Is this some kind of test statistic?

HYPOTHESIS TESTING IN PYTHON

James Chapman
Content Developer, DataCamp

datacamp

Two-sample problems

- Compare sample statistics across groups of a variable
- `converted_comp` is a numerical variable
- `age_first_code_cut` is a categorical variable with levels (`"child"` and `"adult"`)
- Are users who first programmed as a child compensated higher than those that started as adults?

Hypotheses

H_0: The mean compensation (in USD) is **the same** for those that coded first as a child and those that coded first as an adult.

H_0: $\mu_{child} = \mu_{adult}$

H_0: $\mu_{child} - \mu_{adult} = 0$

H_A: The mean compensation (in USD) is **greater** for those that coded first as a child compared to those that coded first as an adult.

H_A: $\mu_{child} > \mu_{adult}$

H_A: $\mu_{child} - \mu_{adult} > 0$

Calculating groupwise summary statistics

```
stack_overflow.groupby('age_first_code_cut')['converted_comp'].mean()
```

```
age_first_code_cut
adult    111313.311047
child    132419.570621
Name: converted_comp, dtype: float64
```

Test statistics

- Sample mean estimates the population mean
- $\bar{x}$ - a sample mean
- $\bar{x}_{child}$ - sample mean compensation for coding first as a child
- $\bar{x}_{adult}$ - sample mean compensation for coding first as an adult
- $\bar{x}_{child} - \bar{x}_{adult}$ - a *test statistic*
- z-score - a (standardized) test statistic

Standardizing the test statistic

$$z = \frac{\text{sample stat} - \text{population parameter}}{\text{standard error}}$$

$$t = \frac{\text{difference in sample stats} - \text{difference in population parameters}}{\text{standard error}}$$

$$t = \frac{(\bar{x}_{\text{child}} - \bar{x}_{\text{adult}}) - (\mu_{\text{child}} - \mu_{\text{adult}})}{SE(\bar{x}_{\text{child}} - \bar{x}_{\text{adult}})}$$

Standard error

$$SE(\bar{x}_{\text{child}} - \bar{x}_{\text{adult}}) \approx \sqrt{\frac{s^2_{\text{child}}}{n_{\text{child}}} + \frac{s^2_{\text{adult}}}{n_{\text{adult}}}}$$

s is the standard deviation of the variable

n is the sample size (number of observations/rows in sample)

Assuming the null hypothesis is true

$$t = \frac{(\bar{x}_{\text{child}} - \bar{x}_{\text{adult}}) - (\mu_{\text{child}} - \mu_{\text{adult}})}{SE(\bar{x}_{\text{child}} - \bar{x}_{\text{adult}})}$$

$$H_0\colon \mu_{\text{child}} - \mu_{\text{adult}} = 0 \quad \rightarrow \quad t = \frac{(\bar{x}_{\text{child}} - \bar{x}_{\text{adult}})}{SE(\bar{x}_{\text{child}} - \bar{x}_{\text{adult}})}$$

$$t = \frac{(\bar{x}_{\text{child}} - \bar{x}_{\text{adult}})}{\sqrt{\frac{s^2_{\text{child}}}{n_{\text{child}}} + \frac{s^2_{\text{adult}}}{n_{\text{adult}}}}}$$

Calculations assuming the null hypothesis is true

```
xbar = stack_overflow.groupby('age_first_code_cut')['converted_comp'].mean()
```

```
adult    111313.311047
child    132419.570621
Name: converted_comp, dtype: float64 age_first_code_cut
```

```
s = stack_overflow.groupby('age_first_code_cut')['converted_comp'].std()
```

```
adult    271546.521729
child    255585.240115
Name: converted_comp, dtype: float64 age_first_code_cut
```

```
n = stack_overflow.groupby('age_first_code_cut')['converted_comp'].count()
```

```
adult    1376
child     885
Name: converted_comp, dtype: int64
```

Calculating the test statistic

$$t = \frac{(\bar{x}_{\text{child}} - \bar{x}_{\text{adult}})}{\sqrt{\frac{s^2_{\text{child}}}{n_{\text{child}}} + \frac{s^2_{\text{adult}}}{n_{\text{adult}}}}}$$

```
import numpy as np
numerator = xbar_child - xbar_adult
denominator = np.sqrt(s_child ** 2 / n_child + s_adult ** 2 / n_adult)
t_stat = numerator / denominator
```

```
1.8699313316221844
```

Let's practice!

