

머신러닝 라이브러리 기초와 실습 2

CONTENTS

01

scikit-learn (사이킷런)

01-1 사이킷런이란?

01-2 데이터셋 (dataset)

01-3 전처리 (preprocessing)



machine learning in Python

01 scikit-learn (사이킷런)

사이킷런(scikit-learn)이란?

'머신러닝 이해하기 - 이론과 실습' 에서 scikit-learn를 사용했습니다.

```
[ ] # sklearn의 tree에서 decision tree classifier 라이브러리를 불러옵니다  
    from sklearn.tree import DecisionTreeClassifier
```




사이킷런(scikit-learn)이란?

scikit-learn.org

- Classification(분류)
- Regression(회귀)
- Clustering(군집화)
- Dimensionality reduction(차원 감소)
- Model selection(모델 선택) - ex: 학습용 데이터와 테스트 데이터로 분리
- Preprocessing(전처리)

J.P.Morgan, Spotify 등의 기업들이 사용

사이킷런(scikit-learn)이란?

scikit-learn 특징

- 다양한 머신러닝 알고리즘을 구현한 파이썬 라이브러리
- 심플하고 일관성 있는 **API**, 유용한 온라인 문서, 풍부한 예제
- 머신러닝을 위한 쉽고 효율적인 개발 라이브러리 제공
- 다양한 머신러닝 관련 알고리즘과 개발을 위한 프레임워크와 **API** 제공
- 많은 사람들이 사용하며 다양한 환경에서 검증된 라이브러리

모듈

설명

sklearn.datasets**내장된 예제 데이터 세트**

sklearn.preprocessing

다양한 데이터 전처리 기능 제공 (변환, 정규화, 스케일링 등)

sklearn.feature_selection

특징(feature)를 선택할 수 있는 기능 제공

sklearn.feature_extraction

특징(feature) 추출에 사용

sklearn.decomposition

차원 축소 관련 알고리즘 지원 (PCA, NMF, Truncated SVD 등)

sklearn.model_selection

교차 검증을 위해 데이터를 학습/테스트용으로 분리, 최적 파라미터를 추출하는 API 제공 (GridSearch 등)

sklearn.metrics

분류, 회귀, 클러스터링, Pairwise에 대한 다양한 성능 측정 방법 제공 (Accuracy, Precision, Recall, ROC-AUC, RMSE 등)

sklearn.pipeline

특징 처리 등의 변환과 ML 알고리즘 학습, 예측 등을 묶어서 실행할 수 있는 유틸리티 제공

sklearn.linear_model

선형 회귀, 릿지(Ridge), 라쏘(Lasso), 로지스틱 회귀 등 회귀 관련 알고리즘과 SGD(Stochastic Gradient Descent) 알고리즘 제공

sklearn.svm

서포트 벡터 머신 알고리즘 제공

sklearn.neighbors

최근접 이웃 알고리즘 제공 (k-NN 등)

sklearn.naive_bayes

나이브 베이즈 알고리즘 제공 (가우시안 NB, 다항 분포 NB 등)

sklearn.tree**의사 결정 트리 알고리즘 제공**

sklearn.ensemble

앙상블 알고리즘 제공 (Random Forest, AdaBoost, GradientBoost 등)

sklearn.cluster

비지도 클러스터링 알고리즘 제공 (k-Means, 계층형 클러스터링, DBSCAN 등)

사이킷런(scikit-learn)이란?

API 사용 방법

API란? API는 두 소프트웨어 사이의 (가상적인) 계약

(만약 사용을 하는 측의 소프트웨어가 미리 정해진 형식으로 인풋을 제공하면 인풋을 받는 소프트웨어의 기능을 확장한 것처럼 출력을 사용하는 측의 소프트웨어에게 돌려줍니다.)

