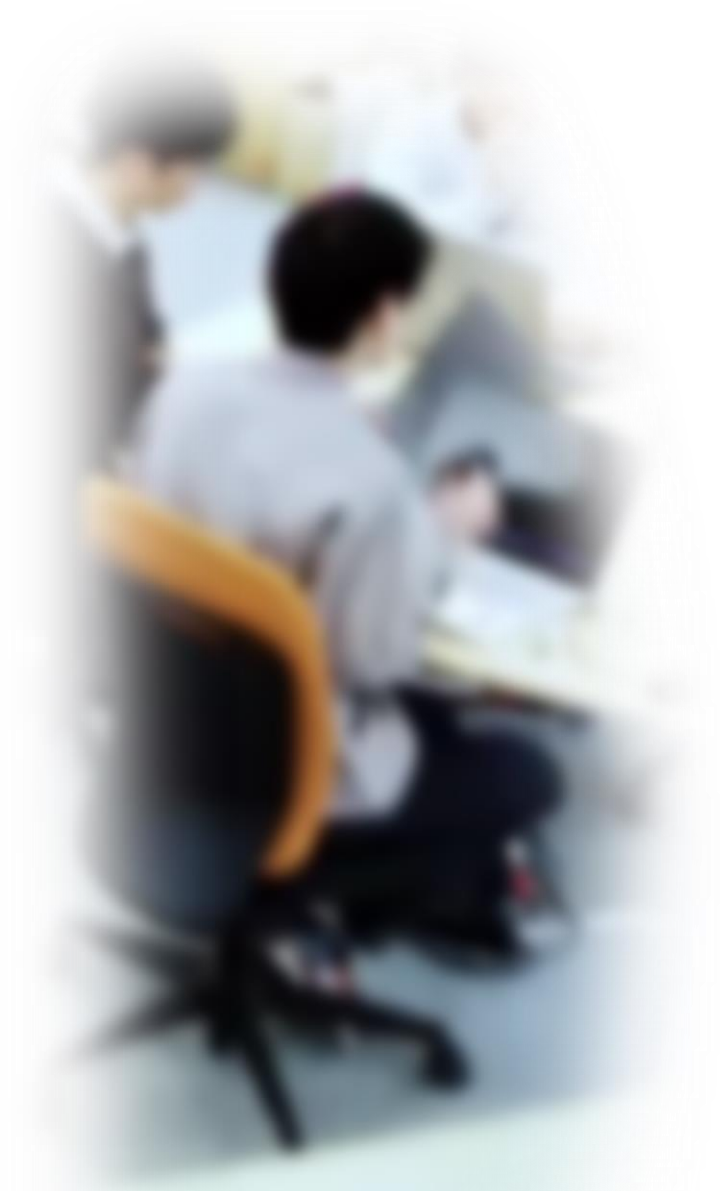
pd-1. データサイエンス，散布図，平均，分布

（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

CC BY NC SA



アウトライン

1. データサイエンス
2. 散布図
3. 平均
4. 分布、密度

1. データサイエンス

データサイエンスでできること

- **データサイエンス**は、**データから正しい知見や結論を導き出す**ための学問であり、**さまざまな分野で活用されている**
  - ビジネス分野：顧客の嗜好やニーズを分析し、マーケティング戦略の立案を行う
  - 医療分野：病気の早期発見や効果的な治療法の開発を行う
  - 工学分野：製品品質の改善や予測保全など、生産の最適化を行う

- **情報化社会**において、**多くのデータが生み出されている。**データサイエンスは、**将来の活躍につながる**
- データサイエンスは、**大量のデータを扱う**もの。**機械学習**など**人工知能**や**情報処理**とも大きく**関連**する。**さまざまな分野でデータを活用する実力**につながる。

研究レポートの書くべき６つの要素

データサイエンスのスキルは、**研究レポートの作成**でも有用

① **問題：研究の背景と目的を明確に説明し、どのような問題に対処するかを示す**

② **仮説**：研究で提起する仮説は、**問題を解決するための自分のアイデアや仮説**です。

③ **実験手順：研究の手順を明確に示す。**

④ **結果**：**研究の結果を正確かつ明確に示すこと**が必要です。グラフや表などを用いて視覚的に表現すると、結果が分かりやすくなります。また、研究結果に関連する数値や統計的な情報も提供しましょう。

⑤ **考察**：**研究結果に基づいた分析**です。

⑥ **引用文献：関連する先行研究や参考にした文献を明示**

2. 散布図

Python のリスト

リストは，**順序のついたデータの並び**

a = [100, 200, 300]

b = [1, "apple", True]　　異なる種類のデータ混在可

```
a = [100, 200, 300]
print(a)
```

```
[100, 200, 300]
```

```
b = [1, "apple", True]
print(b)
```

```
[1, 'apple', True]
```

Python のデータフレーム

データフレームは，**表形式のデータ．**リストなどから作成可能

```
import pandas as pd
d = [[1, 'apple', 100],
        [2, 'orange', 300],
        [3, 'orange', 200]]
df = pd.DataFrame(d, columns=['id', 'name', 'price'])
```

```
import pandas as pd
d = [[1, 'apple', 100],
        [2, 'orange', 300],
        [3, 'orange', 200]]
df = pd.DataFrame(d, columns=['id', 'name', 'price'])
print(df)
```

```
   id    name  price
0   1   apple    100
1   2  orange    300
2   3  orange    200
```

Python の配列（アレイ）

配列は，**データの並び**

```
import numpy as np
a = np.array([[1, 2, 3], [4, 5, 6]])
print(a)
```

