Lecture 12: Hypothesis testing part II

Statistical Methods for Data Science

Yinan Yu

Department of Computer Science and Engineering

December 19, 2024

Today

Learning outcome

- Be able to explain the following hypothesis tests
  - One-sample and two-sample z-test
  - One-sample and two-sample t-test
  - Paired t-test
  - Binomial test (exact, approximate)
  - McNemar's test (exact, approximate)

  For each of these tests, be able to describe a typical setup for the experiment, the general purpose of the test, data produced by the experiment, random variables, parameter of interest, null hypothesis, alternative hypothesis, test statistic, null distribution, the computation of p-value
- Be able to generalize the learning routine to new hypothesis tests
- Be able to compare two classifiers using the paired t-test and McNemar's test for different scenarios

Today

1. Test statistics and hypothesis tests
   - z-test
   - One-sample t-test
   - Two-sample t-test (Welch's t-test - unequal variances)
   - Paired t-test
   - Binomial test
   - McNemar's test
   - Summary
2. Compare two classifiers
3. More exercises
4. Summary

Remark

- In this course, we only consider H_0 **with an equal sign in them**, i.e. the **null distribution is fully specified**.
- For **symmetric null distributions**, e.g. **standard Gaussian distribution**, **student's t distribution**, **binomial distribution with** $p = 0.5$, etc, without loss of generality, we only illustrate examples with the two-tailed alternative hypothesis H_A in this lecture; the one-tailed version can be derived straightforwardly and is not explicitly discussed here.
- For the **exact binomial test with** $p \neq 0.5$, the null distribution is not symmetric; in this case, the computation of the two-tailed p-value is not uniquely defined; in this lecture, we will not go into details for these cases; we will only look at the one-tailed tests for asymmetric binomial null distributions.
- Each hypothesis test includes a **Python code snippet** to help you understand the underlying mechanisms of these tests; you are expected to be able to illustrate your understanding by reproducing these implementations.
- In practice, **built-in libraries** are available for these tests, offering more concise and efficient implementations.

Remark (cont.)

For each of the hypothesis tests we introduce, we present the following components:

- **A typical setup for the experiment**
  - **Test subjects**, e.g. number of samples, number of groups, etc
  - The **experiment** and the **result**
  - The **data type** produced in the result
- **Purpose**: the general purpose of the test
- **Data**: the data produced by the experiment
- **Random variables** and **assumptions**
- **Parameters of interest** and their **estimates**
- **Hypotheses** H_0 and H_A
- **Test statistic**
- **Null distribution**
  - PDF/PMF: description of the PDF/PMF
  - Python: code snippet of the PDF/PMF
- ***p*-value**
  - Definition: expression of the *p*-value
  - Python: code snippet to illustrate the computation of the *p*-value

z-test

One-sample z-test

- **A typical setup for the experiment**:
  - **One sample** of independent test subjects, e.g. a sample of patients, a sample of customers, etc
  - Run the same experiment on each subject and collect the outcomes, e.g. give a new drug to a sample of patients and measure the effect on each individual patient; test a new web design on a sample of customers and record the time they spend on the web page, etc
  - The outcomes contain **one i.i.d. sample** with **continuous numerical values**
- **Purpose**: to test if the mean of the outcomes differs from a predefined constant
- **Data**: $x_1, \cdots, x_N$, e.g. blood pressure after taking a new drug
- **Random variable** and **assumption**: $X_1, \cdots, X_N$
  - X_i i.i.d.
  - X_i Gaussian or large N (CLT)
  - X_i standard deviation σ **known**
- **Parameter of interest**: μ
- **Parameter estimate**: $\bar{x}$, $\bar{X} \sim \mathcal{N}(\mu, \sigma^2/N)$
- **Hypotheses** H_0 and H_A: given c a constant

$$\begin{aligned} H_0: &\quad \mu = c \\ H_A: &\quad \mu \neq c \end{aligned}$$

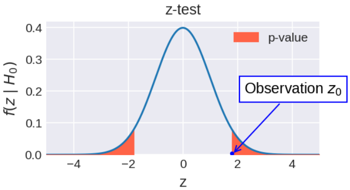
Note: only two-tailed H_A is illustrated here (see page 5).

One-sample z-test (cont.)

- **Test statistic**:

$$z_0 = \frac{\bar{x} - c}{\sigma/\sqrt{N}}$$

- **Null distribution**: standard normal distribution
  - PDF: $f(z \mid H_0) = \frac{1}{\sqrt{2\pi}} e^{-\frac{z^2}{2}}$
  - Python: stats.norm.pdf(z, 0, 1)
- ***p*-value**
  - Definition: $p = 2\min\left(P(Z \leq z_0 \mid H_0), P(Z \geq z_0 \mid H_0)\right)$
  - Python: $2 * \min$ (stats.norm.cdf(z_0, 0, 1), 1-stats.norm.cdf(z_0, 0, 1))



Two-sample z-test

- **A typical setup for the experiment**:
  - **Two samples** of independent test subjects, where the two samples $\mathcal{X}$ and $\mathcal{Y}$ letters with a calligraphic font are typically used to denote sets are independent from one another, e.g. two samples of independent patients, two samples of independent customers, etc
  - Run two sets of experiments A and B on the test subjects from the two samples $\mathcal{X}$ and $\mathcal{Y}$, respectively, and collect the outcomes, e.g. give drug D to patient sample $\mathcal{X}$ and drug E to patient sample $\mathcal{Y}$ and measure the effect on each individual patient; test two web designs on two samples of customers and record the time they spend on the web page, etc
  - The outcomes contain two i.i.d. samples with **continuous numerical values**
- **Purpose**: to test if two alternative options have different effects by testing if the means differ by a constant
- **Data**: $x_1, \cdots, x_{N_X}$ and $y_1, \cdots, y_{N_Y}$, e.g. blood pressure measured after taking two different drugs
- **Random variable** and **assumption**: $X_1, \cdots, X_{N_X}$, $Y_1, \cdots, Y_{N_Y}$
  - X_i and Y_j independent
  - X_i i.i.d.; Y_j i.i.d.
  - X_i Gaussian or large N_X; Y_j Gaussian or large N_Y
  - X_i and Y_j have known standard deviation σ_X and σ_Y, respectively
- **Parameter of interest**: μ_X, μ_Y
- **Parameter estimate**: $\bar{x}$, $\bar{y}$
- **Hypotheses** H_0 and H_A: given c a constant

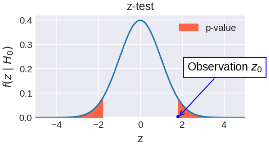
$$
\begin{aligned}
H_0: &\quad \mu_X - \mu_Y = c \\
H_A: &\quad \mu_X - \mu_Y \neq c
\end{aligned}
$$

Two-sample z-test (cont.)

