

Pythonの使い方

2020.04.21

スタートアップゼミ #3

M1 須賀拓実

Pythonとは

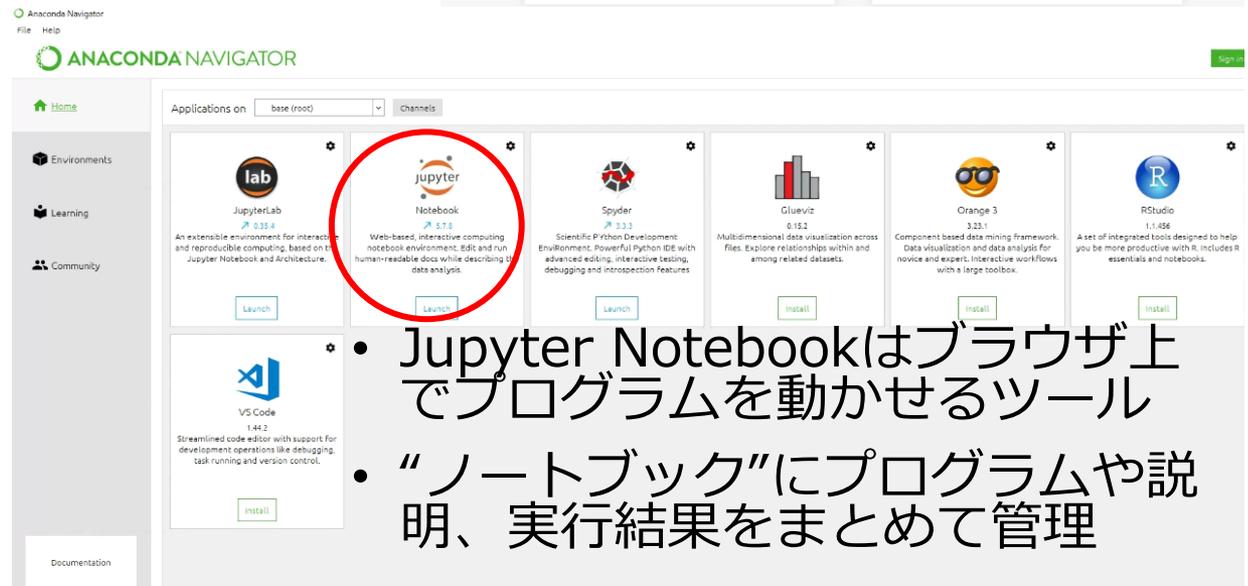
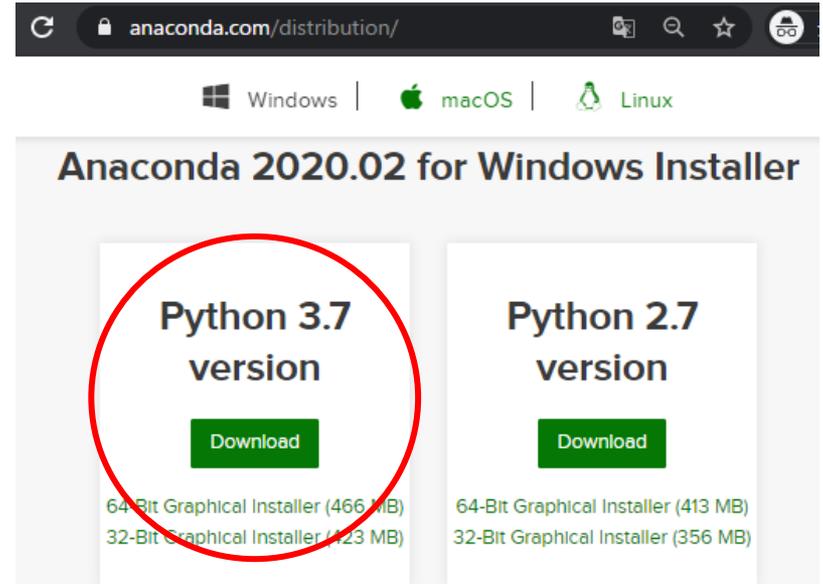
- Google3大言語の1つ(C++, Java, Python)
- コードがシンプルで扱いやすく設計、可読性が高い
- 標準ライブラリや関数など豊富で大規模なツール
- ブラウザからGPU利用で高速計算も可能(Google Colab)
- IDE(統合開発環境：編集しやすくなるソフト)：PyCharmなど
- Anacondaを入れれば簡単にPythonが始められる