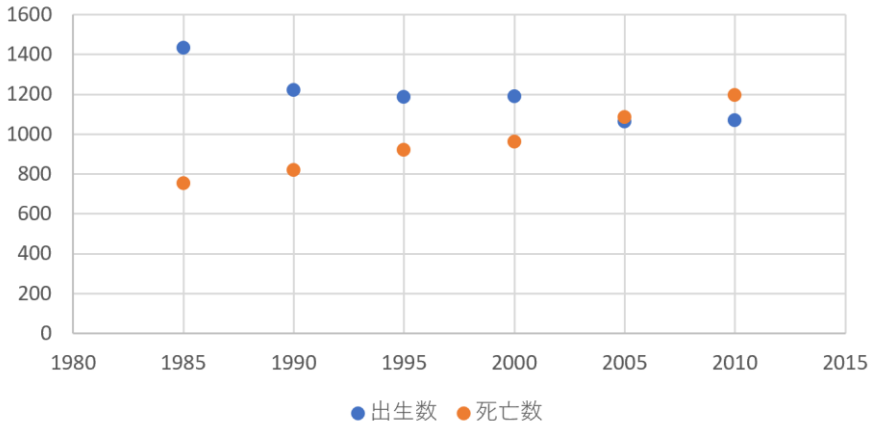
２次元（縦横）、３次元（縦横高さ）など、
高い次元の配列を扱うことができる

```
import numpy as np
a = np.array([[1, 2, 3], [4, 5, 6]])
print(a)
```

```
[[1 2 3]
 [4 5 6]]
```

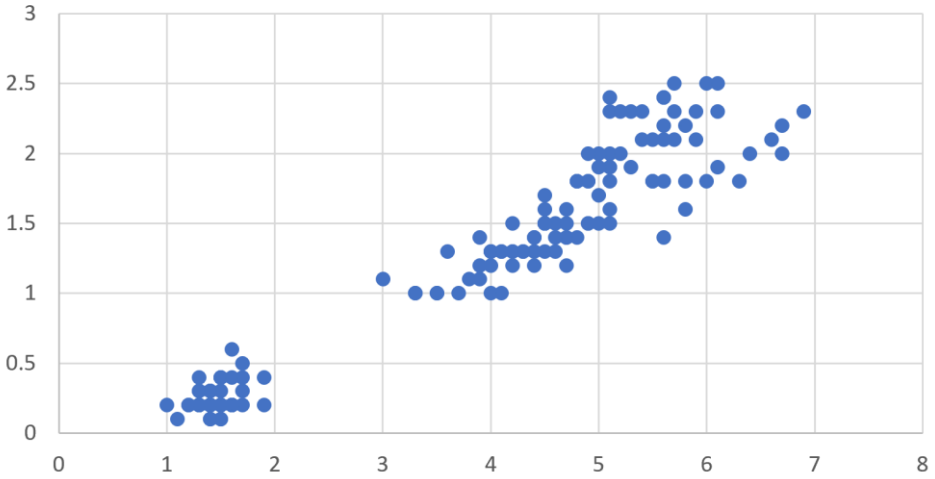
散布図の用途

- 時間変化



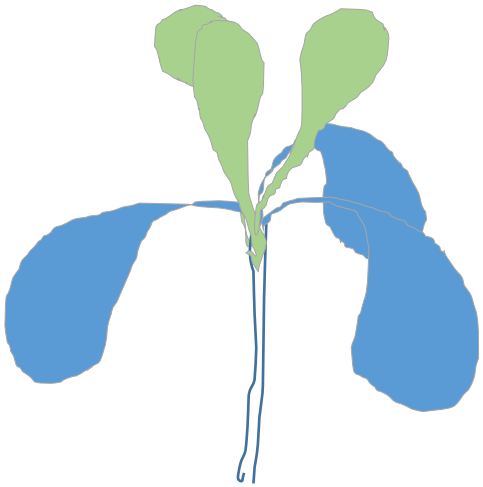
横軸は時間.
散布図から,
時間変化を読み取る

- 分布



横軸と縦軸は, ２つの量.
散布図から,
２つの量の間の**関係**を見る

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

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)
```

コード

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

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]
```

Python による散布図の描画

① ライブラリのインポート

```
import matplotlib.pyplot as plt
import japanize_matplotlib
```

② データの準備
ここでは，Iris データセットを使用

```
from sklearn.datasets import load_iris
iris = load_iris()
x = iris.data
y = iris.target
```

③ 散布図の描画

```
plt.scatter(x[:, 0], x[:, 1])
```

④ ラベルの設定

```
plt.xlabel(iris.feature_names[0])
plt.ylabel(iris.feature_names[1])
plt.title('Iris データセット')
```

⑤ 描画結果の表示

```
plt.show()
```

japanize_matplotlib は，
凡例やラベルで日本語を
使いたいとき

Iris データセットの散布図（Google Colaboratory）

```
!pip install japanize-matplotlib
```

機能のインストール

```
import matplotlib.pyplot as plt
import japanize_matplotlib
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 データセット')
# 表示
plt.show()
```

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）

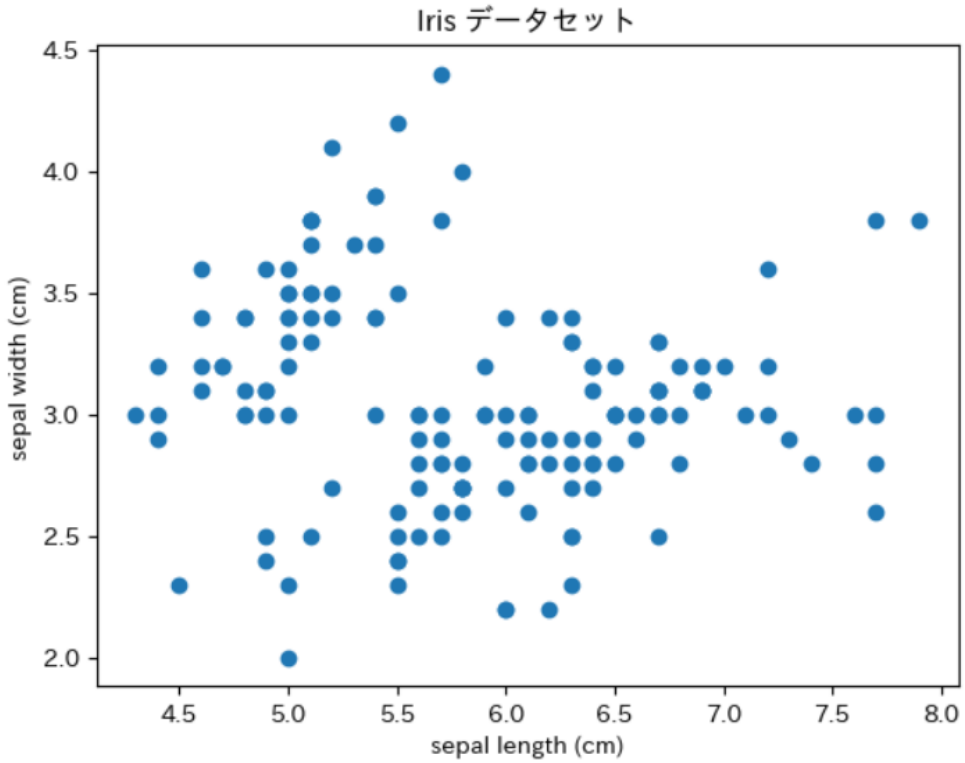
散布図の描画
（0列1列が縦横に）

ラベルと凡例の設定

描画結果の表示

Iris データセットの散布図（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 データセット')
# 表示
plt.show()
```



Iris データセットの散布図、色付き（Google Colaboratory）

