Pythonではじめる機械学習
1.4 必要なライブラリとツール

15T4032G 芝山直希

2018年4月13日

scikit-learn 以外の必要なライブラリ

scikit-learn が依存しているライブラリ

- NumPy
- SciPy

その他のライブラリ

- pandas
- matplotlib

開発環境

- Jupyter Notebook

1.4.1 Jupyter Notebook

- ブラウザ上でコードを実行するインタラクティブな環境
- Python を含む多数のプログラミング言語をサポート
- 探索的なデータ解析に有用なツール
- データサイエンティストに広く利用される

1.4.2 NumPy

Python での科学技術計算において、基本的なライブラリの一つ

- 高レベルの数学関数が含まれる
  - 多次元配列
  - 線形代数
  - フーリエ変換
  - 疑似乱数生成器
- scikit-learn では、NumPy の多次元配列 ndarray[1] を入力時のデータ構造としている
- 同じ配列内に異なる型のデータは混在できない

課題等でも頻出するため、使用例を省略

---

[1]頻出するため、(NumPy) 配列とも表記する。

1.4.3 SciPy

Python で科学技術計算を行うための関数を集めたライブラリ

- 代表的な機能
  - 高度な線形代数ルーチン
  - 数学関数の最適化
  - 特殊数学関数
  - 統計分布

scikit-learn において、アルゴリズムの実装に用いられている

scipy.sparse で表現できる疎行列が非常に有効

疎行列

ほとんどの要素が0である行列。巨大な疎行列を扱う場合、CSR[2]やCOO[3]など、記録方式を工夫しメモリ消費量を大幅に減らす必要がある。

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Figure: 疎行列の例

[2] COOのうち、行の情報を圧縮した方式

[3] 0でない要素に対し、行、列、値の組を記録する方式

scipy.sparseの使用例

前ページの図で示した疎行列を作成している。
この例ではNumPyをnpとしてインポートし、scipy.sparseをインポートしている。

In[4]:

```
data = np.ones(4)
row_indices = np.arange(4)
col_indices = np.arange(4)
eye_coo = sparse.coo_matrix((data, (row_indices, col_indices)))
print("COO representation:\n{}".format(eye_coo))
```

Out[4]:

```
COO representation:  COO表現
  (0, 0)    1.0
  (1, 1)    1.0
  (2, 2)    1.0
  (3, 3)    1.0
```

