農学生命情報科学特論 I

ICT や IoT 等の先端技術を活用し、効率よく高品質生産を可能にするスマート農業への取り組みは世界的に進められています。その基礎を支えている技術の一つがプログラミング言語。なかでも、習得しやすくかつ応用範囲の広い Python がとくに注目されています。本科目では、農学や分子生物学などの分野で利用される Python の最新事例を紹介しながら、Python の基礎文法の講義を行います。

孫　建強　https://aabbdd.jp/
農研機構・農業情報研究センター

June
08 17:15–20:30

基本構文

第 1 回目の授業では、プログラミング言語の基本であるデータ構造とアルゴリズムを簡単に紹介してから、Python の基本構文を紹介する。Python のスカラー、リスト、ディクショナリ、条件構文と繰り返し構文を取り上げる。

June
15 17:15–20:30

文字列処理

バイオインフォマティックスの分野において、塩基配列やアミノ酸配列などの文字列からなるデータを扱うことが多い。第 2 回目の授業では、Python を利用した文字列処理を紹介し、FASTA や GFF などのファイルから情報を抽出する方法を取り上げる。

June
22 17:15–20:30

データ解析

Python で CSV や TSV ファイルに保存された数値データを扱うときに、NumPy や Pandas ライブラリーを使用する。第 3 回目の授業では、NumPy や Pandas の機能を紹介し、ベクトルや行列のデータ処理を中心に取り上げる。

June
29 17:15–20:30

データ可視化

Python で数値データを可視化するときに matplotlib ライブラリーを使用する。第 4 回目の授業では、matplotlib の基本的な使い方を紹介し、全回の授業を復習しながらデータの可視化を行う。

成績評価

- 授業の始めに出席確認を 1 回だけを行う。出席 1 回につき 1 点を与える。
- 出席確認後、レポート課題を示す。毎回 3 問出題し、授業全体を通して合計 12 問出題する。1 問 8 点とする。
- 授業を聞かなくても、レポート課題を解けそうであれば、遠慮なく退室していただいて構いません。

レポート課題の提出方法

- レポート課題を Jupyter Notebook / Jupyter Lab 上で解き、そのファイル（.ipynb）を メールで提出する。

  ✉ report＠aabbdd.jp

- 注意事項
  - メールの件名を「特論I課題2」としてください。
  - メールの本文に何も書かないでください。
  - ファイル名を "英語氏名-2.ipynb"（例： ShinosukeNohara-2.ipynb）としてください。
  - ファイル中に、名前・所属・研究内容などの個人情報・機密情報を書かないでください。個人情報・機密情報を含むレポートを零点とする。
  - 締め切りは 6 月 30 日。

レポート課題（第 2 回）

問題 1

FASTA 形式のテキストファイル ft.fa に FT 遺伝子の塩基配列が記載されている。ft.fa を読み込み、FT 遺伝子の塩基配列のうち G の割合と C の割合を計算するプログラムを作成せよ。

問題 2

GenBank 形式のテキストファイル AF152096.gb の中から塩基配列を抜き出し、FASTA 形式のファイル（ AF152096.fa）に変換せよ。

問題 3

PDB/mmCIF 形式のテキストファイル 1alk.cif に 1ALK タンパク質の立体情報が記載されている。1alk.cif を読み込み、A 鎖（全アミノ酸の C_α 原子）の重心を求めるプログラムを作成せよ。

レポート課題（第 2 回）解説

FASTA 形式

FASTA 形式は、遺伝子の配列情報を記載するために定義されたテキストファイルの形式（フォーマット）である。「>」から始まる行は、ヘッダー行と呼ばれ、遺伝子名や遺伝子 ID が記載されている。ヘッダー行以降には、その遺伝子の塩基配列が記載されている。

```
>FT|AF152096.1
ACAAAAACAAGTAAAACAGAAACAATCAACACAGAGAAACCACCTGTTTGTTCAAGATCAAAGATGTCTA
TAAATATAAGAGACCCTCTTATAGTAAGCAGAGTTGTTGGAGACGTTCTTGATCCGTTTAATAGATCAAT
CACTCTAAAGGTTACTTATGGCCAAAGAGAGGTGACTAATGGCTTGGATCTAAGGCCTTCTCAGGTTCAA
AACAAGCCAAGAGTTGAGATTGGTGGAGAAGACCTCAGGAACTTCTATACTTTGGTTCTTTCACTTGAAC
TCCCTTTTGTCTCTTTTCTTCTTTAGTTTCTTCAGTGCTTCTTACACCTTCTTTTTTAAAATAGAAATTA
TTTTCCTTTTTTGGGGTATACTGAAAATATTTCTTGGGCATGCAGAGACCTTGGTTAAAAATGCCCCACG
```

https://aabbdd.jp/notes/data/ft.fa

レポート課題（第 2 回）解説

GenBank 形式

GenBank 形式のファイルはアミノ酸配列あるいは核酸配列およびそのアノテーションを記録するためのファイル形式（フォーマット）である。この形式において、配列の名前は ACCESSION レコードに記載され、配列そのものは ORIGIN レコードに記載されている。配列の最後の文字の次の行に「//」で終わる。

```
LOCUS       AF152096                 2483 bp    DNA     linear   PLN 22-DEC-1999
DEFINITION  Arabidopsis thaliana flowering locus T (FT) gene, complete cds.
ACCESSION   AF152096
VERSION     AF152096.1
...
ORIGIN
        1 acaaaaacaa gtaaaacaga aacaatcaac acagagaaac cacctgtttg ttcaagatca
       61 aagatgtcta taaatataag agaccctctt atagtaagca gagttgttgg ag
//
```

https://aabbdd.jp/notes/data/AF152096.gb

レポート課題（第 2 回）解説

PDB / mmCIF 形式

PDB / mmCIF 形式は、タンパク質の立体情報を記載するために定義されたテキストファイルの形式（フォーマット）である。各原子の立体座標は ATOM から始まる行に記載されている。例えば、A 鎖の C_α 原子の立体座標は (66.663, 52.938, 2.184) と記載されいてる。

```
ATOM   1    N  N    . THR A 1 1   ? 67.510  52.432 1.100  1.00 53.47 ? 1   THR A N    1
ATOM   2    C  CA   . THR A 1 1   ? 66.663  52.938 2.184  1.00 53.80 ? 1   THR A CA   1
ATOM   3    C  C    . THR A 1 1   ? 66.660  51.955 3.361  1.00 53.28 ? 1   THR A C    1
ATOM   4    O  O    . THR A 1 1   ? 65.713  51.931 4.167  1.00 53.34 ? 1   THR A O    1
ATOM   5    C  CB   . THR A 1 1   ? 65.183  53.192 1.684  1.00 54.32 ? 1   THR A CB   1
ATOM   6    O  OG1  . THR A 1 1   ? 64.737  51.920 1.096  1.00 54.92 ? 1   THR A OG1  1
ATOM   7    C  CG2  . THR A 1 1   ? 65.042  54.354 0.698  1.00 54.82 ? 1   THR A CG2  1
ATOM   8    N  N    . PRO A 1 2   ? 67.726  51.174 3.426  1.00 52.46 ? 2   PRO A N    1
ATOM   9    C  CA   . PRO A 1 2   ? 67.891  50.164 4.478  1.00 51.28 ? 2   PRO A CA   1
```

https://aabbdd.jp/notes/data/1alk.cif.txt

農学生命情報科学特論Ⅰ

②

◯ テキスト処理

◯ ファイル処理

農学生命情報科学特論Ⅰ

②

- テキスト処理
- ファイル処理

テキストデータ

テキストデータはアルファベットや仮名などの文字からなるデータを指す。Python のオブジェクトには、数値の他にアルファベット・漢字・仮名などの文字を代入することもできる。文字をオブジェクトに代入するとき、その文字が、オブジェクトの名前ではなく、文字のデータであることを明示するために、文字データの両側を引用符で囲む。

```
s = 'a'


t = 'B'


u = 't'
u
# 't'


v = t
v
# 'B'
```

t が引用符で囲まれているため、文字データとしての t がオブジェクト u に代入される。

t が引用符で囲まれていないため、t はオブジェクトである。オブジェクト t の内容がオブジェクト v に代入される。

テキストデータ

単語や文章などのような文字列も、1 つのオブジェクトに代入できる。この場合、1 つのオブジェクトに、複数の文字データが保存されている、と捉えることができるため、このオブジェクトをリストのように扱うことができる。添字を指定して、特定の位置の文字を取り出したり、スライスして部分文字列を取り出したりすることができる。また、for 構文を使用して、文字列中の文字を 1 つずつ順に取り出すこともできる。

```
s = 'Smile. Tomorrow will be worse.'

s[0]
# 'S'



s[7:15]
# 'Tommorow'



for t in s:
    print(t)
# 'S'
# 'm'
# 'i'
# 'l'
# 'e'
# ......
```

テキストデータ

文字列を保持しているオブジェクトに対して、足算が定義されている。そのため、+ 演算子を使用することで、複数の文字列を連結することができる。

```
s1 = 'Smile.'
s2 = 'Tomorrow will be worse.'
s3 = ' '

s = s1 + s2
s
# 'Smile.Tomorrow will be worse.'


s = s1 + s3 + s2
s
# 'Smile. Tomorrow will be worse.'


s = s1 + ' ' + s2
s
# 'Smile. Tomorrow will be worse.'
```

問題 T1-1

赤色で書かれたオブジェクトが保持している値を答えよ。

```
s1 = 'anything that can go wrong'
s2 = 'it will go wrong'



s1[:8]



s2[3:]



s3 = s1 + ',' + s2[3:]
s3
```

```
s1 = 'left to themselves'
s2 = 'things tend to go'
s3 = 'from bad to better'



s = s1 + ', ' + s2
s = s + ' ' + s3[:12] + 'worse.'
s
```

問題 T1-2

与えられた塩基配列の中から ATG を検索し、ATG が見つかればその位置を、そうでなければ -1 を出力するプログラムを for/while 構文や if 構文で作成せよ。

```
s = 'CCACAGTCATGTGTCAGTCGTAAGT'



# 8
```

```
s = 'CATTGTGTCACAGCCAGTCGTAAGT'



# -1
```

問題 T1-3

与えられた塩基配列中の A、C、G、T の出現回数および出現確率を求めよ。

```
s = 'AAAGGTCTTT'



# A: 3, G: 2, C: 1, T: 4
```

与えられた塩基配列に含まれる AA、AC、・・・、TT のような 2 文字パターンの出現回数を求めよ。

```
s = 'AAAGGTCTTT'
     ==
      ==
       ==


# AA: 2, AG: 1, GG: 1, GT: 1,
# TC: 1, CT: 1, TT: 2
```

問題 T1-4

for/while 構文や if 構文を使用して、与えられた塩基配列の相補鎖を求めよ。

```
s = 'AAAGGTC'



c
# 'TTTCCAG'
```

for/while 構文や if 構文を使用して、与えられた塩基配列の逆相補鎖を求めよ。

```
s = 'AAAGGTC'



r
# 'GACCTTT'
```

文字列操作関数

文字列の操作には、連結、分割、検索、置換などがある。これらの操作は、次の関数（文字列メソッド）で行える。

`a + b`	文字列 a の後ろに文字列 b を連結する。
`a.split(',')`	コンマ , を区切り文字として、文字列を分割して、部分文字列のリストに変換する。
`a.find('ATG')`	文字列 a の先頭から ATG を探す。見つかればその位置の添字を返し、そうでなければ -1 を返す。
`a.replace('TAG', '*')`	文字列 a 中ににある部分文字列 TAG を * に置き換える。

```
s = '21,31,41,51,61'


s.split(',')
# ['21', '31', '41', '51', '61']



s.find('41')
# 6


s.find('71')
# -1


s.replace(',', ';')
# '21;31;41;51;61'
```

文字と数値の交互変換

Python の世界で、文字と数値の扱い方が異なる。文字と文字の足し算では文字同士が連結され、数値と数値の足し算では数学的に和が計算される。文字と数値の足し算は定義されておらず、エラーが出る。

```
a = '12'
b = 21


a + b
# TypeError: can only concatenate str
(not "int") to str
```

文字列 a に、整数 b を連結できないことを示すエラー。

```
b + a
# TypeError: unsupported operand type(s)
for +: 'int' and 'str'
```

整数 b に文字列 a を足せないことを示すエラー。

文字から数値への変換

文字列を整数または小数に変換することができる。整数への変換は int 関数を利用し、小数への変換は float 関数を利用する。これらの関数は、すべての文字列を整数・小数に変換できるわけではない。変換できない場合は、エラーが起こる。

```
a1 = '12'
a2 = '12.345'
a3 = '1.23e6'

int(a1)
# 12

int(a2)
# ValueError: invalid literal for int()
with base 10: '12.345'

float(a1)
# 12.0

float(a2)
# 12.345

float(a3)
# 1230000.0
```