HYPOTHESIS TESTING IN PYTHON

Time for t

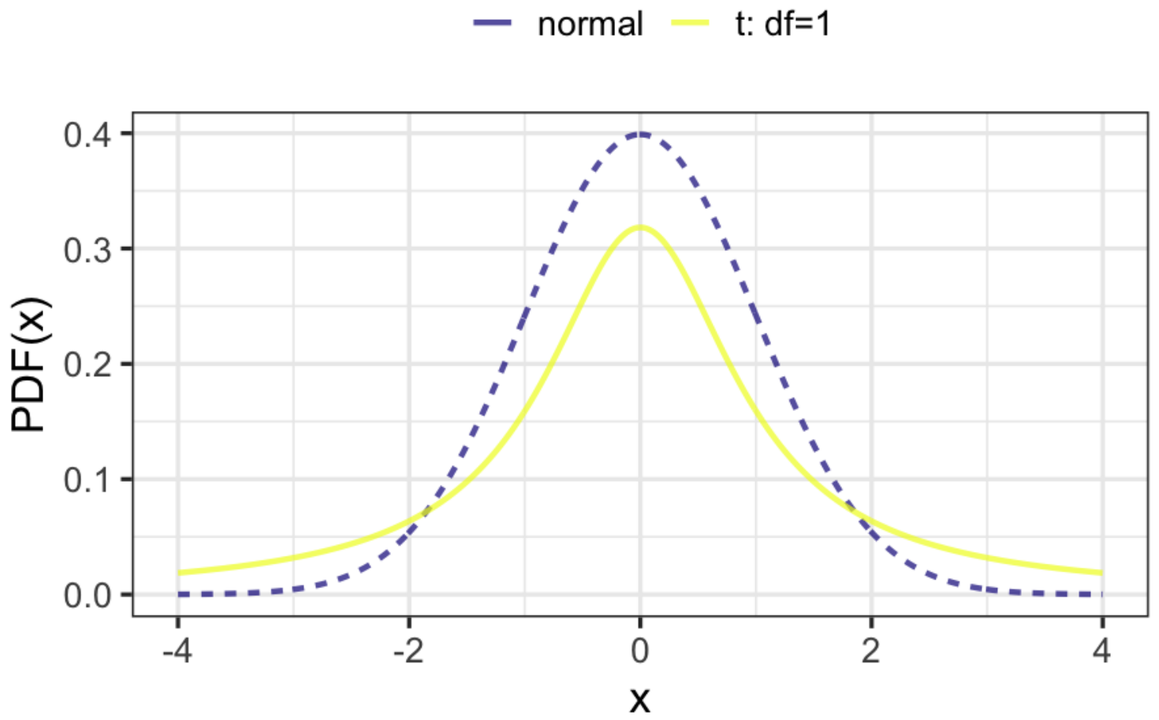
HYPOTHESIS TESTING IN PYTHON

James Chapman
Content Developer, DataCamp

datacamp

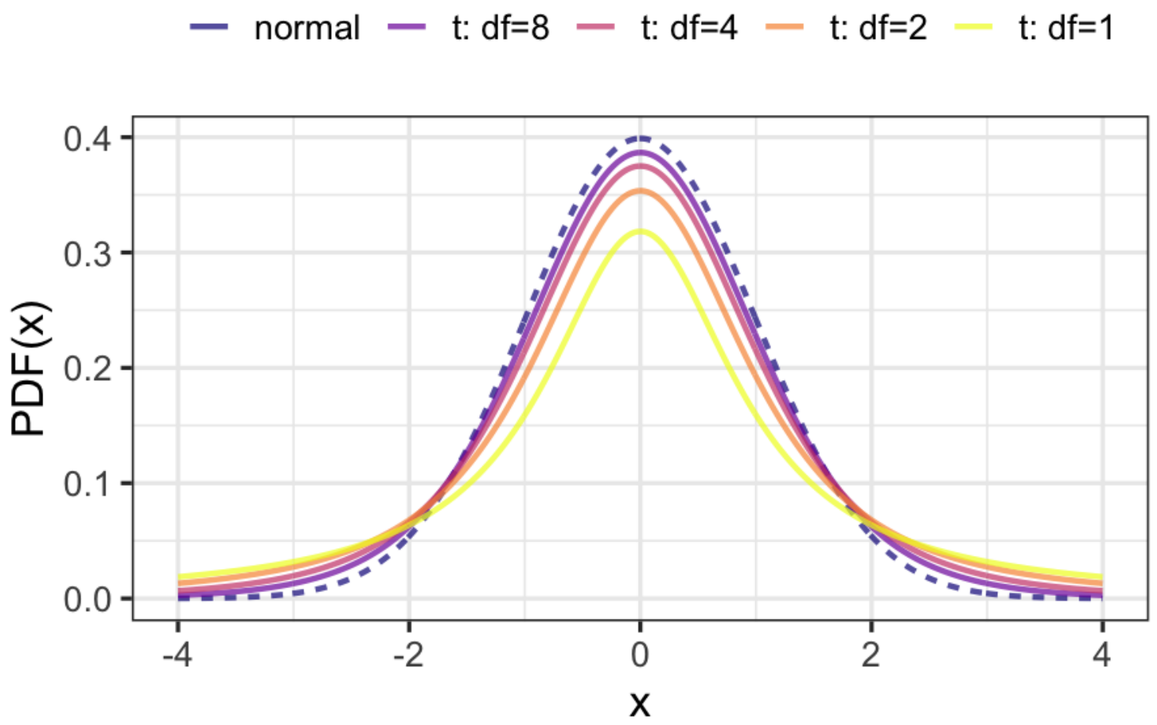
t-distributions

- t statistic follows a t-distribution
- Have a parameter named *degrees of freedom*, or *df*
- Look like normal distributions, with fatter tails



Degrees of freedom

- Larger degrees of freedom $\rightarrow$ t-distribution gets closer to the normal distribution
- Normal distribution $\rightarrow$ t-distribution with infinite df
- Degrees of freedom: maximum number of logically independent values in the data sample



Calculating degrees of freedom

- Dataset has 5 independent observations
- Four of the values are 2, 6, 8, and 5
- The sample mean is 5
- The last value must be 4
- Here, there are 4 degrees of freedom

- $df = n_{child} + n_{adult} - 2$

Hypotheses

H_0: The mean compensation (in USD) is **the same** for those that coded first as a child and those that coded first as an adult

H_A: The mean compensation (in USD) is **greater** for those that coded first as a child compared to those that coded first as an adult

