

# ニコニコAIスクール 第1回

## 講義概要

18/01/06

講師：八木 拓真

$$y = f(x)$$

出力

“2”

学習器

入力



- ▶ 明確な定義はないが、概ね次のような見方が多くの賛同を得ている：

"Field of study that gives computers the ability to learn without being explicitly programmed" –  
Arthur Samuel (1959)

- ▶ コンピュータが未知のパターンから、人間が明確に指示することなく期待する分類や構造を導き出すことができれば、「学習」したと考える

→人間の学習 (e.g. 人間や動物が過去の経験から環境に適応すること) とは意味合いがやや異なることに注意

おすすめ新商品情報



## 推薦システム



## 画像認識・自然言語処理



## 機械制御



音声認識

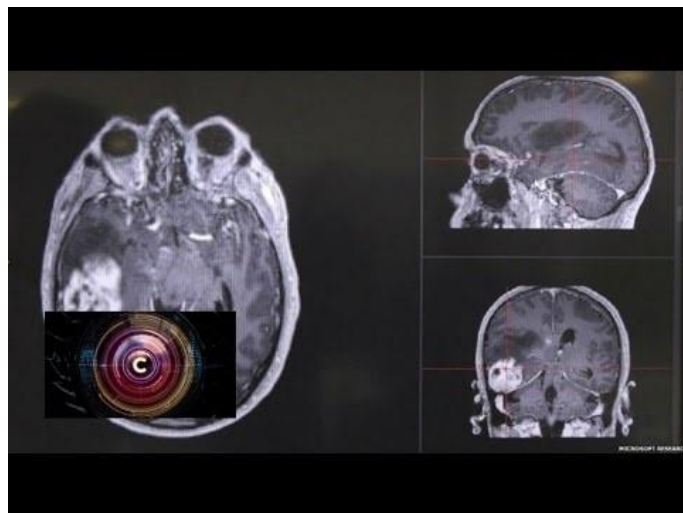
将棋・囲碁

この10年で  
目覚ましい進化



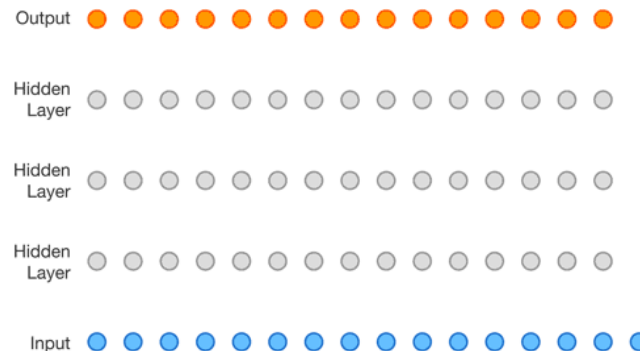
horse → zebra

## 画像のスタイル変換



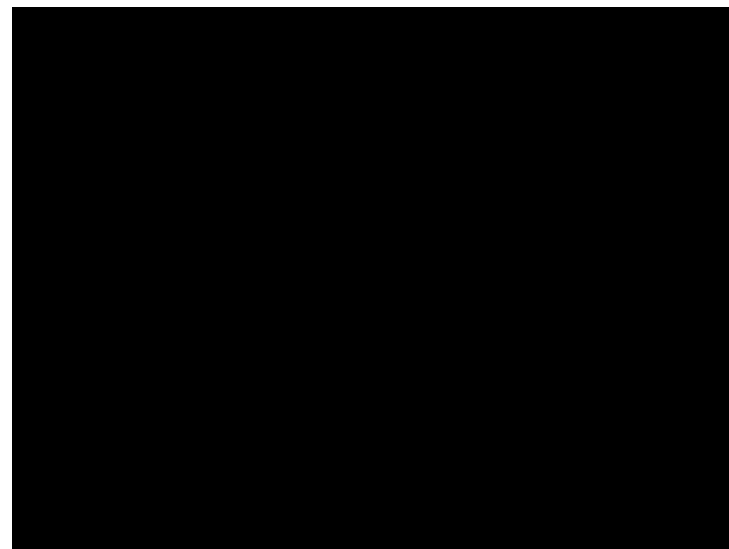
## 脳腫瘍セグメンテーション

<https://www.microsoft.com/en-us/research/project/medical-image-analysis/>



## 音声合成

<https://deepmind.com/blog/wavenet-launches-google-assistant/>



“好奇心”に基づくゲームプレイ

- ▶ 強化学習の普及
  - AlphaGo Zero:自己学習で (プロの棋譜を使わずに) プロを超える性能を発揮
  - ロボット駆動・ビジョンなどへの応用が盛んに
- ▶ Deep Bayesian Learning
  - ベイズ学習の理論を深層学習に導入し、不確定性に対する一貫した枠組みを提供
- ▶ Meta Learning
  - 学習器の学習を制御するmeta-learnerの導入
- ▶ Generative Adversarial Networks (GANs)
  - 画像生成から映像生成へ
- ▶ その他：非ユークリッド幾何、解釈性、汎化誤差の解析

実システムへの適用のみならず、研究も加速度的に進展中  
(主要国際会議※の発表数は2017年だけで4000本超)



**Rosenblattのパーセプトロン (1958)**  
**現在のニューラルネットワークの原型**

**Hubel & Wiesel**  
**単純型・複雑型細胞 (1962)**





## ニューラルネット (NN)

1943: ニューロンの数理モデル  
1952: 微分方程式モデル (H-Hモデル)  
1958: パーセプトロン (Rosenblatt)  
1962: 単純・複雑型細胞 (Hubel & Wiesel) ~  
1967: 誤差逆伝播法の原型 (甘利俊一) ゆ  
1979: ネオコグニトロン (福島邦彦) る  
1986: 誤差逆伝播法の確立 (Rumelhart) や  
1989: 畳み込みニューラルネット (LeCun) か  
1997: Long Short-Term Memory (Schmidhuber) な  
