Erasmus
School of
Economics

Statistics for Data Science
Lecture 3

Dennis Fok (Econometric Institute)

September – October, 2025

Erasmus University Rotterdam

Before next time

Assignment for next week

- Finish/Reread Chapter 2
- Read Chapter 3 on testing (skip ANOVA and Multi-Arm Bandits)
- Reconsider/finish the in-class assignments
- Look at examples in book
- You can already start working on the "final" assignment (will be on Canvas early next week)

Hypothesis testing

Statistical testing – General idea

Common statistical question: are two "things" different?

1. Formulate hypotheses
   - Null hypothesis H_0:
     → nothing special happens (no difference)
   - Alternative hypothesis H_a:
     → "something happened" (there is difference)

Hypothesis design

Hypotheses:

- need to be falsifiable
- are often stated as "nothing interesting happens"

→ See whether data provides evidence to reject (null) hypothesis (falsification)

Statistical testing – General idea

2. Collect data
3. Calculate some statistic (known as the *test statistic*)
4. See whether obtained value is "extreme" if H_0 would be true (so we assume that H_0 is correct)
   - if extreme $\rightarrow$ reject H_0
   - if not extreme $\rightarrow$ do not reject H_0

Notes:

- Can **never** conclude with certainty whether H_0 (or H_a) is correct!
- Never say "we accept H_0" or "H_0 is true"
- Also keep economic/general significance in mind!

What is extreme?

Decision rule: Reject H_0 when result is *extreme*!
$\rightarrow$ what is extreme?

- Extreme = unlikely under H_0 (remember: H_0 codes some assumption(s))
- Need a "model" under H_0 to work out probabilities!
- How unlikely is "unlikely"?
  $\rightarrow$ Choice to be made by researcher

Significance level (α) to define "unlikely"

- Usually set at 5%
- Reject if statistic is in α% tail of the distribution under H_0
- If H_0 correct: we still reject in α% of cases!

Potential errors in hypothesis testing

	Conclusion	
	Not reject	Reject
H_0 true	😁	Type I error
H_0 not true	Type II error	😁

- Pr[Type I error] = significance level = α%
- Pr[Type II error]: not always the same, want to minimize this
- Power of test = 1−Pr[Type II error], depends on
  - sample size
  - true "state of the world" (values of parameters)
  - properties of test

Strategies to perform tests

Central concept

- Calculate statistic
- Compare to distribution under H_0 (to check "extreme/not extreme")

Strategy I: Critical values

1. Choose significance level
2. Obtain critical values
3. Calculate statistic
4. Reject if statistic is beyond critical value

Strategy II: p-values

1. Calculate statistic
2. Obtain probability of equal or more evidence against H_0 (if H_0 is true)
   $\rightarrow$ =p-value
3. Reject if p-value $<$ significance level

Strategies to perform tests

Strategy with p-values is preferred

- Report p-value
- Reader can choose own significance level and conclude
- Shows "size" of evidence

Testing means: t-test

t-test on mean

Given

- $X_1, X_2, \ldots, X_n$ independent and identically distributed $N(\mu, \sigma^2)$
- μ and σ^2 unknown

Hypothesis

$$H_0 : \mu = \mu_0, H_a : \mu \neq \mu_0$$

(μ_0 is some **known** value, often 0)

From earlier we know (if H_0 true)

$$\frac{\bar{X} - \mu_0}{\sqrt{\frac{1}{n}s^2}} \sim t_{n-1}$$

Testing procedure

Calculate t-statistic $\frac{\bar{X}-\mu_0}{\sqrt{\frac{1}{n}s^2}}$

Strategy I: Critical values

- Compare t-statistic to percentiles of the t-distribution
- Reject if t-stat outside

$$[t^{\alpha/2}(n-1), t^{1-\alpha/2}(n-1)]$$

$\rightarrow$ `stats.t(n-1).ppf([0.025,0.975])`

Strategy II: p-values

- Calculate probability of *more extreme* outcome under H_0

$$\begin{aligned} &\Pr[t(n-1) > |\text{t-stat}|] + \Pr[t(n-1) < -|\text{t-stat}|] \\ &= 2\Pr[t(n-1) < -|\text{t-stat}|] \end{aligned}$$

- `2*stats.t(n-1).cdf(-abs(tstat))`

Python one-sample t-test (double sided): stats.ttest_1samp()

