

# Analyzing difference in means A/B tests

A/B TESTING IN PYTHON



**Moe Lotfy, PhD**

Principal Data Science Manager

# Framework for difference in means

- Calculate required sample size
- Run experiment and perform sanity checks

$$H_0 : \mu_B - \mu_A = 0$$

$$H_1 : \mu_B - \mu_A \neq 0$$

- Calculate the metrics per variant
- Analyze the difference using t-test

- If p-value  $< \alpha$ 
  - Reject Null hypothesis
- If p-value  $> \alpha$ 
  - Fail to reject Null hypothesis

```
checkout.groupby('checkout_page')['time_on_page'].mean()
```

```
checkout_page
A    44.668527
B    42.723772
C    42.223772
```

# Pingouin t-test

```
checkout.groupby('checkout_page')['time_on_page'].mean()
```

```
checkout_page
A      44.668527
B      42.723772
C      42.223772
```

```
ttest = pingouin.ttest(x=checkout[checkout['checkout_page']=='C']['time_on_page'],
                       y=checkout[checkout['checkout_page']=='B']['time_on_page'],
                       paired=False,
                       alternative="two-sided")

print(ttest)
```

	T	dof	alternative	p-val	CI95%	cohen-d	BF10	power
T-test	-1.995423	5998	two-sided	0.046042	[-0.99, -0.01]	0.051522	0.212	0.514054

# Pingouin pairwise

```
pairwise = pingouin.pairwise_tests(data = checkout,
                                   dv = "time_on_page",
                                   between = "checkout_page",
                                   padjust = "bonf")

print(pairwise)
```

	Contrast	A	B	Paired	Parametric	T	dof	alternative	\
0	checkout_page	A	B	False	True	7.026673	5998.0	two-sided	
1	checkout_page	A	C	False	True	8.833244	5998.0	two-sided	
2	checkout_page	B	C	False	True	1.995423	5998.0	two-sided	

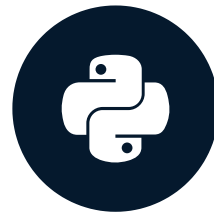
	p-unc	p-corr	p-adjust	BF10	hedges
0	2.349604e-12	7.048812e-12	bonf	1.305e+09	0.181405
1	1.316118e-18	3.948354e-18	bonf	1.811e+15	0.228045
2	4.604195e-02	1.381258e-01	bonf	0.212	0.051515

# Let's practice!

A/B TESTING IN PYTHON

# Non-parametric statistical tests

A/B TESTING IN PYTHON



**Moe Lotfy, PhD**

Principal Data Science Manager

# Parametric tests assumptions

## 1. Random sampling

- Data is randomly sampled from the population.
- Investigate the data collection/sampling process.

## 2. Independence

- Each observation/data point is independent.
- Not accounting for dependencies inflates error rates.

## 3. Normality

- Normally distributed data.
- Large "enough" sample size.
  - Two sample t-test  $n \geq 30$  in each group.
  - Two sample proportions test:  $\geq 10$  successes and  $\geq 10$  failures in each group.

# Mann-Whitney U test

- Non-parametric test for statistical significance
- Determines if two independent samples have the same parent distribution
- Rank sum test
- Unpaired data



# Mann-Whitney U test in python

```
# Calculate the mean and count of time on page by variant
print(checkout.groupby('checkout_page')['time_on_page'].agg({'mean', 'count'}))
```

	mean	count
checkout_page		
A	44.668527	3000
B	42.723772	3000
C	42.223772	3000

```
# Set random seed for repeatability
np.random.seed(40)

# Take a random sample of size 25 from each variant
ToP_samp_A = checkout[checkout['checkout_page'] == 'A'].sample(25)['time_on_page']
ToP_samp_B = checkout[checkout['checkout_page'] == 'B'].sample(25)['time_on_page']
```

# Mann-Whitney U test in python

```
# Run a Mann-Whitney U test
mwu_test = pingouin.mwu(x=ToP_samp_A,
                        y=ToP_samp_B,
                        alternative='two-sided')

# Print the test results
print(mwu_test)
```

	U-val	alternative	p-val	RBC	CLES
MWU	441.0	two-sided	0.013007	-0.4112	0.7056

# Chi-square test of independence

- Free from parametric test assumptions
- Tests whether two or more categorical variables are independent
  - **Null hypothesis:** The variables are independent.
  - **Alternative hypothesis:** The variables are not independent.

