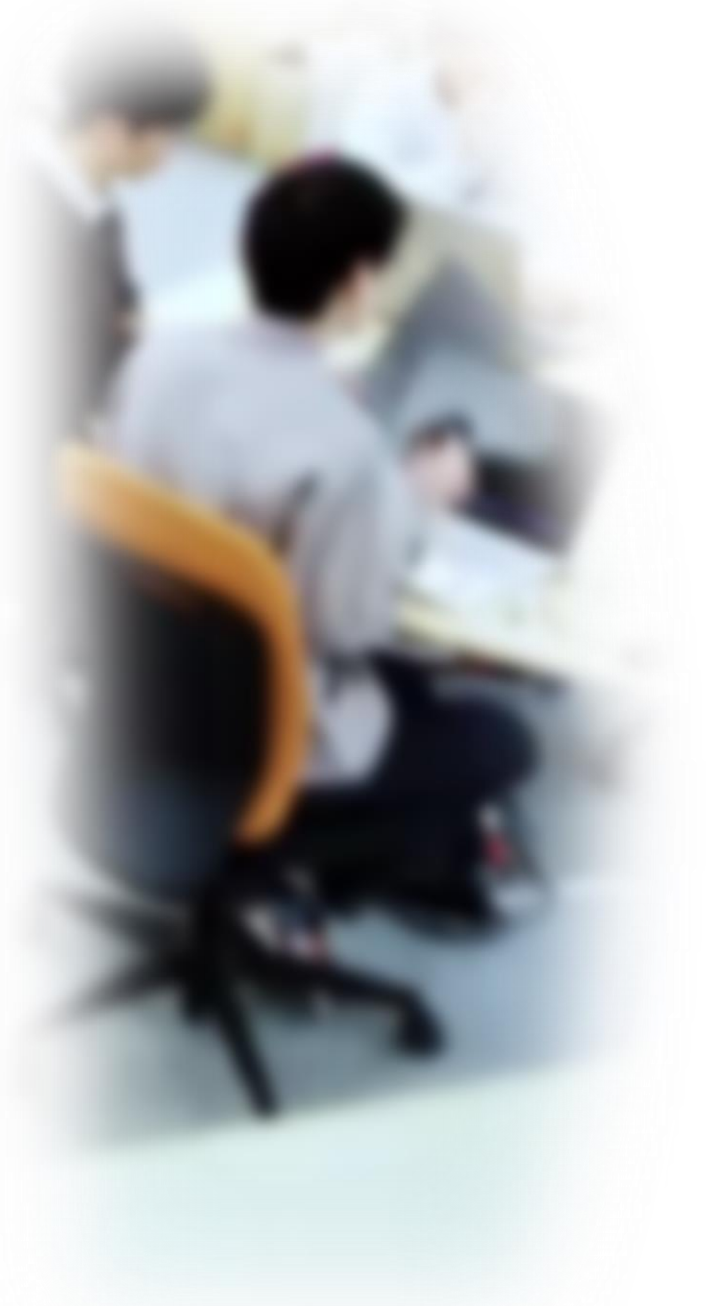
pd-3. 相関，相関係数，ｔ検定

（Python によるデータサイエンス演習）

URL: https://www.kkaneko.jp/cc/pd/index.html

金子邦彦



アウトライン

1. 相関と相関係数
2. Iris データセットでの相関係数の算出
3. 母集団と標本
4. t 検定

相関

- **相関**は，2つの変数の間に関連性があるかを示す
- **相関がある**場合，**一方が変化すると，もう一方も変化する傾向**にある

【相関ありの場合】

- X が**増える**と，Y が**増える**傾向がある（**正の相関**）

　　勉強時間が増えると、得点が上がる

- X が**増える**と，Y が**減る**傾向がある（**負の相関**）

　　ガソリン代が上がると、車の利用が減る

【相関なしの場合】

X と Y に関係がない

　　足のサイズと勉強時間に関係がない

相関係数

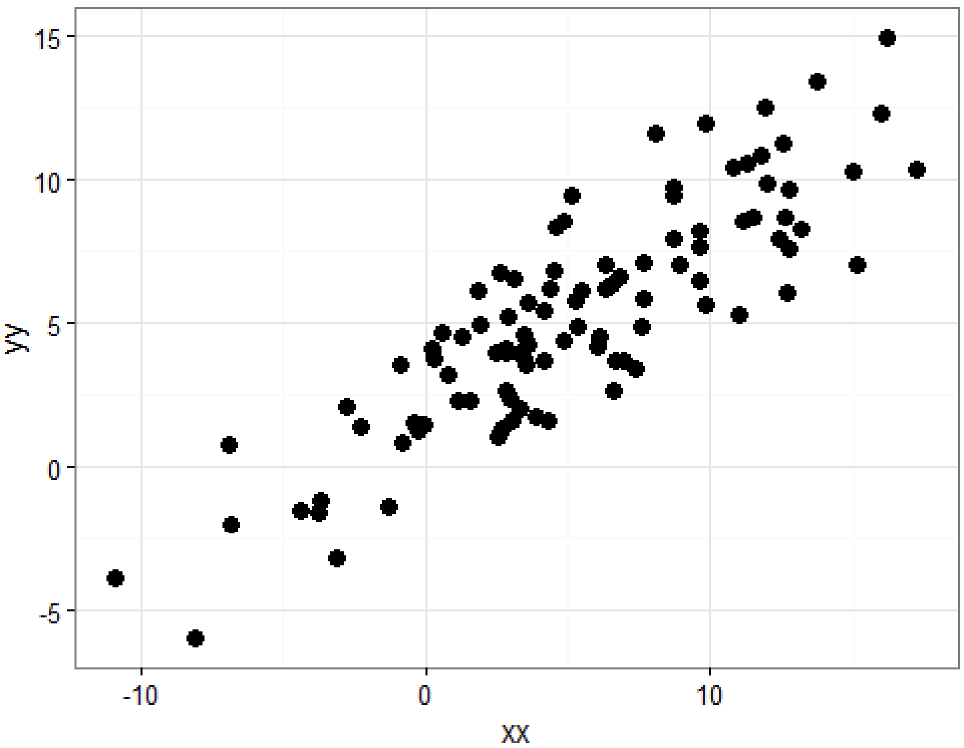
- **相関係数**は，**相関**を算出した**数値．範囲は－１から１まで**
- 相関係数を算出することで、変数間の関係の分析ができる