数値から文字への変換

数値から文字への変換は str 関数を使用する。基本的に、すべての数値を文字に変換できる。桁数の多い小数を文字に変換するとき、桁落ちが発生する場合がある。

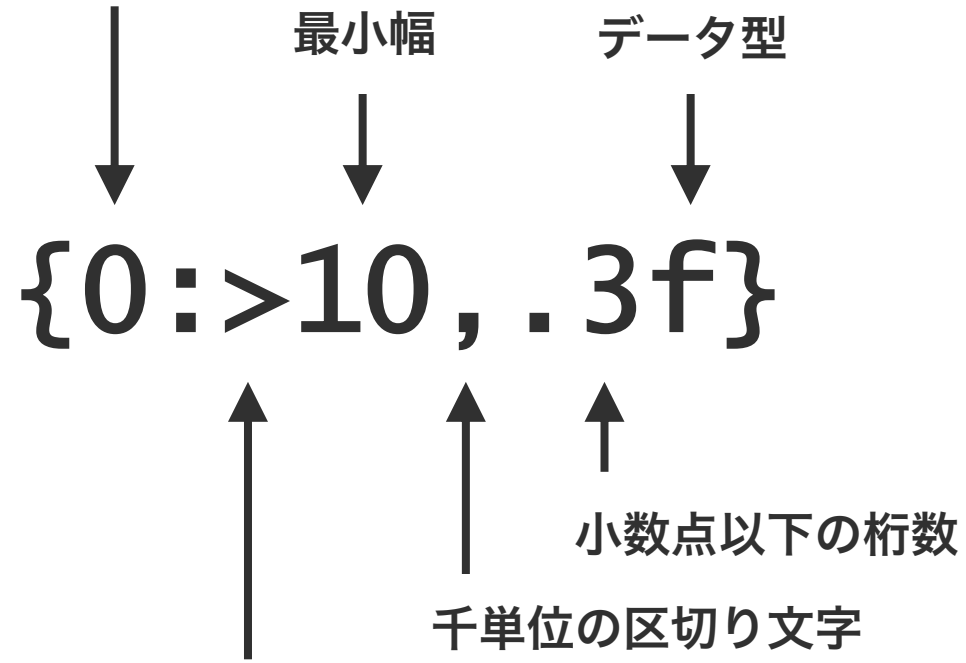
```
b1 = 12
b2 = 12.345
b3 = 12.34567890123456789 01
b4 = 1.23e6


str(b1)
# '12'


str(b2)
# '12.345'


str(b3)
# '12.345678901234567'


str(b4)
# '1230000.0'
```

数値から文字への変換

参考

数値から文字への変換は、str 関数のほかに format 関数も用意されている。format 関数を使うことで、有効桁数を指定したり、ゼロ埋めしたりすることができる。

埋め込みインデックス



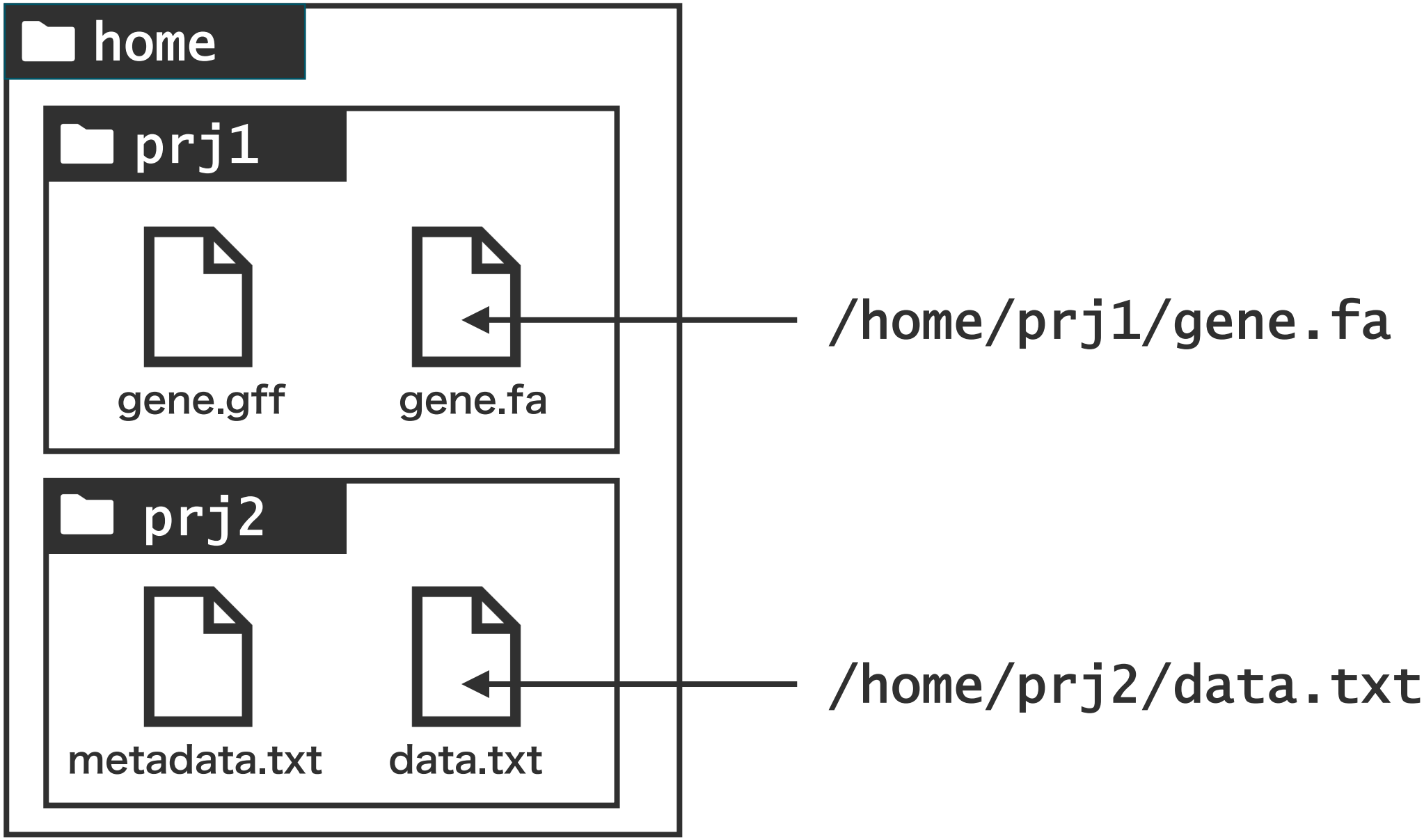

```
b1 = 12
b2 = 12.3456789


'{:4d}'.format(b1)
# '  12'

'{:04d}'.format(b1)
# '0012'

'{:.3f}'.format(b1)
# '12.000'

'{:.3f}'.format(b2)
# '12.345'

'{:.2e}'.format(b2)
# '1.23e+01'
```

農学生命情報科学特論Ⅰ

②

- テキスト処理
- ファイル処理

ファイルパス

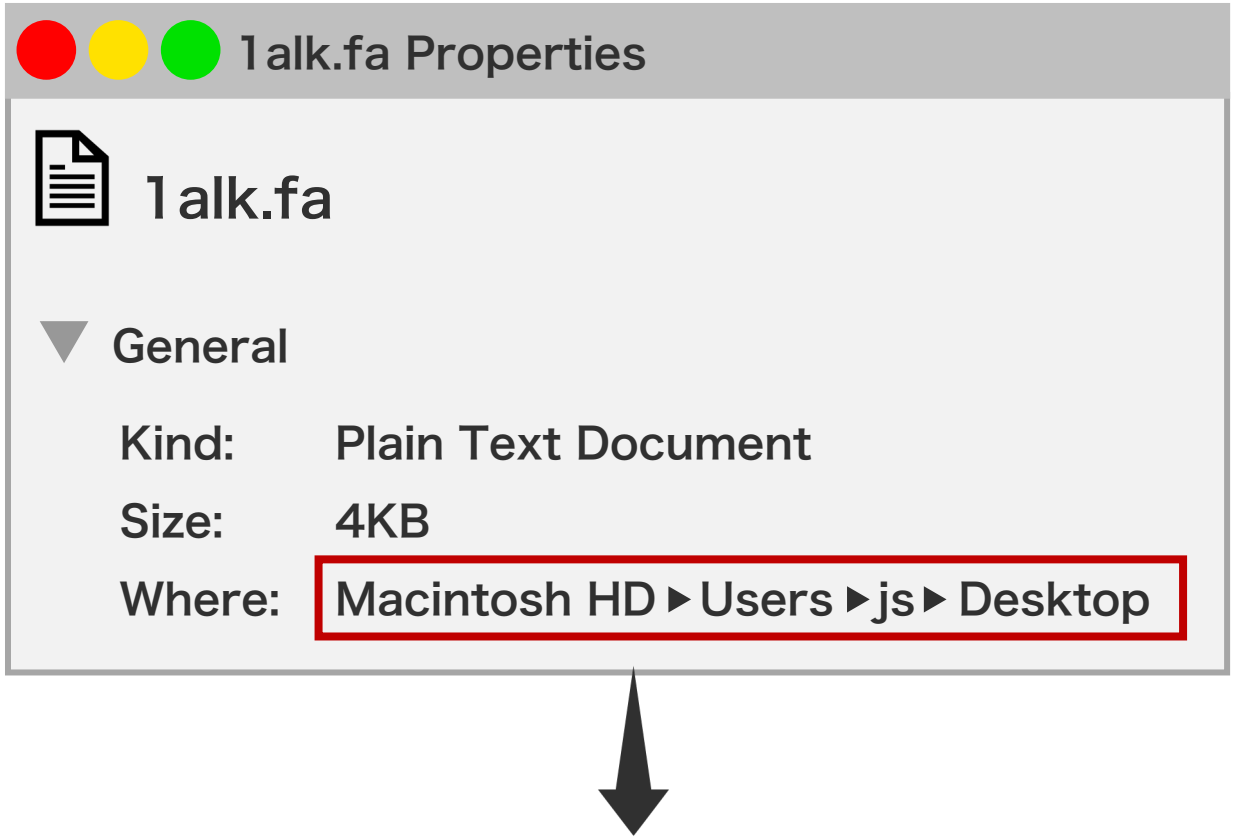

home
prj1
gene.gff
gene.fa
prj2
metadata.txt
data.txt
/home/prj1/gene.fa
/home/prj2/data.txt

ファイルパス（Macintosh）

1. パスを調べたいファイルを右クリックし、「Get Info」を選ぶ。

2. 「General」タブの「Where」項目にファイルへのパスが記載されている。

    - ファイルパスの「Macintosh HD」は最上層を表し、パスを書くとき「/」と書く。
    - 小さい三角形はフォルダの包含関係を表すので、パスを書くときは「/」と書く。