Anaconda

- Pythonパッケージなどを提供するプラットフォーム

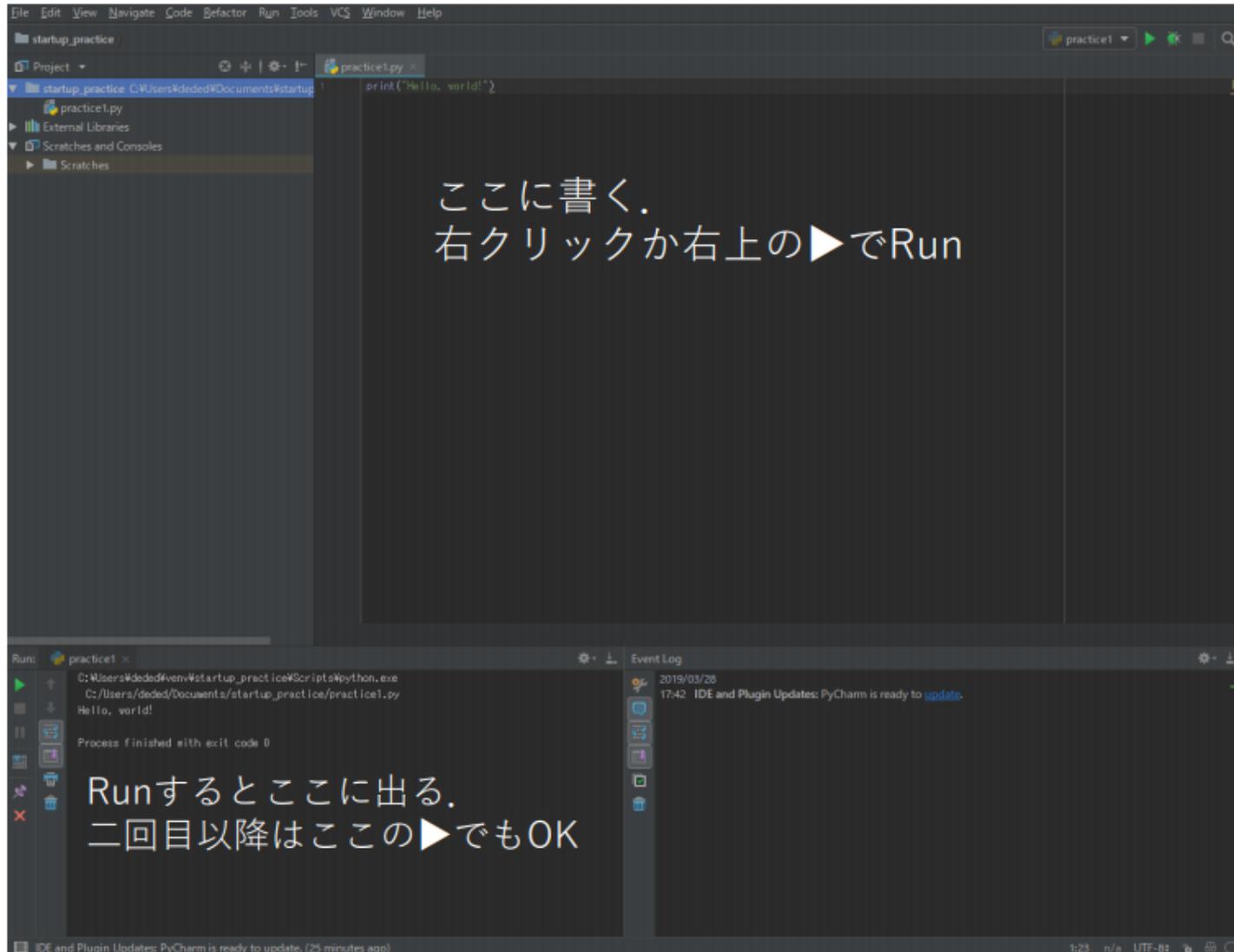
パッケージ：numpy, pandasなど

- Pythonの開発環境やエディターも同時にインストールされる
- 自分のOSに合わせてコンパイル済のパッケージを得られる