１に近い値：　相関あり．正の相関

０に近い値：　相関なし

－１に近い値：　相関なし．負の相関

正の相関



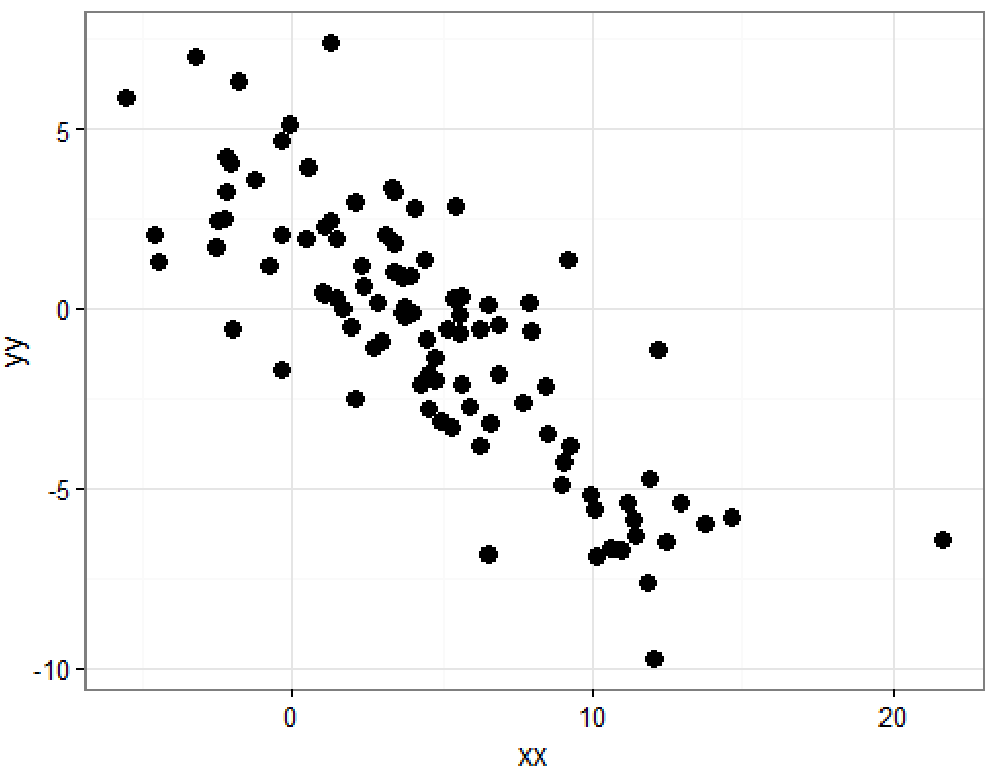
２つの変数 xx, yy に相関がある

xx の値が**<u>増える</u>**と
yy の値が**<u>増える</u>**傾向がある
（正の相関）

相関係数の算出結果

0.8620027 (1 に近い値)

負の相関



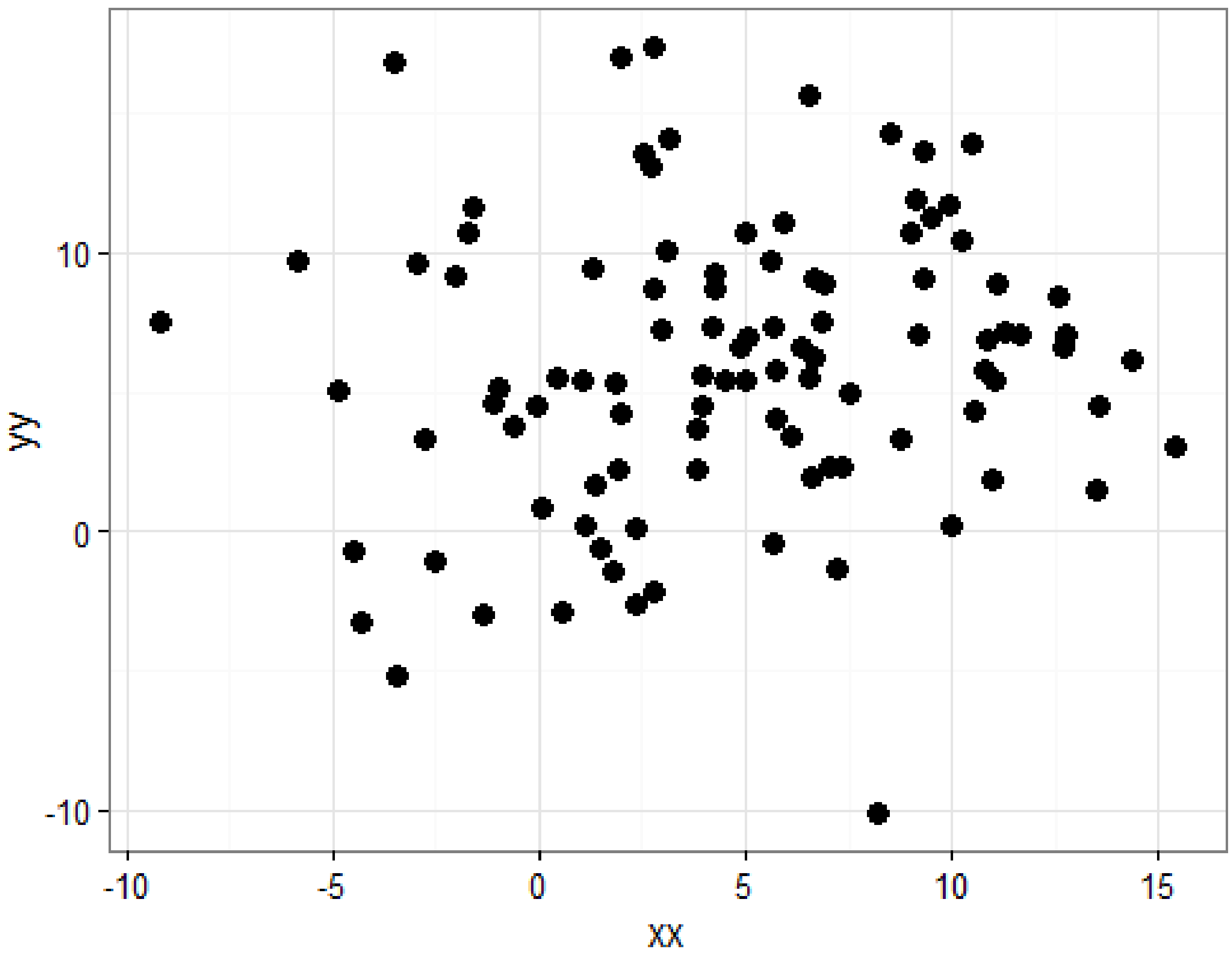
２つの変数 xx, yy に相関がある

xx の値が**増える**と
yy の値が**減る**傾向がある
（負の相関）

相関係数の算出結果

-0.8502535 (-1 に近い値)