```
!pip install japanize-matplotlib
```

機能のインストール

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

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）

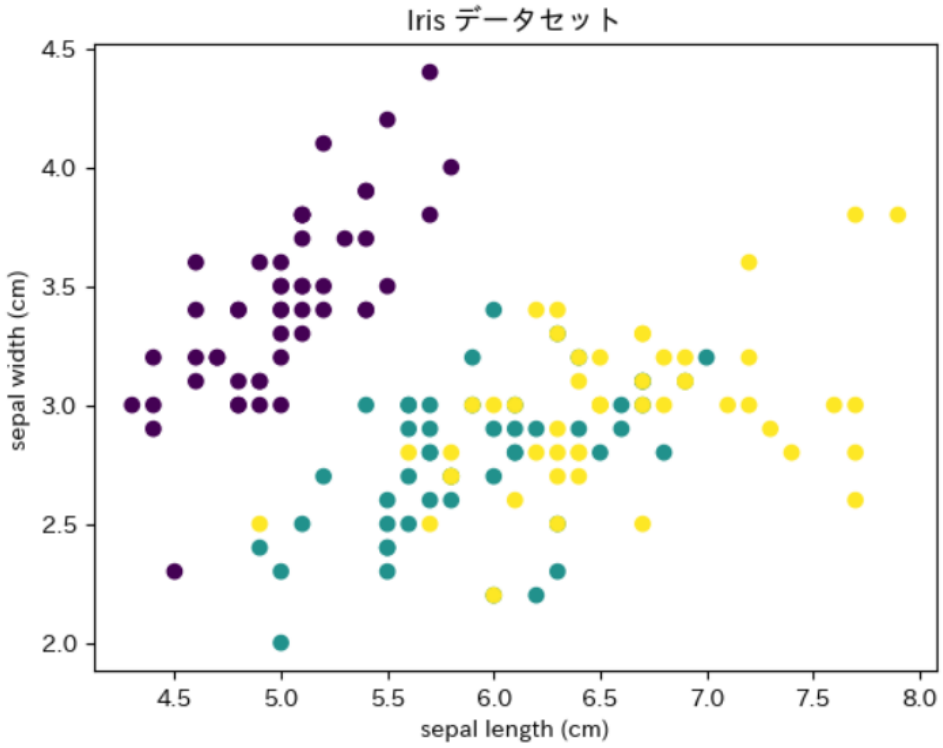
散布図の描画
（0列1列が縦横に。
c=y で色を設定）

ラベルと凡例の設定

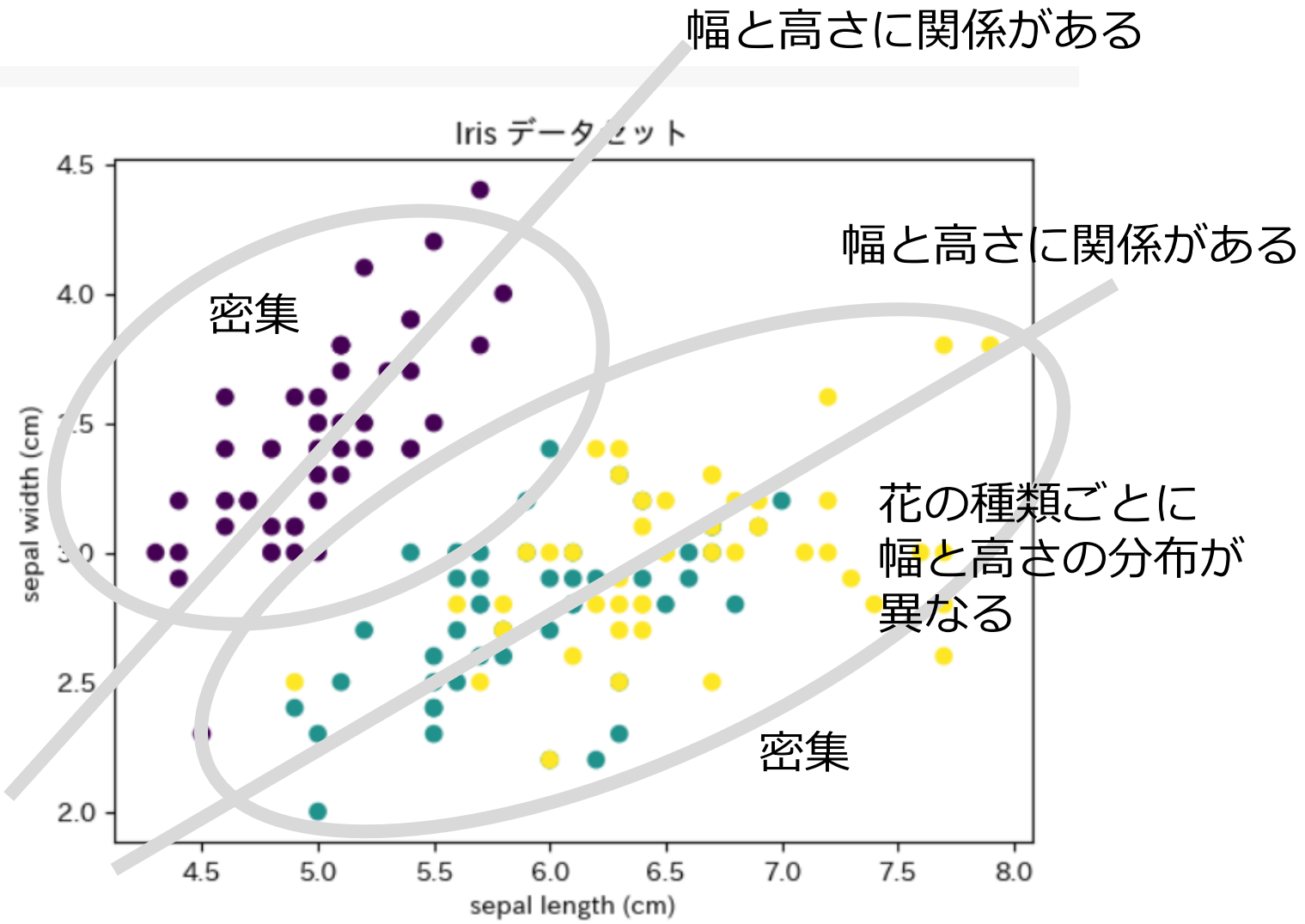
表示

Iris データセットの散布図、色付き（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], c=y)
# ラベル
plt.xlabel(iris.feature_names[0])
plt.ylabel(iris.feature_names[1])
plt.title('Iris データセット')
# 表示
plt.show()
```