/Users/js/Desktop/1alk.fa

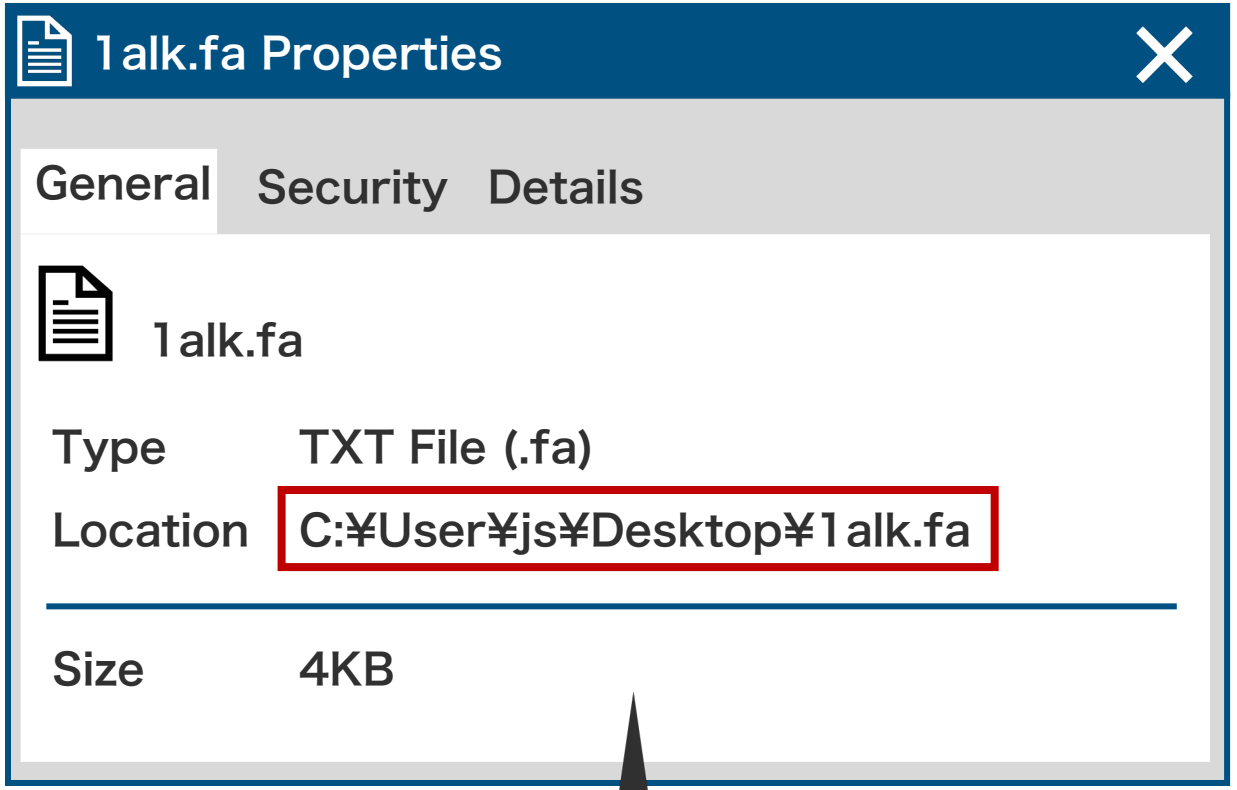
ファイルパス（Windows）

1. パスを調べたいファイルを右クリックし、「Properties」を選ぶ。

2. 「General」タブの「Location」項目にファイルへのパスが記載されている。

   - **日本語環境の場合、「＼」が「¥」として表示される。**どちらも Windows コンピューター内部では 0x5C として認識されている。

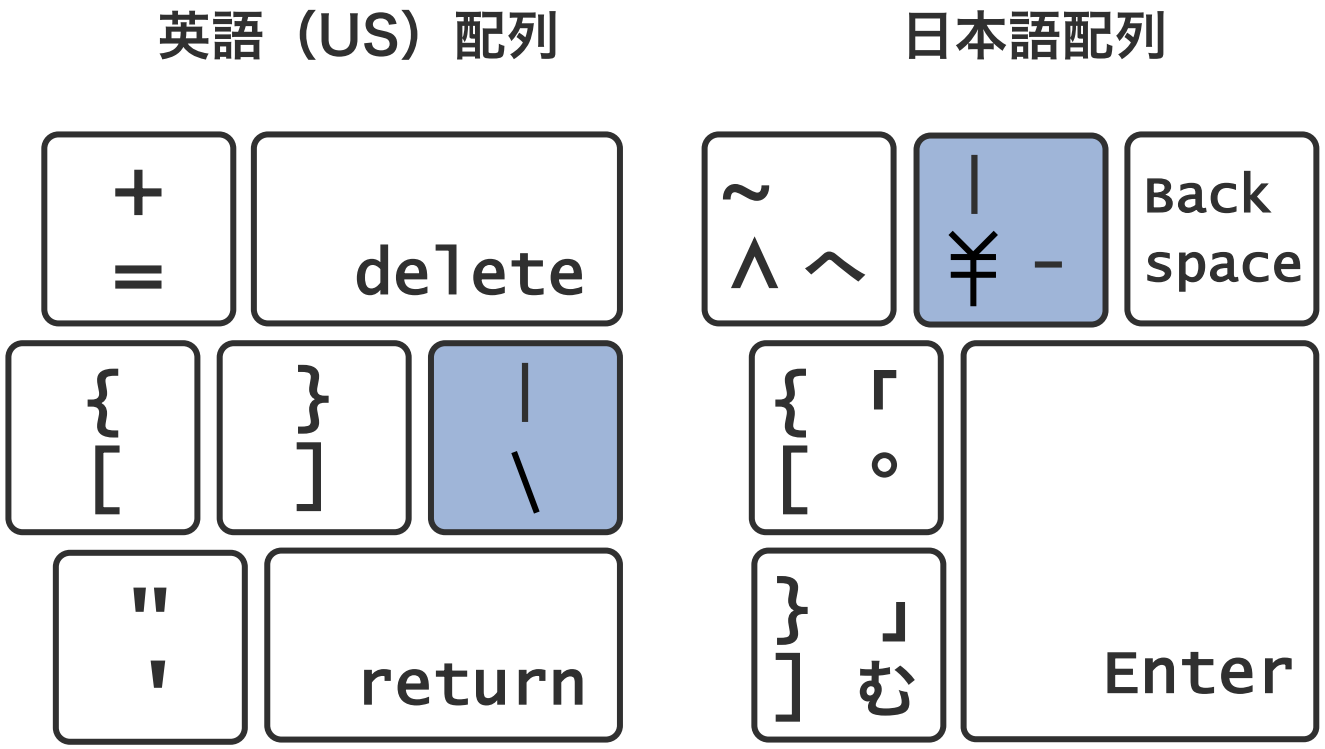
   - パスとして使用するとき、Location 欄に書かれているパス中の**「＼」を「/」に置き換える。**



C:/User/js/Desktop/1alk.fa

バックスラッシュ（＼）・円マーク（¥）

バックスラッシュは、OS の種類、言語環境や文字コードによって、ディスプレイでの表示が異なる。日本語環境のシステムであれば円マーク「¥」として表示され、それ以外の言語環境では「＼」として表示される。バックスラッシュと円マークは、表示が異なるものの、コンピューター上では同一のコード 0x5C として扱われる。



macOS 日本語環境を使用している場合は、＼ と ¥ の両方を入力できる。Option を押しながら＼または¥キーを押すことで、＼と¥の入力の切り替えができる。

ファイル

Python でファイルを読み込むには、open 関数を使用する。open 関数に、ファイルへのパスとともにオープン・モードを与えて使う。

モード	意味
r	読み込みモード。ファイルが存在しない場合はエラーになる。
w	書き込みモード。ファイルが存在しない場合は新規作成される。ファイルが存在する場合は、既存のファイルを削除したうえで新規作成する。
a	追記モード。既存のファイルの最後に追記する。ファイルが存在しない場合は新規作成される。

https://aabbdd.jp/notes/data/1alk.fa

```
f = '1alk.fa'

with open(f, 'r') as fh:
    for line in fh:
        print(line)
```

```
# >1ALK:A|PDBID|CHAI...
# TPEMPVLENRAAQGNITA...
# TGQYTHYALNKKTGKPDY...
# AHVTSRKCYGPSATSQKC...
```

"IOError: [Errno 2] No such file or directory '1alk.fa'" のようなエラーが起きた場合、ファイルのパスを確認してください。ファイルをダウンロードする際に、"1alk.fa" が勝手に "1alk.fa.txt" に名前変更されることがある。

ファイル読み込み

パス f に保存されているファイルを、r モードで開き、ファイルハンドルにセットアップする。

```
f = '1alk.fa'


with open(f, 'r') as fh:

    for line in fh:

        print(line)


# >1ALK:A|PDBID|CHAI...
# TPEMPVLENRAAQGNITA...
# TGQYTHYALNKKTGKPDY...
# AHVTSRKCYGPSATSQKC...
```

https://aabbdd.jp/notes/data/1alk.fa

ファイル読み込み

ファイルの内容がリストに変換されてファイルハンドルに代入される※。ファイルの i 行目の文字列情報が、リストの i 番目の要素に代入されている。そのため、for 構文を利用して、リストの先頭から要素を順に取り出せば、ファイルの内容を 1 行目から順に取り出せるようになる。

```
>1ALK
TPEMPV
TGQYTH
AHVTSR
```

```
fh = [
  '>1ALK\n',
  'TPEMPV\n',
  'TGQYTH\n',
  'AHVTSR\n'
]
```

```
f = '1alk.fa'

with open(f, 'r') as fh:
    for line in fh:
        print(line)

# >1ALK:A|PDBID|CHAI...
# TPEMPVLENRAAQGNITA...
# TGQYTHYALNKKTGKPDY...
# AHVTSRKCYGPSATSQKC...
```

※ 正確には、ファイルの何行の何文字目まで読み込んだかの位置情報（ファイルポインター）がファイルハンドルに保存される。ファイルハンドルに対して for 構文を適用すると、ポインターを自動的に 1 行ずつ進めさせることができる。また、read メソッドと for 構文を組み合わせることで、ポインターを 1 文字ずつ進めさせることもできる。

ファイル読み込み

```
f = '1alk.fa'


with open(f, 'r') as fh:

    for line in fh:

        print(line)


# >1ALK:A|PDBID|CHAI...
# TPEMPVLENRAAQGNITA...
# TGQYTHYALNKKTGKPDY...
# AHVTSRKCYGPSATSQKC...
```

すべての行の読み込みが終わり、ファイルが閉じら ▶
れる。

エスケープシーケンス

Python では、特殊な用途向けに、いくつかの特殊文字が定義されている。例えば、改行とタブがその特殊文字にあたる。改行文字は、人には見えないが、コンピュータが改行として認識するために必要な特殊文字である。限られたアルファベットと数字の中で、このような特殊文字を表すためには、既存文字を組み合わせて表現する。一般に、バックスラッシュに 1 文字を足して、特殊文字を表現していることが多い。例えば、改行ならば \n、タブならば \t のように表現する。

```
s = 'Smile. nTomorrow will be worse.'
s
# 'Smile. nTomorrow will be worse.'



s = 'Smile. \nTomorrow will be worse.'
s
# 'Smile.'
# 'Tomorrow will be worse.'



s = 'Smile. \\nTomorrow will be worse.'
s
# 'Smile. \nTomorrow will be worse.'
```

問題 F1-1

リスト fh の要素のうち、「>」から始まる要素の個数を求めよ。

```
fh = ['>1ALK:A',
      'TPEMPVL',
      'TGQYTHA',
      '>1ALK:B',
      'TPEMPVL',
      'TGQYTHA']
```

問題 F1-2

1alk.fa ファイルを読み込んで、「>」から始まる行の行数を求めよ。

```
f = '1alk.fa'
n = 0
```

https://aabbdd.jp/notes/data/1alk.fa

問題 F1-3

ft.fa ファイルは FASTA 形式のテキストファイルである。このファイルは「>」から始まる行に遺伝子名が記載され、それ以降の行に遺伝子の塩基配列が記載されている。ft.fa に記載されいてる遺伝子の塩基配列の長さを求めよ。

```
f = 'ft.fa'
l = 0
```

小文字エル l、大文字アイ I、数字 1 はフォントによって区別しにくい場合がある。

https://aabbdd.jp/notes/data/ft.fa

問題 F1-4

ft.fa に記載された塩基配列について、A の出現確率を求めよ。

```
f = 'ft.fa'
pA = 0
```

https://aabbdd.jp/notes/data/ft.fa

問題 F1-5

diversity_galapagos.txt は、タブ区切りのテキストファイルであり、ガラパゴス島における種の多様性データが記載されている。このファイルを読み込み、面積（Area 列）の最も大きい島の名前（Island 列）を答えよ。ただし、このテキストファイルには、「#」から始まるコメント行とデータの属性を示すヘッダー行が含まれていることに注意。

```
f = 'diversity_galapagos.txt'
island_name = ''
```

https://aabbdd.jp/notes/data/diversity_galapagos.txt

問題 F1-6

diversity_galapagos.txt は、タブ区切りのテキストファイルであり、ガラパゴス島における種の多様性データが記載されている。面積（Area）あたりの種数（Species）が最も大きい大きい島の名前とそのときの面積当たりの種数を求めよ。

```
f = 'diversity_galapagos.txt'
island_name = ''
island_dens = 0
```

https://aabbdd.jp/notes/data/diversity_galapagos.txt

ファイル書き込み

ファイルの書き込みには w および a モードが定義されている。このほかにも w+ や a+ などのモードも定義されているが、初めは w モードのみ使えればよい。

モード	意味
r	読み込みモード。ファイルが存在しない場合はエラーになる。
w	書き込みモード。ファイルが存在しない場合は新規作成される。ファイルが存在する場合は、既存のファイルを削除したうえで新規作成する。
a	追記モード。既存のファイルの最後に追記する。ファイルが存在しない場合は新規作成される。

```
f = 'output.fa'


with open(f, 'w') as outfh:

    outfh.write('abcdefg')
    outfh.write('1234567')
```

ファイル書き込み

パス f を書き込みモードで開き、ファイルハンドル ▶ にセットアップする。

```
f = 'output.fa'


with open(f, 'w') as outfh:

    outfh.write('abcdefg')
    outfh.write('1234567')
```

ファイル書き込み

abcdefg をファイルに書き込む ▶

```
f = 'output.fa'


with open(f, 'w') as outfh:

    outfh.write('abcdefg')
    outfh.write('1234567')
```

ファイル書き込み

1234567 をファイルに書き込む ▶

```
f = 'output.fa'


with open(f, 'w') as outfh:

    outfh.write('abcdefg')
    outfh.write('1234567')
```

ファイル書き込み

書き込みが終了し、ファイルが閉じられる。 ▶

```
f = 'output.fa'


with open(f, 'w') as outfh:

    outfh.write('abcdefg')
    outfh.write('1234567')
```

問題 F1-7

ft.fa ファイルは FASTA 形式のテキストファイルである。このファイルは「>」から始まる行に、塩基配列の名前が記載され、それ以降の行は塩基配列が大文字で記載されている。配列の名前を変更しないで、塩基配列をすべて小文字に変換し、これらの情報を新しいファイル new_ft.fasta に保存するプログラムを作成せよ。

```
>ft
ACCGTGFA
ATTTGCTA
CACATTTA
AAAC
```

```
>ft
accgtgfa
atttgcta
cacattta
aaac
```