相関なし

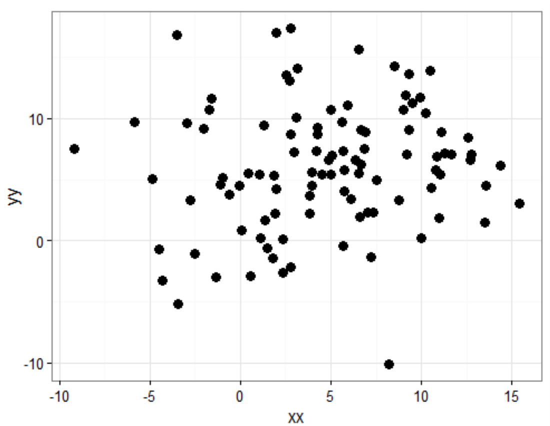


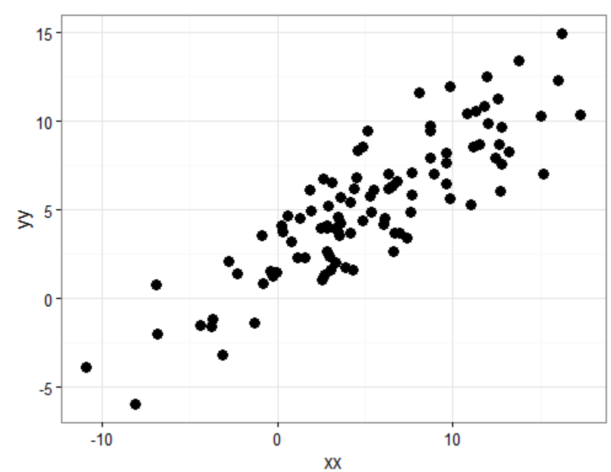
相関係数の算出結果

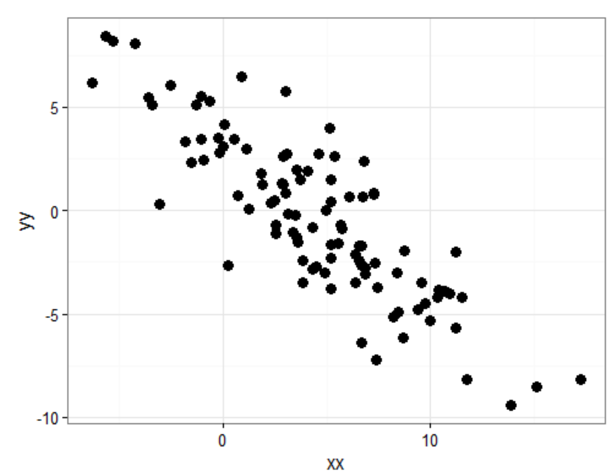
0.1252164 (0 に近い値)

相関係数のまとめ

- １に近い値：　相関あり．正の相関
- ０に近い値：　相関なし
- ー１に近い値：　相関なし．負の相関







0.1252164 (0 に近い値)

相関なし

0.8620027 (1 に近い値)

正の相関

-0.8502535 (-1 に近い値)

負の相関

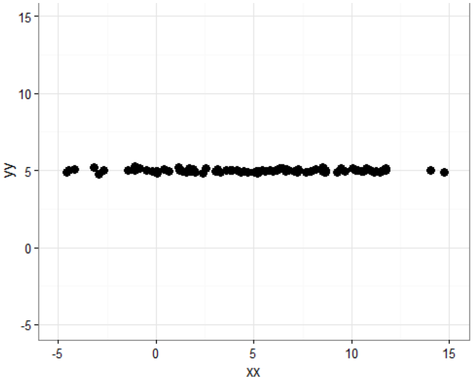
相関係数の活用例

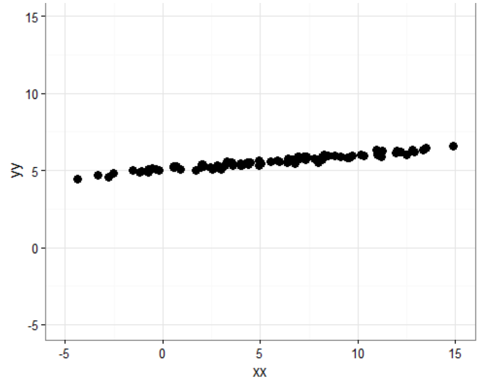
２つの量の間の関係性の分析

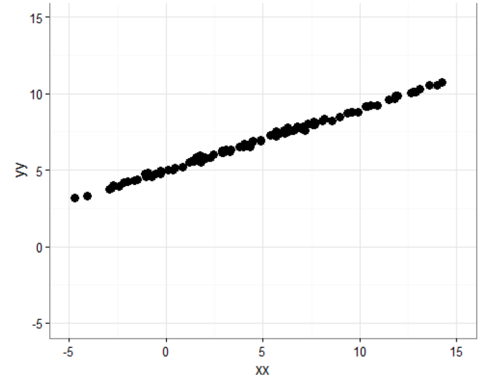
- 広告を増やすと，売上高が増えそうか
- 相関が高い複数の金融商品を扱うと，リスクが高いか
- 遺伝子と疾患に関係がありそうか

相関係数の性質

「**相関の強弱**」の尺度である． 「傾き」では**ない**








```
> cor(d9$xx, d9$yy)
[1] -0.06027409
```

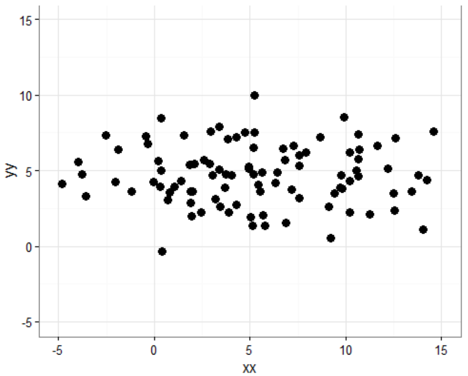
```
> cor(d10$xx, d10$yy)
[1] 0.9743955
```

1 に近い値

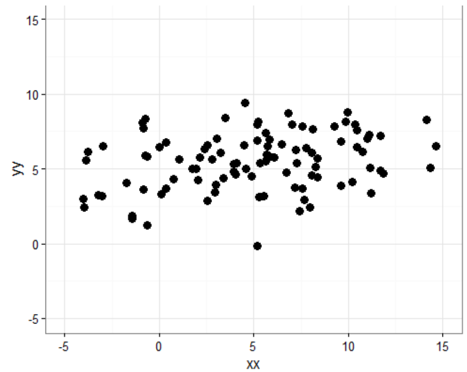
```
> cor(d11$xx, d11$yy)
[1] 0.998944
>
```