分布から読み取れること



縦軸は**花びらの幅**

横軸は**花びらの長さ**

散布図

	A	B	C
1	年次	出生数	死亡数
2	1985	1432	752
3	1990	1222	820
4	1995	1187	922
5	2000	1191	962
6	2005	1063	1084
7	2010	1071	1197
8			

元データ

Python による散布図の描画（データが複数列あるとき）

① ライブラリのインポート

```
import matplotlib.pyplot as plt
import japanize_matplotlib
```

japanize_matplotlib は，凡例やラベルで日本語を使いたいとき

② データの準備

```
x = [1985, 1990, 1995, 2000, 2005, 2010]
y1 = [1432, 1222, 1187, 1191, 1063, 1071]
y2 = [752, 820, 922, 962, 1084, 1197]
```

③ 散布図の描画

```
plt.plot(x, y1, 'o', label='出生数')
plt.plot(x, y2, 'o', label='死亡数')
```

グラフ化したいデータが複数列あるときは plt.plot
散布図は o，折れ線は o-

④ ラベルと凡例の設定

```
plt.xlabel('年次')
plt.ylabel('数（単位：１０００人）')
plt.title('出生数と死亡数')
plt.legend()
```

⑤ 描画結果の表示

```
plt.show()
```

散布図（Google Colaboratory）

```
!pip install japanize-matplotlib
```

機能のインストール

```
import matplotlib.pyplot as plt
import japanize_matplotlib

x = [1985, 1990, 1995, 2000, 2005, 2010]
y1 = [1432, 1222, 1187, 1191, 1063, 1071]
y2 = [752, 820, 922, 962, 1084, 1197]
# グラフ
plt.plot(x, y1, 'o', label='出生数')
plt.plot(x, y2, 'o', label='死亡数')
# ラベル
plt.xlabel('年次')
plt.ylabel('数（単位：１０００人）')
plt.title('出生数と死亡数')
# 凡例
plt.legend()
# 表示
plt.show()
```

ライブラリのインポート

データの準備
（Python のリスト）

散布図の描画
（**o** は**散布図**）

ラベルと凡例の設定

描画結果の表示

コード

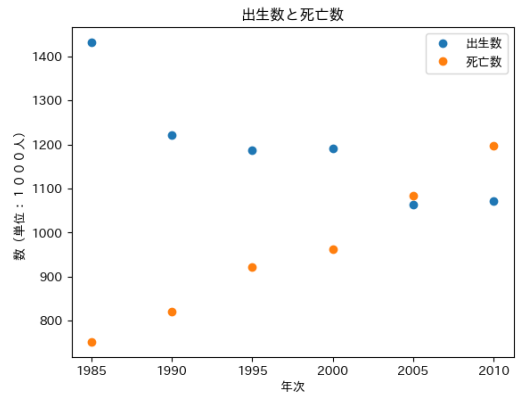
散布図（Google Colaboratory）

＋コード　＋テキスト　ドライブにコピー

```
!pip install japanize-matplotlib
```

```
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple/
Collecting japanize-matplotlib
  Downloading japanize-matplotlib-1.1.3.tar.gz (4.1 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 4.1/4.1 MB 37.1 MB/s eta 0:00:00
  Preparing metadata (setup.py) ... done
Requirement already satisfied: matplotlib in /usr/local/lib/python3.9/dist-packages (from japanize-matplotlib) (3.7.1)
Requirement already satisfied: fonttools>=4.22.0 in /usr/local/lib/python3.9/dist-packages (from matplotlib->japanize-matplotlib) (4.39.3)
Requirement already satisfied: contourpy>=1.0.1 in /usr/local/lib/python3.9/dist-packages (from matplotlib->japanize-matplotlib) (1.0.7)
Requirement already satisfied: kiwisolver>=1.0.1 in /usr/local/lib/python3.9/dist-packages (from matplotlib->japanize-matplotlib) (1.4.4)
Requirement already satisfied: numpy>=1.20 in /usr/local/lib/python3.9/dist-packages (from matplotlib->japanize-matplotlib) (1.22.4)
Requirement already satisfied: pillow>=6.2.0 in /usr/local/lib/python3.9/dist-packages (from matplotlib->japanize-matplotlib) (8.4.0)
Requirement already satisfied: importlib-resources>=3.2.0 in /usr/local/lib/python3.9/dist-packages (from matplotlib->japanize-matplotlib) (5.12.0)
Requirement already satisfied: pyparsing>=2.3.1 in /usr/local/lib/python3.9/dist-packages (from matplotlib->japanize-matplotlib) (3.0.9)
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.9/dist-packages (from matplotlib->japanize-matplotlib) (0.11.0)
Requirement already satisfied: python-dateutil>=2.7 in /usr/local/lib/python3.9/dist-packages (from matplotlib->japanize-matplotlib) (2.8.2)
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.9/dist-packages (from matplotlib->japanize-matplotlib) (23.1)
Requirement already satisfied: zipp>=3.1.0 in /usr/local/lib/python3.9/dist-packages (from importlib-resources>=3.2.0->matplotlib->japanize-matplotlib) (3.15.0)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.9/dist-packages (from python-dateutil>=2.7->matplotlib->japanize-matplotlib) (1.16.0)
Building wheels for collected packages: japanize-matplotlib
  Building wheel for japanize-matplotlib (setup.py) ... done
  Created wheel for japanize-matplotlib: filename=japanize_matplotlib-1.1.3-py3-none-any.whl size=4120274 sha256=d1ea6d730a960057315dc4a18150eb57eb8d0fcaa3779eb99b81d80918b68095
  Stored in directory: /root/.cache/pip/wheels/91/8f/c2/83055ad0c9591b0a094730aa7cb2cc12fedacbcd2241baf534
Successfully built japanize-matplotlib
Installing collected packages: japanize-matplotlib
Successfully installed japanize-matplotlib-1.1.3
```

```
import matplotlib.pyplot as plt
import japanize_matplotlib

x = [1985, 1990, 1995, 2000, 2005, 2010]
y1 = [1432, 1222, 1187, 1191, 1063, 1071]
y2 = [752, 820, 922, 962, 1084, 1197]
# グラフ
plt.plot(x, y1, 'o', label='出生数')
plt.plot(x, y2, 'o', label='死亡数')
# ラベル
plt.xlabel('年次')
plt.ylabel('数（単位：１０００人）')
plt.title('出生数と死亡数')
# 凡例
plt.legend()
# 表示
plt.show()
```