- 지도 학습: 대체로 **predict()** 메서드를 사용해 알려지지 않은 데이터에 대한 레이블 예측
- 비지도 학습: 대체로 **transform()**이나 **predict()** 메서드를 사용해 데이터의 속성을 변환하거나 추론

데이터셋(Datasets) 얻기

머신러닝을 시작할 때, 간단하게 데이터셋을 얻어서 알고리즘을 테스트해 보는 것이 머신러닝을 이해하는데 있어 매우 유용합니다. 간단한 데이터셋으로 원리를 이해한 후, 실제 생활에서 얻을 수 있는 더 큰 데이터셋을 가지고 작업하는 것이 좋습니다.

우선 머신러닝을 연습하기 위해, 간단한 데이터셋을 얻을 수 있는 곳은 다음과 같습니다.

- 사이킷런의 빌트인 데이터셋
- 캐글(Kaggle) 데이터셋
- UCI(캘리포니아 대학, 알바인) 머신러닝 저장소

데이터셋(Datasets) 얻기

캐글(Kaggle) 데이터셋

<https://www.kaggle.com/datasets>

kaggle

The screenshot displays the Kaggle website's 'Datasets' page. The left sidebar contains navigation options: Home, Compete, Data (highlighted), Notebooks, Communities, Courses, and More. The main header includes a search bar, 'Sign In', and 'Register' buttons. The 'Datasets' section features a sub-header 'Engage With Dataset Tasks' and a description: 'You can now actively engage with datasets with thousands of tasks! Help the community by creating and solving Tasks on datasets!'. Below this are buttons for 'Tackle a new task' and 'See Details'. A search bar indicates 'Search 66,709 datasets'. A list of public datasets is shown, including 'World Health 2020', 'Pfizer Vaccine Tweets', and 'Android smartphones high accuracy GNSS datasets'. On the right, the 'Open Tasks' section lists challenges such as 'Exploratory & Statistical Analysis', 'Analysis', 'Fix latitude and longitude data', and 'Train a model to check if a CXR ...'.

데이터셋(Datasets) 얻기

UCI(캘리포니아 대학, 얼바인) 머신러닝
저장소

<https://archive.ics.uci.edu/ml/index.php>

UCI Machine Learning Repository
Center for Machine Learning and Intelligent Systems

[About](#) [Citation Policy](#) [Donate a Data Set](#) [Contact](#)

Search: [See](#)

☒ Repository ☐ Web [View ALL Data Sets](#)

Welcome to the UC Irvine Machine Learning Repository!

We currently maintain 559 data sets as a service to the machine learning community. You may [view all data sets](#) through our searchable interface. For a general overview of the Repository, please visit our [About page](#). For information about citing data sets in publications, please read our [citation policy](#). If you wish to donate a data set, please consult our [donation policy](#). For any other questions, feel free to [contact the Repository librarians](#).

Supported By: In Collaboration With:

Latest News:

- 09-24-2018: Welcome to the new Repository admins Dheeru Dua and Efi Karra Taniskidou!
- 04-04-2013: Welcome to the new Repository admins Kevin Bache and Moshe Lichman!
- 03-01-2010: Note from donor regarding Netflix data
- 10-16-2009: Two new data sets have been added.
- 09-14-2009: Several data sets have been added.
- 03-24-2008: New data sets have been added!
- 06-25-2007: Two new data sets have been added: UJI Pen Characters, MAGIC Gamma Telescope

Featured Data Set: Hepatitis

Task: Classification
Data Type: Multivariate
Attributes: 19
Instances: 155

Newest Data Sets:

- 10-03-2020: Codon usage
- 09-03-2020: Intelligent Media Accelerometer and Gyroscope (IM-AccGyro) Dataset
- 07-22-2020: Facebook Large Page-Page Network
- 07-17-2020: Amphibians
- 07-12-2020: Early stage diabetes risk prediction dataset
- 06-28-2020: Taiwanese Bankruptcy Prediction
- 06-20-2020: South German Credit (UPDATE)

Most Popular Data Sets (hits since 2007):

- 3748368: Iris
- 2033228: Adult
- 1572692: Wine
- 1406266: Heart Disease
- 1406166: Breast Cancer Wisconsin (Diagnostic)
- 1401257: Wine Quality
- 1365943: Bank Marketing