# Chi-square test in python

## Homepage signup rates A/B test

**Null:** There is no significant difference in signup rates between landing page designs C and D

**Alternative:** There is no significant difference in signup rates between them

```
# Calculate the number of users in groups C and D
n_C = homepage[homepage['landing_page'] == 'C']['user_id'].nunique()
n_D = homepage[homepage['landing_page'] == 'D']['user_id'].nunique()
```

```
# Compute unique signups in each group
signup_C = homepage[homepage['landing_page'] == 'C'].groupby('user_id')['signup'].max().sum()
no_signup_C = n_C - signup_C
signup_D = homepage[homepage['landing_page'] == 'D'].groupby('user_id')['signup'].max().sum()
no_signup_D = n_D - signup_D
```

# Chi-square test in python

```
# Create the signups table
table = [[signup_C, no_signup_C], [signup_D, no_signup_D]]
print('Group C signup rate:', round(signup_C/n_C, 3))
print('Group D signup rate:', round(signup_D/n_D, 3))

# Calculate p-value
print('p-value=', stats.chi2_contingency(table, correction=False)[1])
```

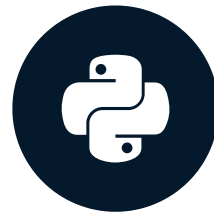
```
Group C signup rate: 0.064
Group D signup rate: 0.048
p-value= 0.009165
```

# Let's practice!

A/B TESTING IN PYTHON

# Ratio metrics and the delta method

A/B TESTING IN PYTHON



**Moe Lotfy, PhD**

Principal Data Science Manager

# Ratio metrics A/B testing

- Mean metrics

$$\text{Mean order value} = \frac{\text{Total orders value}}{\# \text{ users}}$$

$$\text{Mean Time-on-page} = \frac{\text{Total time on page}}{\# \text{ users}}$$

- **Unit of analysis:**
  - The entity being analyzed in an A/B test
  - Denominator in ratio metrics
- **Randomization unit:**
  - The subject randomly allocated to each variant



# Ratio metrics A/B testing

- Per-user Ratio metrics

$$\text{CTR} = \frac{\text{clicks}}{\text{page views}} = \frac{\frac{\text{clicks}}{\text{users}}}{\frac{\text{page views}}{\text{users}}}$$

$$\text{Revenue per session} = \frac{\text{revenue}}{\text{sessions}} = \frac{\frac{\text{revenue}}{\text{users}}}{\frac{\text{sessions}}{\text{users}}}$$

# Delta method motivation

```
print(checkout.groupby('checkout_page')[['order_value', 'purchased']].agg({'sum', 'count', 'mean'}))
```

	order_value		purchased	
	mean	sum	count	mean
checkout_page				
A	24.956437	61417.791564	2461	0.820333
B	29.876202	75915.430125	2541	0.847000
C	34.917589	90890.484142	2603	0.867667

```
checkout.groupby('checkout_page')['order_value'].sum() /  
checkout.groupby('checkout_page')['purchased'].count()
```

```
checkout_page  
A    20.472597  
B    25.305143  
C    30.296828  
dtype: float64
```

# Delta method variance

- Delta method ratio metrics variance estimation:{

$$\text{Var} \frac{X}{Y} \approx \frac{1}{E[Y]^2} \text{Var} X + \frac{E[X]^2}{E[Y]^4} \text{Var} Y - 2 \frac{E[X]}{E[Y]^3} \text{cov}(X, Y) \quad ^1$$

```
# Delta method variance of ratio metric
def var_delta(x,y):
    x_bar = np.mean(x)
    y_bar = np.mean(y)
    x_var = np.var(x,ddof=1)
    y_var = np.var(y,ddof=1)
    cov_xy = np.cov(x,y,ddof=1)[0][1]
    # Note that we divide by len(x) here because the denominator of the test statistic is standard error (=sqrt(var/n))
    var_ratio = (x_var/y_bar**2 + y_var*(x_bar**2/y_bar**4) - 2*cov_xy*(x_bar/y_bar**3))/len(x)
    return var_ratio
```

<sup>1</sup> Budylin, Roman & Drutsa, Alexey & Katsev, Ilya & Tsoy, Valeriya. (2018). Consistent Transformation of Ratio Metrics for Efficient Online Controlled Experiments. 55-63. 10.1145/3159652.3159699.