1 に近い値

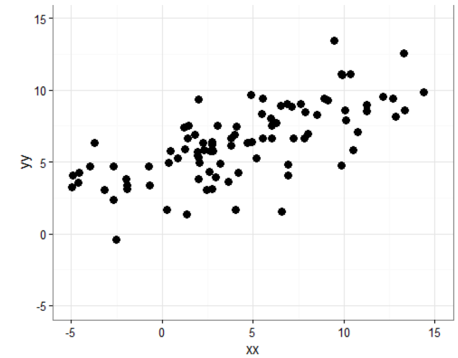
相関係数の例




```
> cor(d12$xx, d12$yy)
[1] -0.02723688
>
```




```
> cor(d13$xx, d13$yy)
[1] 0.2808435
>
```




```
> cor(d14$xx, d14$yy)
[1] 0.7268933
>
```

ここまでのまとめ

- **相関がある**場合，**一方が変化すると，もう一方も変化する傾向**にある
  - **１に近い値：　相関<u>あり</u>．正の相関**
  - **０に近い値：　相関なし**
  - **－１に近い値：　相関<u>なし</u>．負の相関**

- ３つ以上の変数があるとき、相関係数は多数求まる

　変数　　A, B, C　に対して

　　A と B の相関係数,

　　B と C の相関係数,

　　C と A の相関係数

4-2 Iris データセットでの相関係数の算出

Python による相関係数の算出

① ライブラリのインポート

```
import numpy as np
```

② データの準備
（ここでは、h, w を準備）

```
h = np.array([160, 170, 160, 180, 170])
w = np.array([60, 70, 50, 70, 60])
```

③ numpy の corrcoef を用いて相関係数行列を算出

```
print(np.corrcoef(h, w))
```

2×2 の行列が表示される．**右上、左下**が**相関係数**（同じ値）

```
[[1.         0.78571429]
 [0.78571429 1.        ]]
```

Python による相関係数の算出（Google Colaboratory）

```
import numpy as np

h = np.array([160, 170, 160, 180, 170])
w = np.array([60, 70, 50, 70, 60])

print(np.corrcoef(h, w))
```

ライブラリのインポート

データの準備

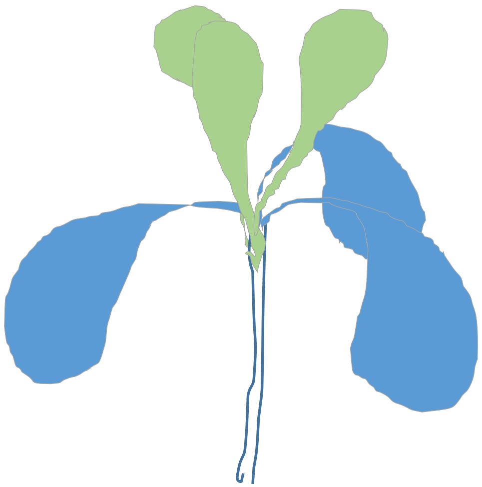
相関係数の算出

コード

```
import numpy as np
h = np.array([160, 170, 160, 180, 170])
w = np.array([60, 70, 50, 70, 60])
print(np.corrcoef(h, w))
```

```
[[1.         0.78571429]
 [0.78571429 1.        ]]
```

アヤメ属 (Iris)



- 多年草
- 世界に 150種. 日本に 9種.
- 花被片は 6個
- 外花被片（がいかひへん） Sepal
  3個（大型で下に垂れる）
- 内花被片（ないかひへん） Petal
  3個（直立する）

Iris データセット

```
[[5.1 3.5 1.4 0.2]
 [4.9 3.  1.4 0.2]
 [4.7 3.2 1.3 0.2]
 [4.6 3.1 1.5 0.2]
 [5.  3.6 1.4 0.2]
 [5.4 3.9 1.7 0.4]
 [4.6 3.4 1.4 0.3]
 [5.  3.4 1.5 0.2]
 [4.4 2.9 1.4 0.2]
 [4.9 3.1 1.5 0.1]
 [5.4 3.7 1.5 0.2]
 [4.8 3.4 1.6 0.2]
 [4.8 3.  1.4 0.1]
 [4.3 3.  1.1 0.1]
 [5.8 4.  1.2 0.2]
 [5.7 4.4 1.5 0.4]
 [5.4 3.9 1.3 0.4]
 [5.1 3.5 1.4 0.3]
 [5.7 3.8 1.7 0.3]
 [5.1 3.8 1.5 0.3]
 [5.4 3.4 1.7 0.2]
 [5.1 3.7 1.5 0.4]
 [4.6 3.6 1.  0.2]
```

0列 1列 2列 3列

```
[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 2
 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
 2 2]
```

Iris データセット

- **3種のアヤメ**の外花被辺、内花被片の幅と長さを計測したデータセット

  Iris setosa

  Iris versicolor

  Iris virginica

- データ数は 150 (50 × 3)
- 作成者：Ronald Fisher
- 作成年：1936

今から行うこと

```
[[5.1 3.5 1.4 0.2]
 [4.9 3.  1.4 0.2]
 [4.7 3.2 1.3 0.2]
 [4.6 3.1 1.5 0.2]
 [5.  3.6 1.4 0.2]
 [5.4 3.9 1.7 0.4]
 [4.6 3.4 1.4 0.3]
 [5.  3.4 1.5 0.2]
 [4.4 2.9 1.4 0.2]
 [4.9 3.1 1.5 0.1]
 [5.4 3.7 1.5 0.2]
 [4.8 3.4 1.6 0.2]
 [4.8 3.  1.4 0.1]
 [4.3 3.  1.1 0.1]
 [5.8 4.  1.2 0.2]
 [5.7 4.4 1.5 0.4]
 [5.4 3.9 1.3 0.4]
 [5.1 3.5 1.4 0.3]
 [5.7 3.8 1.7 0.3]
 [5.1 3.8 1.5 0.3]
 [5.4 3.4 1.7 0.2]
 [5.1 3.7 1.5 0.4]
 [4.6 3.6 1.  0.2]
```

0列 1列 2列 3列

Iris データセット

元データは４次元

４次元を２次元に削減する

２つの方法

①Iris データセットの０，１列目

②主成分分析により２次元に次元削減

Iris データセットのロード（Google Colaboratory）

```
import matplotlib.pyplot as plt
from sklearn.datasets import load_iris
iris = load_iris()
x = iris.data
y = iris.target
print(x)
print(y)
```