- **Test statistic**:

$$z_0 = \frac{\bar{x} - \bar{y} - c}{\sqrt{\frac{\sigma_X^2}{N_X} + \frac{\sigma_Y^2}{N_Y}}}$$

  Hint: $\bar{X} - \bar{Y} \sim \mathcal{N}\left(\mu_X - \mu_Y, \sigma_X^2/N_X + \sigma_Y^2/N_Y\right)$
- **Null distribution**: standard normal distribution
  - PDF: $f(z \mid H_0) = \frac{1}{\sqrt{2\pi}} e^{-\frac{z^2}{2}}$
  - Python: stats.norm.pdf(z, 0, 1)
- ***p*-value**
  - Definition: $p = 2\min\left(P(Z \leq z_0 \mid H_0), P(Z \geq z_0 \mid H_0)\right)$
  - Python: $2 * \min$ (stats.norm.cdf(z_0, 0, 1), 1-stats.norm.cdf(z_0, 0, 1))



One-sample t-test

One-sample t-test

- **A typical setup for the experiment** (same as one-sample z-test):
  - One sample of independent test subjects, e.g. a sample of patients, a sample of customers, etc
  - Run the same experiment on each subject and collect the outcomes, e.g. give a new drug to a sample of patients and measure the effect on each individual patient; test a new web design on a sample of customers and record the time they spend on the web page, etc
  - The outcomes contain one i.i.d. sample with **continuous numerical values**
- **Purpose**: to test if the mean of the outcomes differs from a predefined constant
- **Data**: $x_1, \cdots, x_N$, e.g. blood pressure after taking a new drug
- **Random variable** and **assumption**: $X_1, \cdots, X_N$
  - X_i i.i.d.
  - X_i Gaussian or large N
  - X_i standard deviation σ **unknown**
- **Parameter of interest**: μ
- **Parameter estimate**: $\bar{x}$
- **Hypotheses** H_0 and H_A: given c a constant

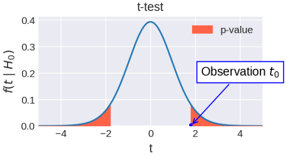
$$\begin{aligned} H_0 : &\quad \mu = c \\ H_A : &\quad \mu \neq c \end{aligned}$$

One-sample t-test (cont.)

- **Test statistic**:

$$t_0 = \frac{\bar{x} - c}{s/\sqrt{N}}$$

  where $s = \sqrt{\frac{1}{N-1}\sum_{i=1}^{N}(x_i - \bar{x})^2}$ is the sample standard deviation
- **Null distribution**:
  - Student's-t distribution with degrees of freedom $df = N - 1$
  - Python: stats.t.pdf$(t, df = N - 1)$
- ***p*-value**:
  - Definition: $p = 2\min\left(P(T \leq t_0 \mid H_0), P(T \geq t_0 \mid H_0)\right)$
  - Python: $2 * \min$ (stats.t.cdf$(t_0, df = N - 1)$, 1-stats.t.cdf$(t_0, df = N - 1)$)



Two-sample t-test (Welch's t-test - unequal variances)

Two-sample t-test

- **A typical setup for the experiment** (same as the two-sample z-test):
  - Two samples of independent test subjects, where the two samples $\mathcal{X}$ and $\mathcal{Y}$ are independent from one another, e.g. two samples of independent patients, two samples of independent customers, etc
  - Run two sets of experiments A and B on the test subjects from the two samples $\mathcal{X}$ and $\mathcal{Y}$, respectively, and collect the outcomes, e.g. give drug D to patient sample $\mathcal{X}$ and drug E to patient sample $\mathcal{Y}$ and measure the effect on each individual patient; test two web designs on two samples of customers and record the time they spend on the web page, etc
  - The outcomes contain two i.i.d. samples with continuous numerical values
- **Purpose**: to test if two alternative options have different effects by testing if the means differ by a constant
- **Data**: $x_1, \cdots, x_{N_X}$ and $y_1, \cdots, y_{N_Y}$, e.g. blood pressure measured after taking two different drugs
- **Random variable** and **assumption**: $X_1, \cdots, X_{N_X}$, $Y_1, \cdots, Y_{N_Y}$
  - X_i and Y_j independent
  - X_i i.i.d.; Y_j i.i.d.
  - X_i Gaussian or large N_X; Y_j Gaussian or large N_Y
  - X_i and Y_j have **unknown** standard deviation σ_X and σ_Y, respectively
- **Parameter of interest**: μ_X, μ_Y
- **Parameter estimate**: $\bar{x}$, $\bar{y}$
- **Hypotheses** H_0 and H_A: given c a constant

$$
\begin{aligned}
H_0: &\quad \mu_X - \mu_Y = c \\
H_A: &\quad \mu_X - \mu_Y \neq c
\end{aligned}
$$

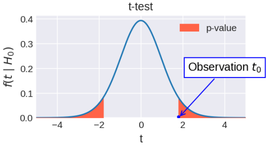
Two-sample t-test (cont.)

- **Test statistic**:

$$t_0 = \frac{\bar{x} - \bar{y} - c}{\sqrt{\frac{s_X^2}{N_X} + \frac{s_Y^2}{N_Y}}}$$

with degrees of freedom $df = \frac{(s_X^2/N_X + s_Y^2/N_Y)^2}{(\frac{s_X^2}{N_X})^2/(N_X-1) + (\frac{s_Y^2}{N_Y})^2/(N_Y-1)}$

- **Null distribution**:
  - Student's-t distribution with degrees of freedom df
  - Python: stats.t.pdf($t, df = df$)
- **p-value**:
  - Definition: $p = 2\min\left(P(T \le t_0 \mid H_0), P(T \ge t_0 \mid H_0)\right)$
  - Python: $2 * \min$ (stats.t.cdf(t_0, df=df), 1-stats.t.cdf(t_0, df=df))