같이하기

데이터셋(Datasets) 사용하기

사이킷런의 샘플 데이터셋 구조

- 일반적으로 딕셔너리 형태로 구성
- **data**: 특징 데이터 세트
- **target**: 분류용은 레이블 값, 회귀용은 숫자 결과값 데이터
- **target_names**: 개별 레이블의 이름 (분류용)
- **feature_names**: 특징 이름
- **DESCR**: 데이터 세트에 대한 설명과 각 특징 설명

같이하기

데이터셋(Datasets) 사용하기

사이킷런의 샘플 데이터셋

샘플 데이터셋을 로드하기 위해, 데이터셋 모듈 `import`. 예제는 Iris 데이터셋

```
▶ from sklearn import datasets  
iris = datasets.load_iris()  
...
```

아이리스 꽃 데이터셋 또는 피셔 아이리스 데이터셋은 영국의 통계 학자이자 생물학자인 로널드 피셔 (Ronald Fisher)가 소개한 다변수 데이터셋입니다. 데이터셋은 3종의 아이리스(Iris)로 된 50개 샘플로 구성되어 있습니다. 각 샘플로부터 4개의 피쳐(features: 피쳐를 우리말로 변수 또는 요인이라고 표현하기도 함)를 측정할 수 있습니다: 꽃받침과 꽃잎의 길이와 너비입니다. 이러한 4가지 피쳐(features)의 결합을 바탕으로 피셔는 종을 서로 구분할 수 있는 선형 판별 모델을 개발했습니다.

```
...
```


같이하기

데이터셋(Datasets) 사용하기

iris의 description(내용)를 출력해봅니다.



```
print(iris.DESCR)
```

```
**Data Set Characteristics:**
```

```
:Number of Instances: 150 (50 in each of three classes)
```

```
:Number of Attributes: 4 numeric, predictive attributes and the class
```

```
:Attribute Information:
```

- sepal length in cm
- sepal width in cm
- petal length in cm
- petal width in cm
- class:
 - Iris-Setosa
 - Iris-Versicolour
 - Iris-Virginica

데이터셋(Datasets) 사용하기

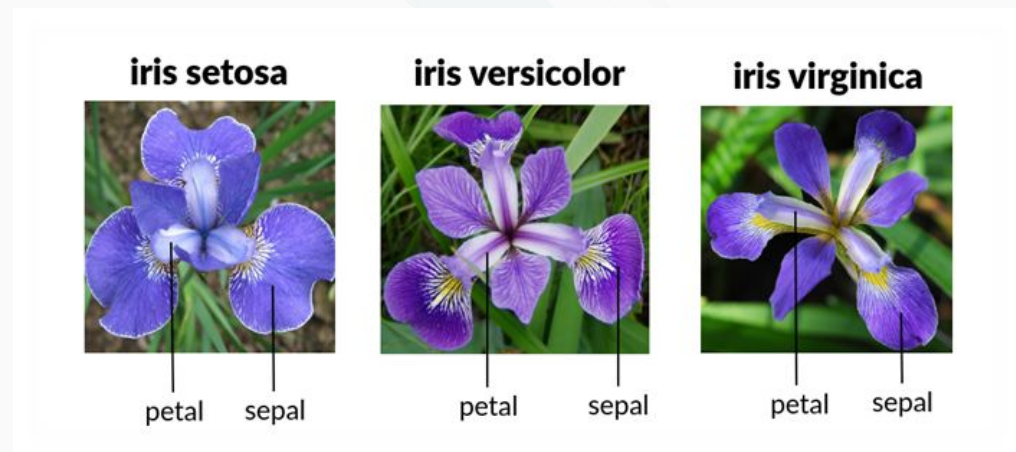
iris의 description(내용)

Number of Instances: 150 (50 in each of three **classes**) 총 150에 각 클래스당 50개씩

Number of Attributes: 4 numeric, predictive attributes and the class 4 개의 숫자, 예측 속성 및 클래스