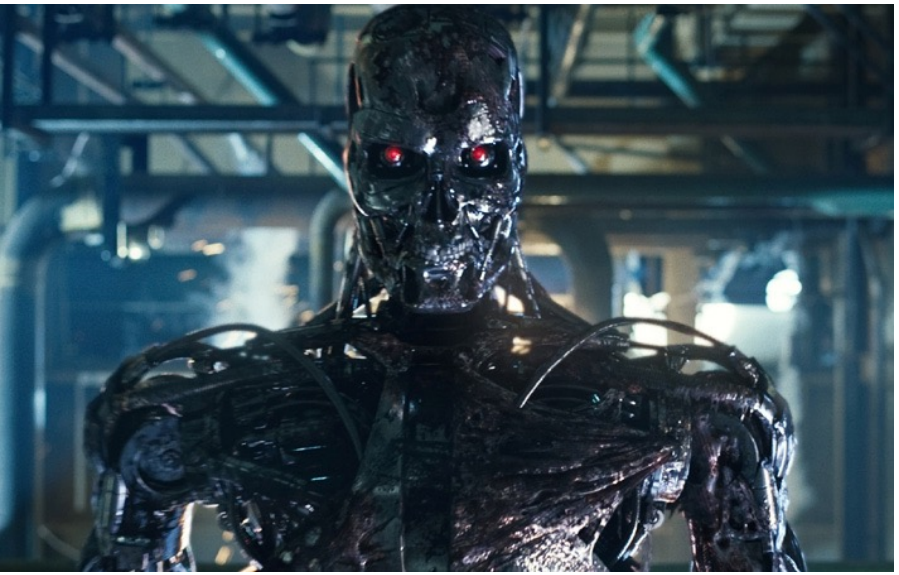
https://aabbdd.jp/notes/data/ft.fa

```
f = 'ft.fa'
l = 0
```

コラム

- 人工知能
- 機械学習
- モデル開発

What society thinks I do



What my friends think I do

What my mom think I do

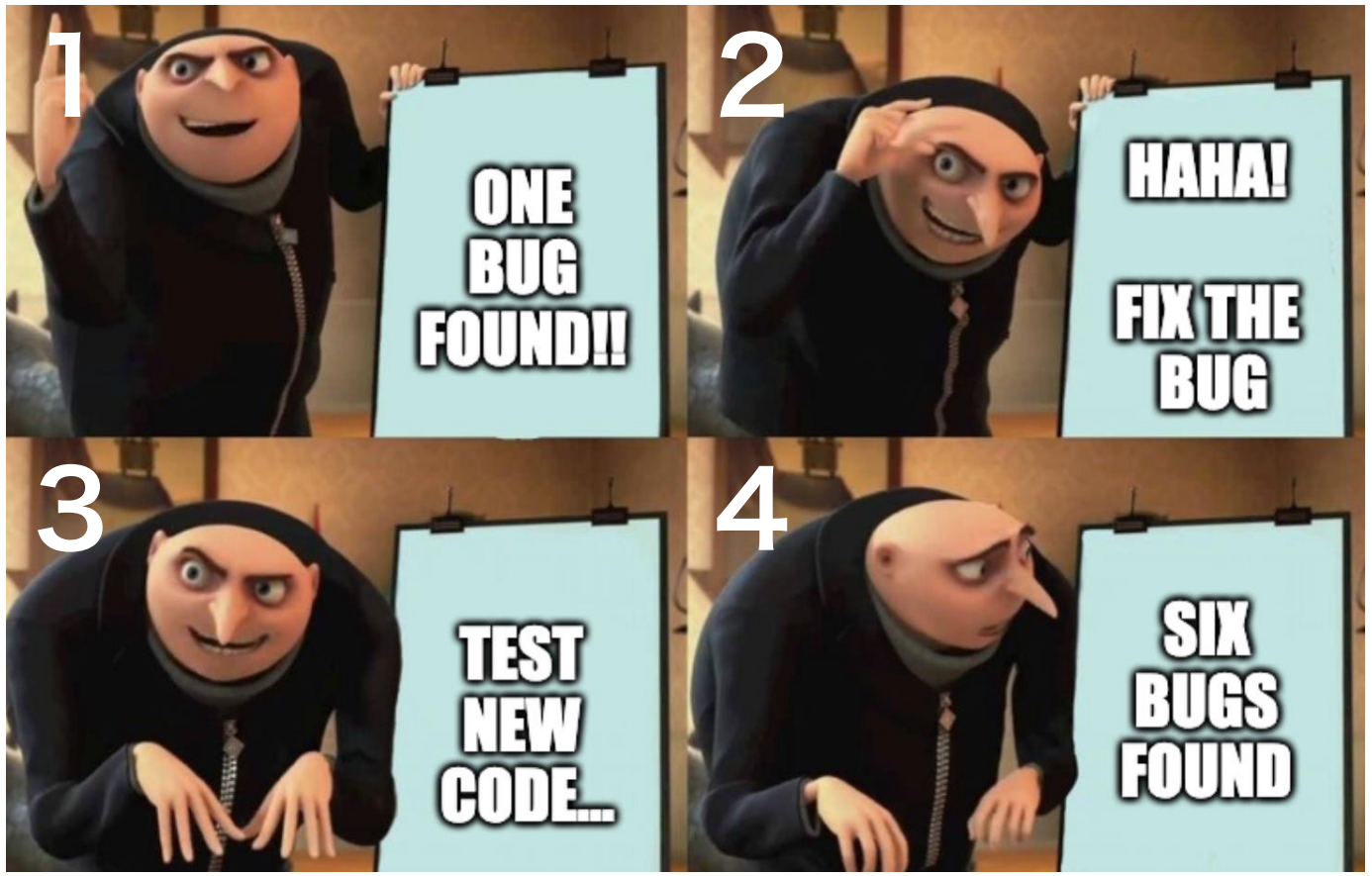
What mathematicians think I do

What I think I do

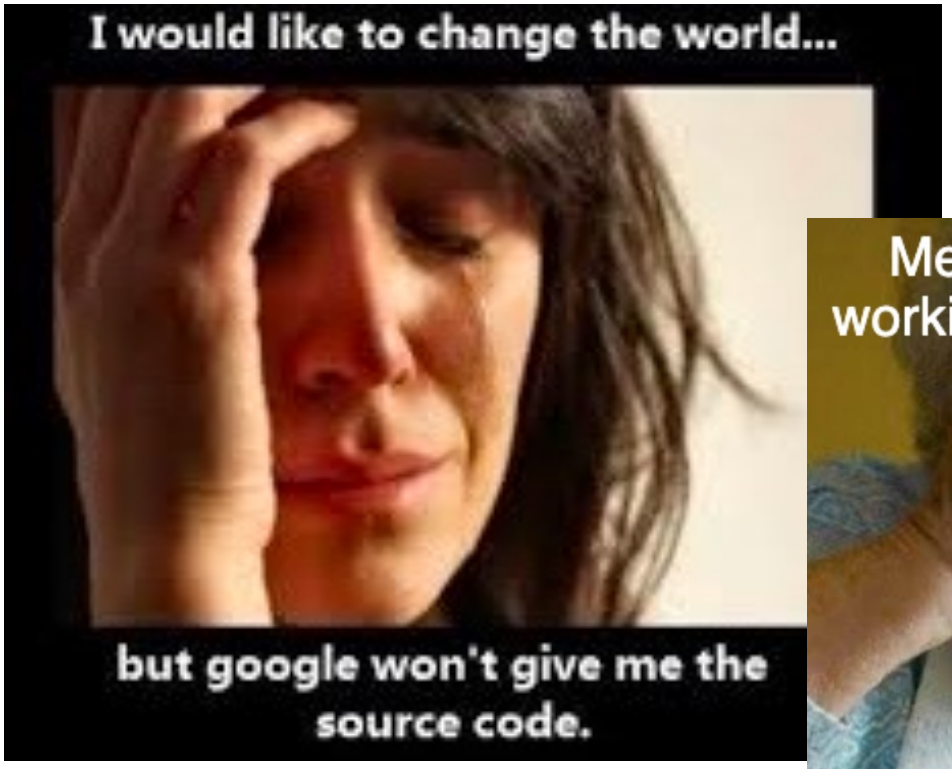
What I actually do

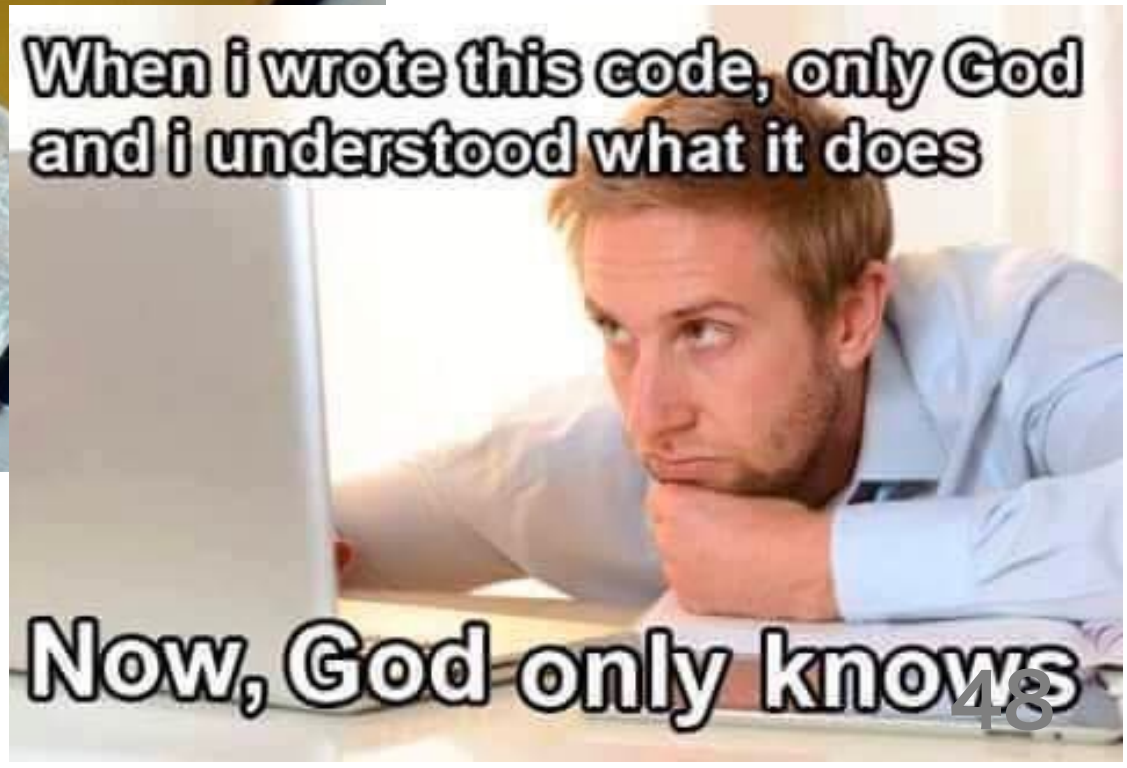
What I actually do ...

```
Python 3.8.2 (default, Apr  8 2021, 23:19:18)
[Clang 12.0.5 (clang-1205.0.22.9)] on darwin
Type "help", "copyright", "credits" or "licen
>>> import torch
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
ModuleNotFoundError: No module named 'torch'
>>>
```



What I actually do ...





What is AI?

What does AI do?