Paired t-test

Paired t-test

- **A typical setup for the experiment**:
  - Typically one sample of independent test subjects, e.g. one sample of independent patients; or two paired samples
  - Run two sets of experiments A and B on all subjects from the sample and collect the outcomes, e.g. measure the blood pressure of a sample of patients **before** giving them a new drug (experiment A); measure the blood pressure of these patients **after** giving them the new drug (experiment B)
  - The outcomes contain two samples with **continuous numerical values**
- **Purpose**: to test if two alternative options have different effects by testing if the mean of their differences differs from a predefined constant
- **Data**: $x_1, \cdots, x_N$, $y_1, \cdots, y_N$
- **Random variable** and **assumption**: $X_1, \cdots, X_N$, $Y_1, \cdots, Y_N$
  - $X_i - Y_i$ i.i.d.
  - $X_i - Y_i \sim \mathcal{N}\left(\mu_{X-Y}, \sigma^2_{X-Y}\right)$ or large N (CLT)
  - standard deviation unknown
- **Parameter of interest**: μ_{X-Y}
- **Parameter estimate**: $m_{X-Y} = \frac{1}{N}\sum_{i=1}^{N}(x_i - y_i)$
- **Hypotheses** H_0 and H_A: given c a constant

$$
\begin{aligned}
H_0 : &\quad \mu_{X-Y} = c \\
H_A : &\quad \mu_{X-Y} \neq c
\end{aligned}
$$

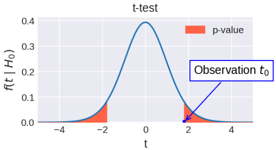
Paired t-test

- **Test statistic**:

$$t_0 = \frac{m_{X-Y} - c}{s_{X-Y}/\sqrt{N}}$$

where $s_{X-Y} = \sqrt{\frac{1}{N-1}\sum_{i=1}^{N}(x_i - y_i - m_{X-Y})^2}$

- **Null distribution**:
  - Student's-t distribution with degrees of freedom $N-1$
  - Python: stats.t.pdf$(t, df = N-1)$
- ***p*-value**:
  - Definition: $p = 2\min\left(P(T \leq t_0 \mid H_0), P(T \geq t_0 \mid H_0)\right)$
  - Python: $2 * \min$ (stats.t.cdf$(t_0, df = N-1)$, 1-stats.t.cdf$(t_0, df = N-1)$)



Exercise 1

- A company claims that a new drug E they have developed can increase the average sleeping hours of people with insomnia. Design three different hypothesis tests to test this statement.

Let's design experiments for running the one-sample t-test, two-sample t-test and paired t-test

Test 1: one-sample t-test

- **Statement**: drug E does not increase the average sleeping hours of people with insomnia; for the one-sample t-test, the average sleeping hours of people with insomnia is a known constant - say, it is 4.5 hours
- **Experiment**: let $N = 40$ people with insomnia take drug E and observe the amount of their sleep
- **Data**: $x_1, \cdots, x_N$ the sleeping hours of people who have taken drug E; **random variable** $X_1, \cdots, X_N$ i.i.d.
- **Parameter of interest**: the mean value μ; **estimate**: sample mean

$$\hat{\mu} = \bar{x} = \frac{1}{N}\sum_{i=1}^{N} x_i$$

- **Null hypothesis** H_0: $H_0 : \mu = 4.5$
- **Significance level** α: set to 0.05

Test 1: one-sample t-test (cont.)

- **Test statistic**:

$$t_0 = \frac{\bar{x} - 4.5}{s/\sqrt{N}}$$

  where $s = \sqrt{\frac{1}{N-1}\sum_{i=1}^{N}(x_i - \bar{x})^2}$
- **Null distribution**:
  - Student's-t distribution with degrees of freedom $df = N - 1$
  - Python: stats.t.pdf($t, df = N - 1$)
- **Alternative hypothesis H_A**: $H_A : \mu \neq 4.5$ - two tailed test

Test 1: one-sample t-test (cont.)

- Run the experiment and collect data

```
# Data in this example is generated using the following command
N = 40
x = stats.norm.rvs(loc=5.2, scale=1.2, size=N, random_state=1)
>> x = [7.14921444 4.4658923  4.5661939  3.91243765
        6.23848916 2.43815356 7.29377412 4.28655172
        5.58284692 4.90075555 6.95452952 2.72783115
        4.81309936 4.73913477 6.56052333 3.88013048
        4.99308615 4.1465699  5.2506565  5.89937826
        3.87925699 6.57366845 6.28190886 5.80299321
        6.28102714 4.37952657 5.05253173 4.07707668
        4.8785343  5.83642656 4.3700071  4.72389577
        4.37539276 4.18575323 4.39450464 5.18480248
        3.85922758 5.48129884 7.19176261 6.09045299]
```

$\Rightarrow \bar{x} = 5.092$

Test 1: one-sample t-test (cont.)

- Compute the test statistic t_0 from data:
  - First, estimate the **nuisance parameter** - the parameter that is **not the parameter of interest**: standard deviation

$$s = \sqrt{\frac{1}{N-1}\sum_{i=1}^{N}(x_i - \bar{x})^2} = 1.172$$

  - Then compute the test statistic

$$t_0 = \frac{\bar{x} - 4.5}{s/\sqrt{N}} = \frac{5.09 - 4.5}{1.172/\sqrt{40}} = 3.197$$

Test 1: one-sample t-test (cont.)

- Compute the p-value:

$$p = 2\min(P(T \leq t_0 \mid H_0), P(T \geq t_0 \mid H_0)) = 0.003$$

- $p < \alpha$: reject H_0

Example implementation in Python: **stats.ttest_1samp(x, 4.5)**

- x is specified on page 24

Test 2: two-sample t-test

- **Statement**: drug E does not increase the average sleeping hours of people with insomnia
- **Experiment**: let $N_X = 40$ people with insomnia take drug E and observe their amount of sleep; observe the sleeping hours of $N_Y = 50$ people with insomnia without taking drug E
- **Data**:
  - $x_1, \cdots, x_{N_X}$ sleeping hours of people with insomnia who have taken drug E; **random variable** $X_1, \cdots, X_{N_X}$ i.i.d.
  - $y_1, \cdots, y_{N_Y}$ sleeping hours of people with insomnia who have not taken drug E; **random variable** $Y_1, \cdots, Y_{N_Y}$ i.i.d.
  - X_i and Y_j independent, for $i = 1, \cdots, N_X$, $j = 1, \cdots, N_Y$

Test 2: two-sample t-test (cont.)

