = len(df2.query('landing_page == "old_page"'))
n_old
```

Out[22]:

145274

e. 在零假设中，使用 p_{new} 转化率模拟 n_{new} 交易，并将这些 n_{new} 1's 与 0's 存储在 **new_page_converted** 中。(提示：可以使用 numpy.random.choice (https://docs.scipy.org/doc/numpy/reference/generated/numpy.random.choice.html)。)

In [23]:

```
new_page_converted = np.random.choice([0, 1], size=n_new, p=[1-p_new, p_new])
```

f. 在零假设中，使用 p_{old} 转化率模拟 n_{old} 交易，并将这些 n_{old} 1's 与 0's 存储在 **old_page_converted** 中。

In [24]:

```
old_page_converted = np.random.choice([0, 1], size=n_old, p=[1-p_old, p_old])
```

g. 在 (e) 与 (f)中找到 p_{new} - p_{old} 模拟值。

In [25]:

```
obs_diff = new_page_converted.mean() - old_page_converted.mean()
obs_diff
```

Out[25]:

0.00083048646206289323

h. 使用**a. 到 g.** 中的计算方法来模拟 10,000个 p_{new} - p_{old} 值，并将这 10,000 个值存储在 **p_diffs** 中。

In [26]:

```
### 每个页面的样本大小要与 ab_data.csv 中的页面大小相同。
### 执行两次页面之间 转化 差异的抽样分布，计算零假设中10000次迭代计算的估计值。

def p_diffs_fun():
    p_diffs = []
    for _ in range(10000):
        new_page_converted = np.random.choice([0, 1], size=n_new, p=[1-p_new, p_new])
        old_page_converted = np.random.choice([0, 1], size=n_old, p=[1-p_old, p_old])
        p_diffs.append(new_page_converted.mean() - old_page_converted.mean())
    return p_diffs
```

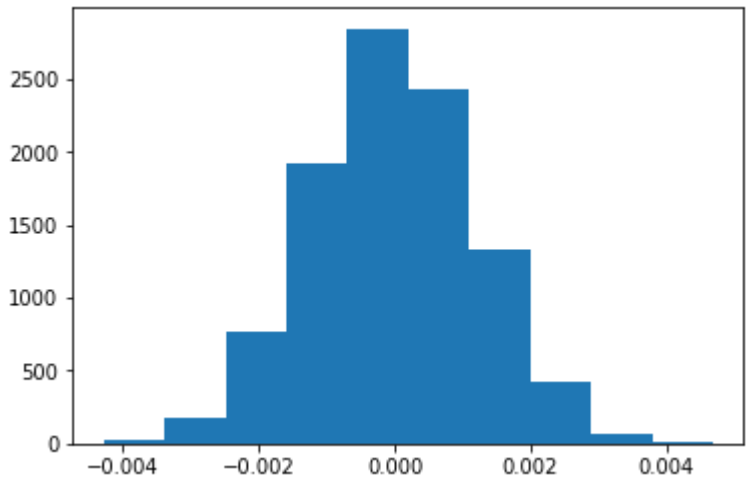
In [27]:

```
p_diffs = p_diffs_fun()
p_diffs = np.array(p_diffs)
```

i. 绘制一个 **p_diffs** 直方图。这个直方图看起来像你所期望的吗？通过回答课堂上的匹配问题，确保你完全理解这里计算出的内容。

In [28]:

```
plt.hist(p_diffs);
```



j. 在**p_diffs**列表的数值中，有多大比例大于 **ab_data.csv** 中观察到的实际差值？

In [29]:

```
new_p = df2.query('group == "treatment"').converted.mean()
old_p = df2.query('group == "control"').converted.mean()
ab_diff = new_p - old_p
```

In [30]:

```
(p_diffs > ab_diff).mean()
```

Out[30]:

0.90000000000000002

k. 用文字解释一下你刚才在 **j.**中计算出来的结果。在科学研究中，这个值是什么？ 这个值在新旧页面中是否有区别呢？

在这里给出你的答案。
这个值是p-value，很显然在p_diffs列表的数值中大于ab_data.csv 中观察到的实际差值大于新的页面在类型I错误率为5%的情况，所以旧页面更合适。

l. 我们也可以使用一个内置程序 （built-in）来实现类似的结果。尽管使用内置程序可能更易于编写代码，但上面的内容是对正确思考统计显著性至关重要的思想的一个预排。填写下面的内容来计算每个页面的转化次数，以及每个页面的访问人数。使用 `n_old` 与 `n_new` 分别引证与旧页面和新页面关联的行数。

In [31]:

```
import statsmodels.api as sm

convert_old = df2.query('group == "control"').converted.sum()
convert_new = df2.query('group == "treatment"').converted.sum()
n_old = len(df2.query('landing_page == "old_page"'))
n_new = len(df2.query('landing_page == "new_page"'))
```

```
D:\Python\Anaconda35\lib\site-packages\statsmodels\compat\pandas.py:56: FutureWarn
ing: The pandas.core.datetools module is deprecated and will be removed in a futur
e version. Please use the pandas.tseries module instead.
  from pandas.core import datetools
```

m. 现在使用 `stats.proportions_ztest` 来计算你的检验统计量与 p-值。这里 (http://knowledgetack.com/python/statsmodels/proportions_ztest/) 是使用内置程序的一个有用链接。

In [32]:

```
z_score, p_value = sm.stats.proportions_ztest([convert_old, convert_new], [n_old, n_new], altern
ative='smaller')
z_score, p_value
```

Out[32]:

```
(1.3109241984234394, 0.90505831275902449)
```

n. 在上一个问题中计算出的z-分和p-值是否意味着新旧页面的转化率？它们与 **j.** 与 **k.** 中的结果一致吗？

在这里给出你的答案。
在上一个问题中计算出的z-分和p-值是意味着新旧页面的转化率,但它们与 **j.** 与 **k.** 中的结果不一致

III - 回归分析法之一

1. 在最后一部分中，你会看到，你在之前的A / B测试中获得的结果也可以通过执行回归来获取。

a. 既然每行的值是转化或不转化，那么在这种情况下，我们应该执行哪种类型的回归？

在这里给出你的答案。
逻辑回归

b. 目标是使用 **statsmodels** 来拟合你在 **a.** 中指定的回归模型，以查看用户收到的不同页面是否存在显著的转化差异。但是，首先，你需要为这个截距创建一个列（ 原文：column），并为每个用户收到的页面创建一个虚拟变量列。添加一个 **截距** 列，一个 **ab_page** 列，当用户接收 **treatment** 时为1， **control** 时为0。

In [33]:

```
df2['intercept'] = 1.
df2['ab_page'] = pd.get_dummies(df2.group)['treatment']
```

In [34]:

```
df2.head()
```

Out[34]:

	user_id	timestamp	group	landing_page	converted	intercept	ab_page
2	661590	2017-01-11 16:55:06.154213	treatment	new_page	0	1.0	1
3	853541	2017-01-08 18:28:03.143765	treatment	new_page	0	1.0	1
6	679687	2017-01-19 03:26:46.940749	treatment	new_page	1	1.0	1
8	817355	2017-01-04 17:58:08.979471	treatment	new_page	1	1.0	1
9	839785	2017-01-15 18:11:06.610965	treatment	new_page	1	1.0	1

c. 使用 **statsmodels** 导入你的回归模型。 实例化该模型，并使用你在 **b.** 中创建的2个列来拟合该模型，用来预测一个用户是否会发生转化。

In [35]:

```
lgm = sm.Logit(df2['converted'], df2[['intercept', 'ab_page']])
results = lgm.fit()
results.summary()
```