画像解析

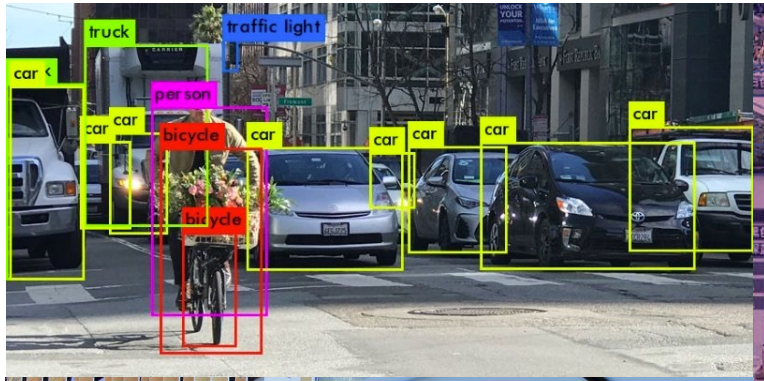
創造

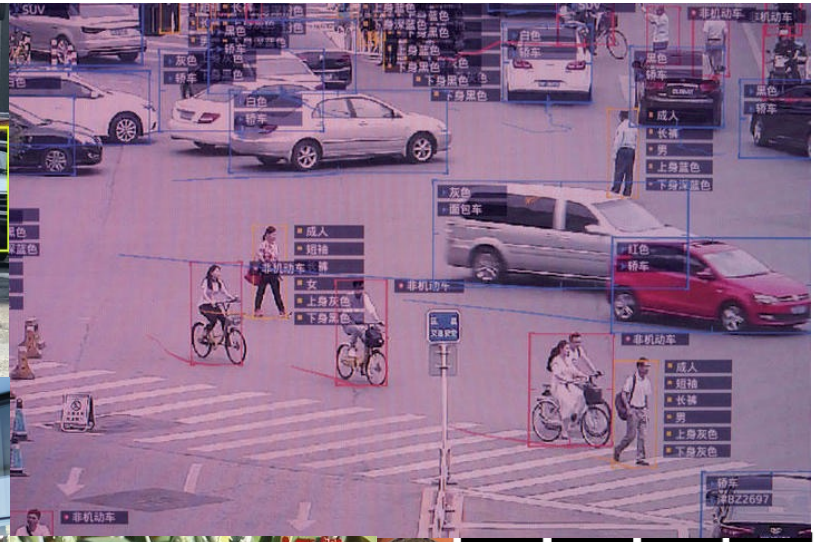
言語処理

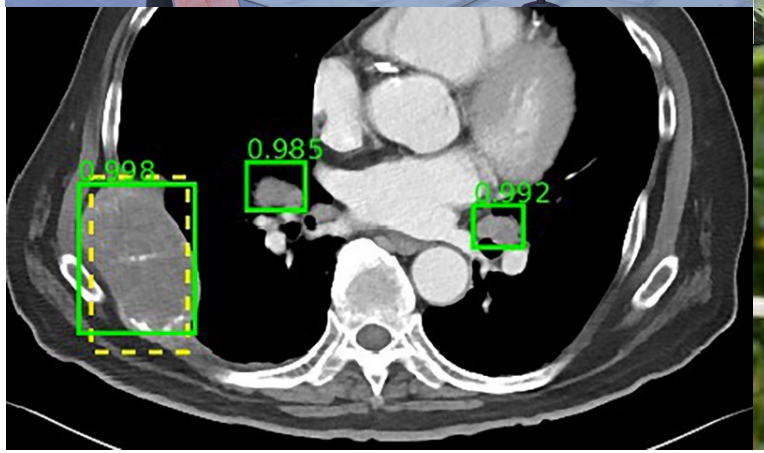
ゲーム

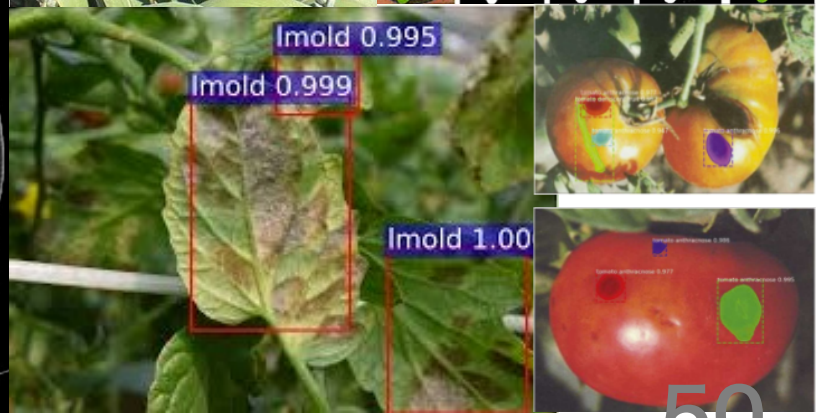
音声認識

制御









画像解析

創造

言語処理

ゲーム

音声認識

制御

DeepL 翻訳ツール Linguee

Macにダウンロード 無料！ ログイン

原文 英語 （自動検出）

訳文 日本語

Using next-generation sequencing technology alone, we have successfully generated and assembled a draft sequence of the giant panda genome. The assembled contigs (2.25 gigabases (Gb)) cover approximately 94% of the whole genome, and the remaining gaps (0.05 Gb) seem to contain carnivore-specific repeats and tandem repeats. Comparisons with the dog and human showed that the panda genome has a lower divergence rate. The assessment of panda genes potentially underlying some of its unique traits indicated that its bamboo diet might be more dependent on its gut microbiome than its own genetic composition. We also identified more than 2.7 million heterozygous single nucleotide polymorphisms in the diploid genome. Our data and analyses provide a foundation for promoting mammalian genetic research, and demonstrate the feasibility for using next-generation sequencing technologies for accurate, cost-effective and rapid de novo assembly of large eukaryotic genomes.

文書の翻訳

968 / 5000

次世代シークエンシング技術を用いて、ジャイアントパンダゲノムのドラフト配列の作成・構築に成功しました。組み立てたコンティグ（2.25GB）は全ゲノムの約94%をカバーしており、残りのギャップ（0.05GB）には肉食動物特有の繰り返しやタンデムリピートが含まれていると考えられます。また、イヌやヒトとの比較から、パンダのゲノムは発散率が低いことがわかりました。パンダのユニークな形質のいくつかの基礎となる可能性のある遺伝子の評価は、その竹の食事が、その遺伝子構成よりも腸内マイクロバイオームに依存している可能性があることを示した。また、二倍体ゲノムには270万以上のヘテロ接合一塩基多型が確認されました。我々のデータと分析は、哺乳類の遺伝研究を促進するための基盤を提供し、大型真核生物ゲノムの正確で費用対効果の高い迅速なde novoアセンブリのために次世代シーケンシング技術を使用することの実現可能性を実証している。

IBM Watson™

画像解析

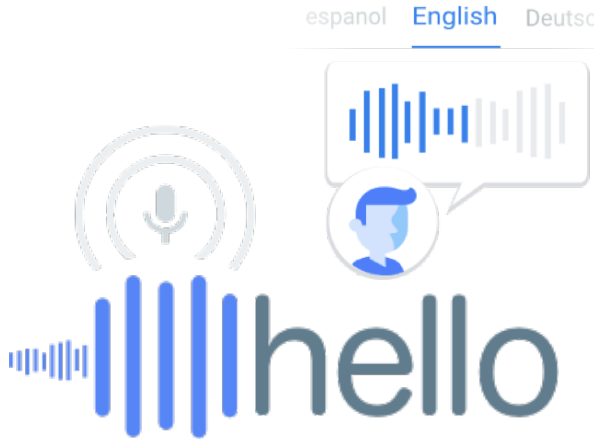
創造

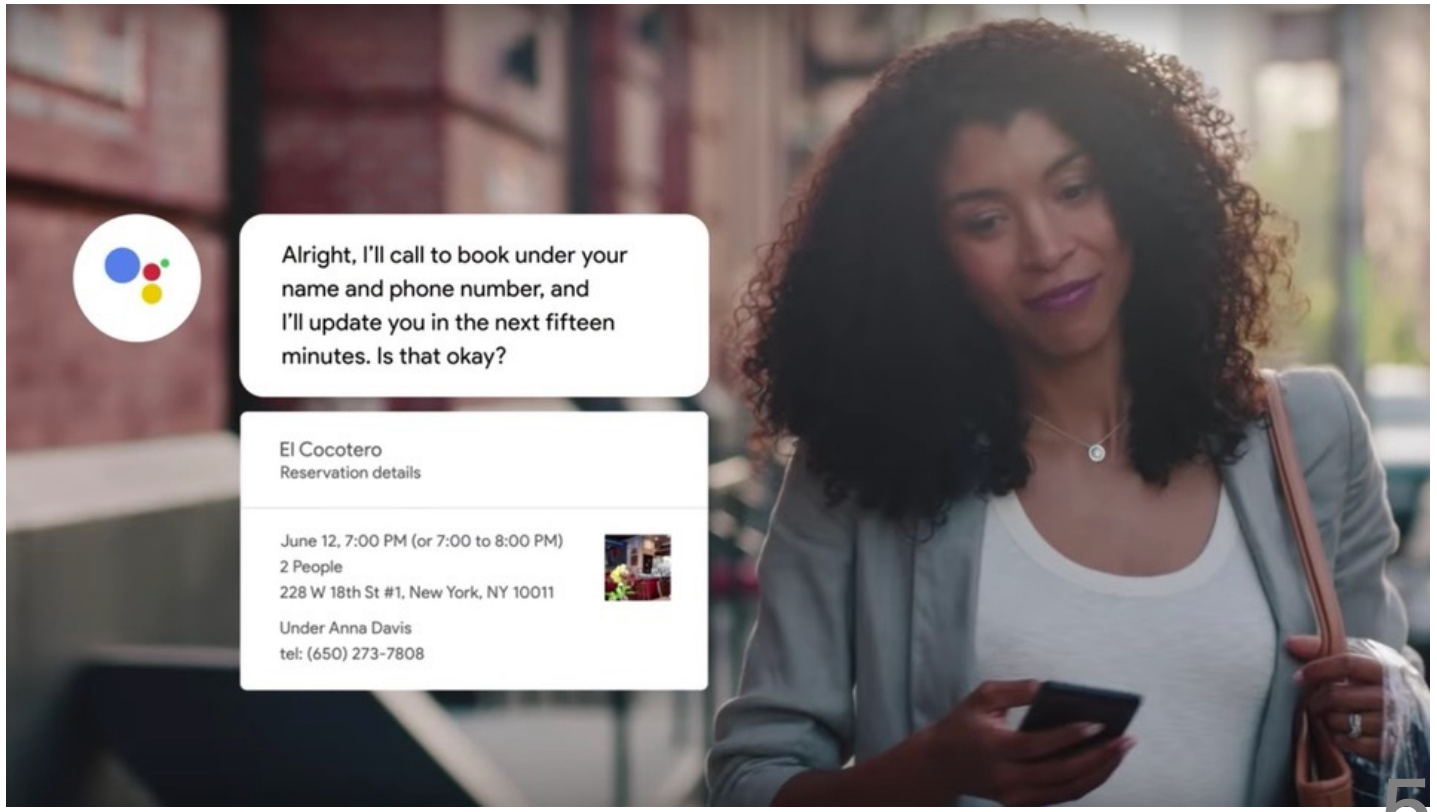
言語処理

ゲーム

音声認識

制御





画像解析

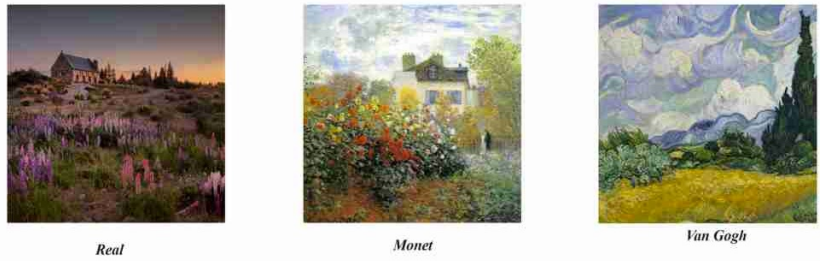
創造

言語処理

ゲーム

音声認識

制御



画像解析

創造

言語処理

ゲーム

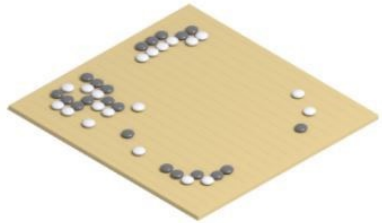
音声認識

制御

Chess

Shogi

Go



画像解析

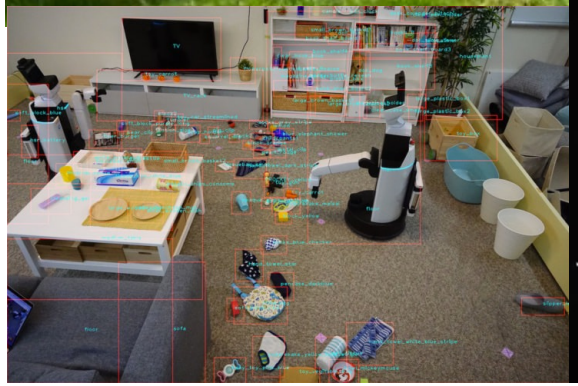
創造

言語処理

ゲーム

音声認識

制御



社会問題　人工知能　機械学習

社会問題に対し規律・秩序を制定し、問題解決の最善方策を探索する。

コラム

- 人工知能
- **機械学習**
- モデル開発

機械学習

教師あり学習

教師あり学習は、訓練データとして特徴量と正解ラベルがペアとして与えられるときに行う学習方法である。機械が特徴量と正解ラベルの関係性を探索し、最適な関数で両者を結びつける。教師あり学習は、分類問題および回帰問題などに応用される。

- ニューラルネットワーク
- ロジスティック回帰
- サポートベクトルマシン（SVM）
- 決定木
- ランダムフォレスト
- k 近傍法
- 線形回帰
- スパース回帰

教師なし学習

教師なし学習は、訓練データとして特徴量のみが与えられるときに行う学習方法である。機械が大量なデータ（特徴量）を解釈し、データに隠されたパターンを抽出して、グループ分けしたりする。教師なし学習は、クラスタリングや次元削減・特徴抽出、外れ値検出などに適用される。

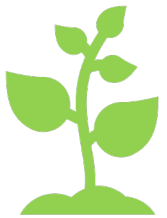
- 階層型クラスタリング
- k-means
- 自己組織化モデル（SOM）
- トピックモデル（pLSI, LDA）
- 主成分分析（PCA）
- 線形判別分析（LDA）