μ_0

```
# Example of one-sample t test
from scipy import stats
data = stats.norm(0.2, 1.0).rvs(size=500) # Generate some test data
res = stats.ttest_1samp(data, popmean = 0.25) # Run the test
display(res) # Show the test result
res.confidence_interval() # Bonus:  get a confidence interval around mean
```

Example output (edited a bit)

```
TtestResult(statistic=np.float64(-3.11), pvalue=np.float64(0.0020),
df=np.int64(499))
and
ConfidenceInterval(low=np.float64(0.022), high=np.float64(0.198))
```

Compare to α (here: $0.002 < 0.05 \rightarrow$ reject $H_0 : \mu = 0.25$)

Power of t-test

Test statistic:

$$\frac{(\bar{X} - \mu_0)}{\sqrt{\frac{1}{n}s^2}}$$

If $\mu \neq \mu_0$

- Want to reject H_0
- Need test statistic to be extreme
- Want large power of test

Power is large if

- $\bar{X}$ large (so μ very different from μ_0)
- n large
- s^2 small (so small σ^2)

$\rightarrow$ only sample size (n) can be controlled
$\rightarrow$ small differences are (of course) hard to detect

Sample size determination

Given standardized effect size $= \frac{\mu - \mu_0}{\sigma}$, where

- $\mu - \mu_0$: considered difference
- σ^2: variance

Can determine:

- power given n and standardized effect size
  Example:
  `tp = sm.stats.TTestPower()`
  `tp.power(stdeffect, nobs=.., alpha=..)`
- needed n for obtaining desired power and given std. effect size
  Example: `tp.solve_power(effect_size=.., power=.., alpha=..)`

using `import statsmodels.api as sm`

Assignment

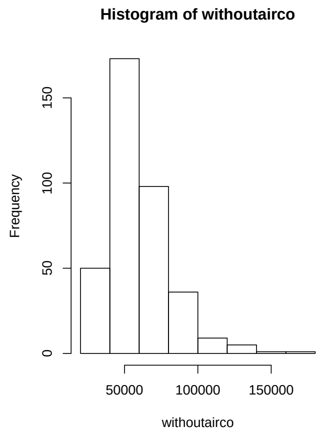
In-class assignment 3.1

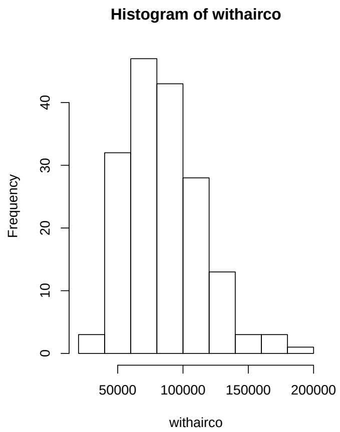
- Generate 100 observations from $N(0.05, 1)$
- Calculate mean
- Perform t-test for $H_0 : \mu = 0$ using `stats.ttest_1samp()`
- What do you conclude? (repeat the above 3 steps a couple of times)
- Calculate the necessary sample size to have power=0.5 for the above situation using
- Advanced: Create a plot of power vs. sample size for different distances between true μ and tested μ (given $\sigma^2 = 1$). You can use `tp.plot_power(..)`

Comparing samples

Comparing samples

Common research question:
Is there a difference between **two** samples?





Comparing samples

- Make sure that you are observing what "needs to be observed" ($\pm$ random treatment)
- Visually compare the two samples
- Focus on summary statistics first (eg. `.mean()` and `.var()`)

```
with_airco = df[df.airco == 1]
wo_airco = df[df.airco == 0]

print(f"Without:  mean={wo_airco.price.mean()}, var={wo_airco.price.var()}")
print(f"With:  mean={with_airco.price.mean()}, var={with_airco.price.var()}")
```

Output:

```
Without:  mean=59884.85254691689, var=455341800.98626363
With:  mean=85880.58959537573, var=810167352.2317516
```

Perform statistical tests

Possible tests

- Is the variance the same?
- Is the mean the same?
  - Variant 1: independent observations
    Sub-variants:
    - if variances are equal
    - if variances are unequal
  - Variant 2: matched/dependent observations

$\rightarrow$ First consider variance

Test on equal variance

Given:

- $X_1, \ldots, X_n$ independent and identically distributed $N(\mu_1, \sigma_1^2)$
- $Y_1, \ldots, Y_m$ independent and identically distributed $N(\mu_2, \sigma_2^2)$
- X_i and Y_j independent

$\rightarrow \mu_1, \mu_2$ and σ_1^2, σ_2^2 are all unknown!