2012: 深層学習の成功 (ILSVRC) 交流  
~

## 学習理論

有史以前: 確率論・統計学・最適化理論  
1973: Dirichlet過程 (Ferguson)  
1974: 赤池情報量規準 (赤池弘次)  
1977: EMアルゴリズム  
1984: PAC学習 (Valiant)  
1992: 非線形SVM・カーネル法 (Vapnik)  
1995: AdaBoost (Freund and Schapire)  
1996: Lasso (Tibshirani)  
2003: トピックモデル (Blei)  
2010: WAIC (渡辺澄夫)

## 現代: NNと学習理論の融合

- ・ 神経回路の模倣に端を発するNN
- ・ 数理統計学の延長としての学習理論

が融合したのが現在「機械学習」と呼ばれているもの  
(さらに情報理論・記号处理的AIも合流)



## モデルとプログラムの橋渡し

数式で記述されたモデルを、正しくプログラムとして書き下す練習をする

### (1) 数式の理解

数式的に何が起きているか

$$y = \sigma(W^T \mathbf{x})$$

$\Leftrightarrow$

### (2) 実装の理解

数式をどのようにしてコードに書き起こすか

```
F.Sigmoid(  
np.dot(w.T, x))
```

## ▶ 機械学習を用いたデータ解析

- Pythonの文法と書き方を理解し、データ解析に必要な手続きを記述できる
- numpyを用いて行列演算を扱える
- 基本的な機械学習アルゴリズムの性質を理解し、自分のデータに適用することができる

## ▶ 全脳アーキテクチャ開発のための要素技術

- 深層学習の主要なアルゴリズムを学ぶ
- Neuro-Inspiredな数理モデルを、ライブラリの力を借りながら実装できる

## ▶ 体系的な統計学入門

- 小島寛之『完全独習 統計学入門』
- 東京大学教養学部統計学教室『統計学入門』

## ▶ やや高度な統計モデリング

- 久保拓弥『データ解析のための統計モデリング入門』

## ▶ 機械学習アルゴリズムの数理的側面

- 高村大也『言語処理のための機械学習入門』

## ▶ その他参考資料：

- Sebastian Raschka『Python機械学習プログラミング』
- 斎藤 康毅『ゼロから作るDeep Learning』
- 岡谷 貴之『深層学習』
- Coursera Machine Learning (Andrew Ng)
- スタンフォード大の講義資料 (CS229, CS231n)

## ▶ 講義 (90分)

- 前回の復習 (5分)
- 講義と解説 (55分)
- 基礎演習 (30分、colaboratoryでサンプルを動かす)

## ▶ 休憩 (10分)

## ▶ 演習 (90分)

- 課題説明 (5分)
- 演習 (60分)
- 解説・コードレビュー (15分)
- フィードバック・次回予告 (10分)

# Python ・ Numpy入門

- ▶ 1994年 (1.0) に登場
- ▶ 数値計算からGUIまでサポートする汎用言語
- ▶ 動的型付け (記述が簡潔)
- ▶ インタプリタ方式 (コンパイル不要)
- ▶ 同じ機能を実現するとき、なるべく同じ書き方になることを意図した設計

There should be one—and preferably only one—obvious way to do it.