- **Parameter of interest**:
  - The mean value of the sleeping hours of people with insomnia after taking drug E μ_E; **estimate**: sample mean $\hat{\mu}_E = \bar{x} = \frac{1}{N_X}\sum_{i=1}^{N_X} x_i$
  - The mean value of the sleeping hours of people with insomnia without taking drug E μ_0; **estimate**: sample mean $\hat{\mu}_0 = \bar{y} = \frac{1}{N_Y}\sum_{i=1}^{N_Y} y_i$
- **Null hypothesis H_0**: $H_0 : \mu_E - \mu_0 = 0$
- **Test statistic**:

$$t_0 = \frac{\bar{x} - \bar{y}}{\sqrt{\frac{s_X^2}{N_X} + \frac{s_Y^2}{N_Y}}}$$

with degrees of freedom $df = \frac{(s_X^2/N_X + s_Y^2/N_Y)^2}{(\frac{s_X^2}{N_X})^2/(N_X-1) + (\frac{s_Y^2}{N_Y})^2/(N_Y-1)}$, where

$s_X = \sqrt{\frac{1}{N_X-1}\sum_{i=1}^{N_X}(x_i - \bar{x})^2}$ and $s_Y = \sqrt{\frac{1}{N_Y-1}\sum_{i=1}^{N_Y}(y_i - \bar{y})^2}$

Test 2: two-sample t-test (cont.)

- **Null distribution**:
  - Student's-t distribution with degrees of freedom **df** (cf. page 28)
  - Python: stats.t.pdf($t, df =$ **df**)
- **Alternative hypothesis H_A**: $H_A : \mu_E - \mu_0 \neq 0$ - two tailed test
- **Significance level α**: set to 0.05

Test 2: two-sample t-test (cont.)

- Run the experiment and collect data: x is the same data as page 24

```
# Data y in this example is generated using the following command
y = stats.norm.rvs(loc=4.5, scale=0.9, size=50, random_state=2)
>> y = [4.12491794 4.44935986 2.57742351 5.97624373 2.88590797
        3.74242737 4.95259328 3.37924072 3.547843   3.68189315
        4.99630864 6.56298721 4.53738545 3.4938671  4.98515249
        3.96345627 4.48278255 5.5575011  3.82691615 4.50812273
        3.7097029  4.35920925 4.73091341 3.61009886 4.19506023
        4.28743437 3.92611049 3.43114894 3.2209045  4.36185432
        4.25784874 6.50823011 2.30870918 4.60145385 4.83340008
        5.72367048 4.95167149 3.74020767 4.50000879 4.98811731
        4.21784262 5.19391056 2.81871841 6.0580662  5.82091021
        4.1978904  5.0502067  4.54317353 3.75377824 4.5789392 ]
```

  Parameter estimate:
  - **Parameter of interest**: $\bar{x} = 5.092$, $\bar{y} = 4.374$
  - **Nuisance parameter**:

$$s_X = \sqrt{\frac{1}{N_X - 1}\sum_{i=1}^{N_X}(x_i - \bar{x})^2} = 1.172, \;\; s_Y = \sqrt{\frac{1}{N_Y - 1}\sum_{i=1}^{N_Y}(y_i - \bar{y})^2} = 0.946$$

Test 2: two-sample t-test (cont.)

- Compute the test statistic t_0 from data:
  - Then compute the test statistic

$$t_0 = \frac{\bar{x} - \bar{y}}{\sqrt{\frac{s_X^2}{N_X} + \frac{s_Y^2}{N_Y}}} = 3.142$$

- Compute the p-value:

$$p = 2\min(P(T \leq t_0 \mid H_0), P(T \geq t_0 \mid H_0)) = 0.002$$

- $p < \alpha$: reject H_0

Test 2: two-sample t-test (cont.)

- In this two-sample t-test, we do not assume equal variance for X_i and Y_j; this type of two-sample t-test is also called **Welch's t-test**
- Example implementation in Python:

  stats.ttest_ind(x, y, equal_var=False)

  where equal_var=False means we do not assume equal variance for x and y

Test 3: paired t-test

- **Statement**: drug E does not increase the average sleeping hours of people with insomnia
- **Experiment**: let $N = 40$ people with insomnia take drug E and observe their amount of sleep before and after taking drug E
- **Data**: let $z_1, \cdots, z_N$ and $x_1, \cdots, x_N$ be the sleeping hours of people before and after taking drug E, respectively; **random variable** $X_1 - Z_1, \cdots, X_N - Z_N$ i.i.d.
- **Parameter of interest**: the mean value of the difference μ_{X-Z}; **estimate**: sample mean $\hat{\mu}_{X-Z} = \frac{1}{N}\sum_{i=1}^{N} x_i - z_i$
- **Null hypothesis H_0**: $H_0 : \mu_{X-Z} = 0$

Test 3: paired t-test (cont.)

- **Test statistic**:

$$t_0 = \frac{\hat{\mu}_{X-Z}}{s_{X-Z}/\sqrt{N}}$$

  where $s_{X-Z} = \sqrt{\frac{1}{N-1}\sum_{i=1}^{N}(x_i - z_i - \hat{\mu}_{X-Z})^2}$
- **Null distribution**:
  - Student's-t distribution with degrees of freedom $df = N - 1$
  - Python: stats.t.pdf$(t, df = N - 1)$
- **Alternative hypothesis H_A**: $H_A : \mu_{X-Z} \neq 0$ - two tailed test
- **Significance level α**: set to 0.05

Test 3: paired t-test (cont.)

- Run the experiment and collect data: x is the same data as page 24

```
# Data z in this example is generated using the following command
N = 40
z = stats.norm.rvs(loc=4.5, scale=0.9, size=N, random_state=0)
>> z = [6.08764711 4.86014149 5.38086419 6.51680388 6.18080219
        3.62044991 5.35507958 4.36377851 4.40710303 4.86953865
        4.62963921 5.80884616 5.18493395 4.60950751 4.89947691
        4.80030689 5.84467117 4.31535756 4.78176093 3.73131383
        2.20230917 5.08825674 5.27799258 3.83205148 6.54277916
        3.19107089 4.54118267 4.33153453 5.87950129 5.82242289
        4.63945268 4.84034627 3.70099283 2.71728318 4.18687907
        4.64071407 5.60726161 5.58214186 4.15140586 4.22792752]
```

Parameter estimate:

- **Parameter of interest**: $\Rightarrow \hat{\mu}_{X-Z} = 0.311$
- **Nuisance parameter**: $s_{X-Z} = \sqrt{\frac{1}{N-1}\sum_{i=1}^{N}(x_i - z_i - \hat{\mu}_{X-Z})^2} = 1.313$

Test 3: paired t-test (cont.)