Use a **right-tailed test**

Significance level

$\alpha = 0.1$

If $p \leq \alpha$ then reject H_0.

Calculating p-values: one proportion vs. a value

```
from scipy.stats import norm
1 - norm.cdf(z_score)
```

$$SE(\bar{x}_{\text{child}} - \bar{x}_{\text{adult}}) \approx \sqrt{\frac{s^2_{\text{child}}}{n_{\text{child}}} + \frac{s^2_{\text{adult}}}{n_{\text{adult}}}}$$

- z-statistic: needed when using one sample statistic to estimate a population parameter
- t-statistic: needed when using multiple sample statistics to estimate a population parameter

Calculating p-values: two means from different groups

```
numerator = xbar_child - xbar_adult
denominator = np.sqrt(s_child ** 2 / n_child + s_adult ** 2 / n_adult)
t_stat = numerator / denominator
```

```
1.8699313316221844
```

```
degrees_of_freedom = n_child + n_adult - 2
```

```
2259
```

Calculating p-values: two means from different groups

- Use t-distribution CDF not normal CDF

```
from scipy.stats import t
1 - t.cdf(t_stat, df=degrees_of_freedom)
```

```
0.030811302165157595
```

- Evidence that Stack Overflow data scientists who started coding as a child earn more.

Let's practice!