※ファイルへの保存も可能

折れ線（Google Colaboratory）

```
!pip install japanize-matplotlib
```

機能のインストール

```
import matplotlib.pyplot as plt
import japanize_matplotlib

x = [1985, 1990, 1995, 2000, 2005, 2010]
y1 = [1432, 1222, 1187, 1191, 1063, 1071]
y2 = [752, 820, 922, 962, 1084, 1197]
# グラフ
plt.plot(x, y1, 'o-', label='出生数')
plt.plot(x, y2, 'o-', label='死亡数')
# ラベル
plt.xlabel('年次')
plt.ylabel('数（単位：１０００人）')
plt.title('出生数と死亡数')
# 凡例
plt.legend()
# 表示
plt.show()
```

ライブラリのインポート

データの準備
（Python のリスト）

散布図の描画
（**o-** は**折れ線**）

ラベルと凡例の設定

描画結果の表示

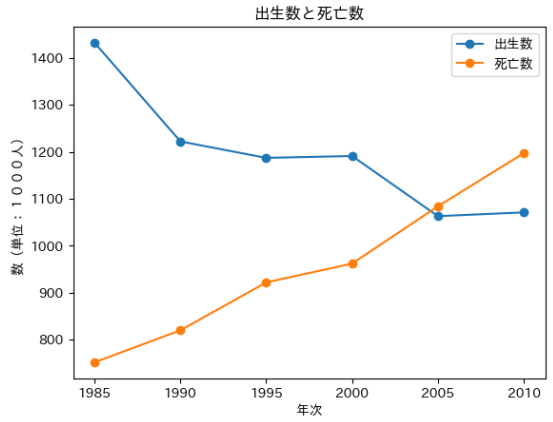
コード

折れ線（Google Colaboratory）

```
!pip install japanize-matplotlib
```

```
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple/
Collecting japanize-matplotlib
  Downloading japanize-matplotlib-1.1.3.tar.gz (4.1 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 4.1/4.1 MB 37.1 MB/s eta 0:00:00
  Preparing metadata (setup.py) ... done
Requirement already satisfied: matplotlib in /usr/local/lib/python3.9/dist-packages (from japanize-matplotlib) (3.7.1)
Requirement already satisfied: fonttools>=4.22.0 in /usr/local/lib/python3.9/dist-packages (from matplotlib->japanize-matplotlib) (4.39.3)
Requirement already satisfied: contourpy>=1.0.1 in /usr/local/lib/python3.9/dist-packages (from matplotlib->japanize-matplotlib) (1.0.7)
Requirement already satisfied: kiwisolver>=1.0.1 in /usr/local/lib/python3.9/dist-packages (from matplotlib->japanize-matplotlib) (1.4.4)
Requirement already satisfied: numpy>=1.20 in /usr/local/lib/python3.9/dist-packages (from matplotlib->japanize-matplotlib) (1.22.4)
Requirement already satisfied: pillow>=6.2.0 in /usr/local/lib/python3.9/dist-packages (from matplotlib->japanize-matplotlib) (8.4.0)
Requirement already satisfied: importlib-resources>=3.2.0 in /usr/local/lib/python3.9/dist-packages (from matplotlib->japanize-matplotlib) (5.12.0)
Requirement already satisfied: pyparsing>=2.3.1 in /usr/local/lib/python3.9/dist-packages (from matplotlib->japanize-matplotlib) (3.0.9)
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.9/dist-packages (from matplotlib->japanize-matplotlib) (0.11.0)
Requirement already satisfied: python-dateutil>=2.7 in /usr/local/lib/python3.9/dist-packages (from matplotlib->japanize-matplotlib) (2.8.2)
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.9/dist-packages (from matplotlib->japanize-matplotlib) (23.1)
Requirement already satisfied: zipp>=3.1.0 in /usr/local/lib/python3.9/dist-packages (from importlib-resources>=3.2.0->matplotlib->japanize-matplotlib) (3.15.0)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.9/dist-packages (from python-dateutil>=2.7->matplotlib->japanize-matplotlib) (1.16.0)
Building wheels for collected packages: japanize-matplotlib
  Building wheel for japanize-matplotlib (setup.py) ... done
  Created wheel for japanize-matplotlib: filename=japanize_matplotlib-1.1.3-py3-none-any.whl size=4120274 sha256=d1ea6d730a9600573l5dc4a18150eb57eb9d0fcaa3779eb99b81d80918b68095
  Stored in directory: /root/.cache/pip/wheels/91/8f/c2/83055ad0c9591b0a094730aa7cb2cc12fedacbcd2241baf534
Successfully built japanize-matplotlib
Installing collected packages: japanize-matplotlib
Successfully installed japanize-matplotlib-1.1.3
```

```
[6] import matplotlib.pyplot as plt
    import japanize_matplotlib

    x = [1985, 1990, 1995, 2000, 2005, 2010]
    y1 = [1432, 1222, 1187, 1191, 1063, 1071]
    y2 = [752, 820, 922, 962, 1084, 1197]
    # グラフ
    plt.plot(x, y1, 'o-', label='出生数')
    plt.plot(x, y2, 'o-', label='死亡数')
    # ラベル
    plt.xlabel('年次')
    plt.ylabel('数（単位：１０００人）')
    plt.title('出生数と死亡数')
    # 凡例
    plt.legend()
    # 表示
    plt.show()
```