- Compute the test statistic t_0 from data:

$$t_0 = \frac{\hat{\mu}_{X-Z}}{s_{X-Z}/\sqrt{N}} = 1.499$$

- Compute the p-value:

$$p = 2\min(P(T \leq t_0 \mid H_0), P(T \geq t_0 \mid H_0)) = 0.142$$

- $p > \alpha$: fail to reject H_0

Example implementation in Python: stats.ttest_rel(x, z)

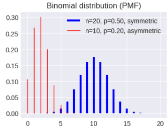
Exercise 2

- One of the tests you have designed is a two-sample test. After the experiments, you realized the test subjects being selected in the second group are twins of the first group. (and they both have insomnia. Duh!) Would that be a problem? Can you still use the result somehow?
- Solution:
    - The two-sample test is the two-sample t-test (see page 27); cannot use the result as is since the two samples are not independent
    - As a potential solution, we can match related subjects in the first group and the second group to create a paired data set $(x_1, y_1), \cdots, (x_N, y_N)$, i.e. x_i and y_i in each pair are related to each other
    - Apply the paired t-test on the new data set $(x_1, y_1), \cdots, (x_N, y_N)$

Binomial test

Binomial distribution

- Discrete distribution
- Applies to discrete numerical data - the number of successes from n independent Bernoulli trials with probability of success p
- Example: You try to catch 10 ducks one by one; (they need their cuddles!) the success rate of catching a duck is $p = 20\%$; what is the probably of catching k ducks successfully, where $k = 0, 1, \cdots, 10$?
- PMF:
  - Equation
  $$f_X(k \mid n, p) = P(X = k) = \binom{n}{k} p^k (1-p)^{n-k}, \; k = 0, \cdots, n, \; p \in [0, 1]$$
  where $\binom{n}{k} = \frac{n!}{k!(n-k)!}$ is the binomial coefficient (choose k from n)
  - Shape
    - When $p = 0.5$, the PMF is symmetric
    - When $p \neq 0.5$, the PMF is asymmetric



- Parameters: p and n; n is typically known

Binomial test

- **A typical setup for the experiment**:
  - One sample of independent test subjects, e.g. one sample of independent patients
  - Run the same experiment on all subjects from the sample and collect the outcomes, e.g. give a new drug to a sample of patients and measure how many patients are cured
  - The outcomes contain one sample with **nominal categorical values with two categories**, which are then summarized into one **discrete numerical value** - the number of "success" cases
- **Purpose**: to test if the proportion of "success" differs from a predefined constant
- **Data**: N independent Bernoulli trials x_i with k_0 "success" outcomes, e.g. the number of cured patients within the sample of size N
- **Random variable** and **assumption**: $X_i \sim Bernoulli(p)$, $\mathrm{K} \sim Binomial(N, p)$ with known N and unknown success rate p
- **Parameter of interest**: p
- **Parameter estimate**: $\hat{p} = \frac{k_0}{N}$
- **Null hypothesis**: given π a constant,

$$H_0 : p = \pi$$

(exact) Binomial test (cont.)

- **Test statistic**: k_0
- **Null distribution**:

$$P(X = k) = \binom{N}{k} \pi^k (1-\pi)^{N-k}$$

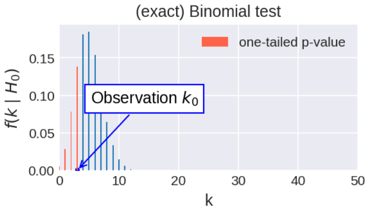
  - Binomial distribution with parameters N and π
  - Python: stats.binom.pmf(k, N, π)
- As discussed in the remarks (cf. page 5), we only introduce the following scenarios:
  - One-tailed (left) binomial test with any $\pi \in (0, 1)$
  - One-tailed (right) binomial test with any $\pi \in (0, 1)$
  - Two-tailed binomial test with $\pi = 0.5$, where the null distribution is symmetric

(exact) One-tailed (left) binomial test

- **Hypotheses H_0 and H_A**:

$$H_0: \quad p = \pi$$
$$H_A: \quad p < \pi$$

- ***p*-value**:
  - Definition: $P(K \leq k_0 \mid H_0)$
  - Python: stats.binom.cdf(k_0, n=N, p=π)

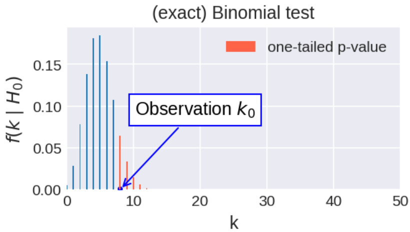


(exact) One-tailed (right) binomial test

- **Hypotheses** H_0 and H_A:

$$H_0: \quad p = \pi$$
$$H_A: \quad p > \pi$$

- p-**value**:
  - Definition: $P(K \geq k_0 \mid H_0)$
  - Python: $1 - \text{stats.binom.cdf}(k_0, n = N, p = \pi) + \text{stats.binom.pmf}(k_0, n = N, p = \pi)$

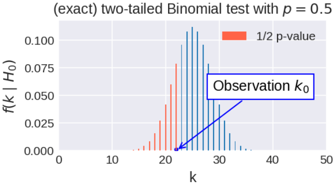


(exact) Two-tailed binomial test

- **Hypotheses** H_0 and H_A:

$$\begin{aligned} H_0 : &\quad p = 0.5 \\ H_A : &\quad p \neq 0.5 \end{aligned}$$

- ***p*-value**:
  - Definition: $2 \min \left(P(K \leq k_0 \mid H_0), P(K \geq k_0 \mid H_0) \right)$
  - Python:
    - $c = \text{stats.binom.cdf}(k_0, n = N, p = 0.5)$
    - $2 * \min \left(c, 1 - c + \text{stats.binom.pmf}(k_0, n = N, p = 0.5) \right)$



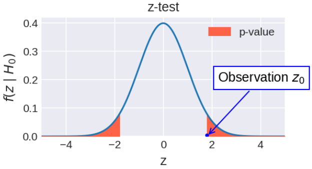
(large N) Binomial test

Same set up as page 40, but for large N, the Binomial distribution can be approximated using a Gaussian distribution $K \sim \mathcal{N}(N\pi, N\pi(1-\pi))$