HYPOTHESIS TESTING IN PYTHON

Pairing is caring

HYPOTHESIS TESTING IN PYTHON

James Chapman
Content Developer, DataCamp

datacamp

US Republican presidents dataset

	state	county	repub_percent_08	repub_percent_12
0	Alabama	Hale	38.957877	37.139882
1	Arkansas	Nevada	56.726272	58.983452
2	California	Lake	38.896719	39.331367
3	California	Ventura	42.923190	45.250693
..	...	...	...	...
96	Wisconsin	La Crosse	37.490904	40.577038
97	Wisconsin	Lafayette	38.104967	41.675050
98	Wyoming	Weston	76.684241	83.983328
99	Alaska	District 34	77.063259	40.789626

[100 rows x 4 columns]

100 rows; each row represents county-level votes in a presidential election.

[1] https://dataverse.harvard.edu/dataset.xhtml?persistentId=doi:10.7910/DVN/VOQCHQ

Hypotheses

Question: Was the percentage of Republican candidate votes lower in 2008 than 2012?

H_0: $\mu_{2008} - \mu_{2012} = 0$

H_A: $\mu_{2008} - \mu_{2012} < 0$

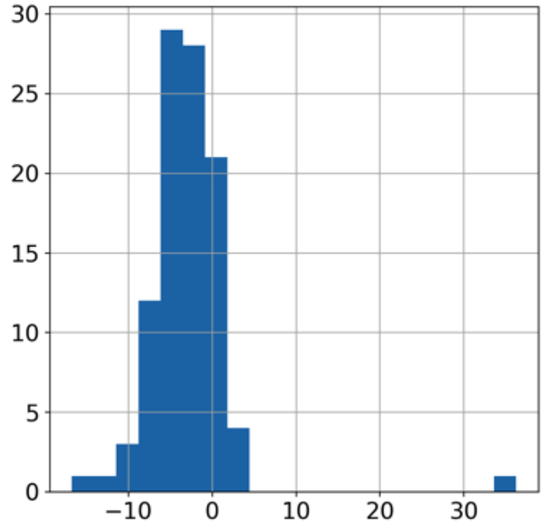
Set $\alpha = 0.05$ significance level.

- Data is **paired** → each voter percentage refers to the same county
  - Want to capture voting patterns in model

From two samples to one

```
sample_data = repub_votes_potus_08_12
sample_data['diff'] = sample_data['repub_percent_08'] - sample_data['repub_percent_12']
```

```
import matplotlib.pyplot as plt
sample_data['diff'].hist(bins=20)
```



Calculate sample statistics of the difference

```
xbar_diff = sample_data['diff'].mean()
```

```
-2.877109041242944
```

Revised hypotheses

Old hypotheses:

H_0: $\mu_{2008} - \mu_{2012} = 0$

H_A: $\mu_{2008} - \mu_{2012} < 0$

$$t = \frac{\bar{x}_{\text{diff}} - \mu_{\text{diff}}}{\sqrt{\frac{s^2_{diff}}{n_{\text{diff}}}}}$$

$$df = n_{diff} - 1$$

New hypotheses:

H_0: $\mu_{\text{diff}} = 0$

H_A: $\mu_{\text{diff}} < 0$

Calculating the p-value

```
n_diff = len(sample_data)
```

```
100
```

```
s_diff = sample_data['diff'].std()
```

```
t_stat = (xbar_diff-0) / np.sqrt(s_diff**2/n_diff)
```

```
-5.601043121928489
```

```
degrees_of_freedom = n_diff - 1
```

```
99
```

$$t = \frac{\bar{x}_{\text{diff}} - \mu_{\text{diff}}}{\sqrt{\frac{s^2_{\text{diff}}}{n_{\text{diff}}}}}$$

$$df = n_{\text{diff}} - 1$$

```
from scipy.stats import t
p_value = t.cdf(t_stat, df=n_diff-1)
```

```
9.572537285272411e-08
```

Testing differences between two means using ttest()

```
import pingouin
pingouin.ttest(x=sample_data['diff'],
               y=0,
               alternative="less")
```

```
              T  dof alternative         p-val          CI95%   cohen-d  \
T-test -5.601043   99        less  9.572537e-08  [-inf, -2.02]  0.560104

              BF10  power
T-test  1.323e+05    1.0
```