Hypothesis to test:

$$H_0 : \sigma_1^2 = \sigma_2^2$$

against alternative

$$H_a : \sigma_1^2 > \sigma_2^2 \text{ (or } \sigma_1^2 \neq \sigma_2^2)$$

Fisher's F test – theory

We know:

- $\frac{\sum_{i=1}^{n}(X_i - \bar{X})^2}{\sigma_1^2} \sim \chi^2(n-1)$
- $\frac{\sum_{i=1}^{m}(Y_i - \bar{Y})^2}{\sigma_2^2} \sim \chi^2(m-1)$
- and both terms statistically independent (Q: why?)

Hence:

$$\frac{\sum_i (X_i - \bar{X})^2/[\sigma_1^2(n-1)]}{\sum_i (Y_i - \bar{Y})^2/[\sigma_2^2(m-1)]} \sim F(n-1, m-1)$$

Under $H_0 : \sigma_1^2 = \sigma_2^2$ we therefore have

$$\frac{s_X^2}{s_Y^2} = \frac{\sum_{i=1}^{n}(X_i - \bar{X})^2/(n-1)}{\sum_{i=1}^{m}(Y_i - \bar{Y})^2/(m-1)} \sim F(n-1, m-1)$$

Performing Fisher's F test

Steps within this procedure:

- Calculate ratio of (estimated) variances (hypothesised large/hyp. small)
- If true variances are equal $\rightarrow$ ratio should be close to 1
- Ratio $\sim F(n-1, m-1)$
- Check whether ratio is in 5% tail(s) of F-distribution
- p-value: probability of finding a more extreme statistic if H_0 is true

In Python:

```
pvalue = 1-stats.f(n1-1,n2-1).cdf(var1/var2)
```

Note: `scipy` has Barttlet's test and the Fligner-Killeen tests for equal variance: these are more robust to non-normality

t-test for equal means

Consider

- $X_1, \ldots, X_n$ independent and identically distributed $N(\mu_1, \sigma_X^2)$
- $Y_1, \ldots, Y_m$ independent and identically distributed $N(\mu_2, \sigma_Y^2)$
- X_i and Y_j independent

Hypothesis

$$H_0 : \mu_1 = \mu_2$$

against

$$H_a : \mu_1 \neq \mu_2$$

t-test for equal means

We know:

- $\bar{X} \sim N(\mu_1, \frac{1}{n}\sigma_X^2)$ and $\bar{Y} \sim N(\mu_2, \frac{1}{m}\sigma_Y^2)$
- $\bar{X}$ and $\bar{Y}$ independent
- Therefore $\bar{X} - \bar{Y} \sim N(\mu_1 - \mu_2, \frac{1}{n}\sigma_X^2 + \frac{1}{m}\sigma_Y^2)$

$\rightarrow$ Need to estimate variance(s)!

Equal variance

- Estimate pooled variance σ^2: s^2
- t-statistic

$$\frac{(\bar{X} - \bar{Y})}{\sqrt{(\frac{1}{n} + \frac{1}{m})s^2}} \sim t(n + m - 2)$$

Unequal variance

- Separately estimate var(X) and var(Y) $\rightarrow s_1^2$ and s_2^2
- t-statistic

$$\frac{(\bar{X} - \bar{Y})}{\sqrt{\frac{1}{n}s_1^2 + \frac{1}{m}s_2^2}}$$

- Distribution is not exactly t, but approximations exist

Implementation

`scipy.stats.ttest_ind(x, y, equal_var=True)` or
`scipy.stats.ttest_ind(x, y, equal_var=False)`

More than 2 groups

What if more than 2 groups to compare?

- Translate the problem to a linear regression problem (see also next week)
- (Use ANOVA methods)

Assignment

In-class assignment 3.2

Compare prices of houses with airco to houses without airco

- Test whether the variance of the prices is the same for both samples
- Test whether the mean of the prices is the same
  - Use t-test (which one?)
    $\rightarrow$ Use the result from the variance test
- Do the same for log(price)
  $\rightarrow$ Why could this be smart?