- **Test statistic**:

$$z_0 = \frac{k_0 - N\pi}{\sqrt{N\pi(1-\pi)}}$$

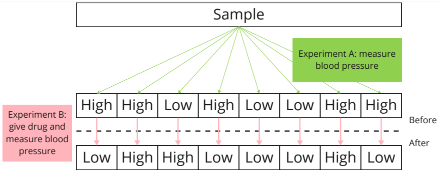
- **Null distribution**: standard Gaussian distribution
  - PDF: $f(z \mid H_0) = \frac{1}{\sqrt{2\pi}} e^{-\frac{z^2}{2}}$
  - Python: stats.norm.pdf(z, 0, 1)
- ***p*-value**:
  - Definition: $p = 2\min\left(P(Z \leq z_0 \mid H_0), P(Z \geq z_0 \mid H_0)\right)$
  - Python: $2 * \min$ (stats.norm.cdf(z_0, 0, 1), 1-stats.norm.cdf(z_0, 0, 1))



McNemar's test

McNemar's test

- **A typical setup for the experiment**:
  - **One sample** of independent test subjects, e.g. one sample of independent patients
  - Within the sample, there are **two groups**; each subject belongs to one and only one group, e.g. within the sample of patients, we have one group with high blood pressure and another group with normal blood pressure
  - Run two sets of experiments A and B on all test subjects from the sample and collect the outcomes, e.g. measure the blood pressure (high or normal) of the patients **before** giving them a new drug (experiment A); measure the blood pressure (high or normal) of the patients **after** giving them the new drug (experiment B)
  - The result contains two paired samples with **nomimal categorical values with two categories** measured from each test subject, e.g. high blood pressure and normal blood pressure, which are summarized in a contingency table



McNemar's test (cont.)

- **Purpose**: to test if an action have different effects on two different **groups**
- **Data**: N independent Bernoulli trials with outcomes $x_1, \cdots, x_N$ and $y_1, \cdots, y_N$ for the two experiments, respectively; $x_i, y_i \in \{0, 1\}$

	$x_i = 0$	$x_i = 1$	
$y_j = 0$	n_{00}	n_{10}	$n_{00} + n_{10}$
$y_j = 1$	n_{01}	n_{11}	$n_{01} + n_{11}$
	$n_{00} + n_{01}$	$n_{10} + n_{11}$	N

  where n_{mn} is the count of $x_i = m$ and $y_j = n$
- **Random variable** and **assumption**: i.i.d. $X_i \sim Bernoulli(p_X)$ and i.i.d. $Y_i \sim Bernoulli(p_Y)$

McNemar's test (cont.)

Example

- A company is trying to determine the effectiveness of a drug on lowering blood pressure
- The company tested the drug on a sample of 229 **independent** patients
- There are **two groups** within this sample: a high blood pressure group (112 patients) and a normal blood pressure group (117 patients); each patient belongs to one and only one of these two groups
- The blood pressure of each patient is measured **before** (to determine the group) and **after** (to determine the effect) taking the drug
- The data is summarized as follows:

	Before (high blood pressure)	Before (normal blood pressure)	
After (high blood pressure)	90	15	105
After (normal blood pressure)	22	102	124
	112	117	229

(small discordance $n_{01} + n_{10}$) McNemar's test (cont.)

	$x_i = 0$	$x_i = 1$	
$y_j = 0$	n_{00}	n_{10}	$n_{00} + n_{10}$
$y_j = 1$	n_{01}	n_{11}	$n_{01} + n_{11}$
	$n_{00} + n_{01}$	$n_{10} + n_{11}$	N

- **Parameter of interest**: discordance

$$p = \min\left(P(X_i = 0, Y_i = 1 \mid X_i \neq Y_i), P(X_i = 1, Y_i = 0 \mid X_i \neq Y_i)\right)$$

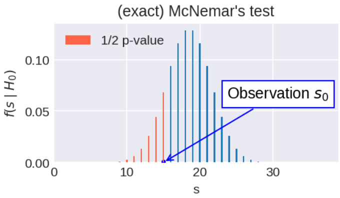
- **Parameter estimate**: $\hat{p} = \frac{\min(n_{01}, n_{10})}{n_{01} + n_{10}}$
- **Hypotheses** H_0 and H_A:

$$\begin{aligned} H_0 : \quad & p = 0.5 \\ H_A : \quad & p \neq 0.5 \end{aligned}$$

  - H_0: the drug does not have any effect on blood pressure control
  - H_1: the drug has effect on blood pressure control

(small discordance $n_{01} + n_{10}$) McNemar's test (cont.)

- **Test statistic**: $s_0 = \min(n_{01}, n_{10})$
- **Null distribution**:
  - Binomial distribution with parameters $(n_{01} + n_{10}, 0.5)$
  - Python: stats.binom.pmf(s, $n_{01} + n_{10}$, 0.5)
- ***p*-value**:
  - Definition: $p = 2P(S \leq s_0 \mid H_0)$
  - Python: $2 *$ stats.binom.cdf$(s_0, n_{01} + n_{10}, 0.5)$



(large discordance $n_{01} + n_{10}$) McNemar's test

	$x_i = 0$	$x_i = 1$	
$y_j = 0$	n_{00}	n_{10}	$n_{00} + n_{10}$
$y_j = 1$	n_{01}	n_{11}	$n_{01} + n_{11}$
	$n_{00} + n_{01}$	$n_{10} + n_{11}$	N

Same set up as page 48, but with large $n_{01} + n_{10}$, e.g. $n_{01} + n_{10} > 25$

- **Parameter of interest**: **discordance** (note: it is different from the previous definition (cf. page 50)) $p_{01} = P(X = 0, Y = 1)$ and $p_{10} = P(X = 1, Y = 0)$
- **Parameter estimate**: $\hat{p}_{01} = \frac{n_{01}}{N}$ and $\hat{p}_{10} = \frac{n_{10}}{N}$
- **Hypotheses** H_0 and H_A:

$$\begin{aligned} H_0: &\quad p_{01} = p_{10} \\ H_A: &\quad p_{01} \neq p_{10} \end{aligned}$$

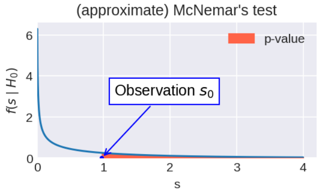
- **Test statistic**:

$$s_0 = \frac{(|n_{01} - n_{10}| - 1)^2}{n_{01} + n_{10}}$$

Note: "-1" is called the continuity correction (https://en.wikipedia.org/wiki/Continuity_correction)

(large discordance $n_{01} + n_{10}$) McNemar's test (cont.)