ライブラリのインポート

x, y にロード、
（x は２次元の配列、
y は１次元の配列）

表示

コード

Iris データセットのロード（Google Colaboratory）

```
import matplotlib.pyplot as plt
from sklearn.datasets import load_iris
iris = load_iris()
x = iris.data
y = iris.target
print(x)
print(y)
```

```
[[5.1 3.5 1.4 0.2]
 [4.9 3.  1.4 0.2]
 [4.7 3.2 1.3 0.2]
 [4.6 3.1 1.5 0.2]
 [5.  3.6 1.4 0.2]
 [5.4 3.9 1.7 0.4]
 [4.6 3.4 1.4 0.3]
 [5.  3.4 1.5 0.2]
 [4.4 2.9 1.4 0.2]
 [4.9 3.1 1.5 0.1]
 [5.4 3.7 1.5 0.2]
 [4.8 3.4 1.6 0.2]
 [4.8 3.  1.4 0.1]
 [4.3 3.  1.1 0.1]
 [5.8 4.  1.2 0.2]
 [5.7 4.4 1.5 0.4]
 [5.4 3.9 1.3 0.4]
 [5.1 3.5 1.4 0.3]
```

Iris データセットの０，１列目（Google Colaboratory）

```
import matplotlib.pyplot as plt
from sklearn.datasets import load_iris
iris = load_iris()
x = iris.data
y = iris.target
# グラフ
plt.scatter(x[:, 0], x[:, 1])
# ラベル
plt.xlabel(iris.feature_names[0])
plt.ylabel(iris.feature_names[1])
plt.title('Iris dataset')
# 表示
plt.show()
```

ライブラリのインポート

Irisデータセットをロード

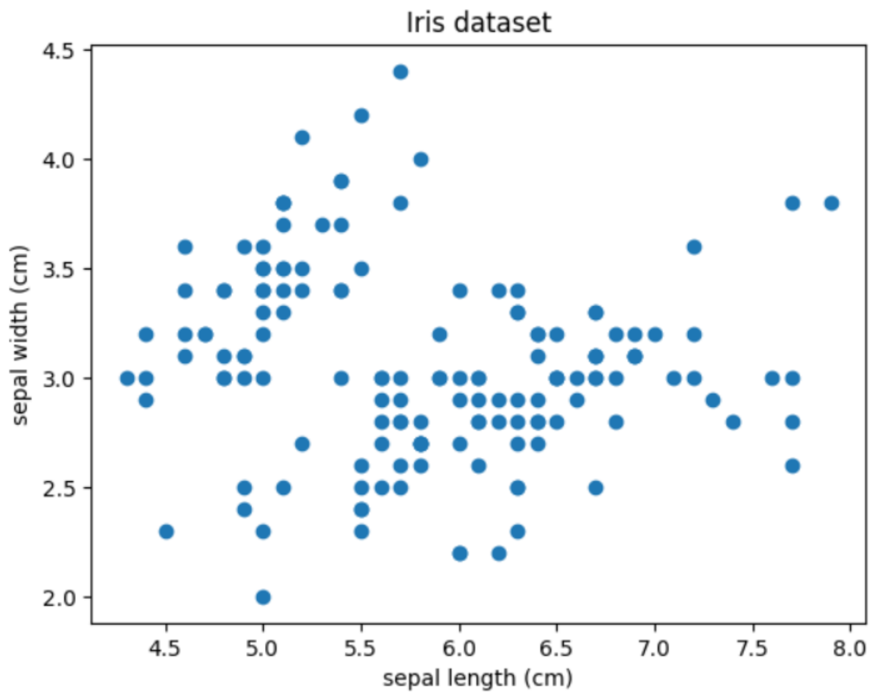
グラフの設定

（0列1列が縦横に）

表示

Iris データセットの０，１列目 （Google Colaboratory）

```
import matplotlib.pyplot as plt
from sklearn.datasets import load_iris
iris = load_iris()
x = iris.data
y = iris.target
# グラフ
plt.scatter(x[:, 0], x[:, 1])
# ラベル
plt.xlabel(iris.feature_names[0])
plt.ylabel(iris.feature_names[1])
plt.title('Iris dataset')
# 表示
plt.show()
```



Iris データセットの０，１列目の相関係数（Google Colaboratory）

```
import matplotlib.pyplot as plt
import numpy as np
from sklearn.datasets import load_iris
iris = load_iris()
x = iris.data
y = iris.target
np.corrcoef(x[:, 0], x[:, 1])
```

ライブラリのインポート

Irisデータセットをロード

相関係数の算出

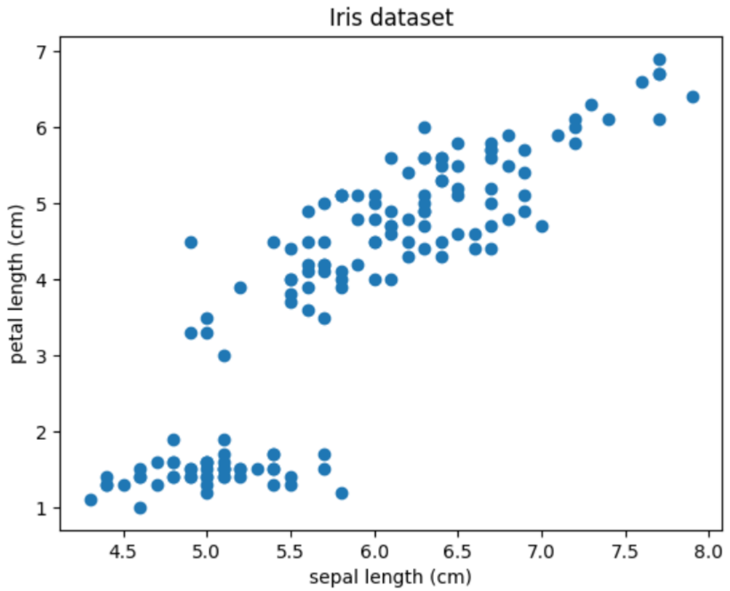
```
import matplotlib.pyplot as plt
import numpy as np
from sklearn.datasets import load_iris
iris = load_iris()
x = iris.data
y = iris.target
np.corrcoef(x[:, 0], x[:, 1])
```

```
array([[ 1.        , -0.11756978],
       [-0.11756978,  1.        ]])
```