[1] Details on Returns from pingouin.ttest() are available in the API docs for pingouin at https://pingouin-stats.org/generated/pingouin.ttest.html#pingouin.ttest.

ttest() with paired=True

```
pingouin.ttest(x=sample_data['repub_percent_08'],
               y=sample_data['repub_percent_12'],
               paired=True,
               alternative="less")
```

```
              T  dof alternative         p-val          CI95%   cohen-d  \
T-test -5.601043   99        less  9.572537e-08  [-inf, -2.02]  0.217364

              BF10     power
T-test  1.323e+05  0.696338
```

Unpaired ttest()

```
pingouin.ttest(x=sample_data['repub_percent_08'],
               y=sample_data['repub_percent_12'],
               paired=False, # The default
               alternative="less")
```

```
              T  dof alternative     p-val         CI95%   cohen-d   BF10  \
T-test -1.536997  198        less  0.062945  [-inf, 0.22]  0.217364  0.927

            power
T-test   0.454972
```

- Unpaired t-tests on paired data increases the chances of false negative errors

Let's practice!

HYPOTHESIS TESTING IN PYTHON

P-hacked to pieces

HYPOTHESIS TESTING IN PYTHON

James Chapman
Content Developer, DataCamp

datacamp

Job satisfaction: 5 categories

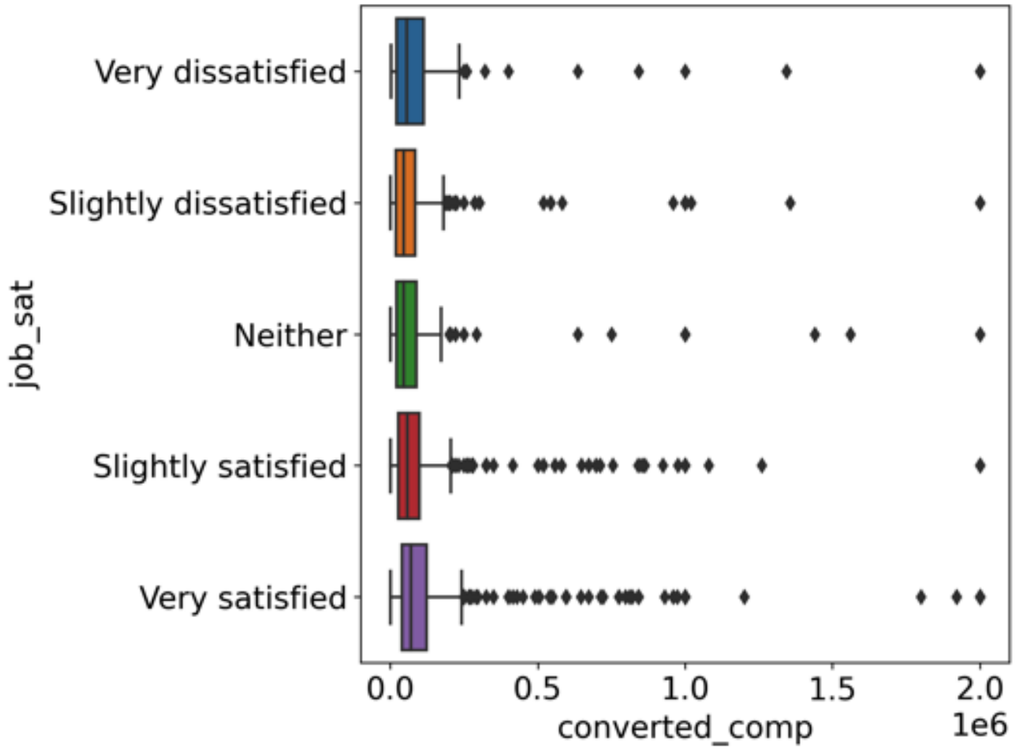
```
stack_overflow['job_sat'].value_counts()
```

```
Very satisfied            879
Slightly satisfied        680
Slightly dissatisfied     342
Neither                   201
Very dissatisfied         159
Name: job_sat, dtype: int64
```