- **Null distribution**:
  - Chi-squared distribution with $df = 1$
  - Python: stats.chi2.pdf(s, $df = 1$)
- ***p*-value**:
  - Definition: $P(S \geq s_0 \mid H_0)$
  - Python: 1-stats.chi2.cdf(s_0, $df = 1$)



Exercise 3

- Run both the exact McNemar's test and the approximate McNemar's test on the data set provided on page 49

Test 1: exact McNemar's test

	Before (high blood pressure)	Before (normal blood pressure)	
After (high blood pressure)	n_{00} (90)	n_{10} (15)	105
After (normal blood pressure)	n_{01} (22)	n_{11} (102)	124
	112	117	229

- **Data**: contingency table
  - n_{01}: high blood pressure $\overset{\text{take drug}}{\rightarrow}$ normal blood pressure
  - n_{10}: normal blood pressure $\overset{\text{take drug}}{\rightarrow}$ high blood pressure
  - We want to test if there is a significant difference between $\frac{n_{01}}{n_{01}+n_{10}}$ and $\frac{n_{10}}{n_{01}+n_{10}}$
- **Parameter estimate**: $\hat{p} = \frac{\min(n_{01},n_{10})}{n_{01}+n_{10}}$
- **Hypotheses** H_0 and H_A:

$$
\begin{aligned}
H_0: &\quad p = 0.5\\
H_A: &\quad p \neq 0.5
\end{aligned}
$$

- **Significance level** α: 0.05
- **Collected data**: $n_{01} = 22$, $n_{10} = 15$
- **Test statistic**: $s_0 = \min(n_{01}, n_{10}) = \min(22, 15) = 15$

Test 1: exact McNemar's test (cont.)

- **Null distribution**:
  - Binomial distribution with parameters $(n_{01} + n_{10}, 0.5)$
  - Python: stats.binom.pmf(s, $n_{01} + n_{10}$, 0.5)
- *p*-**value**: $2P(S \leq 15 \mid H_0) = 0.3239$
- p-value $> \alpha$: fail to reject H_0

Example implementation in Python:

```
from statsmodels.stats import contingency_tables
contingency_tables.mcnemar(table=[[n00, n10], [n01, n11]],
                              exact=True)
```

Args:

- table: the contingency table
- exact=True: exact test

Test 2: approximate McNemar's test

- **Data**: contingency table
  - n_{01}: high blood pressure $\overset{\text{take drug}}{\rightarrow}$ normal blood pressure
  - n_{10}: normal blood pressure $\overset{\text{take drug}}{\rightarrow}$ high blood pressure
  - We want to test if there is a significant difference between $\frac{n_{01}}{N}$ and $\frac{n_{10}}{N}$
- **Parameter estimate**: $\hat{p}_{01} = \frac{n_{01}}{N}$ and $\hat{p}_{10} = \frac{n_{10}}{N}$
- **Hypotheses H_0 and H_A**:

$$\begin{aligned} H_0 : &\quad p_{01} = p_{10} \\ H_A : &\quad p_{01} \neq p_{10} \end{aligned}$$

- **Significance level**: $\alpha = 0.05$
- **Collected data**: $n_{01} = 22$, $n_{10} = 15$, $n_{00} = 90$, $n_{11} = 102$
- **Test statistic**:

$$s_0 = \frac{(|n_{01} - n_{10}| - 1)^2}{n_{01} + n_{10}} = \frac{(|22 - 15| - 1)^2}{22 + 15} = 0.973$$

Test 2: approximate McNemar's test (cont.)

- **Null distribution**:
  - Chi-squared distribution with $df = 1$
  - Python: stats.chi2.pdf(s, $df = 1$)
- **p-value**: $P(S \geq s_0 \mid H_0) = P(S \geq 0.973 \mid H_0) = 0.324$
- p-value $> \alpha$: fail to reject H_0

Example implementation in Python:

```
from statsmodels.stats import contingency_tables
contingency_tables.mcnemar(table=[[n00, n10], [n01, n11]],
                            exact=False,
                            correction=True)
```

Args:

- table: the contingency table
- exact=False: approximate test
- correction=True: continuity correction (cf. page 52)

Summary

Statistical tests

	Data discrete/continuous	No. of samples collected for the test	Remark	Test statistic	Null distribution
One-sample z-test	Continuous	1	σ known	$\frac{\bar{x}-c}{\sigma/\sqrt{N}}$	Standard Gaussian
One-sample t-test	Continuous	1	σ unknown	$\frac{\bar{x}-c}{s/\sqrt{N}}$	Student's-t distribution
Two-sample z-test	Continuous	2	σ_X, σ_Y known	$\frac{\bar{x}-\bar{y}-c}{\sqrt{\frac{\sigma_X^2}{N_X}+\frac{\sigma_Y^2}{N_Y}}}$	Standard Gaussian
Two-sample t-test	Continuous	2	σ_X, σ_Y unknown	$\frac{\bar{x}-\bar{y}-c}{\sqrt{\frac{s_X^2}{N_X}+\frac{s_Y^2}{N_Y}}}$	Student's-t distribution
Paired t-test	Continuous	1 or 2 (paired)	σ_X, σ_Y unknown	$\frac{m_{X-Y}-c}{s_{X-Y}/\sqrt{N}}$	Student's-t distribution
Binomial test (exact)	Discrete	1	Small N	k_0	Binomial distribution
Binomial test (approximate)	Discrete	1	Large N	$\frac{k_0-N\pi}{\sqrt{N\pi(1-\pi)}}$	Standard Gaussian
McNemar's test (exact)	Discrete	1 (2 groups)	Small $n_{01}+n_{10}$	$\min(n_{01}, n_{10})$	Binomial distribution
McNemar's test (approximate)	Discrete	1 (2 groups)	Large $n_{01}+n_{10}$	$\frac{(\lvert n_{01}-n_{10}\rvert-1)^2}{n_{01}+n_{10}}$	Chi-squared distribution

Today

K-fold cross validation

- **Classifiers**: A and B
- **Data**: evaluation metric; **continuous numerical data**, e.g. accuracies $p_1^A, \cdots, p_K^A$ and $p_1^B, \cdots, p_K^B$ on the K validation sets

	fold 1	fold 2	$\cdots$	fold K
classifier A	p_1^A	p_2^A	$\cdots$	p_K^A
classifier B	p_1^B	p_2^B	$\cdots$	p_K^B
$p_i^A - p_i^B$	$p_1^A - p_1^B$	$p_2^A - p_2^B$	$\cdots$	$p_K^A - p_K^B$