0 に近い値

- Iris データセットの０列目と２列目について、散布図の作成、相関係数の算出を行ってみなさい

```
import matplotlib.pyplot as plt
from sklearn.datasets import load_iris
iris = load_iris()
x = iris.data
y = iris.target
# グラフ
plt.scatter(x[:, 0], x[:, 2])
# ラベル
plt.xlabel(iris.feature_names[0])
plt.ylabel(iris.feature_names[2])
plt.title('Iris dataset')
# 表示
plt.show()
```




```
import matplotlib.pyplot as plt
import numpy as np
from sklearn.datasets import load_iris
iris = load_iris()
x = iris.data
y = iris.target
np.corrcoef(x[:, 0], x[:, 2])
```

```
array([[1.        , 0.87175378],
       [0.87175378, 1.        ]])
```

4-3 母集団と標本

母集団

母集団は，調査や研究の対象となる**全体の集団**のこと

- **母集団**の**把握と理解**が重要

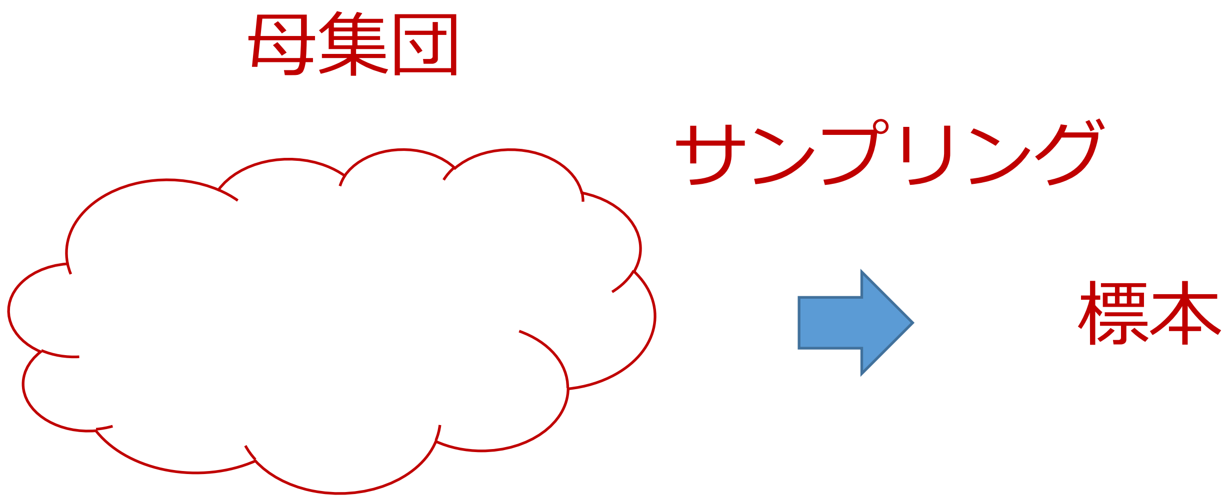
（例）人類全体、２０歳以上の人類全体

サンプリングと標本

- **<u>母集団全体を調べることが困難な場合</u>**、**サンプリング**を適切に行う

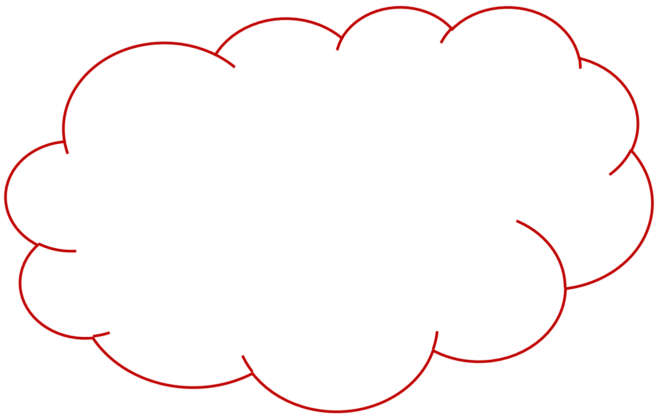
　　　（例）１０００名をランダムに選ぶ

- **サンプリング**は、**<u>母集団から一部を選ぶ</u>**こと。
- 母集団全体を調べるのでなく、**<u>一部を調べる</u>**ことになる。
- **標本**は、**<u>サンプリングで選ばれたもの</u>**のこと。



サンプリングと標本

母集団



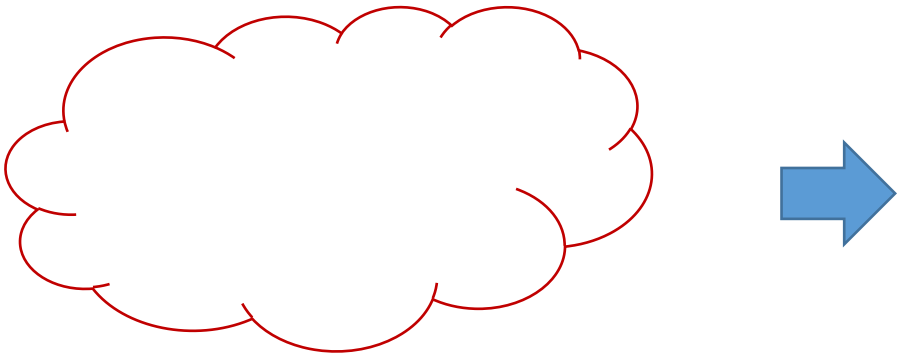
あるときの標本	別の標本
128	118
104	110
124	96
85	85
120	109
平均	平均
112.2	103.6

選ばれた標本によっては、
値が違い、平均なども異なってくる

十分な数の標本が必要

あるときの標本

母集団



128
104
124
85
120

- 標本の大きさが**小さい**と、**結果の信頼性が下がる**
- **十分な数の標本を得る**ことが重要
- 標本の大きさの決定は簡単に決めることができない
- 母集団の特徴、調査や研究の目的に**よって，適切な標本の大きさは変わる**ことに注意しよう

まとめ

- **母集団**：調査や研究の対象となる**全体の集団**
- **サンプリング**：

母集団全体を調べることが困難な場合、**母集団から一部を選ぶサンプリング**を行う。

母集団の特徴や性質を**推測**することが可能となる。

- **標本**：

標本は、母集団からサンプリングで選ばれた**母集団の一部**。

標本から得られたデータを分析し、**母集団全体の性質や傾向を推測**可能。

【注意点】**十分な標本サイズの確保**が必要。ランダムに選択するなどの考慮が重要。

4-4 t 検定

t 検定

t 検定は、**２つの標本**の**平均値**が**統計的に有意に異なるかどうかを**判断するための**統計手法**

【注意点】

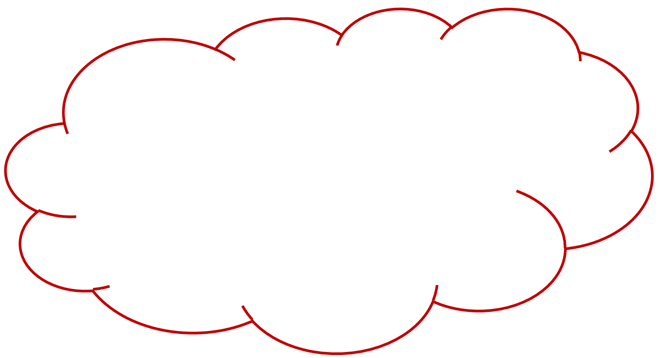
- **標本が正規分布**に従っていること
- **外れ値**が存在する場合は、取り除いたり、適切に修正すること
- **十分な標本サイズ**を確保すること．小さな標本サイズでは、結果の信頼性が下がる可能性がある