Visualizing multiple distributions

Is mean annual compensation different for different levels of job satisfaction?

```
import seaborn as sns
import matplotlib.pyplot as plt
sns.boxplot(x="converted_comp",
            y="job_sat",
            data=stack_overflow)
plt.show()
```



Analysis of variance (ANOVA)

- A test for differences *between* groups

```
alpha = 0.2

pingouin.anova(data=stack_overflow,
               dv="converted_comp",
               between="job_sat")
```

```
    Source  ddof1  ddof2         F     p-unc       np2
0  job_sat      4   2256  4.480485  0.001315  0.007882
```

- `0.001315` $< \alpha$
- At least two categories have *significantly different* compensation

Pairwise tests

- $\mu_{\text{very dissatisfied}} \neq \mu_{\text{slightly dissatisfied}}$
- $\mu_{\text{very dissatisfied}} \neq \mu_{\text{neither}}$
- $\mu_{\text{very dissatisfied}} \neq \mu_{\text{slightly satisfied}}$
- $\mu_{\text{very dissatisfied}} \neq \mu_{\text{very satisfied}}$
- $\mu_{\text{slightly dissatisfied}} \neq \mu_{\text{neither}}$
- $\mu_{\text{slightly dissatisfied}} \neq \mu_{\text{slightly satisfied}}$
- $\mu_{\text{slightly dissatisfied}} \neq \mu_{\text{very satisfied}}$
- $\mu_{\text{neither}} \neq \mu_{\text{slightly satisfied}}$
- $\mu_{\text{neither}} \neq \mu_{\text{very satisfied}}$
- $\mu_{\text{slightly satisfied}} \neq \mu_{\text{very satisfied}}$

Set significance level to $\alpha = 0.2$.

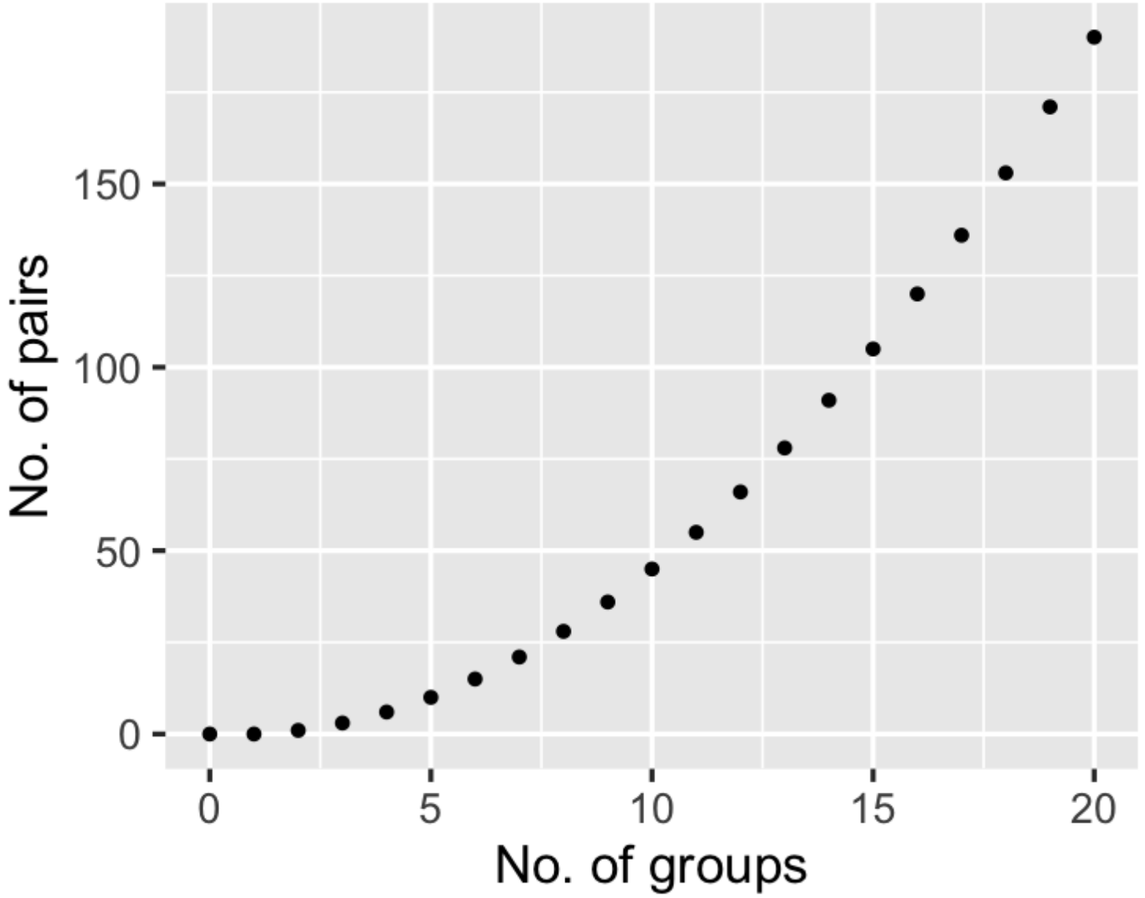
pairwise_tests()