- **Random variable** and **assumption**: $P_1^A, \cdots, P_K^A$, $P_1^B, \cdots, P_K^B$
  - $P_i^A - P_i^B$ i.i.d.
  - $P_i^A - P_i^B \sim \mathcal{N}\left(\mu_{A-B}, \sigma_{A-B}^2\right)$ with unknown standard deviation
- **Parameter of interest**: μ_{A-B}
- **Parameter estimate**: $m_{A-B} = \frac{1}{K}\sum_{i=1}^{K}\left(p_i^A - p_i^B\right)$
- **Hypotheses** H_0 and H_A:

$$\begin{aligned} H_0: &\quad \mu_{A-B} = 0 \\ H_A: &\quad \mu_{A-B} \neq 0 \end{aligned}$$

- **Test statistic**: $t = \frac{m_{A-B}}{s_{A-B}/\sqrt{K}}$, where $s_{A-B} = \sqrt{\frac{1}{K-1}\sum_{i=1}^{K}\left(p_i^A - p_i^B - m_{A-B}\right)^2}$
- **Hypothesis test**: paired t-test

Training-validation split and leave-one-out cross validation

- **Classifiers**: A and B
- **Data**: classifiers A and B tested on the validation data; the outcome x_i^A and x_i^B can be either correct (0) or incorrect (1); **nominal categorical data** with two categories correct or incorrect

	classifier A correct	classifier A incorrect
classifier B correct	n_{00}=count(A correct, B correct)	n_{10}=count(A incorrect, B correct)
classifier B incorrect	n_{01}=count(A correct, B incorrect)	n_{11}=count(A incorrect, B incorrect)

- **Random variable** and **assumption**: $X_i^A \sim Bernoulli(p_A), X_i^B \sim Bernoulli(p_B)$
- **Test statistic**:
  - Small discordance (e.g. $n_{01} + n_{10} < 25$): $s_0 = \min(n_{01}, n_{10})$
  - Large discordance: $s_0 = \frac{(|n_{01} - n_{10}| - 1)^2}{n_{01} + n_{10}}$
- **Hypothesis test**: McNemar's test

Exercise 4

- You have a labeled data set $\mathcal{D} = \{(x_1, y_1), \cdots, (x_N, y_N)\}$
- You developed two classifiers using the 10-fold validation
- Construct a table of the resulting F1 scores for these two classifiers
- Design a hypothesis test to compare these two classifiers

Exercise 5

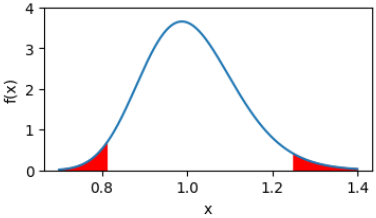
- You have a labeled data set $\mathcal{D} = \{(x_1, y_1), \cdots, (x_N, y_N)\}$
- You developed two classifiers using the training-validation split
- Construct a table of the results for these two classifiers
- Design a hypothesis test to compare these two classifiers

Today

1. Test statistics and hypothesis tests
2. Compare two classifiers
3. More exercises
4. Summary

Exercise 6

- **Setup**: To test equality of variances given two samples from *Gaussian distributions*, we can use the F-test, where the test statistic is $F = \frac{S_X^2}{S_Y^2}$. Let us assume $s_X \geq s_Y$, i.e., we place the larger variance on the numerator. The null distribution is an F-distribution (scipy.stats.f) with parameters $d_1 = N_X - 1$ and $d_2 = N_Y - 1$. Let us perform a two-tailed F-test for $\alpha = 0.05$. The rejection region for each tail is defined by the area corresponding to $\alpha/2$.
- **Questions**: What is the null and alternative hypothesis? Plot the null distribution and indicate the rejection region. Draw a conclusion for a given sample.
- **Answers**:
  - Null hypothesis: $H_0 : \sigma_X^2 = \sigma_Y^2$
  - Alternative hypothesis: $H_A : \sigma_X^2 \neq \sigma_Y^2$
  - Test statistics: $F = \frac{S_X^2}{S_Y^2}$, where S_X and S_Y are sample standard deviations of X and Y, respectively
  - Null distribution: F-distribution (scipy.stats.f) with parameters $d_1 = N_X - 1$ and $d_2 = N_Y - 1$
  - Rejection region: x-axis of the red area (you can show only the right tail)

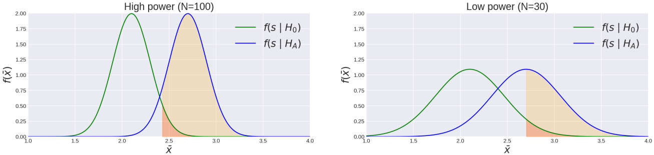


Exercise 7

- Question: company A is trying to sell Duckyphanomin (drug D); what is the sample size needed to achieve 90% power at the alternative $\mu_D = 2.7$ compared to regular diet $\mu_D = 2.1$ (one-sample z-test)
- Hint:
  - Power: correctly rejecting H_0

$$\text{power} = P(\text{reject } H_0 \mid H_A) = 1 - P(\text{type II error})$$

  - CLT: $\bar{X} \sim \mathcal{N}(\mu, \sigma^2/N)$

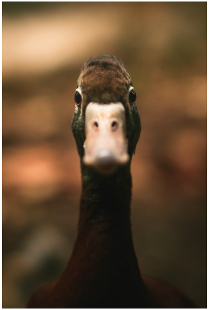


Today

Summary

So far:

- Data types and data containers
- Descriptive data analysis: descriptive statistics, visualization
- Probability distributions, events, random variables, PMF, PDF, parameters
- CDF, Q-Q plot, how to compare two distributions (data vs theoretical, data vs data)
- Modeling
- Parameter estimation: maximum likelihood estimation (MLE) and maximum a posteriori estimation (MAP)
- Classification, multinomial naive Bayes classifier, Gaussian naive Bayes classifier
- Central limit theorem, interval estimation
- Clustering, cluster tendency
- Centroid clustering, k-means, parameter estimation, SSE, Silhouette score
- Gaussian Mixture Models, AIC/BIC
- The EM algorithm
- Hypothesis tests, comparison of two classifiers



I will miss you all...

Happy holidays!