```
Optimization terminated successfully.
         Current function value: 0.366118
         Iterations 6
```

Out[35]:

Logit Regression Results

Dep. Variable:	converted	**No. Observations:**	290584
Model:	Logit	**Df Residuals:**	290582
Method:	MLE	**Df Model:**	1
Date:	Wed, 30 May 2018	**Pseudo R-squ.:**	8.077e-06
Time:	10:15:30	**Log-Likelihood:**	-1.0639e+05
converged:	True	**LL-Null:**	-1.0639e+05
		LLR p-value:	0.1899

	coef	**std err**	**z**	**P>|z|**	**[0.025**	**0.975]**
intercept	-1.9888	0.008	-246.669	0.000	-2.005	-1.973
ab_page	-0.0150	0.011	-1.311	0.190	-0.037	0.007

d. 请在下方提供你的模型摘要，并根据需要使用它来回答下面的问题。

In [36]:

```
1 / np.exp(results.params.ab_page)
```

Out[36]:

```
1.0151020136964775
```

因为ab_page的系数小于1，所以组成员在其他变量不变的情况下每减少一个单位，它的转换率为1.0151倍
又由于ab_page变量对应的 p 值大于 0.05 的通常 alpha 水平，意味着不能拒绝斜率为 0 的零假设，即差异不具有统计学显著性

e. 与 **ab_page** 关联的 p-值是多少？ 为什么它与你在 **II** 中发现的结果不同？

提示: 与你的回归模型相关的零假设与备择假设分别是什么？它们如何与 **Part II** 中的零假设和备择假设做比较？

在这里给出你的答案。
逻辑回归模型ab*page的 p-值是： 0.1899，**Part II**中比较的是新页面和旧页面的差异即$ P*{old} < P_{new}$所以是单尾检验，而**Part III**中回归模型是双尾检验

f. 现在，你一定在考虑其他可能影响用户是否发生转化的因素。讨论为什么考虑将其他因素添加到回归模型中是一个不错的主意。在回归模型中添加附加项有什么弊端吗？

在这里给出你的答案。
在回归模型中添加附加项会有多重共线性问题出现

g. 现在，除了测试不同页面的转化率是否会发生变化之外，还要根据用户居住的国家或地区添加一个 effect 项。你需要导入 **countries.csv** 数据集，并将数据集合并在适当的行上。 这里 (https://pandas.pydata.org/pandas-docs/stable/generated/pandas.DataFrame.join.html) 是链接表格的文档。

这个国家项对转化有影响吗？不要忘记为这些国家的列创建虚拟变量—— **提示: 你将需要为这三个虚拟变量增加两列。** 提供统计输出，并书面回答这个问题。

In [37]:

```
df_c = pd.read_csv('./countries.csv')
```

In [38]:

```
df_c.head()
```

Out[38]:

	user_id	country
0	834778	UK
1	928468	US
2	822059	UK
3	711597	UK
4	710616	UK

In [39]:

```
df_new = df2.merge(df_c, on='user_id', how='left')
df_new.head()
```

Out[39]:

	user_id	timestamp	group	landing_page	converted	intercept	ab_page	c
0	661590	2017-01-11 16:55:06.154213	treatment	new_page	0	1.0	1	U
1	853541	2017-01-08 18:28:03.143765	treatment	new_page	0	1.0	1	U
2	679687	2017-01-19 03:26:46.940749	treatment	new_page	1	1.0	1	C
3	817355	2017-01-04 17:58:08.979471	treatment	new_page	1	1.0	1	U
4	839785	2017-01-15 18:11:06.610965	treatment	new_page	1	1.0	1	C

In [40]:

```
df_new[['CA', 'UK']] = pd.get_dummies(df_new.country)[['CA', 'UK']]
df_new.head()
```

Out[40]:

	user_id	timestamp	group	landing_page	converted	intercept	ab_page
0	661590	2017-01-11 16:55:06.154213	treatment	new_page	0	1.0	1
1	853541	2017-01-08 18:28:03.143765	treatment	new_page	0	1.0	1
2	679687	2017-01-19 03:26:46.940749	treatment	new_page	1	1.0	1
3	817355	2017-01-04 17:58:08.979471	treatment	new_page	1	1.0	1
4	839785	2017-01-15 18:11:06.610965	treatment	new_page	1	1.0	1

In [41]:

```
lgm2 = sm.Logit(df_new.converted, df_new[['intercept', 'CA', 'UK']])
results2 = lgm2.fit()
results2.summary()
```