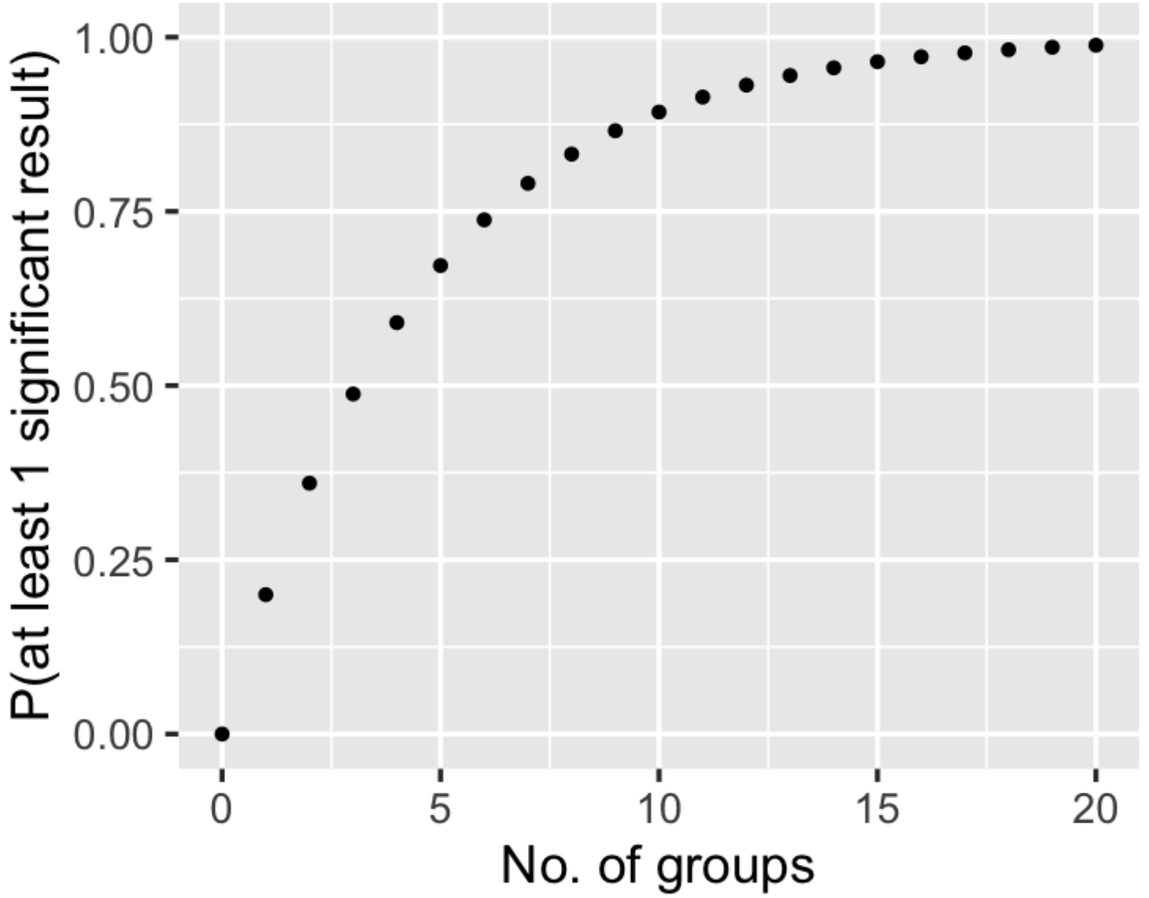
```
pingouin.pairwise_tests(data=stack_overflow,
                        dv="converted_comp",
                        between="job_sat",
                        padjust="none")
```