Dependent samples

Test for means for dependent samples

The two samples can be dependent/related

- Two observations for same individual over time
- Two different variables for sample of firms
- Two different measurements of same concept

$\rightarrow$ The observations are matched

Consider

- $X_1, \ldots, X_n$ independent and identically distributed $N(\mu_1, \sigma_1^2)$
- $Y_1, \ldots, Y_n$ independent and identically distributed $N(\mu_2, \sigma_2^2)$
- X_i and Y_i (perhaps) dependent

$\rightarrow$ Simply look at the differences $X_i - Y_i$ and apply t-test for mean=0!

In Python:
`ttest_rel(X, Y)` (testing related samples)

Deviations from assumptions

Deviations from assumptions

What if data **not** normal?

As before

- If n large
  $\rightarrow$ Central limit theorem:
  - t-stat approx. $N(0,1)$
  - No problem!
- If n not large **and** data not normal
  $\rightarrow$ Do not use t-test!

Alternatives

- Bootstrap-based test (see book + later lecture)
  How does the obtained mean compare to the bootstrap distribution?
- Permutation tests (see book)
- Other non-parametric tests

Non-parametric tests

Try to avoid making assumptions

+ No worries about possibly incorrect assumptions
– Less powerful when assumptions are correct

General idea: use properties that should be true under H_0

Example Wilcoxon signed-rank test (to replace one-sample t-test)

- Sort |observation−hypothesized mean| and assign rank numbers $(1, 2, 3, \ldots, n)$
- Look at the sum of ranks for observations above hyp. mean
- Does not assume a particular distribution
- Can also use it to test for differences across samples

`scipy.stats.wilcoxon()`

Two-sample case: Wilcoxon Rank-Sum test (aka Mann-Witney U test)

Given

- $X_1, \ldots, X_n$ independent and identically distributed
- $Y_1, \ldots, Y_m$ independent and identically distributed
- X_i and Y_j independent

Procedure

1. Merge X and Y and sort
2. Number obs from 1 to $n + m$
3. Sum all ranks corresponding to X observations $\rightarrow R(X)$
4. Sum all ranks corresponding to Y observations $\rightarrow R(Y)$

If H_0 (mean X equals mean Y) is true

- $R(X)$ should be close to $R(Y)$ (corrected for n vs m)
- Compare obtained results to known tables

Mann-Witney U test in Python

Procedure in Python:

- `scipy.stats.mannwhitneyu(X,Y)`
- Automatically calculates p-values
- Also corrects for ties

Non-parametric alternatives for paired taest

- Wilcoxon signed rank test
  `scipy.stats.wilcoxon(x, y)`
- Binomial test
  `scipy.stats.binomtest(failures, n)`
  → "Failures"=Count no. times $X_i > Y_i$: should have Bin(n,0.5) distribution

Bivariate descriptives

Bivariate descriptive statistics

Up to now we have mainly discussed summary statistics on single variables
→ Does not show relations between variables

Simple bivariate measures

- Covariance
- Correlation

→ Indication of relation

Note

- Correlation ≠ Causation
- Sometimes we find spurious correlation

Covariance and correlation

Given

- Random variable X
- Random variable Y

The covariance is

$$\text{Cov}(X, Y) = \text{E}\left[(X - \text{E}[X])(Y - \text{E}[Y])\right]$$

$\rightarrow$ Scale depends on scale of X and Y!

Correlation is defined as

$$\text{Cor}[X, Y] = \frac{\text{Cov}[X, Y]}{\sqrt{\text{Var}[X]\text{Var}[Y]}}$$

Notes

- Correlation is scale free
- $-1 \leq$ correlation ≤ 1
- If X and Y independent $\implies$ $\text{Cor}[X, Y] = 0$ (not the other way around!)

Estimation of correlation

The above definitions are population statistics

- Given data $\rightarrow$ Estimate the correlation (or covariance)
  `scipy.stats.pearsonr(x,y)`
- .. or covariance
  `np.cov(x,y)`: gives covariance matrix, look at [0][1] element
- Also here: there is estimation uncertainty!

