いま再びのscikit-learn

みんなのPython勉強会 # 75

11/9, 2021

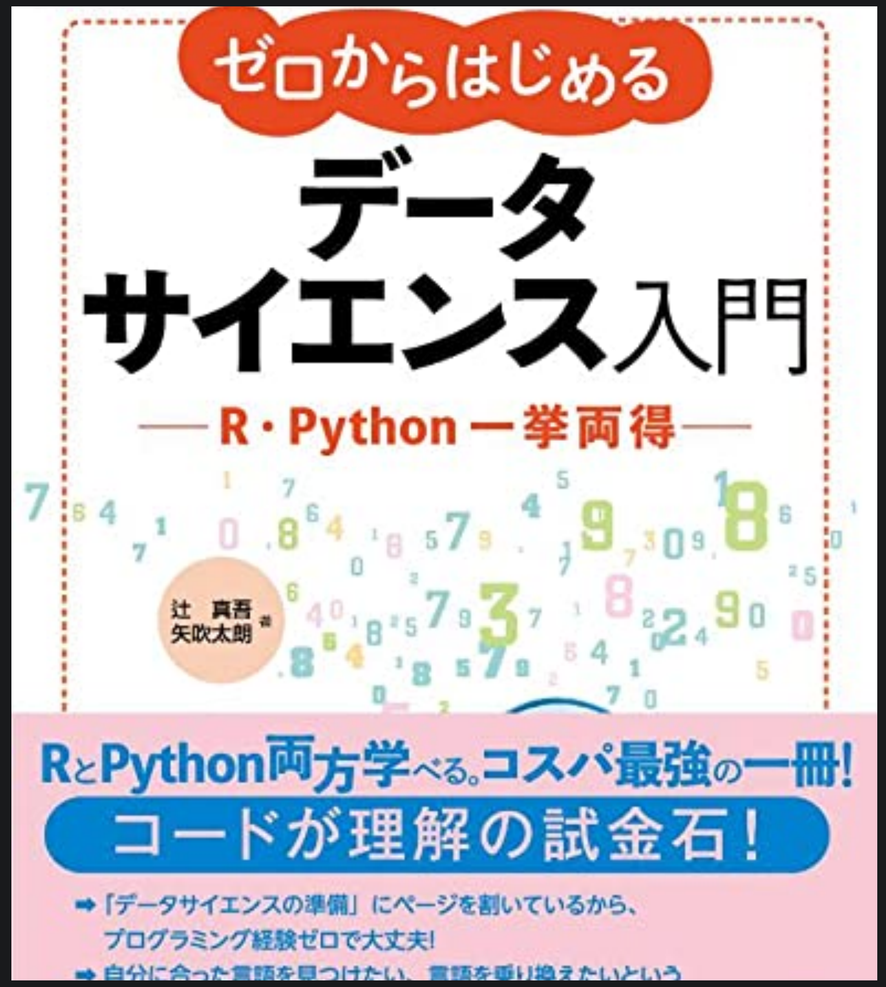
辻真吾（@tsjshg）

おまえ、誰よ？

- 辻真吾（つじしんご）www.tsjshg.info
- 大学の研究所に勤めています（バイオ→エネルギー）
  - 応用データサイエンスという分野を作りたい
- 小学生のころ（MSX2）からコンピュータが好き
  - BASIC, LOGO → C/C++ → Java → Python
- 11/2〜4まで香川県に旅行してきました

www.hanmoto.com/bd/isbn/9784065132326

www.amazon.co.jp/dp/4065132320



実践Data Scienceシリーズ

ゼロからはじめるデータサイエンス入門

– R・Python一挙両得 –

12月9日(木)講談社から発売予定・Amazonで絶賛予約受付中

コンピュータの基礎、シェルの使い方、Docker、簡単な統計まで解説したあと深層学習を含むデータサイエンス全般の話題を網羅

ほぼすべてのコードをRとPythonで丁寧に記述

辻真吾・矢吹太朗 著

共著者とは小学校1年生（足立区立千寿第5小）からの幼馴染み

scikit-learnのはなし

- scikit-learnの基本機能と最近のバージョンのハイライトを紹介
- 機械学習アルゴリズムがまったく分からない方にも伝わるように心がけます（が、少し知っている方が分かりやすいです）
- scikit-learnを毎日ゴリゴリに使っている方は退屈だと思うので、zoomのチャットでマニア情報をお待ちしております
- 機械学習簡単そう自分もやってみよう！と思っていただけると嬉しい

scikit-learnとは？

- Pythonで機械学習アルゴリズムを実行するためのライブラリ
- 次元削減、教師あり学習、教師なし学習、各種前処理やパラメータチューニングまで（ほとんど）なんでもできる
- はじまりは2007年Google Summer of codeでのDavid Cournapeauさん
- 2021年9月にバージョンがそれまでの0.24系から1.0系へ