強化学習

最適化問題

回帰問題

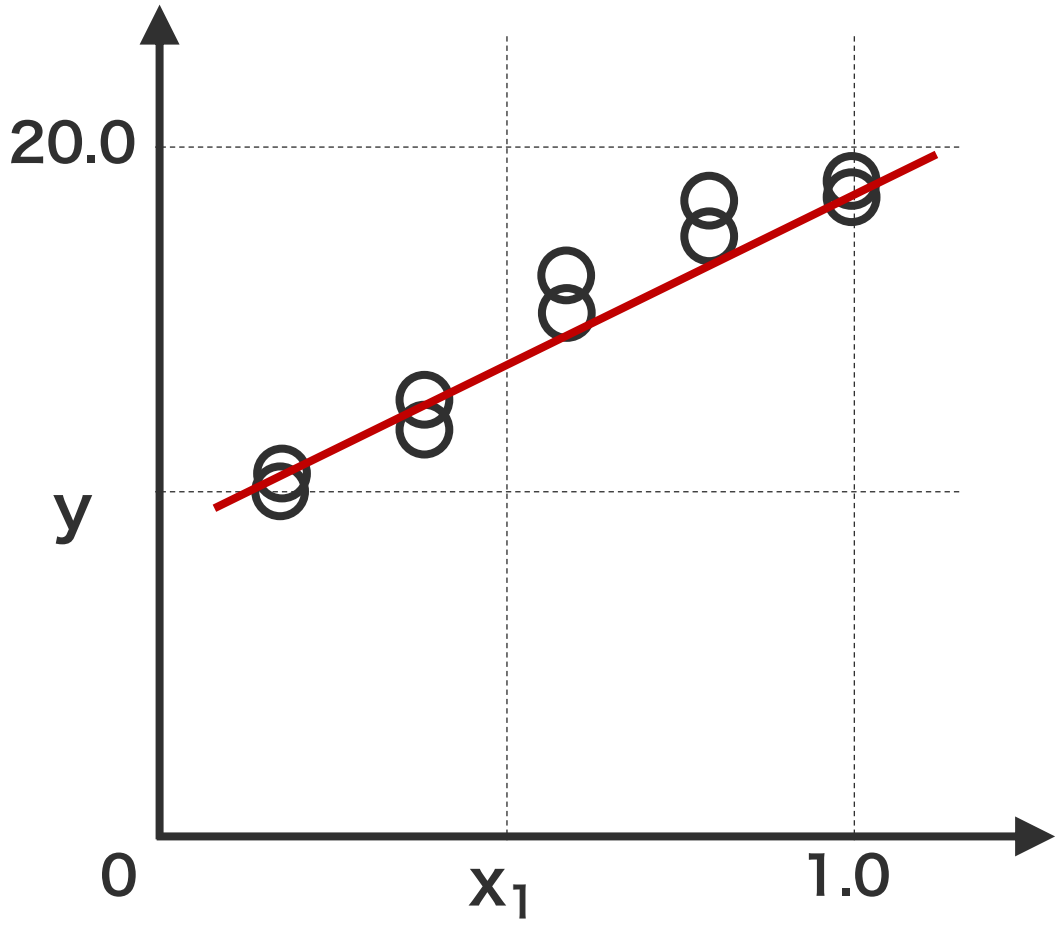
施肥量を利用して収穫量を予測するにはどうすればいいのか？

実験区画	区画 1	区画 2	区画 3	区画 4	区画 5
施肥量	0.2	0.4	0.6	0.8	1.0
収穫量					
	10.2	12.4	16.1	17.8	18.2
					
	10.1	11.9	17.2	18.1	18.0

回帰問題

施肥量を利用して収穫量を予測するにはどうすればいいのか？

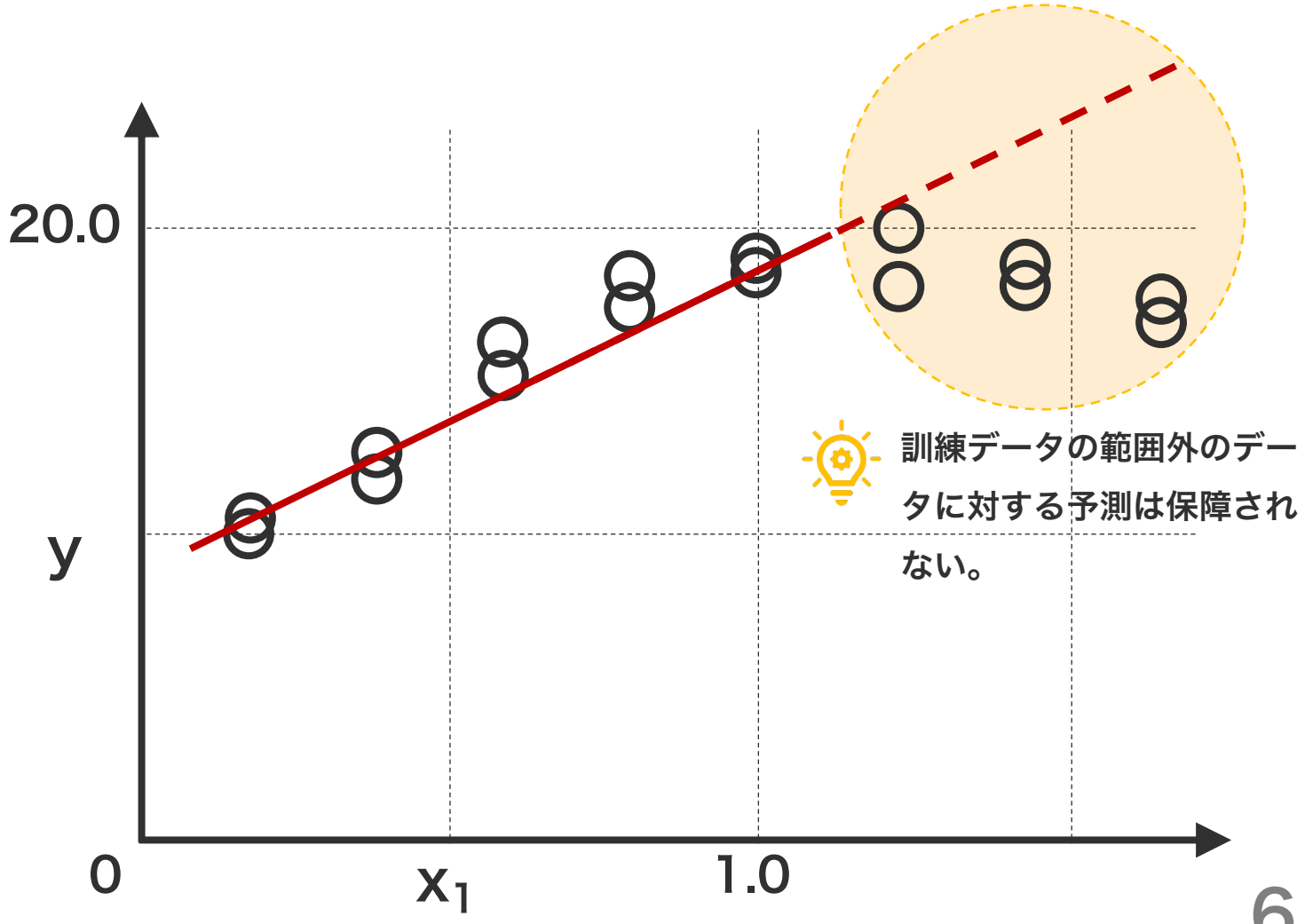
収穫量	施肥料
10.2	0.2
10.1	0.2
12.4	0.4
11.9	0.4
16.1	0.6
17.2	0.6
17.8	0.8
18.1	0.8
18.2	1.0
18.0	1.0



回帰問題

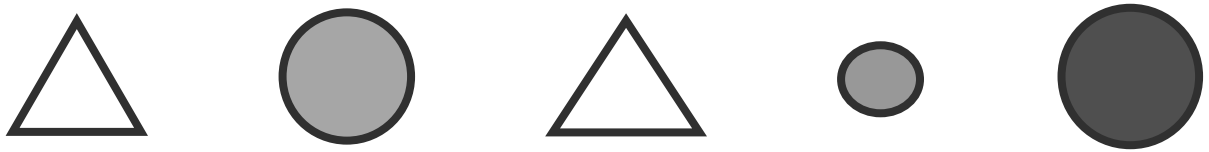
施肥量を利用して収穫量を予測するにはどうすればいいのか？

収穫量	施肥料
10.2	0.2
10.1	0.2
12.4	0.4
11.9	0.4
16.1	0.6
17.2	0.6
17.8	0.8
18.1	0.8
18.2	1.0
18.0	1.0

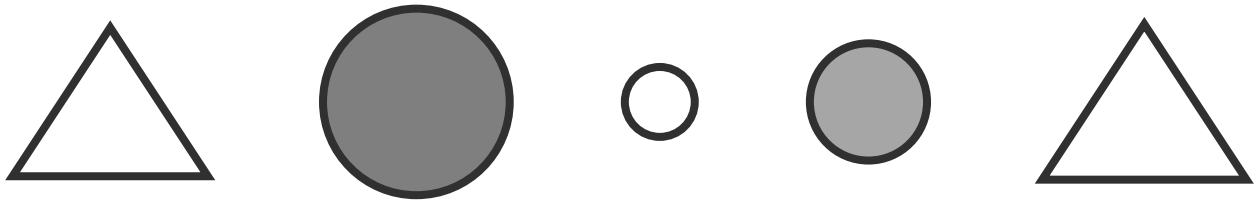


分類問題

円形と三角形を分類したい場合、どの特徴に着目すればいいのか？



x_1	1.0	0.5	1.0	0.4	0.1
x_2	0.8	1.0	0.8	0.9	1.0



x_1	1.0	0.3	1.0	0.5	1.0
x_2	0.8	1.0	1.0	1.0	0.7

色

色に着目すると、三角形は白が多く、円形は黒い場合が多い。ここで、白を 1、黒を 0 とする色の指標を考える。

0 1

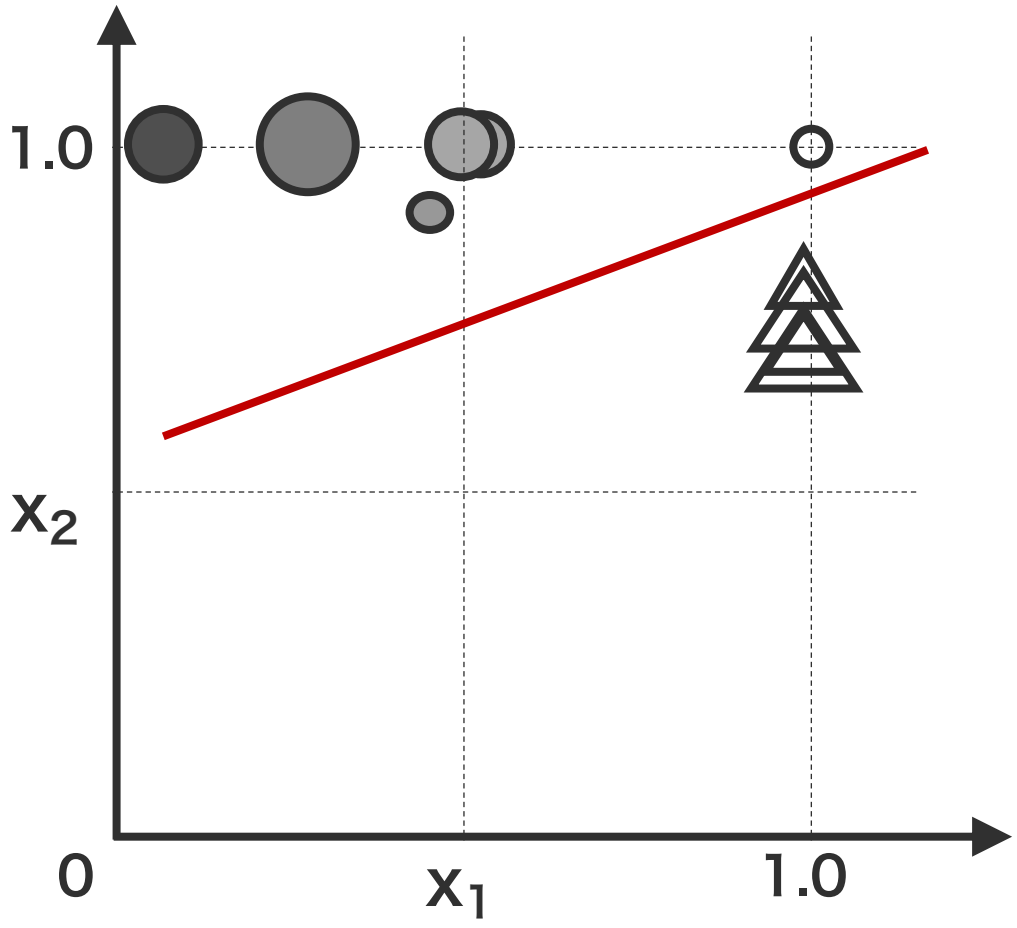
外接円との面積比

分類したい形とその外接円の面積の比に着目すると、三角形の場合は 1 よりも小さく、円形の場合はほぼ 1 になる。

分類問題

円形と三角形を分類したい場合、どの特徴に着目すればいいのか？

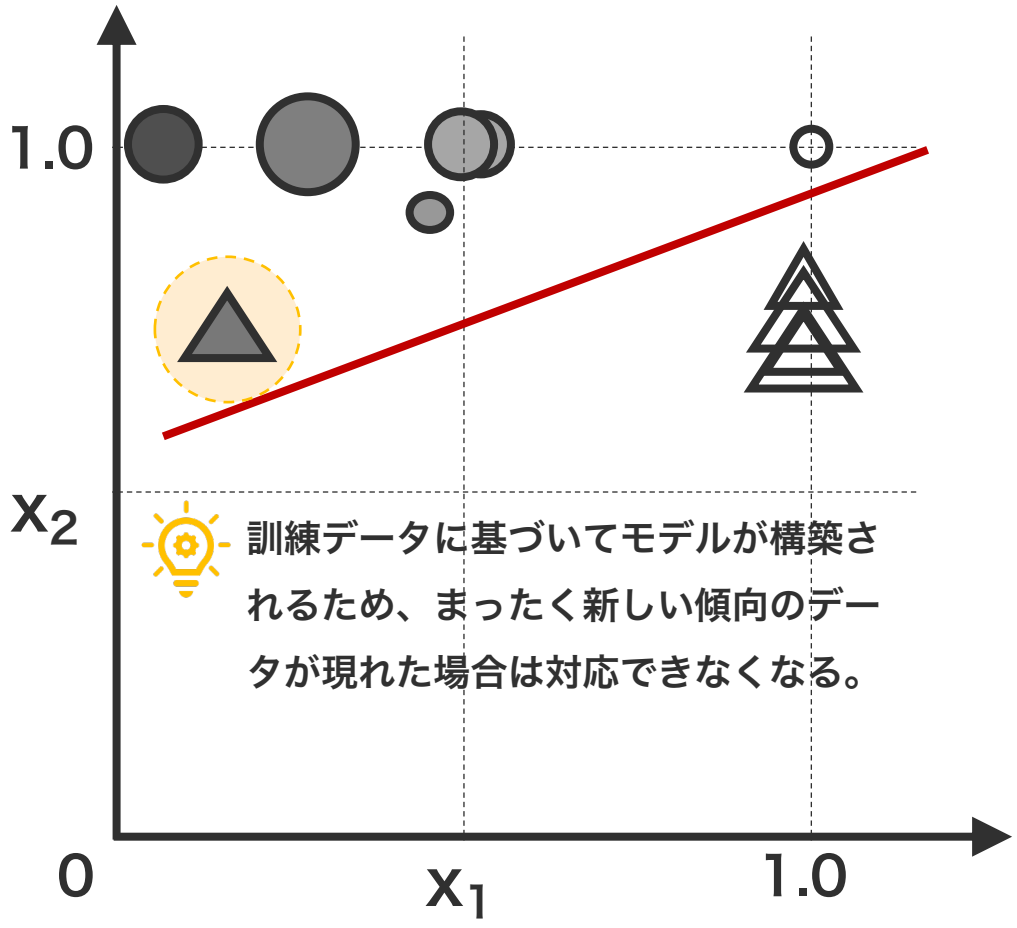
ラベル	X_1	X_2
1	1.0	0.8
0	0.7	1.0
1	1.0	0.8
0	0.8	0.9
0	0.9	1.0
1	1.0	0.8
0	0.8	1.0
0	1.0	1.0
0	0.7	1.0
1	1.0	0.7