- ライブラリの追加にはAnaconda Navigatorを開く



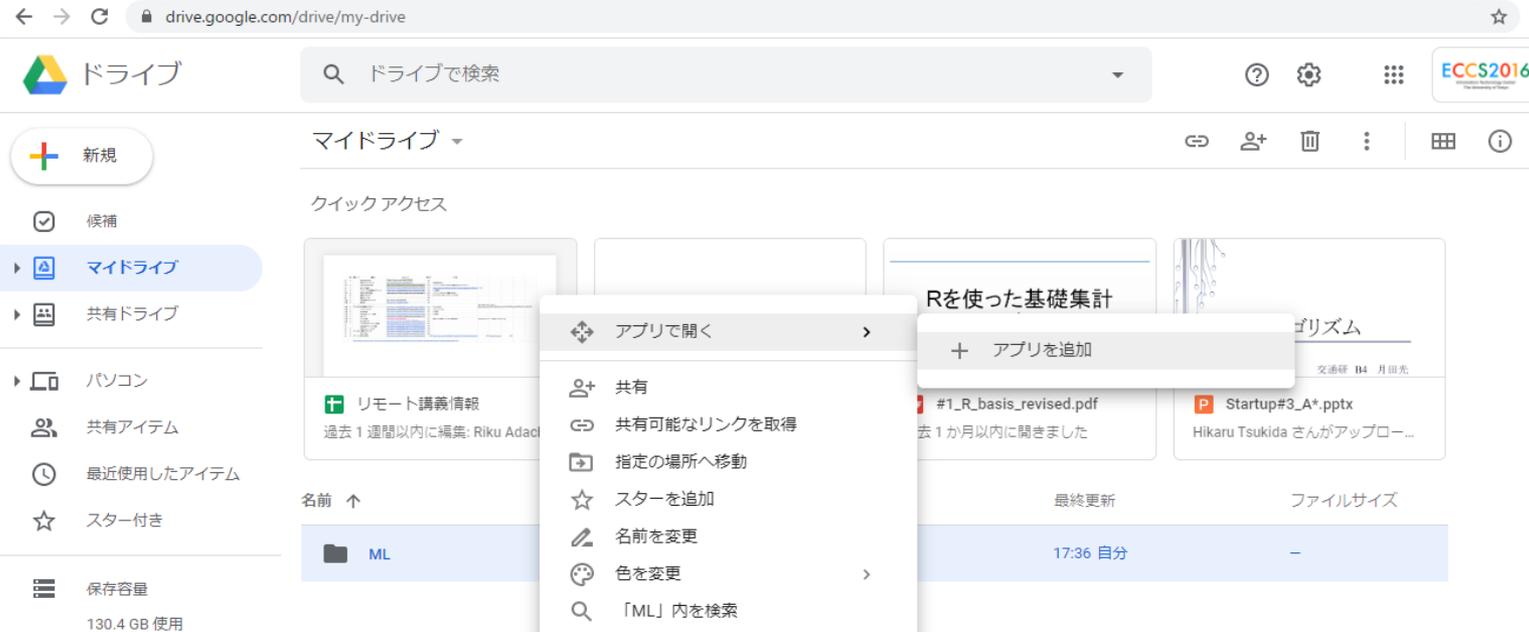
IDE: PyCharm

- 各自インストール。画面はこんな感じ



Google Colab (Colaboratory)