みんな大好きアヤメのデータ

```
import statsmodels.api as sm
iris = sm.datasets.get_rdataset('iris', 'datasets').data
iris
```

	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
0	5.1	3.5	1.4	0.2	setosa
1	4.9	3.0	1.4	0.2	setosa
2	4.7	3.2	1.3	0.2	setosa
3	4.6	3.1	1.5	0.2	setosa
4	5.0	3.6	1.4	0.2	setosa
...	...	...	...	...	...
145	6.7	3.0	5.2	2.3	virginica
146	6.3	2.5	5.0	1.9	virginica
147	6.5	3.0	5.2	2.0	virginica
148	6.2	3.4	5.4	2.3	virginica
149	5.9	3.0	5.1	1.8	virginica

150 rows × 5 columns

setosa

versicolor

virginica

https://en.wikipedia.org/wiki/Iris_flower_data_set

統一的なインターフェイス

データを訓練データとテストデータに分けたあと2つの予測モデルを作る

```
from sklearn.model_selection import train_test_split

# 説明変数Xと目的変数yに分離
X = iris.iloc[:, :4]
y = iris['Species']
# 訓練データとテストデータに分ける
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=1 / 3)
```

```
from sklearn.svm import SVC
# SVMのインスタンスをデフォルトのパラメータで生成
clf = SVC()
# 訓練
clf.fit(X_train, y_train)
# 予測
clf.predict(X_test)
```

```
from sklearn.ensemble import RandomForestClassifier
# Random Forestsのインスタンスをデフォルトのパラメータで生成
clf = RandomForestClassifier()
# 訓練
clf.fit(X_train, y_train)
# 予測
clf.predict(X_test)
```

次元削減も簡単

説明変数の数を削減

高次元空間で近くにあるモノはより近く、遠くにあるモノはより遠くに配置

アヤメの4次元データをt-SNEを使って2次元に縮約

```
from sklearn.manifold import TSNE

X_tsne = TSNE(n_components=2).fit_transform(X)
```