Can test hypothesis on correlation=0

- `scipy.stats.pearsonr(x,y)` for two-sided alternative
- `scipy.stats.pearsonr(x,y),alternative='less'` or
  `scipy.stats.pearsonr(x,y),alternative='greater'` for one-sided alternatives

Warning!

Be very careful when interpreting correlations

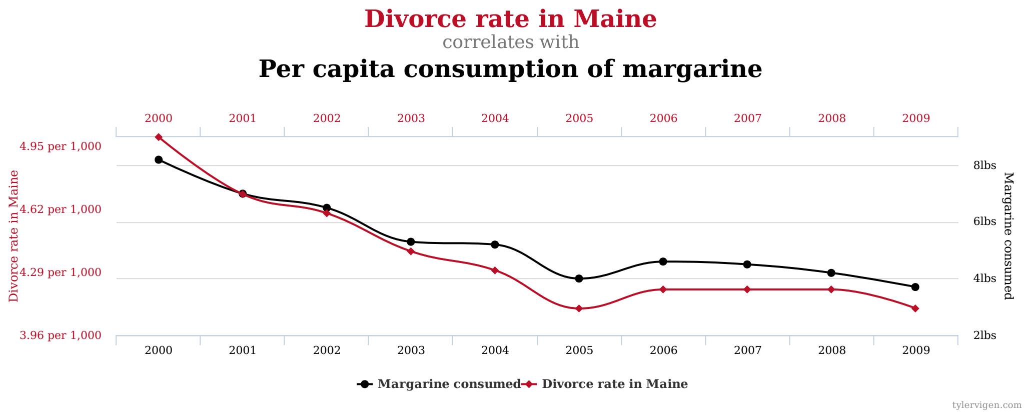
- Direction of effect not given
- Other variables may explain correlation (use partial correlations)

(we cover partial correlations in the context of the linear model)

Advice:

- Correct for time trends
- Think about logical relation between variables
- Think about other related variables

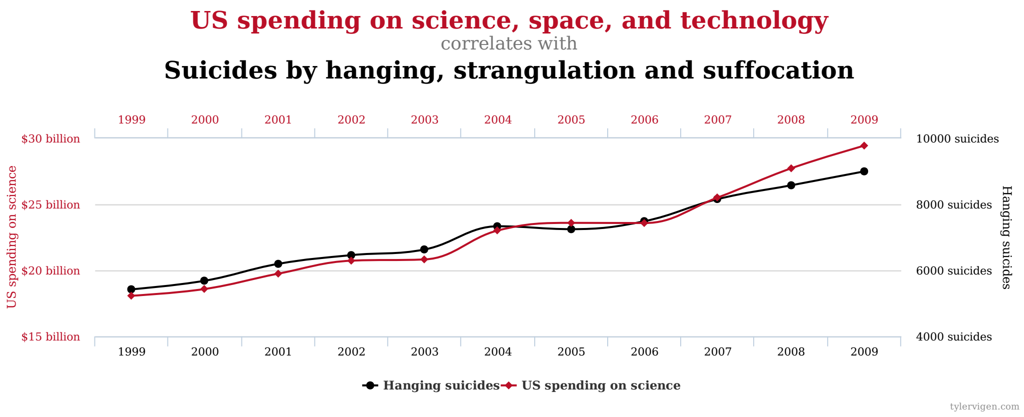
Some examples



Correlation: 0.992558

Source: `http://www.tylervigen.com/spurious-correlations`

Some examples



Correlation: 0.992082
Source: `http://www.tylervigen.com/spurious-correlations`

Other types of correlation

The correlation is a measure of *linear dependence*
→ Also called Pearson correlation

Other measures (to relax the linearity assumption)

- Spearman rank-order correlation
  → Calculate correlation after rank-ordering
- Kendall's tau
  → Alternative measure based on ranks

Python function

- `scipy.stats.spearmanr(x, y)`
- `scipy.stats.kendalltau(x, y)`

Before next time

Assignment for next week

- Reread Chapter 2 & 3 (if needed)
- Read Chapter 4 (main material for next week)
- Reconsider/finish the in-class assignments
- Examples in book
- Small new programming assignment
  - Visualize the correlation between some (continuous) variables in the houseprice data using a scatter plot
  - Calculate the correlation
  - Perform a hypothesis test on this correlation (clearly formulate the hypotheses and the conclusion)
- Work on final assignment