PEP 20 -- The Zen of Python (<https://www.python.org/dev/peps/pep-0020/>)

- ▶ **Numpy/Scipy**をはじめとした数値計算ライブラリ、**多数の深層学習ライブラリ**がPython用の開発されており、C++などと並んで機械学習分野で利用されている

## Python

```
# -*- coding: utf-8 -*-
```

```
def fib(n):  
    result = 0  
    tmp = 0  
    for x in range(n):  
        result += tmp  
        tmp = result
```

```
if __name__ == "__main__":  
    print(fib(6))
```

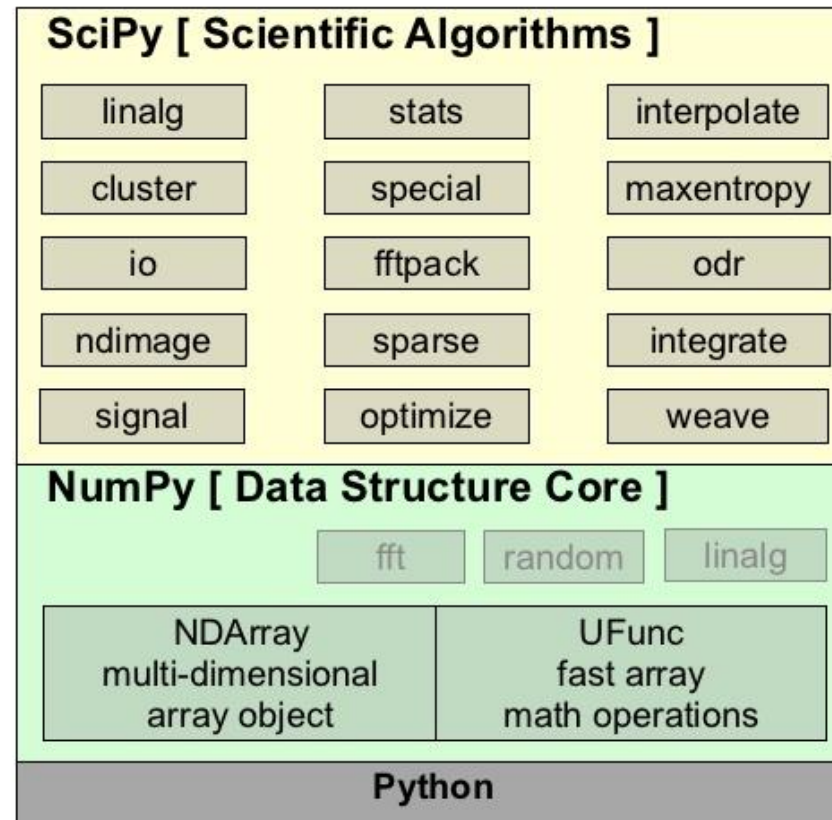
## C++

```
#include <iostream>  
#include <vector>  
  
int fibonacci(int n){  
    std::vector<int> result;  
    result.push_back(0);  
    result.push_back(1);  
    for (int i = 0; i < n; i++){  
        result.push_back(result.at(i) + result.at(i+1));  
    }  
    return result.at(n);  
}  
  
int main(int argc, char** argv){  
    std::cout << fibonacci(6) << "¥n";  
    return 0;  
}
```

括弧ではなく、インデント(段組み)で階層を表す  
行数が少なく、簡潔 (デメリットになる場合も)