※ファイルへの保存も可能

3. 평균

平均

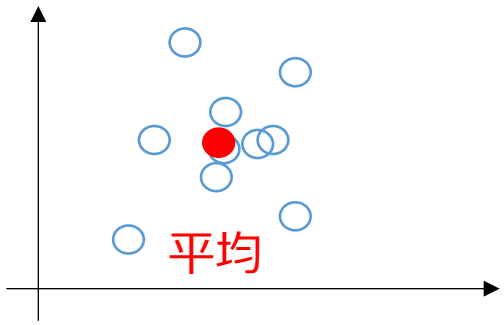
- **平均**は，データの**合計**を，**データの個数**で**割ったもの**

  10, 40, 30, 40 の**平均**: **120** ÷ **4** で **30**

- **複数の値の組**の**平均**を考えることもある

  (10, 5), (40, 10), (30, 5), (40, 20) の平均:

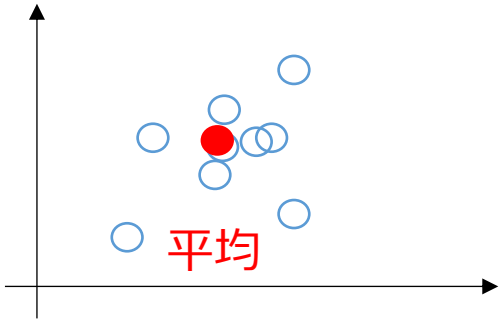
  合計は 120 と 40．4で割って (30, 10)

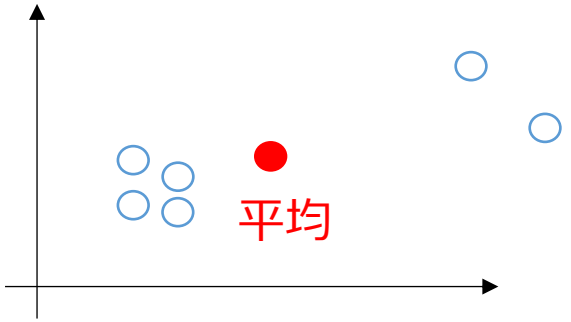


平均は，**データ集合**の**代表**とみることができる場合がある

計測に**誤差**があるとき，
複数の計測を繰り返し，**平均**をとることで，**誤差を軽減**できることも

平均を使うときの注意点





このような平均に，
意味があるでしょうか？

データの分布によっては，**平均**では**役に立たない**こともある．
（平均は万能ではない）

Python による平均の算出

① ライブラリのインポート

import numpy as np

numpy という数値計算ライブラリをインポート．Np という名前で使用できるようにしている

② データの準備

y1 = [10, 40, 30, 40]

リストy1を作成、その中に4つの数値 10, 40, 30, 40 を設定

③ 平均の算出と表示

print(np.mean(y1))

numpyのmeanを使ってリストy1の平均を算出

平均（Google Colaboratory）

```
import numpy as np

y1 = [10, 40, 30, 40]
y2 = [5, 10, 5, 20]
print(np.mean(y1))
print(np.mean(y2))
```

ライブラリのインポート

データ

平均の算出と表示

コード

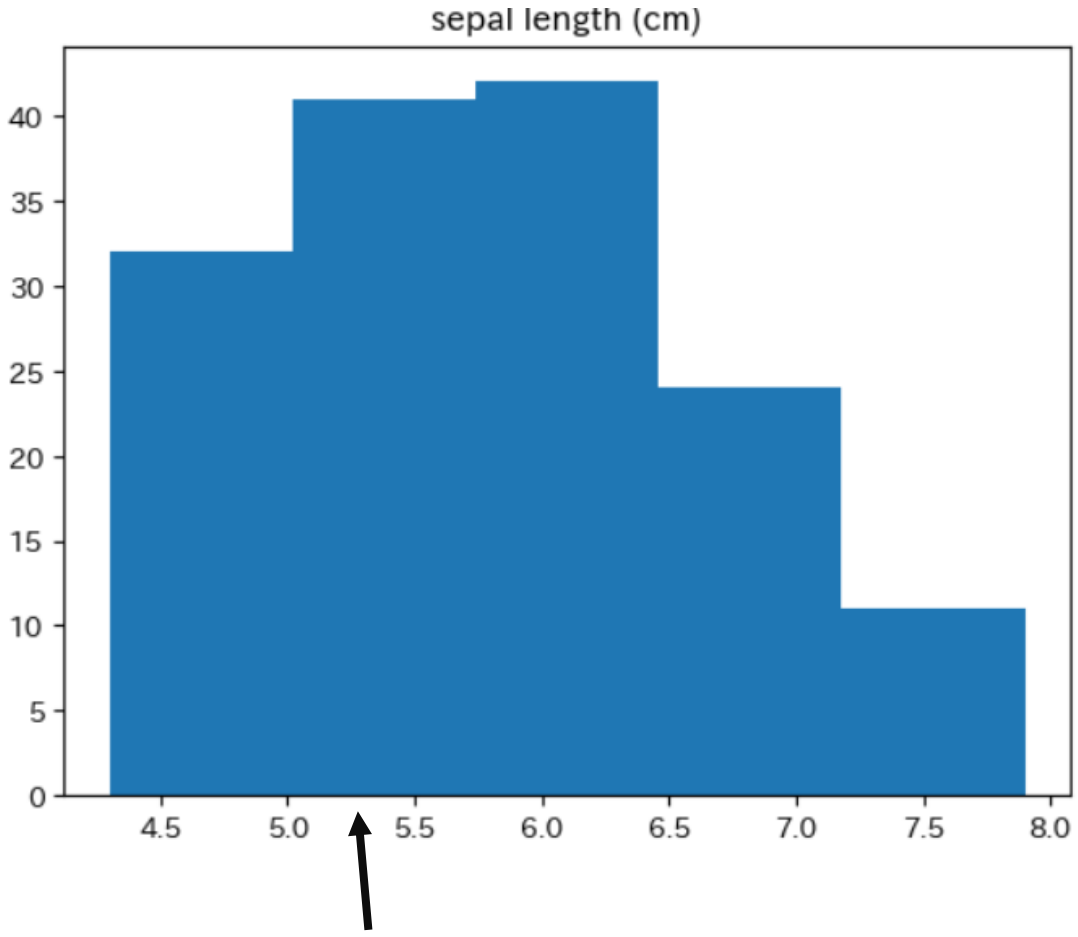
```
import numpy as np

y1 = [10, 40, 30, 40]
y2 = [5, 10, 5, 20]
print(np.mean(y1))
print(np.mean(y2))
```

```
30.0
10.0
```

