PythonとPyDataの基礎

TFUG Utsunomiya #12

お前誰？

- 櫻岡響
- 電気電子システム工学専攻 修士2年
- 音声研に所属
- 言語処理とかやってます

AGENDA

- **Python基礎**
  - 基本のデータ構造
  - 基本の制御構造
- **PyData**
  - Numpyでベクトル計算
  - 正弦波の作図
  - W杯のデータで遊ぶ

AGENDA

- Python基礎
  - **基本のデータ構造**
  - 基本の制御構造
- PyData
  - Numpyでベクトル計算
  - 正弦波の作図
  - W杯のデータで遊ぶ

変数代入とprint

```
a = 'hello World'
print(a)
```

- jupyter notebookの場合セルの最後がprintされる

```
a = 'hello World'
a
```

Pythonのデータ構造

- 数値
- 文字列
- 真偽値
- リスト
- タプル
- 辞書

数値

- int　　1,　　2,　　3
- float　　0.2,　3.4,　5.6
- 主な演算子
  - +　加算
  - −　減算
  - *　乗算
  - /　除算
  - %　剰余
  - **　累乗
- 13と6で全部やってみましょう

文字列

- 'Hello' or "Hello"
- 主な演算子
  - + 結合

```
'Hello' + ' ' + 'World'
```

論理値

- True, False
- 主な演算子
  - and / or / not

```
True and False
```

リスト 1/3

- 配列のようなもの
- 型を混在させて格納できる

```
a = [ 2, 'foo', 6.4, 'bar', True]
a
```

リスト 2/3

- 値の取り出し方
  - a[1:3]みたいな書き方→スライス

```
a[0]
a[1]
a[-1]
a[1:3]
```

リスト 3/3

- 代入
  - 下を実行後のaを見てみましょう

```
a[1] = 'Hello'
a
```

タプル

- リストとほぼ一緒
- 再代入出来ない

```
a = (2, ‘foo’, 6.4, ‘bar’, True)
print(a)
a[0] = 'Hello'
```

辞書

- 連想配列

```
foo = {'a':100, 'b':200, 'c':300}
foo['b']
```

型の変換

- str、 float、boolなどの関数をつかう

```
float(5)
```

AGENDA

- Python基礎
  - 基本のデータ構造
  - **基本の制御構造**
- PyData
  - Numpyでベクトル計算
  - 正弦波の作図
  - W杯のデータで遊ぶ

Pythonの制御構造

- if
- for
- インデントで階層を表す

if

- 条件分岐

```
score = 50
if score < 60:
   print('成績は不可です')
elif score >=90:
   print('成績は秀です')
else:
   print('合格です')
```

for

- 繰り返し
- range(5)に書き換えてみましょう

```
for i in [0,1,2,3,4]:
   print(i)
```

Let’s FizzBuzz

- **Step1.**
  - 0から100までprintするプログラムを書いてください
- **Step2.**
  - 以下の仕様に変更してください
    - 3の倍数のときはFizz
    - 5の倍数のときはBuzz
    - 15の倍数のときはFizzBuzz

AGENDA

- Python基礎
  - 基本のデータ構造
  - 基本の制御構造
- PyData
  - **Numpyでベクトル計算**
  - 正弦波の作図
  - W杯のデータで遊ぶ

そのまえに

関数の実行と属性呼び出し

- 関数の実行
  - `some_func(arg)`
- 属性の呼び出し
  - `some_object.attr`

import

- `import hoge`
  - グローバルにhogeオブジェクトが定義される
- `from hoge import fuga`
  - hogeモジュールにあるfugaオブジェクトを定義

では行きます

Numpy

- 数値計算用ライブラリ
- ベクトル・行列計算ができる

http://www.numpy.org/

```
import numpy as np
```

Numpy配列の作り方 1/2

- numpy.array関数を使う

```
a = np.array([2,4,6])
b = np.array([
   [1,2,3],
   [4,5,6],
   [7,8,9]
])
```

Numpy配列の作り方 2/2

- numpy.arange
  - range関数のNumpy版
- numpy.identity
  - 単位行列
- numpy.zeros
  - ゼロ行列
- numpy.linspace
  - startからstopまでnum個に分割

Numpy配列の演算子など

- `+, -` 加算、減算
- `*` 要素毎の積
- `b.dot(a)` ドット積
- `a.T` 転置

```
b.dot(a).T
```

Universal functions

- Numpy配列に対して適用出来る関数
  - `numpy.sin`
  - `numpy.cos`
  - `numpy.tan`
  - `numpy.exp`
  - `numpy.log`

練習

- 以下のx, yを作ってみてください
  - x : $-\pi$からπを100個に分割
  - y : xの要素それぞれにsinを適用

AGENDA

- Python基礎
  - 基本のデータ構造
  - 基本の制御構造
- PyData
  - Numpyでベクトル計算
  - **正弦波の作図**
  - W杯のデータで遊ぶ

matplotlib

- 作図をするためのライブラリ

https://matplotlib.org/

```
from matplotlib import pyplot as plt
```

正弦波の作図

- さきほどのx、yを使いましょう

```
fig, ax =  plt.subplots(figsize=(12,9))
ax.plot(x,y)
ax.set_xlabel('$x$')
ax.set_ylabel('$sin(x)$')
ax.set_title('$y = sin(x)$')
```

AGENDA

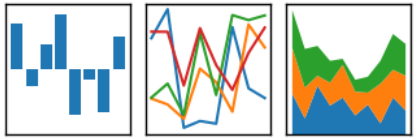
- Python基礎
  - 基本のデータ構造
  - 基本の制御構造
- PyData
  - Numpyでベクトル計算
  - 正弦波の作図
  - **W杯のデータで遊ぶ**

Pandas

- 統計処理のためのライブラリ
- pandas.Series
  - 1次元の名前付きデータ列
- pandas.DataFrame
  - 2次元の分割表

pandas

$y_{it} = \beta' x_{it} + \mu_i + \epsilon_{it}$



https://pandas.pydata.org/

```
import pandas as pd
pd.options.display.max_rows = 10
```

DLお願いします
https://goo.gl/VPjb8K

CSVの読み込み方

- 出典
  - 2018 FIFA World Cup Squads
  - https://www.kaggle.com/cclayford/2018-fifa-world-cup-squads

```
data = pd.read_csv(
   '2018 FIFA World Cup Squads.csv'
)
```

ワールドカップ！
adidas
brazuca

データ抽出 1/2

- 列
  - `data['Team']` -> Series
- 行
  - `data.iloc[123]` -> Series
- 範囲
  - `data.iloc[0:5, 3:6]` -> DataFrame

データ抽出 2/2

- boolのSeriesでの抽出
  - cはなんでしょうか？
  - DataFrame[c]が出来るのがポイント

```
c = (data['Group'] == 'A')
data[c]
```

練習1 /4

- 日本人選手の名前をprintしましょう

練習2 /4

- ゴール数が最多の選手の名前と国をprintしましょう
  - `pandas.Series.idxmax()`

練習3 /4

- W杯登録選手を輩出している上位5クラブを挙げましょう
  - `pandas.Series.value_counts()`

練習4 /4

- 出場試合数(Caps)とゴール数(Goals)を散布図にしましょう
  - `ax.scatter(x,y)`
  - `x,y`は`pandas.Series`でもOK

オマケ

- seaborn
  - matplotlibのラッパー
  - 統計的な作図が簡単に出来る
- Nobebookに入れときました
- matplotlibよりカスタマイズはしにくいので注意