1.4.4 matplotlib

広く利用されている Python の科学技術計算向けグラフ描画ライブラリ

- 高品質な可視化を実現する関数群を含む
- 折れ線グラフ、ヒストグラム、散布図など、様々な形式をサポート

データ・解析結果の様々な視点からの可視化は、重要な洞察を得る上で不可欠

Jupyter Notebook 上では、%matplotlib inline[4] コマンドで、出力結果をブラウザ上に表示できる

[4] %matplotlib notebook でもよい

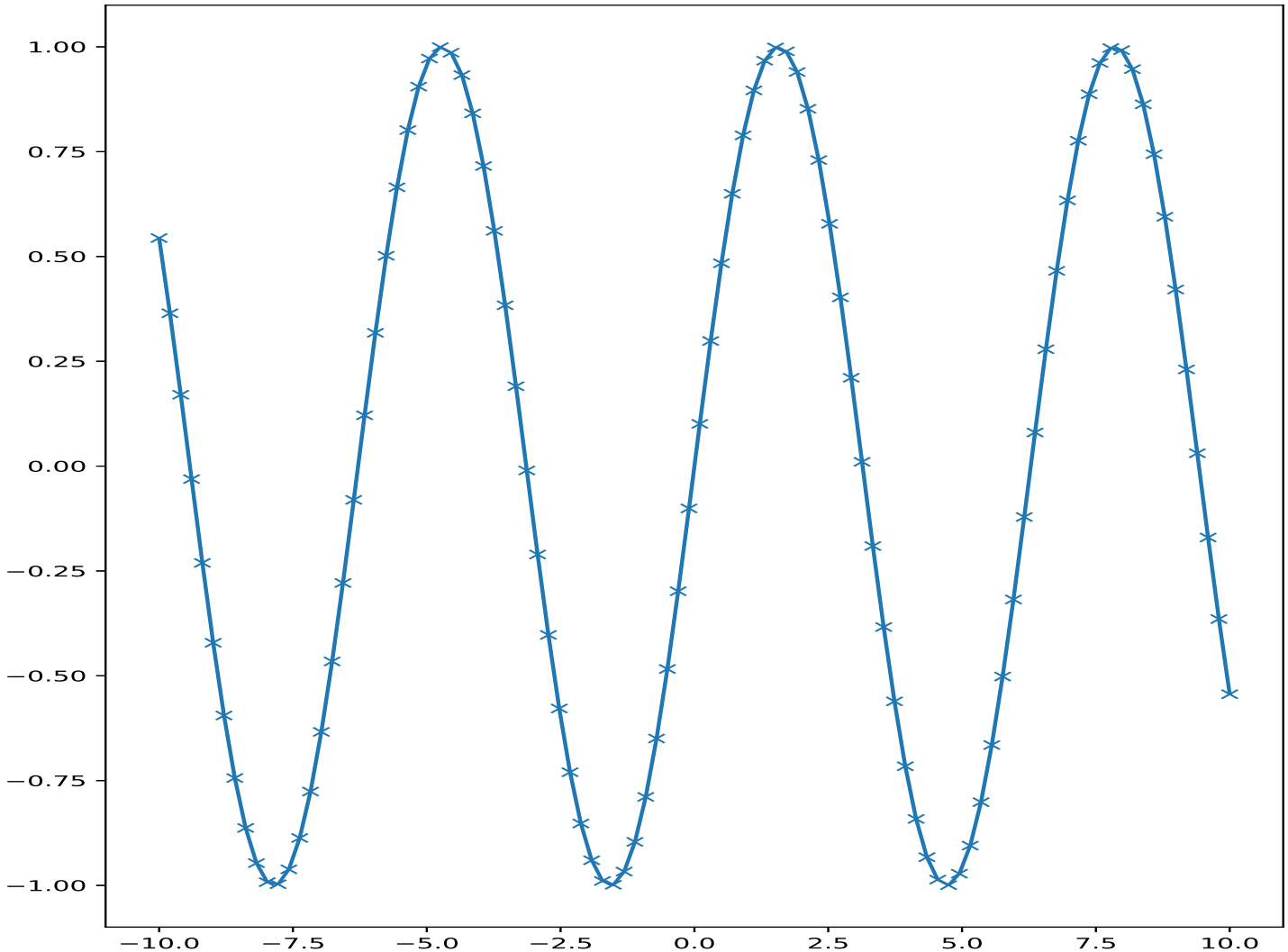
matplotlibの使用例

NumPyをnpとしてインポートしている。この例(とほぼ同等)のコードを実行すると、次ページの画像が得られる。

In[5]:

```
%matplotlib inline
import matplotlib.pyplot as plt

# -10から10までを100ステップに区切った列を配列として生成
x = np.linspace(-10, 10, 100)
# サイン関数を用いて2つ目の配列を生成
y = np.sin(x)
# plot関数は、一方の配列に対して他方の配列をプロットする
plt.plot(x, y, marker="x")
```


1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
−10.0
−7.5
−5.0
−2.5
0.0
2.5
5.0
7.5
10.0

1.4.5 pandas

データ変換・解析のためのライブラリ

- DataFrame という、Excel のスプレッドシートのようなデータ構造を持つ
- DataFrame に対して、SQL のような問い合わせや join などの操作を行える
- DataFrame 内では異なる列に異なる型を設定できる
- 様々なファイルフォーマット及びデータベースからデータを読み込む機能を持つ
  - SQL データベース、Excel ファイル、CSV ファイル等に対応

