

CSE3013-02 / AIE3091-02

소프트웨어 개발도구 및 환경실습

2025년 1학기, 1주차 3/11 화요일

조교(강의자) 소개 및 연락처

- 박한나(메인 조교)
 - 인공지능학과 석사과정
 - 채점 문의: crescent.check@gmail.com
- 이강호
 - 컴퓨터공학과 석사과정
 - 출결 문의: lee7roho@sogang.ac.kr

>예비보고서/결과보고서/과제.. 관련 질문은 ‘사이버캠퍼스(eclass.sogang.ac.kr) - 질의응답’<

[오늘의 실습]

Python환경의 Google Colaboratory (a.k.a. Google Colab)에서
scikit-learn 라이브러리를 이용하여
기본적인 머신러닝 모델 제작

목차

1. `scikit-learn`과 파이썬 라이브러리 사용
2. 간단 요약 Google Colab 사용법
3. 기본적인 `scikit-learn`을 이용한 머신러닝

실습 준비물: Google 계정

👁️... 있으시죠?

Python Library

Library in Python?

- 유용한 기능들을 모아둔 것
- 보통 Python에서는 '패키지 (Package)'의 형태로 제공

파이썬에서는?

1. 표준 라이브러리 (Standard Library)
2. 외부 라이브러리 (Third Party Library)

표준 라이브러리 (Standard Library)

- 설치할 때 기본 제공
- os, sys, pathlib
- datetime, time
- math, random
- re
- ... 많다!
 - [The Python Standard Library](#) 참고


외부 라이브러리 (Third Party Library)

- 따로 pip, conda와 같은 도구를 이용해 설치해야 한다.
 - but, 대부분의 유-명한 외부 라이브러리는 Google Colab에 기본적으로 설치됨.
- numpy, pandas, matplotlib
- pytorch, huggingface
- tqdm ...
- 정말 많다!
 - [PyPI](#) 참고

scikit-learn (sklearn)

- [Scikit-learn](#)
- 외부 라이브러리 (오픈소스, 2007년부터 제작)
- 파이썬에서 머신러닝을 구현할 때 사용
 - Classification: 어떤 class(범주)에 속하는가
 - Regression: 연속된 값을 예측
 - Clustering: 비슷한 집합을 묶기
 - Preprocessing: 머신러닝 알고리즘에 사용하기 위해서 여러 형태로 변환 지원
 - etc

파이썬에서 라이브러리 import 하는 법



```
import numpy

# numpy.함수이름( ) 형태로 사용 가능
# array( ) 함수: numpy형식의 배열 생성
arr = numpy.array([1, 2, 3])

print(arr) # [1 2 3]
```

`import`를 사용합니다.

라이브러리 별칭 지정하기

```
import numpy
```

```
# numpy.함수이름( ) 형태로 사용 가능  
# array( ) 함수: numpy형식의 배열 생성  
arr = numpy.array([1, 2, 3])
```

```
print(arr) # [1 2 3]
```

```
import numpy as np
```

```
# np.함수이름( ) 형태로 사용 가능  
# array( ) 함수: numpy형식의 배열 생성  
arr = np.array([1, 2, 3])
```

```
print(arr) # [1 2 3]
```

사실 둘은 같은 동작을 하는 코드입니다.
오른쪽은 `as` 키워드 사용

라이브러리 별칭 지어주기

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

data = np.random.rand(5) # 0~1 사이 난수 5개 생성
df = pd.DataFrame({"values": data}) # 데이터프레임 생성

plt.figure(figsize=(12, 7))
sns.lineplot(data=df, x=df.index, y="values") # 선 그래프 그리기
plt.show()
```

`as` 키워드를 통해 코드가 간결해질 수 있습니다.
보통 암묵적으로 쓰는 이름이 있습니다.

라이브러리 별칭 지어주기



그래서 이러시면 곤란합니다
(다른 사람이 쉽게 코드를 이해할 수 없음)

나는

`sklearn.model_selection.train_test_split(...)`을 쓰고 싶다

[Install](#)[User Guide](#)[API](#)[Examples](#)[Community](#)[More](#)[RepeatedKfold](#)[RepeatedStratifiedKfold](#)[ShuffleSplit](#)[StratifiedGroupKfold](#)[StratifiedKfold](#)[StratifiedShuffleSplit](#)[TimeSeriesSplit](#)[check_cv](#)[train_test_split](#)[GridSearchCV](#)[HalvingGridSearchCV](#)[HalvingRandomSearchCV](#)

[Home](#) > [API Reference](#) > [sklearn.model_selection](#) > [train_test_split](#)

train_test_split