複数の母集団

- **母集団が複数あるという考え方は重要**

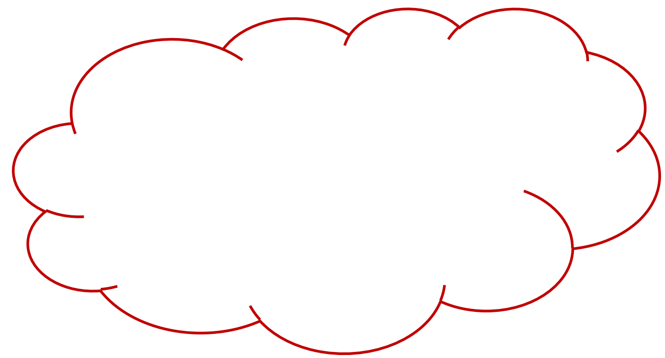
あなたは大学生です。授業Aを受けた人と、授業Aを受けていない人の調査し、比較してみたいと考えました

母集団



授業Aを受けた人

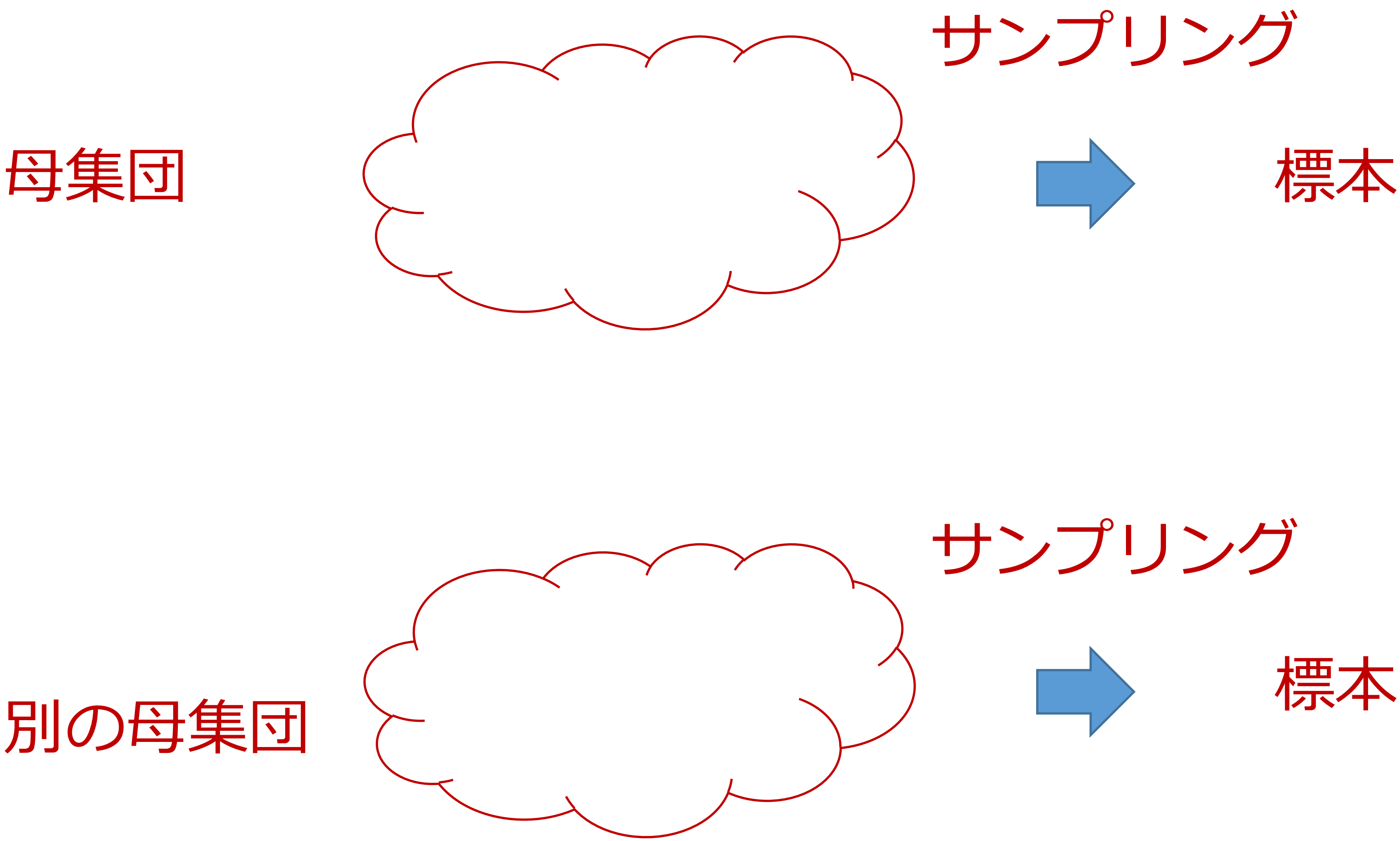
母集団



授業Aを受けていない人

母集団が２つ

２つの母集団と２つの標本



t 検定と p 値

- **t 検定**は、**２つの標本**の**平均値**が**統計的に有意に異なるかどうかを**判断するための**統計手法**

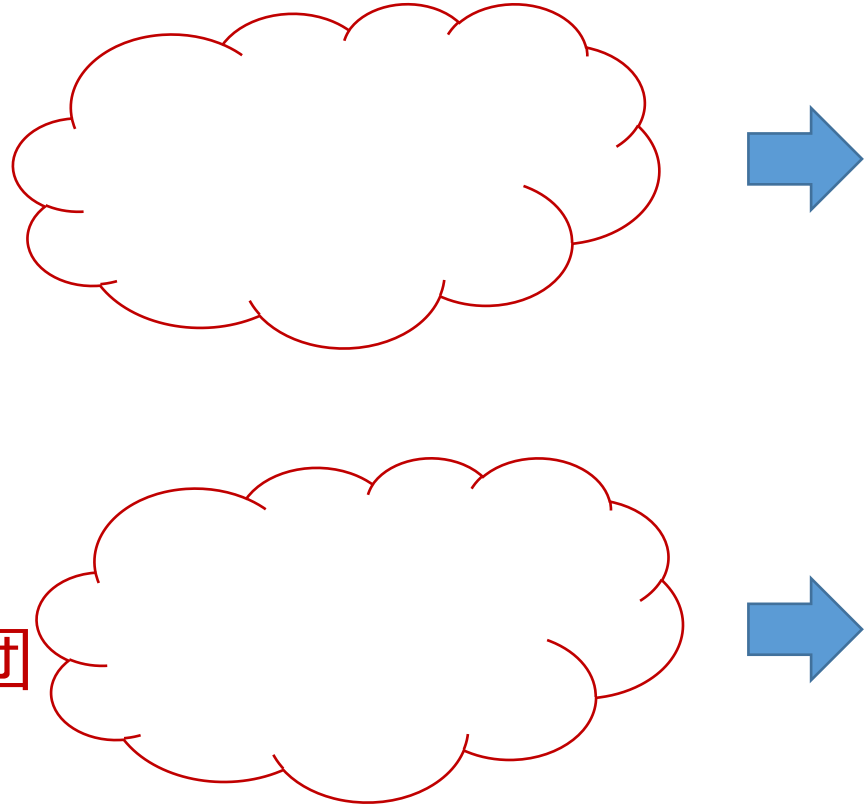
- **p 値**は、**２つの標本の差が偶然による**（有意でない）確率を示す

p 値 = 0.99 のとき．「偶然による確率は９９％」
⇒**有意であるとも有意でないともいえない**

P 値 = 0.0005 のとき、「偶然による確率は０．０５％」
⇒おそらく**有意である**

２つの母集団と２つの標本

母集団



128
104
124
85
120

別の母集団

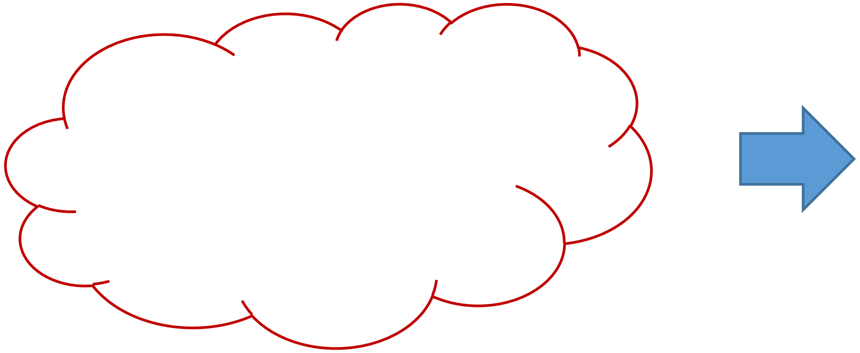
180
191
189
131
130
150

２つの標本から t 検定の
p 値を算出： 0.006908

２つの標本の差が偶然による
（有意でない）**確率が低い**

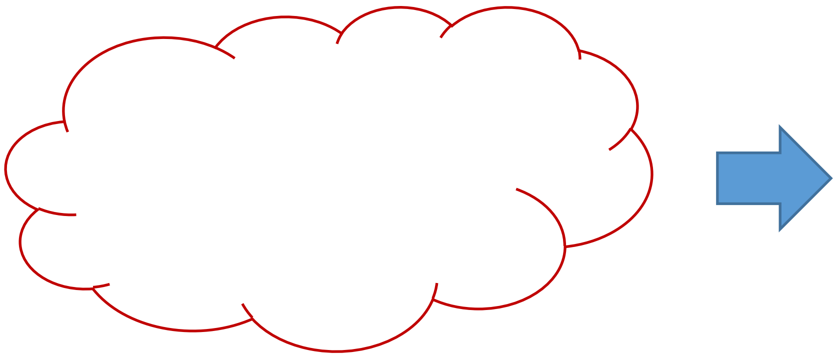
２つの母集団と２つの標本

母集団



128
104
124
85
120

別の母集団



100
106
89
89
105

２つの標本から t 検定の
p 値を算出：0.1541

有意であるとも有意でないとも言えない

p 値と有意性

- **t 検定**の **p 値**は、**２つの標本の差が偶然による**確率を示す
- **p値が小さい**とき「**とても偶然とは思えず、有意である**」と考える
- **p値が大きい**ときは「**偶然であるとも、偶然でないとも言えない**」と考える

ここまでのまとめ

- **t検定**
  - **t検定**は、**2つの標本の平均値の統計的な有意性**を判断する**統計手法**
  - **標本が正規分布**に従い、**外れ値**を適切に扱い、**十分な標本サイズ**を確保することが重要
- **t検定のp値**