Attribute Information: 속성 정보

- sepal length in cm 꽃받침 길이 (cm)
- sepal width in cm 꽃받침 너비 (cm)
- petal length in cm 꽃잎 길이 (cm)
- petal width in cm 꽃잎 너비 (cm)
- class:
 - Iris-Setosa (세토사)
 - Iris-Versicolour (버시 컬러)
 - Iris-Virginica(버지니 카)



데이터셋(Datasets) 사용하기

iris의 data



```
print(iris.data)
```



```
[[5.1 3.5 1.4 0.2]  
 [4.9 3.  1.4 0.2]  
 [4.7 3.2 1.3 0.2]  
 [4.6 3.1 1.5 0.2]  
 [5.  3.6 1.4 0.2]  
 [5.4 3.9 1.7 0.4]  
 [4.6 3.4 1.4 0.3]  
 [5.  3.4 1.5 0.2]]
```

데이터셋(Datasets) 사용하기

iris의 속성 혹은 피쳐 이름 - `.feature_names`



```
print(iris.feature_names)
```

```
['sepal length (cm)', 'sepal width (cm)', 'petal length (cm)', 'petal width (cm)']
```

데이터셋(Datasets) 사용하기

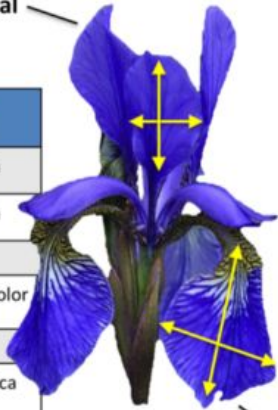
iris의 data

Samples
(instances, observations)

	Sepal length	Sepal width	Petal length	Petal width	Class label
1	5.1	3.5	1.4	0.2	Setosa
2	4.9	3.0	1.4	0.2	Setosa
...					
50	6.4	3.5	4.5	1.2	Versicolor
...					
150	5.9	3.0	5.0	1.8	Virginica

Features
(attributes, measurements, dimensions)

Class labels
(targets)



데이터셋(Datasets) 사용하기

```
iris_label - .target
```

여기서 0은 'setosa'를, 1은 'versicolor'를 2는 'virginica'를 나타냄

```
print(iris.target)
print(iris.target_names)
```

```
[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0  
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0  
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1  
2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2  
2 2]
```

```
['setosa' 'versicolor' 'virginica']
```

(사이킷런의 모든 샘플 데이터가 feature_names, target_names 속성을 지원하는 것은 아닙니다)

데이터셋(Datasets) 사용하기

데이터셋이 눈에 확 잘 보이지 않고, 이대로 사용하면 불편. pandas의 dataframe 사용



```
import pandas as pd  
df = pd.DataFrame(iris.data)  
df.head()
```

	0	1	2	3
0	5.1	3.5	1.4	0.2
1	4.9	3.0	1.4	0.2
2	4.7	3.2	1.3	0.2
3	4.6	3.1	1.5	0.2
4	5.0	3.6	1.4	0.2

데이터셋(Datasets) 사용하기

분류 또는 회귀용 데이터 세트

API	설명
<code>datasets.load_boston()</code>	미국 보스턴의 집에 대한 특징과 가격 데이터 (회귀용)
<code>datasets.load_breast_cancer()</code>	위스콘신 유방암 특징들과 악성/양성 레이블 데이터 (분류용)
<code>datasets.load_diabetes()</code>	당뇨 데이터 (회귀용)
<code>datasets.load_digits()</code>	0에서 9까지 숫자 이미지 픽셀 데이터 (분류용)
<code>datasets.load_iris()</code>	붓꽃에 대한 특징을 가진 데이터 (분류용)