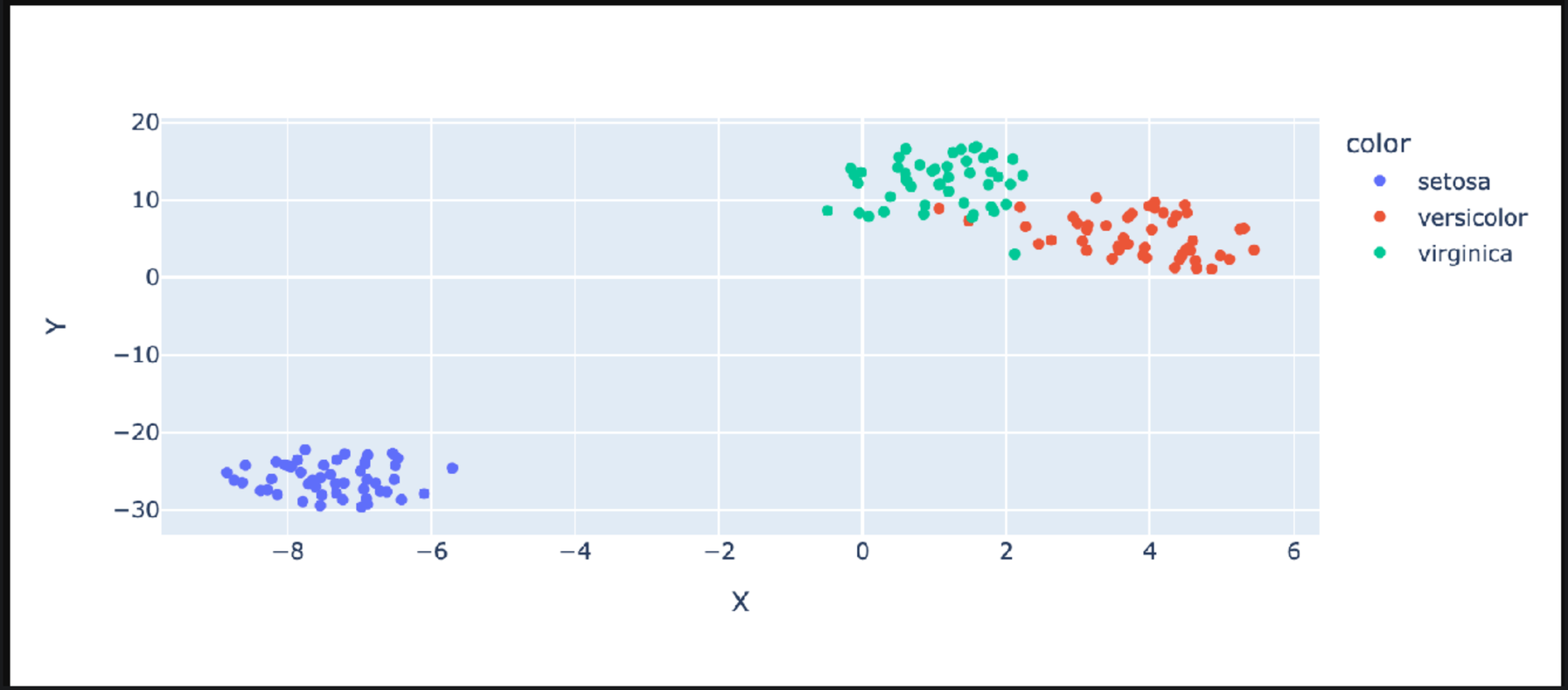
PCA（主成分分析）を使ったやり方

```
from sklearn.decomposition import PCA

X_pca = PCA(n_components=2).fit_transform(X)
```

Plotlyを使った可視化

```
import plotly.express as px
import pandas as pd

tsne_df = pd.DataFrame(X_tsne, columns=['X', 'Y'])
px.scatter(tsne_df, x='X', y='Y', color=y)
```