- Jupyter Notebookをクラウド上で動かす感じ
- 環境構築がほぼ不要(Python, Numpyなどが入っている)
- チーム内での共有が簡単(コードがDriveに保存)
- GPUが無料で使える(PCスペックによらず高速計算)
- 接続できる時間に制限あり、リセットされる
 - 右クリック>アプリで開く>アプリを追加>Colabを検索>インストール
 - 右クリック>その他>Google Colaboratory で新規ファイル作成



Pythonを始める

1. File>New Project>プロジェクト名を入力
2. プロジェクト名を右クリック
3. New>Python File>ファイル名を入力
4. コードを書いて ▶Run

注意

- エンコーディング : UTF-8
- インデント : 4文字分のスペース
- カンマ後ろには半角スペース
- その他、PEP8という規則にしたがう

データ型

Pythonは変数の型を自動で判定

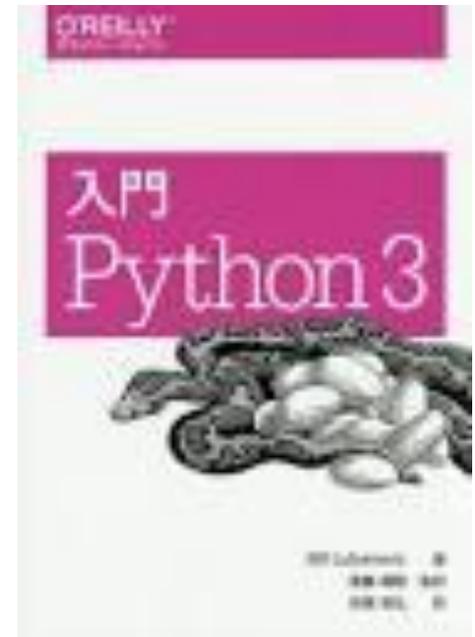
型が異なるもの同士の演算はできないことも

- 整数 `int`
- 長整数 `long`
- 小数 `float`
- 複素数 `complex`
- 真理値 `bool`: `True`, `False`
- 文字列 `str`: `"abcd"` `"10"` (数字も`"`の間に置くと文字列に)
- リスト `list`: `["today", 200, "next", 250]` (リストは変更可能)
- タプル `tuple`: `("today", 200, "next", 250)` (変更不可だが早い)
- 辞書 `dict`: `{"dog":4, "cat":4, "bird":2}` (`"dog"`キーで4が出る)

自学自習

- range()とprint()
- 比較演算子(==, !=, >= など)
- 算術演算(+, *, ** など)
- for文, if文, while文, try文
- データ型の判定
- 関数の定義の仕方

本を1冊読んで、こんなことができる
というのを把握しよう
やってみることが大事！



モジュール、パッケージ、ライブラリ

- モジュール：関数やクラスなどを書いたファイル
- クラス：データと機能を組み合わせる方法を提供
- パッケージ：モジュールをまとめたもの
- ライブラリ：パッケージをまとめたもの