실습

데이터셋(Datasets) 불러오고, 사용하기

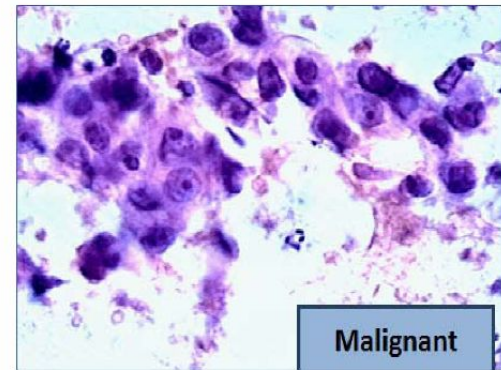
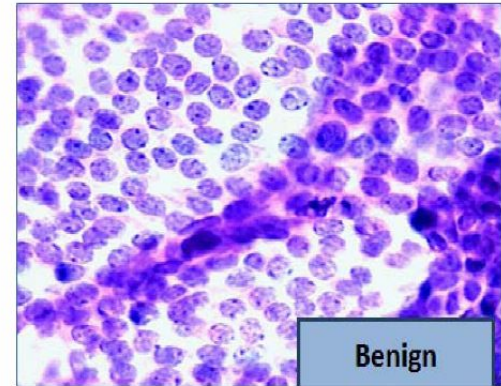
1. `load_breast_cancer()`로 데이터셋을 불러 온 후, 들여다 봅시다. (다음 2 page의 설명 참조)
 - `DESCR`: 데이터 세트에 대한 설명과 각 특징 설명
 - `data`: 특징 데이터 세트
 - `feature_names`: 특징 이름
 - `target`: 분류용은 레이블 값, 회귀용은 숫자 결과값 데이터
 - `target_names`: 개별 레이블의 이름 (분류용)
2. 판다스(pandas)의 `DataFrame`를 사용해서 데이터셋을 변환합니다.
3. 데이터셋이 잘 변환되었는지, 처음 5개의 데이터셋을 살펴봅니다.

UCI data - [https://archive.ics.uci.edu/ml/datasets/Breast+Cancer+Wisconsin+\(Diagnostic\)](https://archive.ics.uci.edu/ml/datasets/Breast+Cancer+Wisconsin+(Diagnostic))

실습(설명)

데이터셋(Datasets) 불러오고, 사용하기

1. UCI ML 위스콘신 유방암 진단 데이터셋의 사본.
2. 특징은 유방 덩어리의 미세 바늘 FNA의 디지털화된 이미지이고, 그것이 이미지에 존재하는 세포핵의 특성을 설명
3. 총 30개의 속성과 **malignant**(악성), **benign**(양성)의 두가지 타겟값을 가지고 있다.



실습(설명)

데이터셋(Datasets)

불러오고, 사용하기

데이터의 이해를 돕기 위해 포함된
32개의 변수에 대한 간략한
설명입니다.

id	환자 식별 번호
diagnosis	양성 여부 (M = 악성, B = 양성)
각 세포에 대한 정보들	
radius	반경 (중심에서 외벽까지 거리들의 평균값)
texture	질감 (Gray-Scale 값들의 표준편차) #gray-scale 값은 광도의 정보를 전달할 수
perimeter	둘레
area	면적
smoothness	매끄러움(반경길이의 국소적 변화)
compactness	조그만 정도(둘레 ² /면적 - 1)
concavity	오목함(윤곽의 오목한 부분의 정도)
points	오목한 점의 수
symmetry	대칭
dimension	프랙탈 차원(해안선근사 -1)
_mean	3 ~ 12 번까지는 평균값을 의미합니다.
_se	13 ~ 22 번까지는 표준오차(Standard Error) 를 의미합니다.
_worst	23 ~ 32 번까지는 각 세포별 구분들에서 제일 큰 3개의 값을 평균낸 값입니다.

Preprocessing(전처리)

Scaling (스케일링/피쳐 스케일링) 이란?

- 입력된 데이터에는 각각의 피쳐에, 해당 피쳐들의 값을 일정한 수준으로 맞춤
- 스케일링 방법:
 - 표준화(standardization)
 - 정규화(normalization)

Preprocessing(전처리)

Scaling (스케일링/피쳐 스케일링) 왜 해야하나?