DataFrameを作る



設定はpip install plotlyのあとシェルからjupyter labextension install jupyterlab-plotly

クラスタリングはPyCaretが便利かも

PythonにおけるAutoMLの1つ　みんなのPython勉強会#73で紹介しました

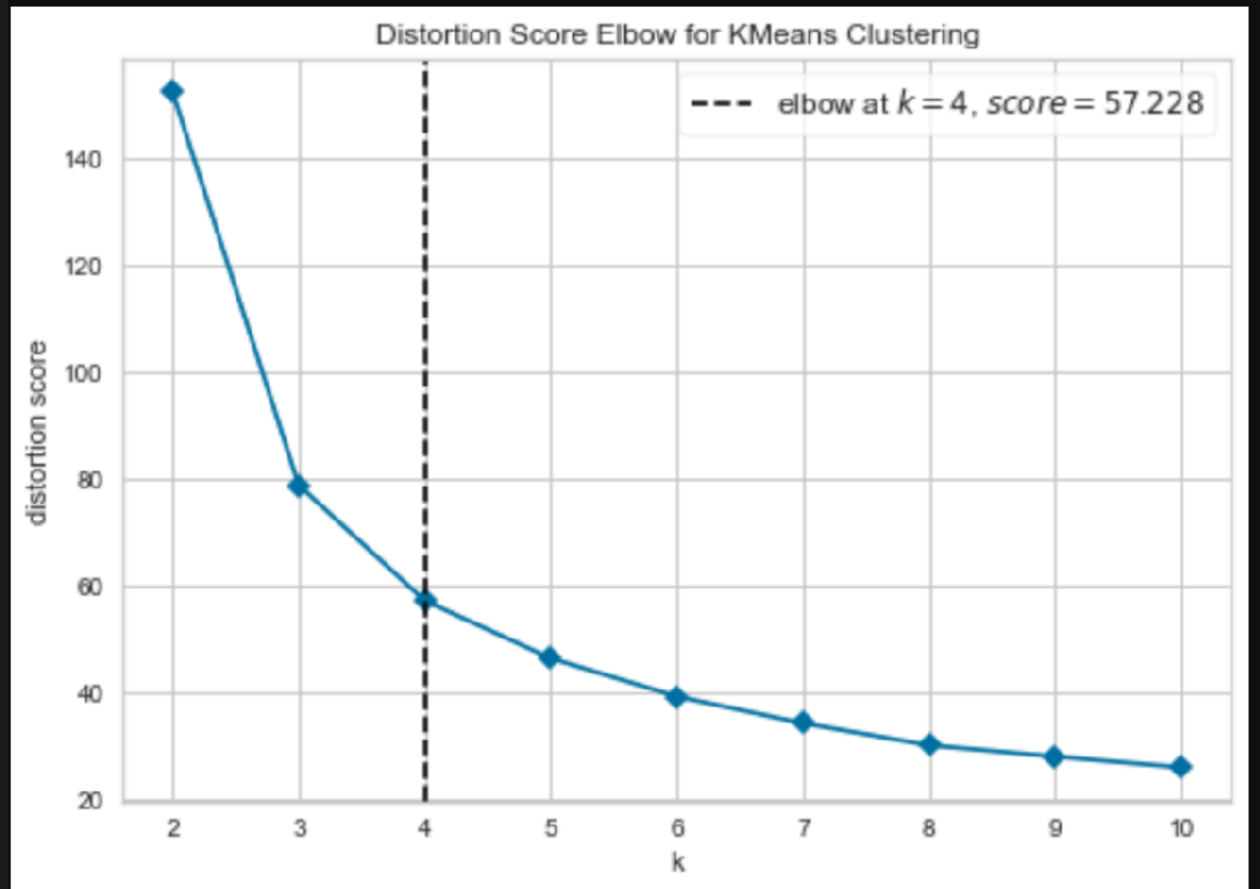
```
from pycaret.clustering import *

exp_name = setup(data=iris, ignore_features=['Species'])
```

↑データをセットしてクラスタリングの環境をセットアップ

K-meansクラスタリングはクラスター数の指定が必要
いくつか妥当か検討する1つの方法にエルボープロットがある→

現行のPyCaret 2.3.4は、scikit-learn 0.23.2を利用

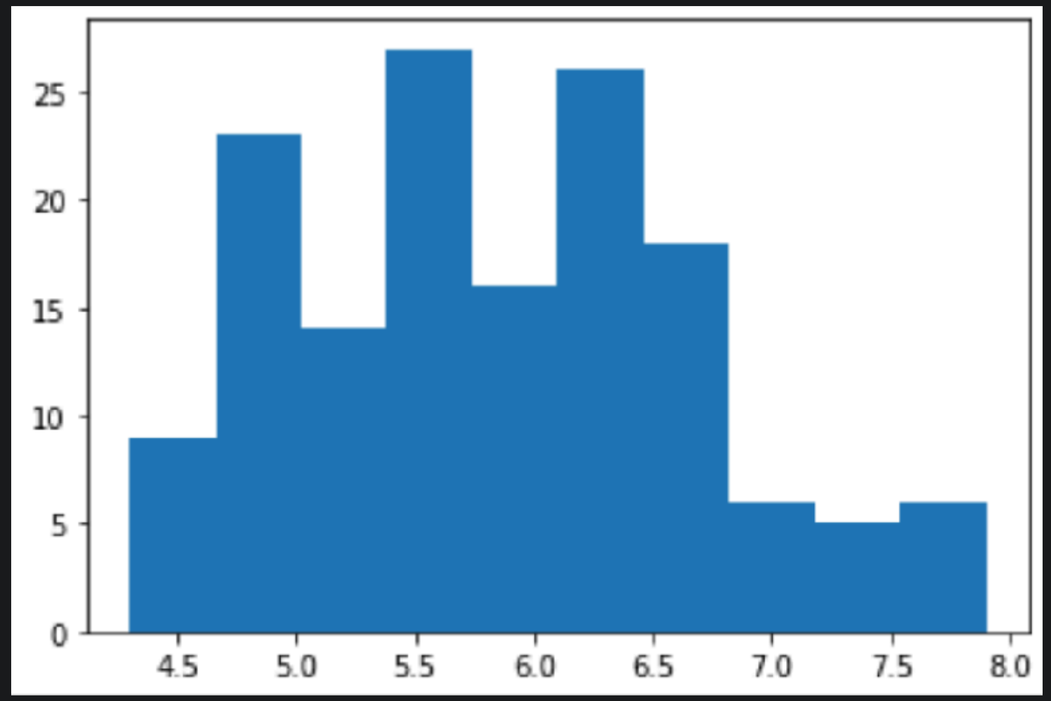
```
kmeans = create_model('kmeans')
plot_model(kmeans, plot='elbow')
```