分類問題

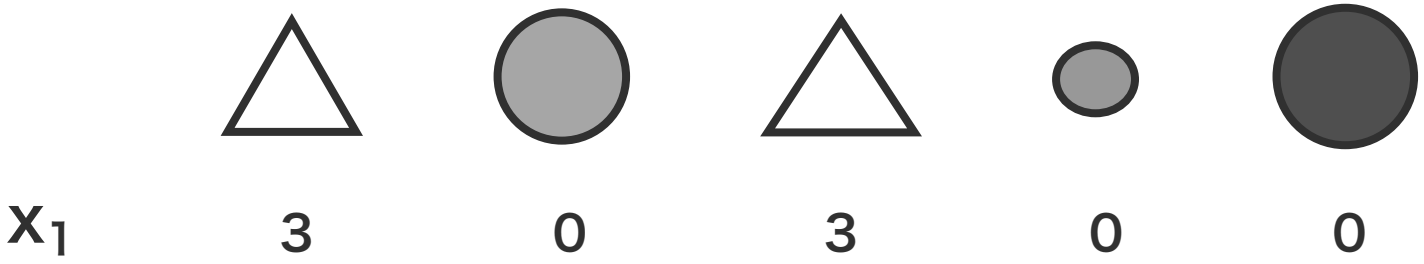
円形と三角形を分類したい場合、どの特徴に着目すればいいのか？

ラベル	X_1	X_2
1	1.0	0.8
0	0.7	1.0
1	1.0	0.8
0	0.8	0.9
0	0.9	1.0
1	1.0	0.8
0	0.8	1.0
0	1.0	1.0
0	0.7	1.0
1	1.0	0.7

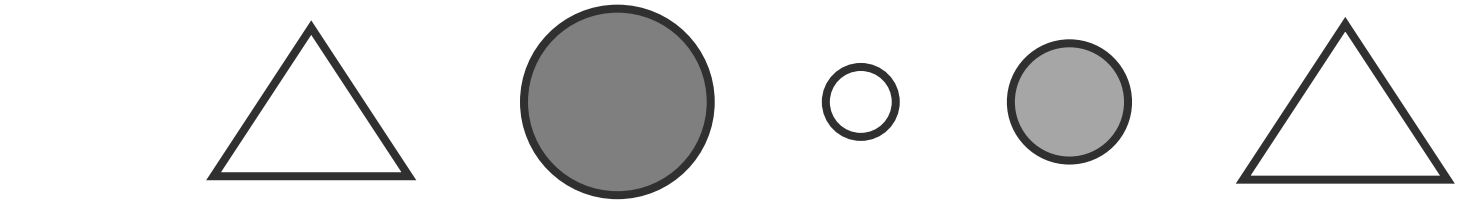


分類問題

円形と三角形を分類したい場合、どの特徴に着目すればいいのか？



x_1	3	0	3	0	0



x_1	3	0	0	0	3

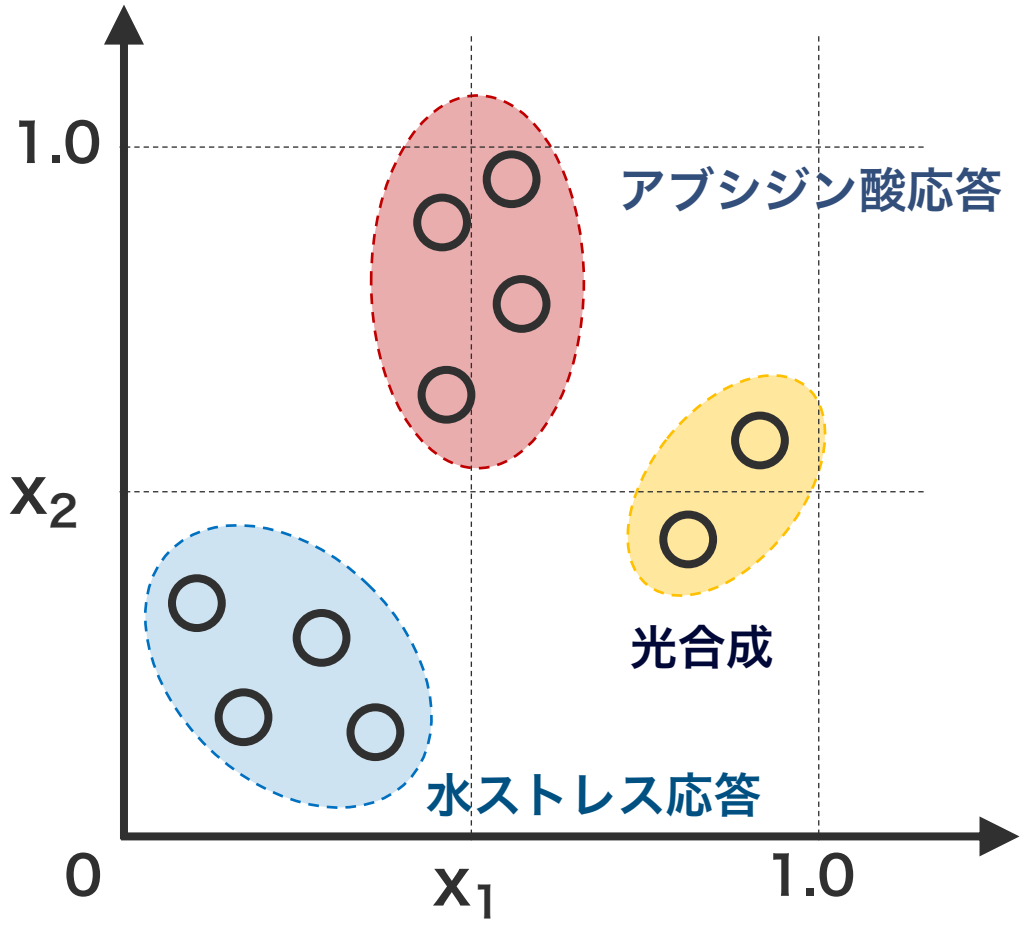
角の数

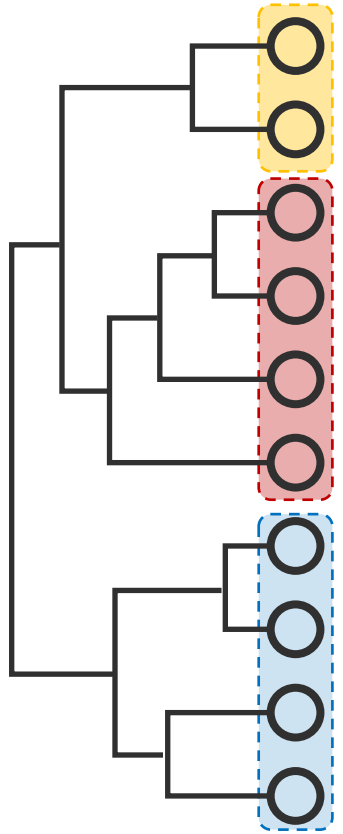
円形の角は 0 個で、三角形の角の数は 3 個である。そのため、角の数を特徴量をして使えば、この 1 つの特徴量だけで円形と三角形を分けられるようになる。

クラスタリング

10 個の遺伝子に対して、乾燥ストレス処理前 X_1 と処理後 X_2 の発現量を測定した。遺伝子間の関係を調べるにはどうすればいいのか？

X_1	X_2
0.12	0.38
0.92	0.57
0.39	0.21
0.52	0.98
0.49	0.88
0.26	0.25
0.34	0.36
0.49	0.61
0.53	0.73
0.82	0.46

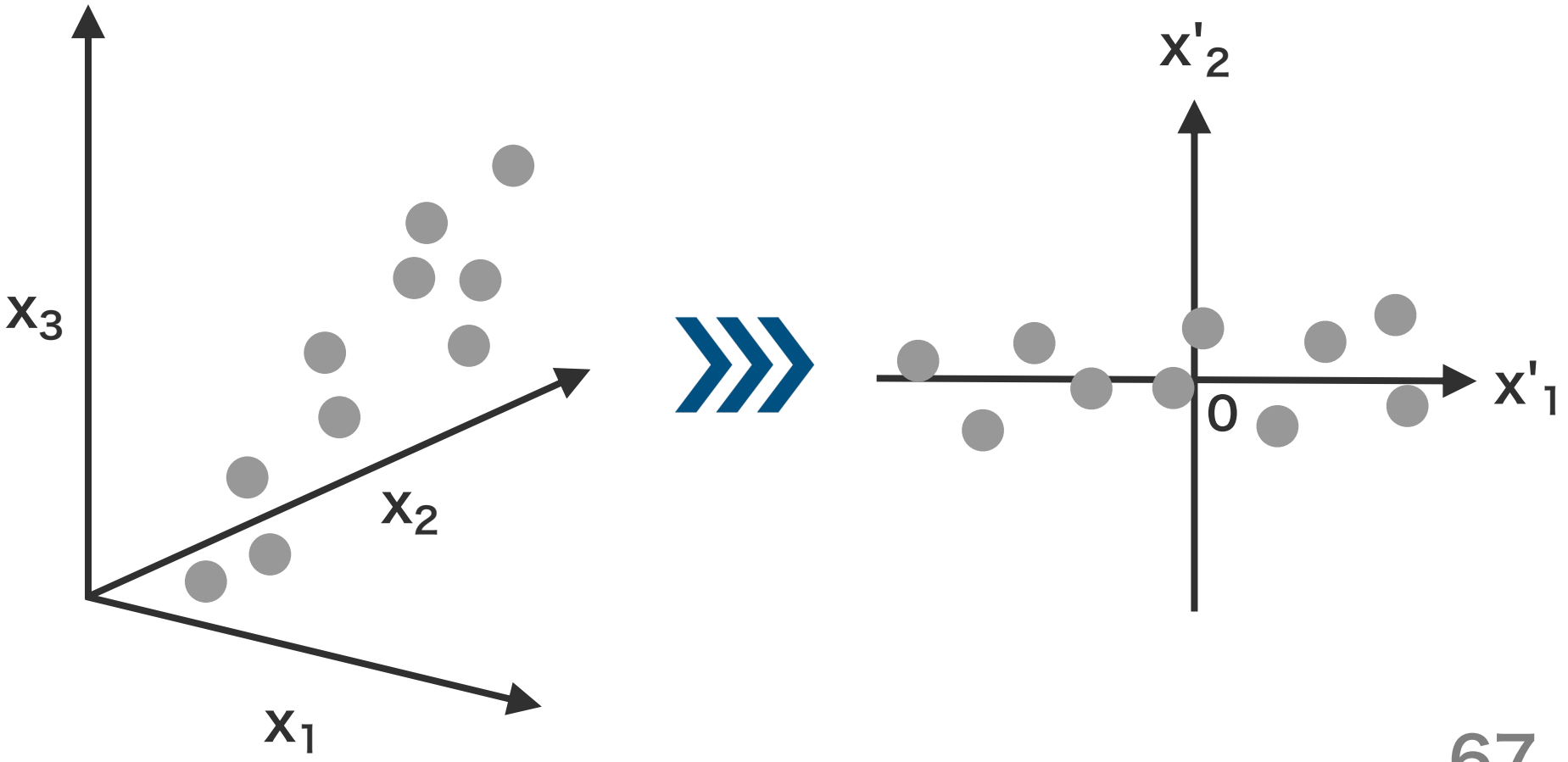




次元削減・特徴量抽出

多次元の情報を二次元の情報に落として図示したい。

X_1	X_2	X_3
0.12	0.38	0.54
0.92	0.57	0.12
0.39	0.21	0.42
0.52	0.98	0.85
0.49	0.88	0.24
0.26	0.25	0.53
0.34	0.36	0.87
0.49	0.61	0.42
0.53	0.73	0.13
0.82	0.46	0.56