- 온도 값들의 평균 -102, 대다수의 값들이 큰 음수
- 진동 값들은 평균 3042, 대다수의 값들이 큰 양수

이 두 피쳐의 값들을 있는 그대로 넣었을 때, 이 두 값의 평균과 분산, 최대 최소 값이 제각각이기 때문에 두 피쳐를 비교하기 어렵고 따라서 학습 성능이 떨어짐

표준화를 통해

- 두 피쳐들의 값들을 평균 0 표준편차 1 인 표준 정규 분포, 혹은
- 값의 범위를 0~1 로 제한하는 정규화

모델의 학습 성능을 높이기 위해, 해당 값들을 일정한 범위 의 값으로 스케일링 필요

기기명	온도	진동
x1	-150	4500
x2	-88	3042
x3	-20	2788
x4	-45	2400

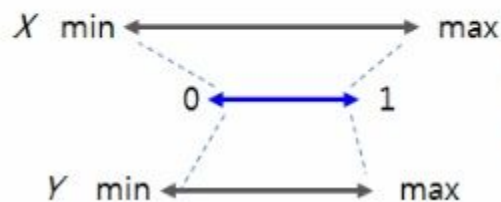
Preprocessing(전처리)

MinMaxScaler란?

- 최소값(Min)과 최대값(Max)을 사용해서 '0~1' 사이의 범위(range)로 데이터를 표준화해주는 '0~1 변환'
- 인공신경망, 딥러닝 할 때 변수들을 '0~1' 범위로 변환해서 사용



최소 최대 '0~1' 범위 표준화
(Scaling features to Min Max '0~1' range)



`sklearn.preprocessing.MinMaxScaler()`
`sklearn.preprocessing.minmax_scale()`

Preprocessing(전처리)

MinMaxScaler()

- sklearn의 preprocessing를 이용해서 사용
- <https://scikit-learn.org/stable/modules/classes.html?highlight=scikit-learn%20preprocessing#module-sklearn.preprocessing>

fit_transform(X)

- MinMaxScaler의 함수 (preprocessing의 함수가 아님) - 데이터에 맞춘 다음 변환
- https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.MinMaxScaler.html#sklearn.preprocessing.MinMaxScaler.fit_transform

같이하기

Preprocessing(전처리)

range[-1, 1]에서 range[0,1]로 바뀐것이 확인됨.

```
import numpy as np
from sklearn import preprocessing
X_train = np.array([[ 1., -1.,  2.],
                    [ 2.,  0.,  0.],
                    [ 0.,  1., -1.]])

min_max_scaler = preprocessing.MinMaxScaler()
X_train_minmax = min_max_scaler.fit_transform(X_train)
X_train_minmax

array([[0.5       , 0.       , 1.       ],
       [1.       , 0.5     , 0.33333333],
       [0.       , 1.       , 0.       ]])
```

실습

Preprocessing(전처리)

1. 넘파이를 사용해서 데이터셋을 만드세요.
2. preprocessing의 MinMaxScaler를 사용해서 scaler를 만드세요.
3. 이 scaler의 fit_transform(data)를 사용해서 데이터셋의 스케일을 변형하세요.
4. 결과물을 출력하세요.

```
1, 2, 3  
4, 11, 6  
7, 11, 9  
10, 11, 12  
13, 14, 15  
16, 17, 18
```

Normalization(정규화)

iris

6.3.3. Normalization

norm='l2'

See code snippet

normalizer.transform(X)

normalizer.transform([[-1., 1., 0.]])

Normalization(정규화)

정규화란?

정규화는 입력된 x 값들을 모두 0과 1사이의 값으로 변환하는 방식

정규화 공식은 간단하다.

$$x_{new} = \frac{x - x_{min}}{x_{max} - x_{min}}$$

Normalization(정규화)

normalize()

- preprocessing에서 사용.
- <https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.normalize.html#sklearn.preprocessing.normalize>

```
sklearn.preprocessing.normalize(X, norm='l2', *, axis=1, copy=True, return_norm=False)
```

[\[source\]](#)

Scale input vectors individually to unit norm (vector length).

Read more in the [User Guide](#).