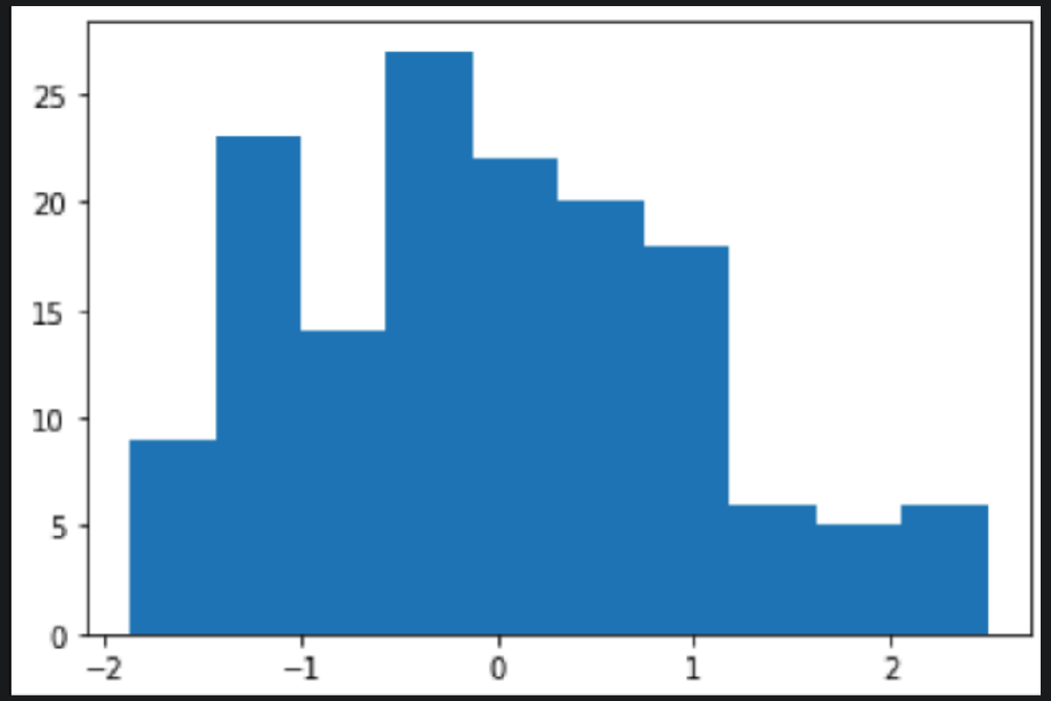
データの標準化

それぞれの説明変数について、平均値を引いて標準偏差で割る

Before



After



ヒストグラムの横軸が変わっている

前処理も簡単にできる（のだけれど・・・）

```
from sklearn.preprocessing import StandardScaler

ss = StandardScaler()
std_X = ss.fit_transform(iris.iloc[:, :4])
std_X
```

入力はpandasのDataFrame

```
array([[-9.00681170e-01,  1.01900435e+00, -1.34022653e+00,
        -1.31544430e+00],
       [-1.14301691e+00, -1.31979479e-01, -1.34022653e+00,
        -1.31544430e+00],
       [-1.38535265e+00,  3.28414053e-01, -1.39706395e+00,
        -1.31544430e+00],
       [-1.50652052e+00,  9.82172869e-02, -1.28338910e+00,
        -1.31544430e+00],
       [-1.02184904e+00,  1.24920112e+00, -1.34022653e+00,
```