4. 分布、密度

ヒストグラム

ヒストグラムは，区間ごとに，データを数え上げたもの

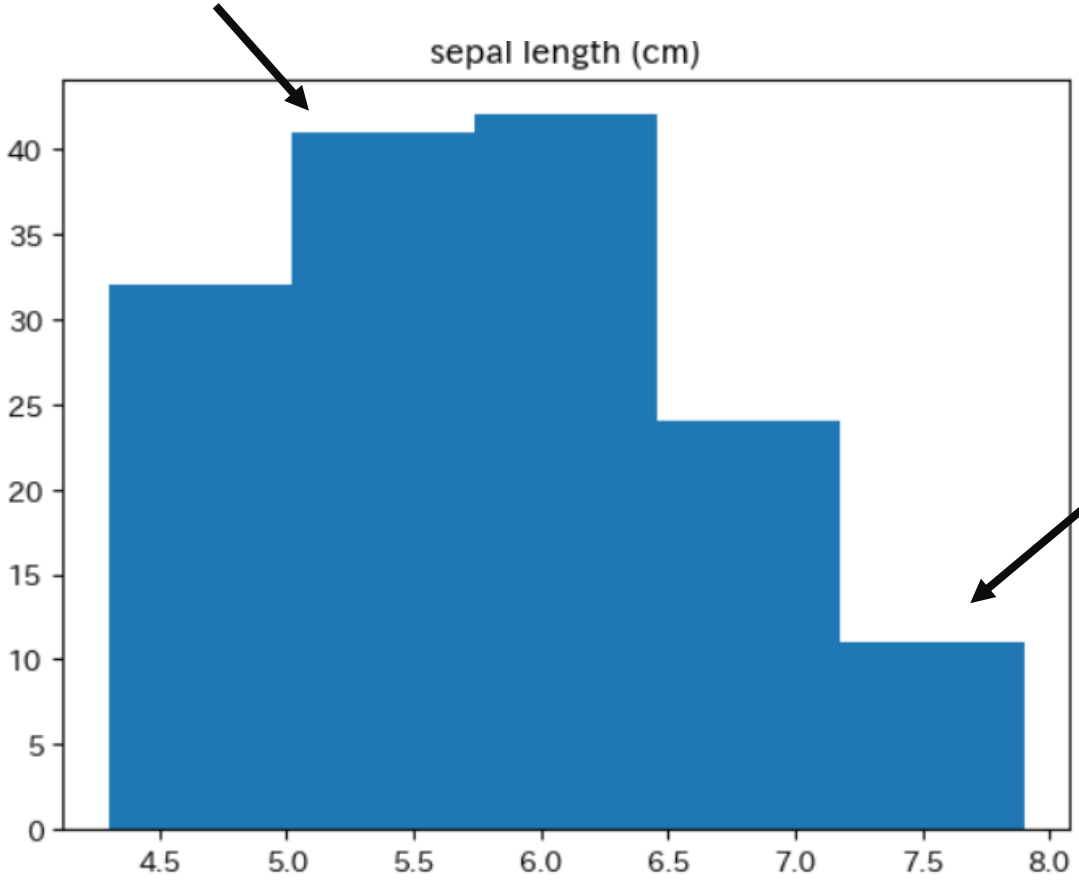


データが何個あるのか

区間

ヒストグラムから読み取れること

密度が高い



密度が低い

【全体の傾向】山が１つある

Python によるヒストグラムの描画

① ライブラリのインポート

```
import matplotlib.pyplot as plt
import japanize_matplotlib
```

japanize_matplotlib は，
凡例やラベルで日本語を
使いたいとき

② データの準備
ここでは，Iris データセットを使用

```
from sklearn.datasets import load_iris
iris = load_iris()
x = iris.data
y = iris.target
```

③ ヒストグラムの描画．「bins=**5**」は帯数を 5に設定．

```
plt.hist(x[:, 0], bins=5)
```

④ ラベルの設定

```
plt.title(iris.feature_names[0])
```

⑤ 描画結果の表示

```
plt.show()
```

Iris データセットのヒストグラム（Google Colaboratory）

```
import matplotlib.pyplot as plt
import japanize_matplotlib
from sklearn.datasets import load_iris
iris = load_iris()
x = iris.data
y = iris.target
# グラフ
plt.hist(x[:, 0], bins=5)
# ラベル
plt.title(iris.feature_names[0])
# 表示
plt.show()
```