Parameters:

X : {array-like, sparse matrix} of shape (n_samples, n_features)

The data to normalize, element by element. scipy.sparse matrices should be in CSR format to avoid an unnecessary copy.

norm : {'l1', 'l2', 'max'}, default='l2'

The norm to use to normalize each non zero sample (or each non-zero feature if axis is 0).

axis : {0, 1}, default=1

axis used to normalize the data along. If 1, independently normalize each sample, otherwise (if 0) normalize each feature.

Normalization(정규화)

예제1 - 마이너스의 값이 있는 경우



```
X = [[ 1., -1., 2.],  
      [ 2., 0., 0.],  
      [ 0., 1., -1.]]
```

```
X_normalized = preprocessing.normalize(X)
```

```
X_normalized
```

```
array([[ 0.40824829, -0.40824829,  0.81649658],  
       [ 1.         ,  0.         ,  0.         ],  
       [ 0.         ,  0.70710678, -0.70710678]])
```

같이하기

Normalization(정규화)

예제2 - digits data - load_digits()

```
[6] from sklearn.datasets import load_digits
     from sklearn import preprocessing
     digits = load_digits()
     digits.data

array([[ 0.,  0.,  5., ...,  0.,  0.,  0.],
       [ 0.,  0.,  0., ..., 10.,  0.,  0.],
       [ 0.,  0.,  0., ..., 16.,  9.,  0.],
       ...,
       [ 0.,  0.,  1., ...,  6.,  0.,  0.],
       [ 0.,  0.,  2., ..., 12.,  0.,  0.],
       [ 0.,  0., 10., ..., 12.,  1.,  0.]])
```

같이하기

Normalization(정규화)

예제2 - digits data - load_digits()

```
array([[ 0.,  0.,  5., ...,  0.,  0.,  0.],
       [ 0.,  0.,  0., ..., 10.,  0.,  0.],
       [ 0.,  0.,  0., ..., 16.,  9.,  0.],
       ...,
       [ 0.,  0.,  1., ...,  6.,  0.,  0.],
       [ 0.,  0.,  2., ..., 12.,  0.,  0.],
       [ 0.,  0., 10., ..., 12.,  1.,  0.]])
```



```
d_normalized = preprocessing.normalize(digits.data)
d_normalized
```

```
array([[0.          , 0.          , 0.09024036, ..., 0.          , 0.          ,
        0.          ],
       [0.          , 0.          , 0.          , ..., 0.15413829, 0.          ,
        0.          ],
       [0.          , 0.          , 0.          , ..., 0.24153867, 0.1358655 ,
        0.          ],
       ...,
       [0.          , 0.          , 0.0140138 , ..., 0.08408278, 0.          ,
        0.          ],
       [0.          , 0.          , 0.03044313, ..., 0.18265877, 0.          ,
        0.          ],
       [0.          , 0.          , 0.14230641, ..., 0.17076769, 0.01423064,
        0.          ]])
```

실습

Normalization(정규화)

- sklearn.datasets와 preprocessing를 불러옵니다(import)
- iris dataset - sklearn.datasets의 load_iris를 사용해서 iris란 변수에 저장
- iris.data 출력 (range 0-6)
- pandas를 불러옵니다(import)
- pandas의 DataFrame를 사용해서 iris_df란 변수에 저장
- iris_df 출력 (range 0-6)
- preprocessing.normalize를 사용해서 데이터셋을 변환후 iris_norm이란 변수에 저장
- iris_norm 출력 (range 0-1)



Reference

https://scikit-learn.org/stable/tutorial/machine_learning_map/index.html

<https://cyan91.tistory.com/38>

<https://medium.com/@Nivitus/iris-flower-classification-machine-learning-d4e337140fa4>

<https://www.semanticscholar.org/paper/Classifying-breast-cancer-types-based-on-fine-data-Ahmad-Yusoff/60e88562a14a03e7c9be580f965180f4c9b83d08>

<https://gomguard.tistory.com/52>

<https://huidea.tistory.com/39>

<https://rfriend.tistory.com/270>

<https://scikit-learn.org/stable/modules/preprocessing.html>

<https://cyan91.tistory.com/40?category=230402>