コラム

- 人工知能
- 機械学習
- モデル開発

データ収集

市場調査を行い、ニーズを確定し、モデル構築に必要なデータを収集する。

モデル構築

入力データを使用して、パラメーターチューニングを行い、最適なモデルを構築する。

ワークフロー構築

収集されたデータに対して欠損値や異常値処理を行い、データを機械学習が使える形に整形し、機械学習アルゴリズムに渡すまでのワークフローを構築する。

実装

機械学習モデルをアプリ化し、スマホや専用機器に実装し、実用化する。

市場調査を行い、ニーズを確定し、モデル構築に必要なデータを収集する。

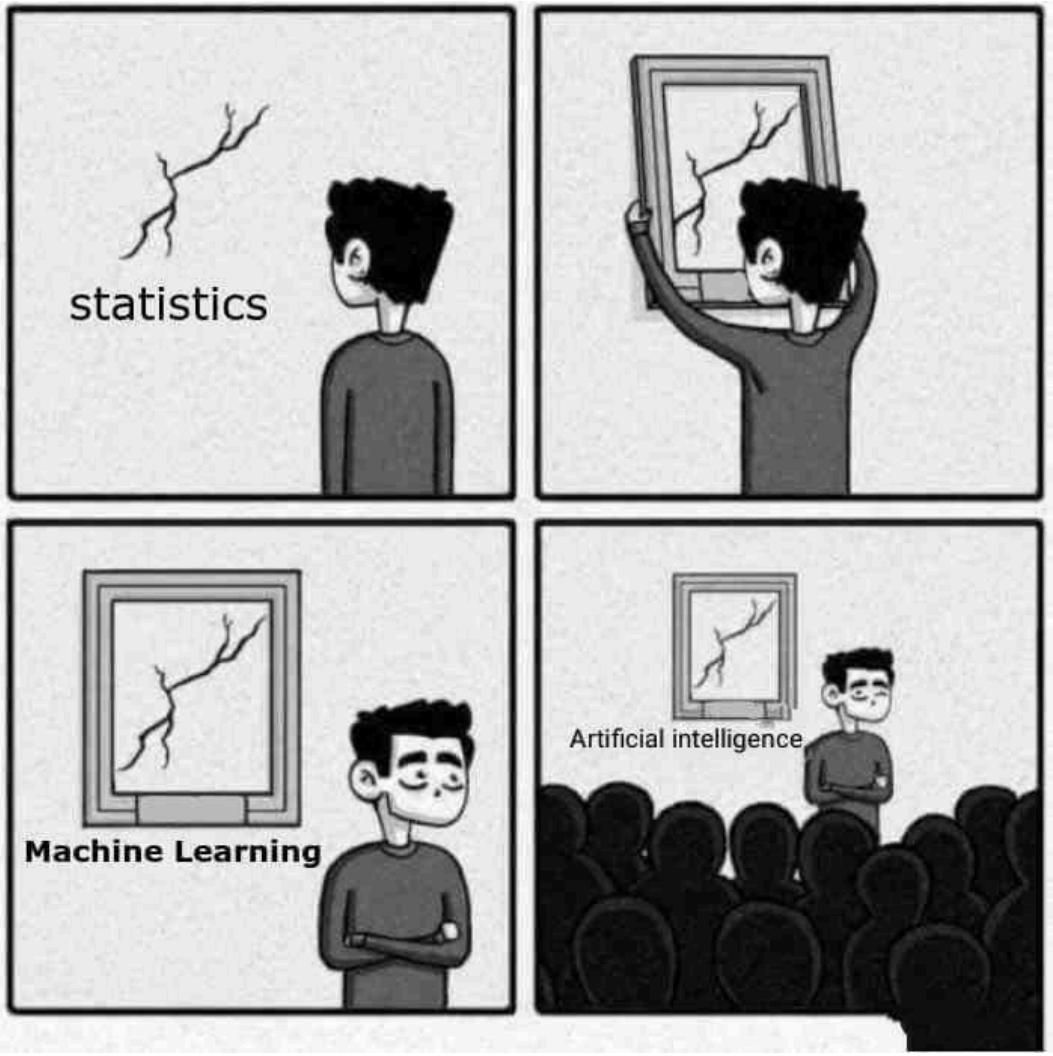
収集されたデータに対して欠損値や異常値処理を行い、データを機械学習が使える形に整形し、機械学習アルゴリズムに渡すまでのワークフローを構築する。

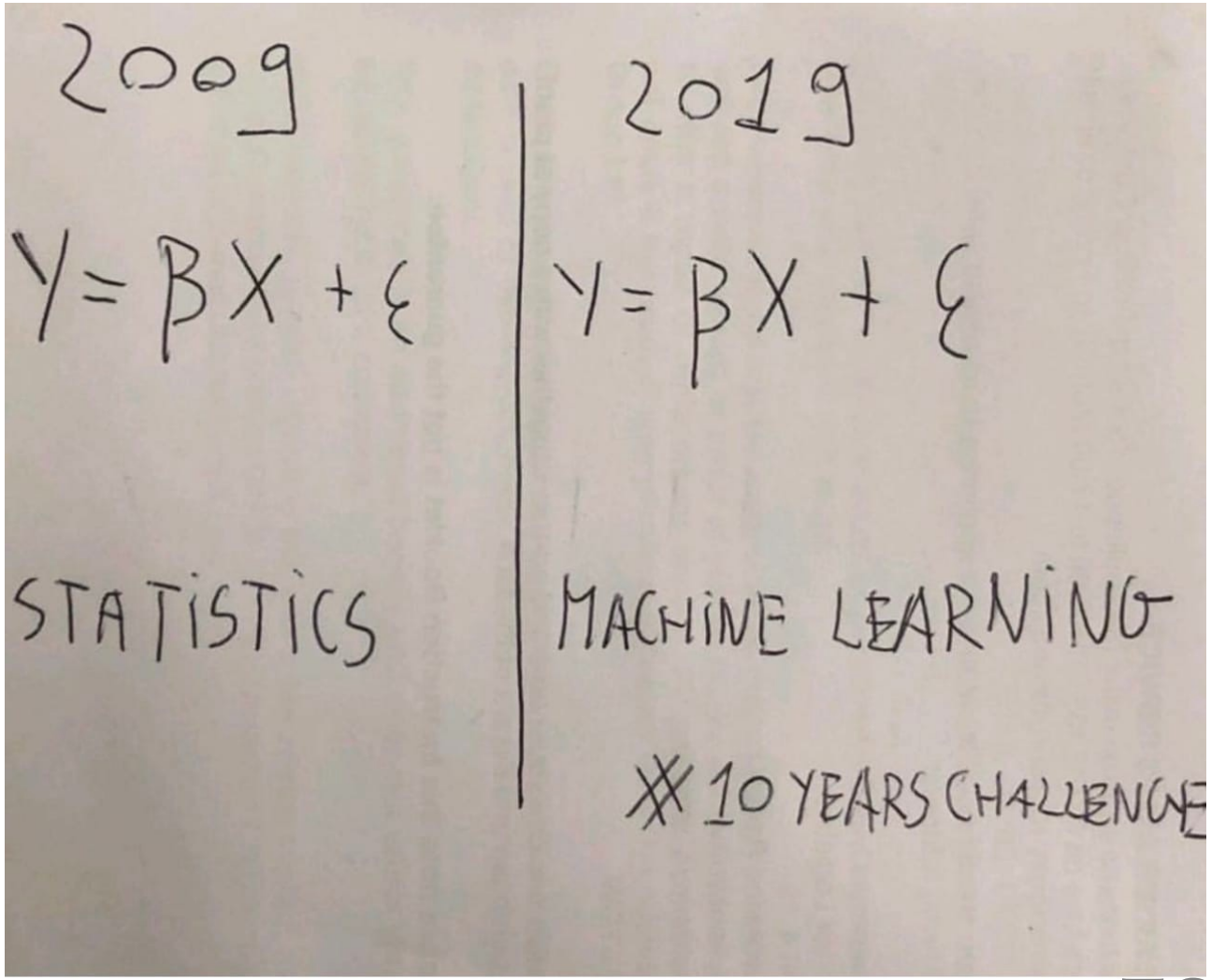
入力データを使用して、パラメーターチューニングを行い、最適なモデルを構築する。

機械学習モデルをアプリ化し、スマホや専用機器に実装し、実用化する。

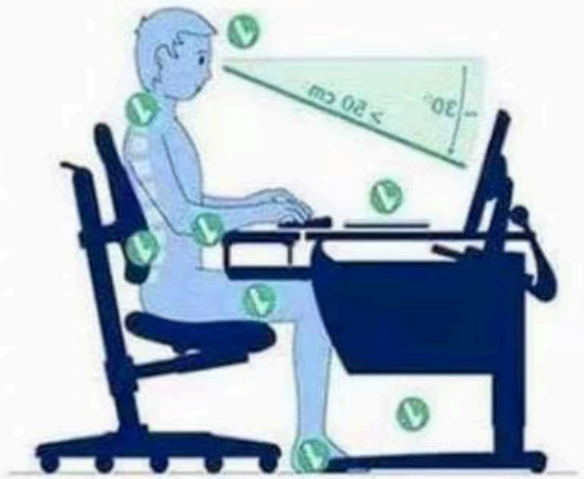
- 解決すべき問題を明確に定義する。
- 問題を解決するためのデータを収集する。
  - 機械学習の成否はデータの量と質に影響される。
  - 継続的なデータ収集フローを制定する。
- 手作業によるデータ分析を行い、効率的な機械学習アルゴリズムを検討する。

データ解析


statistics
Machine Learning
Artificial intelligence

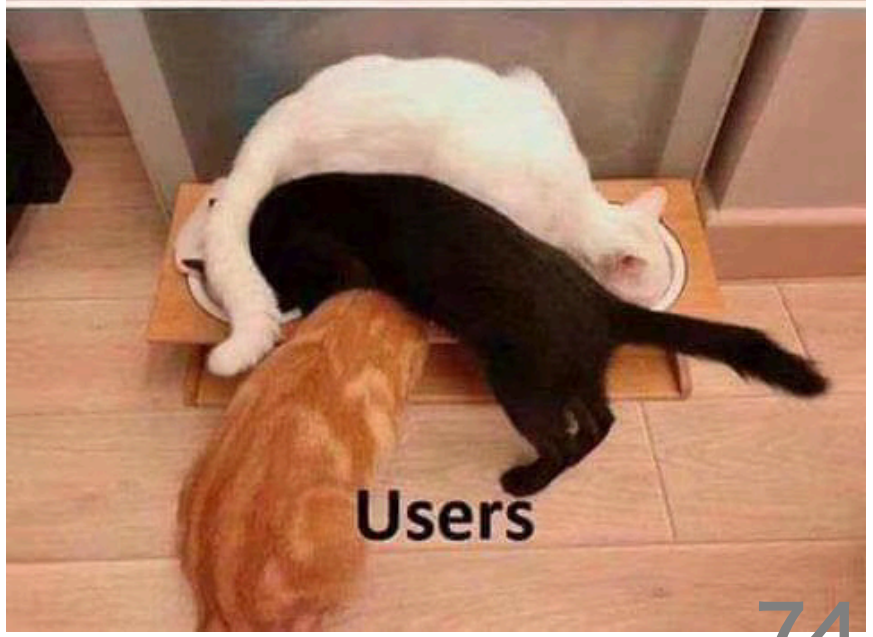

2009
$Y = \beta X + \varepsilon$
STATISTICS
2019
$Y = \beta X + \varepsilon$
MACHINE LEARNING
※10 YEARS CHALLENGE

- 開発時と運用時のシステム運用方法を検討する。
  - 運用時に収集されたデータを学習に再利用できるようになっているか。
  - 運用時に専門家によるチェック過程が組み込まれているか。
- 運用時のソフトウェア・ハードウェアを検討する。
  - どのようなユーザーが、どのような使い方を想定し、どのような API を必要とするのか。



How I'm supposed to sit

How I sactualy sit



- 単一のモデルだけで、問題をすべて解決できない。
  - 複数のモデルを組み合わせて１つの問題を解決することが多い。
- 既存のアルゴリズムを優先的に使う。
  - 既存のアルゴリズムは開発コストが少なく、応用実績も多い。
- 機械学習は必ず成功するとは限らない。

テキスト抽出モデル

EXIT

言語推定モデル

{English: EXIT}

翻訳モデル

{中文: 退出}

画像置換モデル

- アプリケーションへの実装は外注で行う。
  - 研究時間を確保するために、研究要素の少ない実装は外注する。
  - 開発会社の技術力が高く、使いやすくてセキュリティの堅い実装を可能にする。
- ユーザーによる評価やフィードバックを積極的に対応する。

市場調査を行い、ニーズを確定し、モデル構築に必要なデータを収集する。

収集されたデータに対して欠損値や異常値処理を行い、データを機械学習が使える形に整形し、機械学習アルゴリズムに渡すまでのワークフローを構築する。

入力データを使用して、パラメーターチューニングを行い、最適なモデルを構築する。

機械学習モデルをアプリ化し、スマホや専用機器に実装し、実用化する。