```
Optimization terminated successfully.
         Current function value: 0.366116
         Iterations 6
```

Out[41]:

Logit Regression Results

Dep. Variable:	converted	**No. Observations:**	290584
Model:	Logit	**Df Residuals:**	290581
Method:	MLE	**Df Model:**	2
Date:	Wed, 30 May 2018	**Pseudo R-squ.:**	1.521e-05
Time:	10:15:33	**Log-Likelihood:**	-1.0639e+05
converged:	True	**LL-Null:**	-1.0639e+05
		LLR p-value:	0.1984

	coef	**std err**	**z**	**P>|z|**	**[0.025**	**0.975]**
intercept	-1.9967	0.007	-292.314	0.000	-2.010	-1.983
CA	-0.0408	0.027	-1.518	0.129	-0.093	0.012
UK	0.0099	0.013	0.746	0.456	-0.016	0.036

由于变量(CA, UK)对应的 p 值均大于 0.05 的通常 alpha 水平，意味着不能拒绝斜率为 0 的零假设，即差异不具有统计学显著性

h. 虽然你现在已经查看了国家与页面在转化率上的个体性因素，但现在我们要查看页面与国家/地区之间的相互作用，测试其是否会对转化产生重大影响。创建必要的附加列，并拟合一个新的模型。

提供你的摘要结果，以及根据结果得出的结论。

提示：页面与国家/地区的相互作用

```
df3['new_CA'] = df3['new_page'] * df3['CA']
df3['new_UK'] = df3['new_page'] * df3['UK']
```

In [42]:

```
df_new['new_page'] = pd.get_dummies(df_new.landing_page)['new_page']
```

In [43]:

```
df_new.head()
```

Out[43]:

	user_id	timestamp	group	landing_page	converted	intercept	ab_page	c
0	661590	2017-01-11 16:55:06.154213	treatment	new_page	0	1.0	1	U
1	853541	2017-01-08 18:28:03.143765	treatment	new_page	0	1.0	1	U
2	679687	2017-01-19 03:26:46.940749	treatment	new_page	1	1.0	1	C
3	817355	2017-01-04 17:58:08.979471	treatment	new_page	1	1.0	1	U
4	839785	2017-01-15 18:11:06.610965	treatment	new_page	1	1.0	1	C

In [44]:

```
df_new['new_CA'] = df_new['new_page'] * df_new['CA']
df_new['new_UK'] = df_new['new_page'] * df_new['UK']
```

In [45]:

```
lgm3 = sm.Logit(df_new.converted, df_new[['intercept', 'new_page', 'CA', 'UK', 'new_CA', 'new_UK']])
results3 = lgm3.fit()
results3.summary()
```

```
Optimization terminated successfully.
         Current function value: 0.366109
         Iterations 6
```

Out[45]:

Logit Regression Results

Dep. Variable:	converted	**No. Observations:**	290584
Model:	Logit	**Df Residuals:**	290578
Method:	MLE	**Df Model:**	5
Date:	Wed, 30 May 2018	**Pseudo R-squ.:**	3.482e-05
Time:	10:15:35	**Log-Likelihood:**	-1.0639e+05
converged:	True	**LL-Null:**	-1.0639e+05
		LLR p-value:	0.1920

	coef	**std err**	**z**	**P>|z|**	**[0.025**	**0.975]**
intercept	-1.9865	0.010	-206.344	0.000	-2.005	-1.968
new_page	-0.0206	0.014	-1.505	0.132	-0.047	0.006
CA	-0.0175	0.038	-0.465	0.642	-0.091	0.056
UK	-0.0057	0.019	-0.306	0.760	-0.043	0.031
new_CA	-0.0469	0.054	-0.872	0.383	-0.152	0.059
new_UK	0.0314	0.027	1.181	0.238	-0.021	0.084

由于变量(new_page, CA, UK, new_CA, new_UK)对应的 p 值均大于 0.05 的通常 alpha 水平，意味着不能拒绝斜率为 0 的零假设，即差异不具有统计学显著性