	Contrast	A	B	Paired	Parametric	...	dof	alternative	p-unc	BF10	hedges
0	job_sat	Slightly satisfied	Very satisfied	False	True	...	1478.622799	two-sided	0.000064	158.564	-0.192931
1	job_sat	Slightly satisfied	Neither	False	True	...	258.204546	two-sided	0.484088	0.114	-0.068513
2	job_sat	Slightly satisfied	Very dissatisfied	False	True	...	187.153329	two-sided	0.215179	0.208	-0.145624
3	job_sat	Slightly satisfied	Slightly dissatisfied	False	True	...	569.926329	two-sided	0.969491	0.074	-0.002719
4	job_sat	Very satisfied	Neither	False	True	...	328.326639	two-sided	0.097286	0.337	0.120115
5	job_sat	Very satisfied	Very dissatisfied	False	True	...	221.666205	two-sided	0.455627	0.126	0.063479
6	job_sat	Very satisfied	Slightly dissatisfied	False	True	...	821.303063	two-sided	0.002166	7.43	0.173247
7	job_sat	Neither	Very dissatisfied	False	True	...	321.165726	two-sided	0.585481	0.135	-0.058537
8	job_sat	Neither	Slightly dissatisfied	False	True	...	367.730081	two-sided	0.547406	0.118	0.055707
9	job_sat	Very dissatisfied	Slightly dissatisfied	False	True	...	247.570187	two-sided	0.259590	0.197	0.119131

[10 rows x 11 columns]

As the number of groups increases...





Bonferroni correction

```
pingouin.pairwise_tests(data=stack_overflow,
                        dv="converted_comp",
                        between="job_sat",
                        padjust="bonf")
```

	Contrast	A	B	...	p-unc	p-corr	p-adjust	BF10	hedges
0	job_sat	Slightly satisfied	Very satisfied	...	0.000064	0.000638	bonf	158.564	-0.192931
1	job_sat	Slightly satisfied	Neither	...	0.484088	1.000000	bonf	0.114	-0.068513
2	job_sat	Slightly satisfied	Very dissatisfied	...	0.215179	1.000000	bonf	0.208	-0.145624
3	job_sat	Slightly satisfied	Slightly dissatisfied	...	0.969491	1.000000	bonf	0.074	-0.002719
4	job_sat	Very satisfied	Neither	...	0.097286	0.972864	bonf	0.337	0.120115
5	job_sat	Very satisfied	Very dissatisfied	...	0.455627	1.000000	bonf	0.126	0.063479
6	job_sat	Very satisfied	Slightly dissatisfied	...	0.002166	0.021659	bonf	7.43	0.173247
7	job_sat	Neither	Very dissatisfied	...	0.585481	1.000000	bonf	0.135	-0.058537
8	job_sat	Neither	Slightly dissatisfied	...	0.547406	1.000000	bonf	0.118	0.055707
9	job_sat	Very dissatisfied	Slightly dissatisfied	...	0.259590	1.000000	bonf	0.197	0.119131

[10 rows x 11 columns]

More methods

padjust : *string*

Method used for testing and adjustment of pvalues.

- `'none'` : no correction [default]
- `'bonf'` : one-step Bonferroni correction
- `'sidak'` : one-step Sidak correction
- `'holm'` : step-down method using Bonferroni adjustments
- `'fdr_bh'` : Benjamini/Hochberg FDR correction
- `'fdr_by'` : Benjamini/Yekutieli FDR correction

Let's practice!

HYPOTHESIS TESTING IN PYTHON