# Delta method z-test

```
# Delta method ztest calculation
ztest_delta(x_control, y_control, x_treatment, y_treatment, alpha = 0.05)
```

## Input arguments:

- `x_control` : control variant user-level ratio numerator column
- `y_control` : control variant user-level ratio denominator column
- `x_treatment` : treatment variant user-level ratio numerator column
- `y_treatment` : treatment variant user-level ratio denominator column

## Output:

- `mean_control` : control group ratio metric mean
- `mean_treatment` : treatment group ratio metric mean
- `difference` : difference between treatment and control means
- `diff_CI` : confidence interval of the difference in means
- `p-value` : the two-tailed z-test p-value

<sup>1</sup> <https://medium.com/@ahmadnuraziz3/applying-delta-method-for-a-b-tests-analysis-8b1d13411c22>

# Python example

```
# Create DataFrames for per user metrics for variants A and B
A_per_user = pd.DataFrame({'order_value':checkout[checkout['checkout_page']=='A'].groupby('user_id')['order_value'].sum(),
                           'page_view':checkout[checkout['checkout_page']=='A'].groupby('user_id')['user_id'].count()})
B_per_user = pd.DataFrame({'order_value':checkout[checkout['checkout_page']=='B'].groupby('user_id')['order_value'].sum(),
                           'page_view':checkout[checkout['checkout_page']=='B'].groupby('user_id')['user_id'].count()})

# Assign the control and treatment ratio columns
x_control = A_per_user['order_value']
y_control = A_per_user['page_view']
x_treatment = B_per_user['order_value']
y_treatment = B_per_user['page_view']

# Run a z-test for ratio metrics
ztest_delta(x_control,y_control,x_treatment,y_treatment)
```

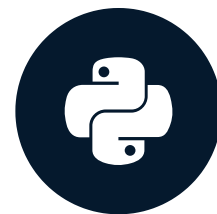
```
{'mean_control': 20.472597188012,
 'mean_treatment': 25.30514337484097,
 'difference': 4.833,
 'diff_CI': '[4.257, 5.408]',
 'p-value': 5.954978880467735e-61}
```

# Let's practice!

A/B TESTING IN PYTHON

# A/B Testing best practices and advanced topics intro

A/B TESTING IN PYTHON



**Moe Lotfy, PhD**  
Principal Data Science Manager

# Best practices

## Avoid peeking

- Avoid making decisions by peeking at the results before reaching the designed sample size, as this inflates error rates similar to multiple comparisons.

## Account for day-of-the-week effects

- Users may behave differently on weekends versus weekdays, so we should include overall behavior.



# Best practices

- **Simplicity/feasibility:**
  - Do we need to build the full feature?
  - Painted door tests
- **Isolation**
  - Change one variable at a time to attribute impact.

# Advanced topics

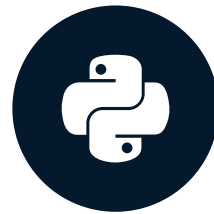
- **Multifactorial design and interaction effects**
  - Measures the isolated effect of each variable
  - Uncovers interaction/synergistic effects
- **Bayesian A/B testing**
  - Incorporates prior data into current experiment
  - Views population parameters as distributions
  - More intuitive understanding of test results
- **SUTVA violation and network effects**
  - One user's assignment in a test impacts others
  - Common in social networks A/B tests
  - One solution: clusters assignment

# Let's practice!

A/B TESTING IN PYTHON

# Wrap-up: A/B testing in python

A/B TESTING IN PYTHON



**Moe Lotfy, PhD**

Principal Data Science Manager

# A/B testing summary

## Chapter 1

- A/B testing steps and use-cases
- Metrics definition and estimation
- `.sample()` , `.corr()` , `pairplot` , `heatmap`

## Chapter 3

- Data cleaning and EDA
- Sanity checks for validation
- Analyzing difference in proportions
- `proportions_ztest` , `proportion_confint`

## Chapter 2

- Formulating A/B testing hypotheses
- Error rates, power, effect size
- Power analysis: sample size estimation
- Multiple comparisons corrections

## Chapter 4

- Analyzing differences in means
- Non-parametric tests
- Delta method for ratio metrics
- Best practices and advanced topics

# Congratulations!

A/B TESTING IN PYTHON