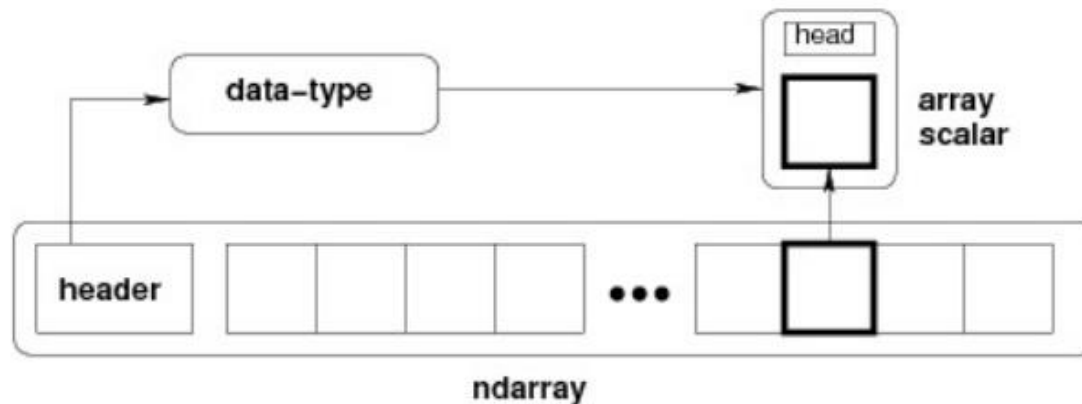
- ▶ Numpy: 高速な数値計算ライブラリ
- ▶ Scipy: numpyを含む、科学技術演算用の各種ライブラリのセット
- ▶ BLASなどの基礎的な数値計算ルーチンの力を借りることで、機械学習に必須のベクトル演算を高速化
- ▶ Pythonで数値計算ができるのは、numpyのおかげ



## NumPy Array

A NumPy array is an N-dimensional homogeneous collection of “items” of the same kind. The kind can be any arbitrary structure of bytes and is specified using the data-type.

0	1	2	3	4	5
10	11	12	13	14	15
20	21	22	23	24	25
30	31	32	33	34	35
40	41	42	43	44	45
50	51	52	53	54	55



## Array Slicing

SLICING WORKS MUCH LIKE  
STANDARD PYTHON SLICING

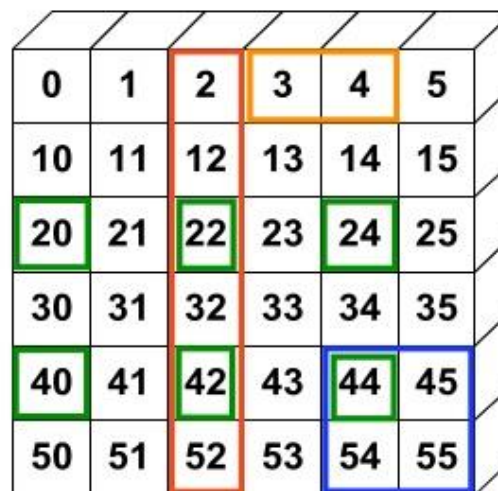
```
>>> a[0,3:5]
array([3, 4])

>>> a[4:,4:]
array([[44, 45],
       [54, 55]])

>>> a[:,2]
array([2, 12, 22, 32, 42, 52])
```

STRIDES ARE ALSO POSSIBLE

```
>>> a[2::2,::2]
array([[20, 22, 24],
       [40, 42, 44]])
```



0	1	2	3	4	5
10	11	12	13	14	15
20	21	22	23	24	25
30	31	32	33	34	35
40	41	42	43	44	45
50	51	52	53	54	55

## Fancy Indexing in 2-D

```
>>> a[(0,1,2,3,4),(1,2,3,4,5)]  
array([ 1, 12, 23, 34, 45])
```

```
>>> a[3:,[0, 2, 5]]  
array([[30, 32, 35],  
       [40, 42, 45],  
       [50, 52, 55]])
```

```
>>> mask = array([1,0,1,0,0,1],  
                 dtype=bool)
```

```
>>> a[mask,2]  
array([2, 22, 52])
```

0	1	2	3	4	5
10	11	12	13	14	15
20	21	22	23	24	25
30	31	32	33	34	35
40	41	42	43	44	45
50	51	52	53	54	55



Unlike slicing, fancy indexing creates copies instead of a view into original array.