モジュールの使い方

pip install モジュール名 ←Rのinstall.packages("")的なやつ

import モジュール名 as 別名 ←Rのlibrary()的なやつ

例 import pandas as pd / import numpy as np

- Pandas: データ解析系ライブラリ
- NumPy: 配列処理系ライブラリ

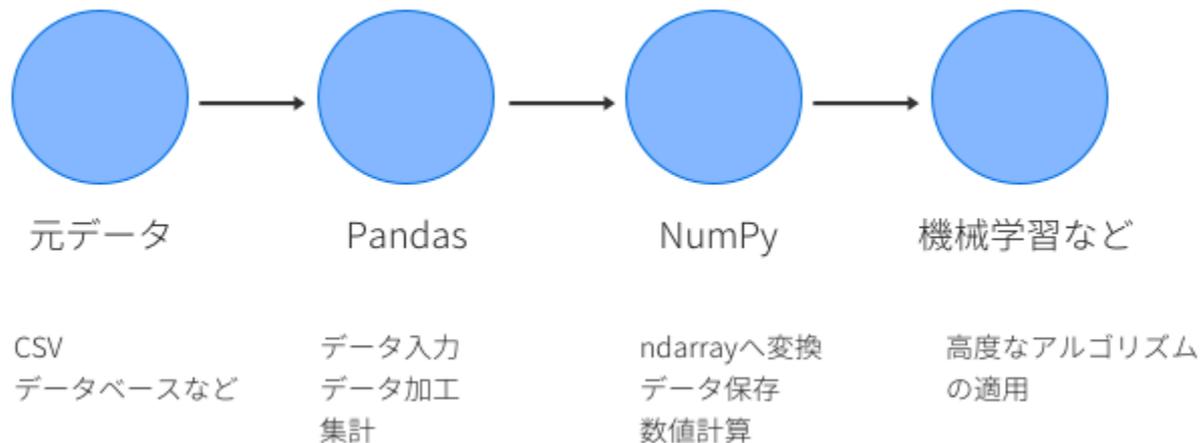
PandasとNumPy

NumPy

- 多次元配列の数値データを扱う
- 計算速度が速い

Pandas

- あらゆるデータを処理する関数が一通りそろろう
- csv読み込み、データ整形、欠損値処理、時系列データの扱い
- Pandasで加工・処理してNumPyに変換し計算を行うのがよい



データの処理

csv読み込み

- `pd.read_csv("sample.csv")`
- `np.loadtxt("sample.csv", delimiter=",", skiprows=1)`

csv書き込み

- `データ名.to_csv("sample.csv", sep=',', header=True)`
- `np.savetxt("sample.csv", データ名, delimiter=",")`

Pandasデータフレーム

- データを行で取り出す：`データ名[始まり行番号:終わり行番号]`
- データを列で取り出す：`データ名["列名"]`
- データを複数列で取り出す：`データ名[['列名1', '列名2']]`
- 条件を満たす行だけ取り出す：
`データ名[データ名["列名"]==10]` など

NaNの処理

- `data4 = data3[data3['A'] == data3['A']]`
でdata3のうち列名AのセルがNaNでない行のみがdata4に入る
- Pythonでは同じものを比べたらTrueが返るが、`NaN==NaN`となるとFalseが返る

```
# dataedit
# パッケージpandasを使うのが便利
import pandas
# 書き込み先ファイル閉じないと回らないので注意
data2 = pandas.read_csv("data2.csv", encoding="utf_8")
# 文字コードのutf-8への変更は、メモ帳で開いて新規保存のときに文字コード指定が簡単
# print(csv)
data3 = data2[data2['A'] != "不明"]
data3 = data3[data3['A'] == data3['A']]
data3 = data3[data3['B'] == data3['B']]
data3 = data3[data3['C'] == data3['C']]
data3 = data3[data3['D'] == data3['D']]
data3 = data3[data3['D'] != "?" ]
print(data3)

# csvファイルとして出力
data3.to_csv("data3.csv")
```

Google Colabを使い倒せ！

- Google Driveをマウント

```
from google.colab import drive  
drive.mount('/content/drive')
```

- Driveのファイルを読み込む

```
root_dir = "/content/drive/My Drive/.../"  
pd.read_csv(root_dir + "sample.csv", sep = ',')
```

- GPUを使う

ランタイム>ランタイムのタイプを変更 で変更可能

NN(ニューラルネットワーク)などはGPUで爆速になる！

- 90分(実際にはもっと短い)の時間制限

回すときは一気に回す

練習

1. I can't speak English. と表示させる
2. 1~100の整数を表示させる
3. 1~100の整数をカンマで区切った1行で表示(1, 2, ..., 100)
4. 1~99の奇数の和
5. フィボナッチ数列を1000まで表示させる
6. 1~100の素数を表示

Python チュートリアル というサイトもおすすめです

解答例

- Google Colabでやってみた