pandasの使用例

端末で実行する場合、IPythonからdisplayをインポートする必要はない。

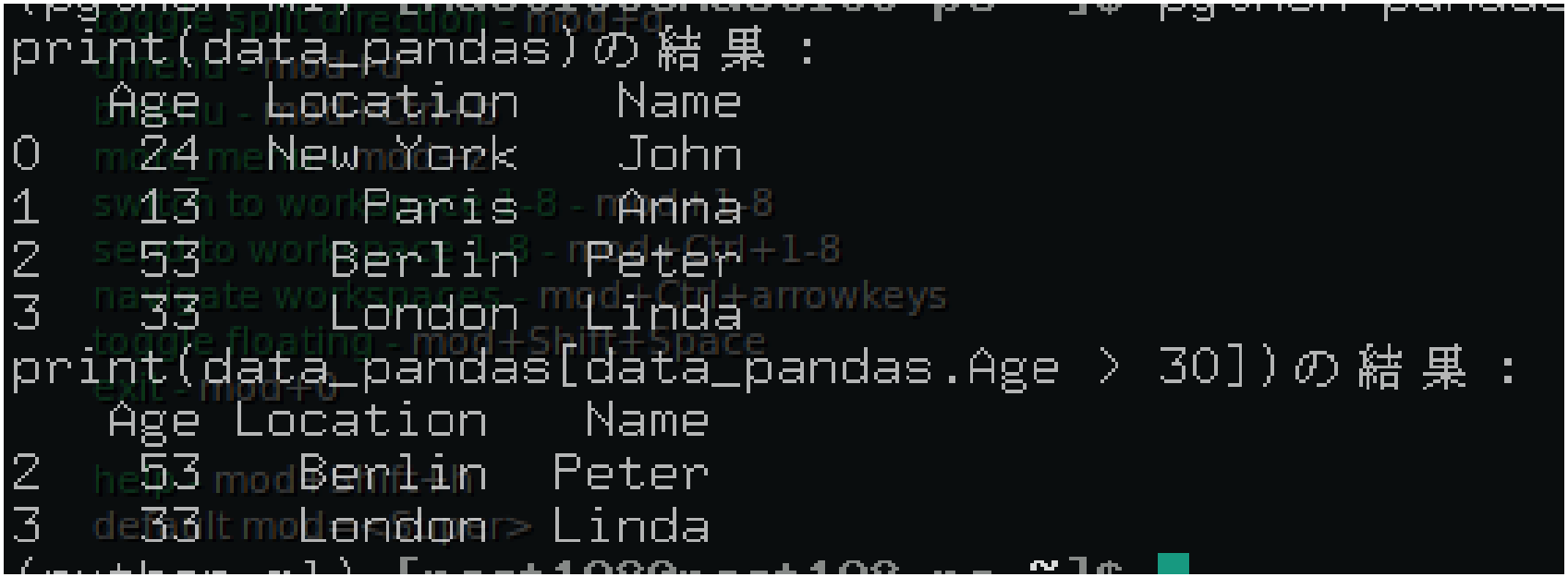
In[6]:

```
import pandas as pd
from IPython import display

# 人を表す簡単なデータセットを作る
data = {'Name': ["John", "Anna", "Peter", "Linda"],
        'Location' : ["New York", "Paris", "Berlin", "London"],
        'Age' : [24, 13, 53, 33]
       }

data_pandas = pd.DataFrame(data)
```

前ページの data_pandas を print すると次の画像[5]上段のようになる。また、画像下段のように問い合わせを行い、該当する要素のみ取り出すことができる。



[5]背景が若干透過しているのは、端末エミュレータの設定である。

1.4.6 mglearn

可読性向上のためのライブラリ 必須ではない

- グラフの描画やデータの読み込み等のコードで、本書が読みにくくならないようにするべく作成された
- GitHub[6] 上で公開されている NoteBook では、既に設置されている
- 必要であれば pip コマンドで導入可能
  - $ pip install mglearn

[6] https://github.com/amueller/introduction_to_ml_with_python

以降前提となるインポート文

```
imoprt numpy as np
import matplotlib.pyplot as plt
import pandas as pd
import mglearn
from IPython import display
```

- Jupyter Notebook 上で実行する場合、マジックコマンド%matplotlib notebook (%matplotlib inline でもよい) を実行しておく
- 他の環境では、図の表示のために plt.show() を呼び出す必要がある