出力がNumPyのarray

```
# 1.0の地味な新機能
ss.feature_names_in_
```

```
array(['Sepal.Length', 'Sepal.Width', 'Petal.Length', 'Petal.Width'],
      dtype=object)
```

pipelineが便利

```
from sklearn.pipeline import make_pipeline
# 標準化の後、SVMを実行するパイプライン
my_pl = make_pipeline(StandardScaler(), SVC())
my_pl.fit(X_train, y_train)
```

```
Pipeline(steps=[('standardscaler', StandardScaler()), ('svc', SVC())])
```

```
my_pl.predict(X_test)
```

データの標準化と予測を全部やってくれる

```
from sklearn import set_config
# Jpyter内で図を出す設定
set_config(display="diagram")
my_pl
```

こちらは、0.23から使えます

Pythonの関数

引数は位置でもキーワードでも指定できる

```python
def my_div(a, b):
    # aをbで割った余りを返す
    return a % b
```

```python
assert my_div(10, 4) == 2
assert my_div(a=10, b=4) == 2
assert my_div(b=4, a=10) == 2
assert my_div(10, b=4) == 2
```

```python
# 1度キーワードで指定したらその後位置には戻れない
my_div(a=10, 4)
```

```
  File "/var/folders/gg/y144ms9d1l51khjw847lxkb80000gn/T/ipykernel_16024/642405458.py", line 2
    my_div(a=10, 4)
                 ^
SyntaxError: positional argument follows keyword argument
```

/より前は位置のみ引数（Python3.8〜）

```python
def my_div(a, b, /):
    # aをbで割った余りを返す
    return a % b
```

関数の定義を変更

```python
assert my_div(10, 4) == 2
assert my_div(a=10, b=4) == 2
assert my_div(b=4, a=10) == 2
assert my_div(10, b=4) == 2
```

この呼び出しだけ許される

下の3つは全エラー

*より後ろはキーワード引数

```python
def my_div(a, *, b):
    # aをbで割った余りを返す
    return a % b
```

関数の定義を変更

```python
assert my_div(10, 4) == 2
assert my_div(a=10, b=4) == 2
assert my_div(b=4, a=10) == 2
assert my_div(10, b=4) == 2
```

これはエラー　bはキーワードのみ

下の3つはOK！

aはどっちでもいい

1.0から引数の指定が厳しくなった

ここ

sklearn.ensemble.RandomForestClassifier

class sklearn.ensemble.RandomForestClassifier(*n_estimators=100*, ***, *criterion='gini'*, *max_depth=None*, *min_samples_split=2*, *min_samples_leaf=1*, *min_weight_fraction_leaf=0.0*, *max_features='auto'*, *max_leaf_nodes=None*, *min_impurity_decrease=0.0*, *bootstrap=True*, *oob_score=False*, *n_jobs=None*, *random_state=None*, *verbose=0*, *warm_start=False*, *class_weight=None*, *ccp_alpha=0.0*, *max_samples=None*) [source]

```
my_forest = RandomForestClassifier(100, criterion='gini')
```

```
my_forest = RandomForestClassifier(100, 'gini')
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
/var/folders/gg/y144ms9d1l51khjw847lxkb80000gn/T/ipykernel_16024/2048377832.py in <module>
----> 1 my_forest = RandomForestClassifier(100, 'gini')

TypeError: __init__() takes from 1 to 2 positional arguments but 3 were given
```

https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html?highlight=random%20forest#sklearn.ensemble.RandomForestClassifier

0.24で実行した場合

```
from sklearn.ensemble import RandomForestClassifier

my_forest = RandomForestClassifier(100, 'gini')
```

```
/Library/Frameworks/Python.framework/Versions/3.9/lib/python3.9/site-packages/sklearn/utils/validati
on.py:70: FutureWarning: Pass criterion=gini as keyword args. From version 1.0 (renaming of 0.25) pa
ssing these as positional arguments will result in an error
  warnings.warn(f"Pass {args_msg} as keyword args. From version "
```

FutureWarning

（いまは許してあげるけど、0.25改め1.0からエラーになるから気を付けてね）

まとめ

- scikit-learnはPythonの機械学習ライブラリ
- さまざまなアルゴリズムに対応していて、統一的なインターフェイスで利用可能
- クラスタリングは、PyCaretも便利（バージョン注意）
- キーワード引数のみによる指定は、1.0の大きな変更点
- 既存アルゴリズムの改善などもあり、ますます頼れるライブラリに

ご清聴ありがとうございました