  - t検定のp値は、**2つの標本の差が偶然**である**確率**
  - p値が低いとき、差が統計的に有意である可能性が高まる
- **p値の解釈**
  - **p値が小さい**とき、「**差は統計的に有意であり、偶然とは考えにくい**」と考える
  - **p値が大きい**ときは、「差は統計的に有意であるとは言いきれない。**偶然であるとも、偶然でないとも言えない**」と考える

Python による t 検定

① ライブラリのインポート

from scipy import stats

② データの準備

（ここでは、sample1、sample2 を準備）

sample1 = [86, 93, 91, 89, 92]

sample2 = [77, 81, 84, 85, 80]

③ scipy の ttest_ind を用いて t 検定を実施

t, p = stats.ttest_ind(sample1, sample2)

④ p 値の表示

print(p)

```
from scipy import stats
sample1 = [86, 93, 91, 89, 92]
sample2 = [77, 81, 84, 85, 80]
t, p = stats.ttest_ind(sample1, sample2)
print(p)
```

```
0.001670538621824343
```

p 値 < 0.05 が，**判断の分かれ目**の目安という考え方も

t 検定（Google Colaboratory）

```
from scipy import stats
sample1 = [86, 93, 91, 89, 92]
sample2 = [77, 81, 84, 85, 80]
t, p = stats.ttest_ind(sample1, sample2)
print(p)
```

ライブラリのインポート
データの準備
t 検定の実施
p 値の表示

コード

```
from scipy import stats
sample1 = [86, 93, 91, 89, 92]
sample2 = [77, 81, 84, 85, 80]
t, p = stats.ttest_ind(sample1, sample2)
print(p)
```

```
0.0016705386218243
```

演習の例

1. サイズが 5 以上の数値データを，2個準備しなさい

データ1	
データ2	

2. 1のデータについて t 検定を行い，その p 値を求めなさい

＜p値＞