# 演習のやり方・進め方

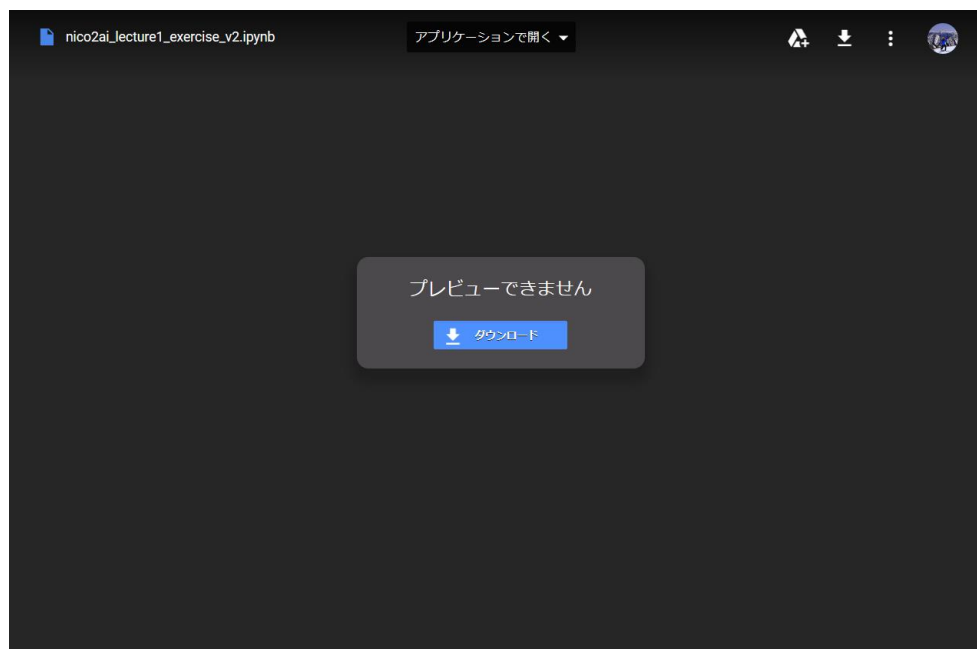
- ▶ クラウド上で動くJupyter notebookの拡張版
- ▶ 基本的にはJupyter notebookと互換性があり、既存のnotebookを動かすことが可能
- ▶ Google Drive上で共有可能で、同時編集も可能
- ▶ Python3にも対応
- ▶ Jupyter notebookにない機能としては、
  - 同時編集・コメント
  - セル単位での共有
  - インライン数式

などがある

- ▶ 公式サポートはChromeのみ (Firefoxでも一応動く)



- ▶ **Google Chrome**で生徒用フォルダへのリンクを開く (Slackで共有)
- ▶ Lecture1->nico2ai\_lecture1\_exercise\_v2.ipynbをクリックし、下記の画面が開いたら右上のAdd to driveボタンをクリックして自身のGoogle driveフォルダに追加



- ▶ 次のURLにアクセス：  
<https://colab.research.google.com/>  
(または <https://goo.gl/U63qpr>)
- ▶ ポップアップを一度cancelして、左上のFile->Open drive notebookを選択
- ▶ My Driveタブを選択し、以下のファイルを開く  
nico2ai\_lecture1\_exercise\_v2.ipynb

- ▶ 本講義では、colaboratory (無料) 及び iLect (有料) を使用予定です。いずれも環境構築なしにPythonを実行できるようになっていますが、手元で実行したい場合もあると思います
- ▶ Windows・Mac・Linux共通で使える方法として、Anaconda+Jupyter Notebookを推奨します
  - Anaconda: <https://www.continuum.io/downloads>
  - Jupyter Notebook:  
<http://jupyter.readthedocs.io/en/latest/install.html>
- ▶ インストールして、コマンドプロンプトまたはターミナル上でディレクトリに移動し、“jupyter notebook”と打てば同じ画面が出てきます

- ▶ Pythonチュートリアル (公式)

<https://docs.python.jp/3/tutorial/>

- ▶ Numpy 100 exercises (英語)

<http://www.labri.fr/perso/nrougier/teaching/numpy.100/>

Numpyの基礎文法を速習できる100個の小演習。

- ▶ Dive Into Python 3 日本語版 (経験者向け)

<http://diveintopython3-ja.rdy.jp/index.html>

- ▶ 統計的機械学習入門

[https://www.nii.ac.jp/userdata/karuizawa/h23/111104\\_3rdlecueda.pdf](https://www.nii.ac.jp/userdata/karuizawa/h23/111104_3rdlecueda.pdf)

- ▶ Numpy Talk at SIAM

<https://www.slideshare.net/enthought/numpy-talk-at-siam>