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）

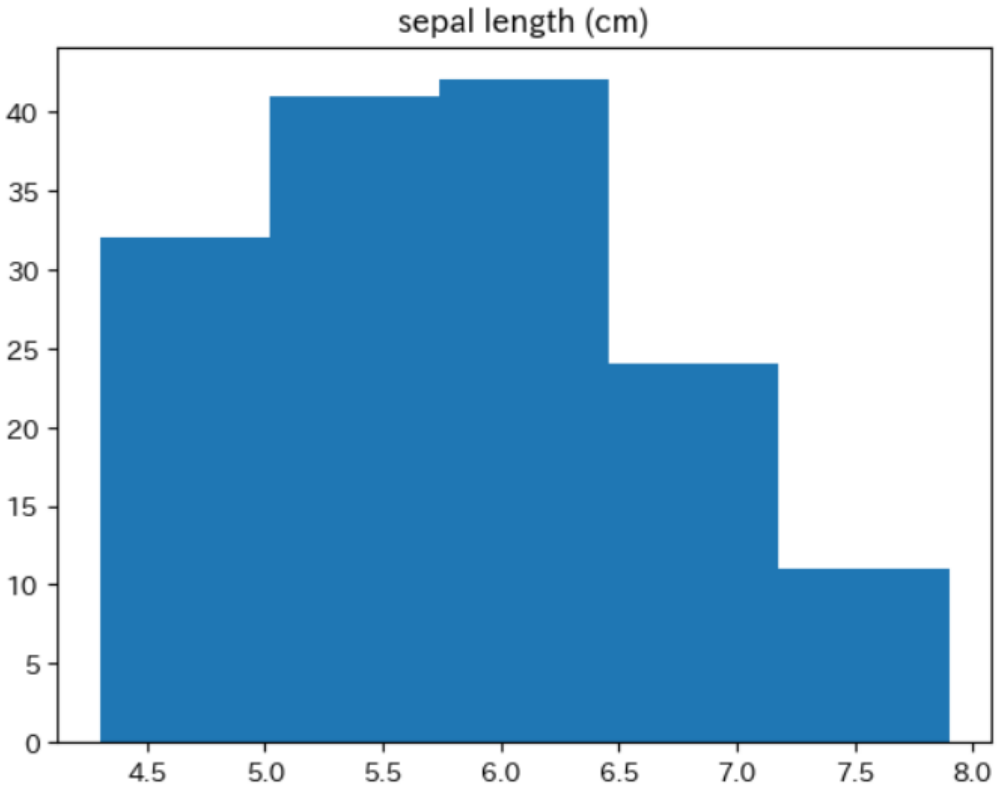
グラフの描画
（0列のヒストグラム）

ラベルの設定

表示

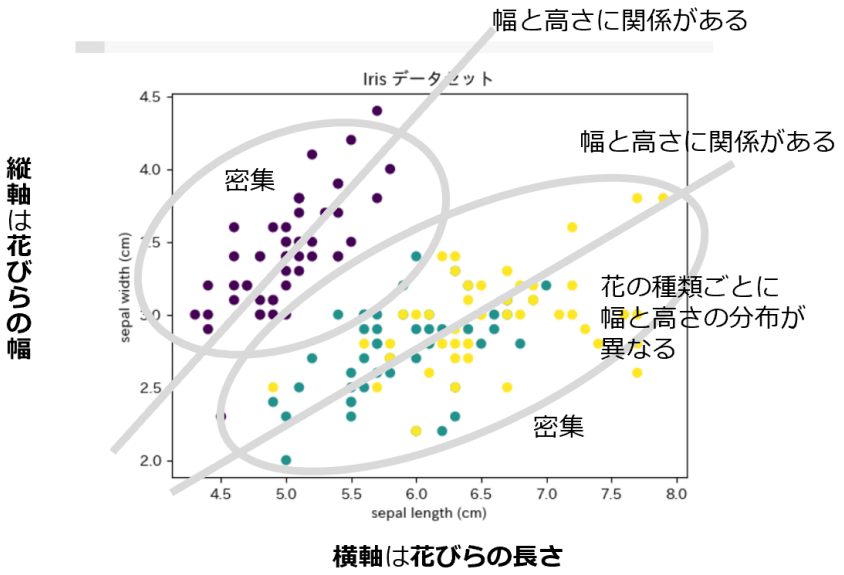
Iris データセットのヒストグラム（Google Colaboratory）

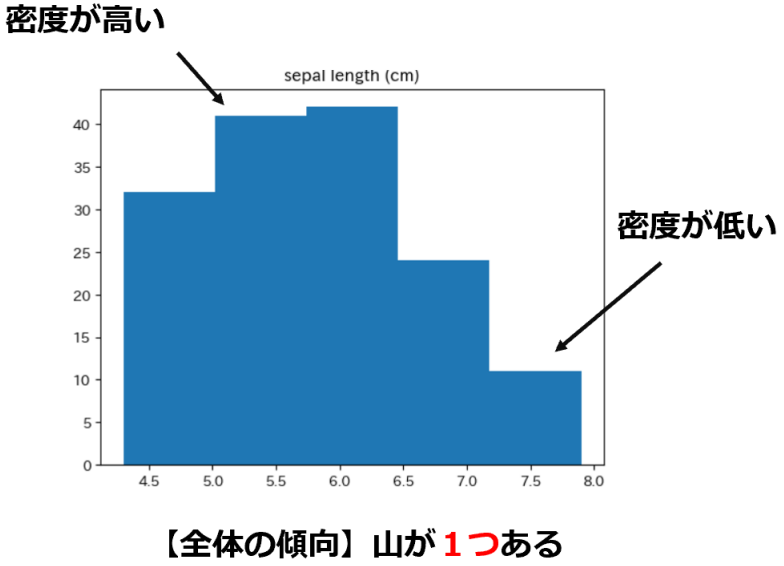
```
import matplotlib.pyplot as plt
from sklearn.datasets import load_iris
iris = load_iris()
x = iris.data
y = iris.target
# グラフ
plt.hist(x[:, 0], bins=5)
# ラベル
plt.title(iris.feature_names[0])
# 表示
plt.show()
```



データサイエンス

- **データサイエンス**は，**データから正しい知見や結論を導く**ための学問





- 数式を使うこともあるが、それがデータサイエンスの全てではない
- **データから有益な情報を引き出す**ことが可能に。
- **大学生にとって、大切なスキル**

まとめ

- **データサイエンス**は、**データから有益な情報を導き出す**ための学問
- Python プログラミング言語は、豊富な機能があり、簡単な文法で扱えるなどの特徴がある。データサイエンス分野でも広く利用されている。

データサイエンスを**理解**し、**実践的な経験を積む**ことは、**将来の活躍に大いに役立つ**。