```
sklearn.model_selection.train_test_split(*arrays, test_size=None,  
train_size=None, random_state=None, shuffle=True, stratify=None)
```

Split arrays or matrices into random train and test subsets.

[\[source\]](#)

Quick utility that wraps input validation, `next(ShuffleSplit().split(X, y))`, and application to input data into a single call for splitting (and optionally subsampling) data into a one-liner.

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

```
from library_name import module_or_func
```




```
from sklearn.model_selection import train_test_split  
  
train, test = train_test_split(data, ...)
```

`from` 키워드를 사용하면
해당 라이브러리의 특정 모듈이나 함수만 불러온다.

여기서는 `sklearn.model_selection`에서 `train_test_split`만 가져온다
(c.f. `model_selection`: `scikit-learn` 라이브러리(패키지)안에 있는 패키지(서브 패키지))


```
from library_name import *
```



```
from sklearn.model_selection import *  
  
train, test = train_test_split(data, ...)
```

``*` 을 사용하면
해당 라이브러리의 특정 모듈이나 함수 **전체**를 불러온다.

[KFold\(...\)](#) 등을 `sklearn.model_selection.` 없이 바로 사용 가능하다
(보통은 비추천, 어떤 함수가 어디서 왔는지 헛갈릴 수 있음)

[정리] 다양한 import 방식

1. import ~: `sklearn.model_selection` 전체 다 가져온다 & 패키지명도 붙여야 함

```
import sklearn.model_selection

train, test = sklearn.model_selection.train_test_split(data, ...)
```


2. from ~ import ~: `sklearn.model_selection`에서 `train_test_split`만 가져온다

```
from sklearn.model_selection import train_test_split

train, test = train_test_split(data, ...)
```

[정리] 다양한 import 방식


1. `import ~: sklearn.model_selection` 전체 다 가져온다 & 패키지명도 붙여야 함



```
import sklearn.model_selection

train, test = sklearn.model_selection.train_test_split(data, ...)
```

2-1. `from ~ import *`: `sklearn.model_selection`에서 전체 다 가져온다
& 패키지명 언급 X



```
from sklearn.model_selection import *

train, test = train_test_split(data, ...)
```

Colab 간단 설명



Colab이란?

Colaboratory(줄여서 'Colab'이라고 함)을 통해 브라우저 내에서 Python 스크립트를 작성하고 실행할 수 있습니다.

- 구성이 필요하지 않음
- 무료로 GPU 사용
- 간편한 공유

예를 들어 다음은 값을 계산하여 변수로 저장하고 결과를 출력하는 간단한 Python 스크립트가 포함된 코드 셀입니다.

✓
0초



```
1 seconds_in_a_day = 24 * 60 * 60  
2 seconds_in_a_day
```



86400

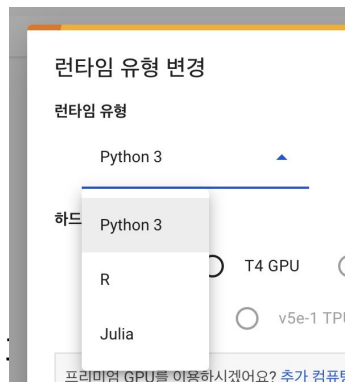
출처: <https://colab.research.google.com/?hl=en>

Google Colaboratory (Colab)

- Google에서 만든 브라우저에서 코드를 작성하고 실행할 수 있게 해주는 Jupyter notebook 서비스
 - Python 3, R, Julia 지원
- Jupyter notebook을 로컬에서 사용하기 위해 수행해야 하는 설치 작업을 할 필요가 없어 편리

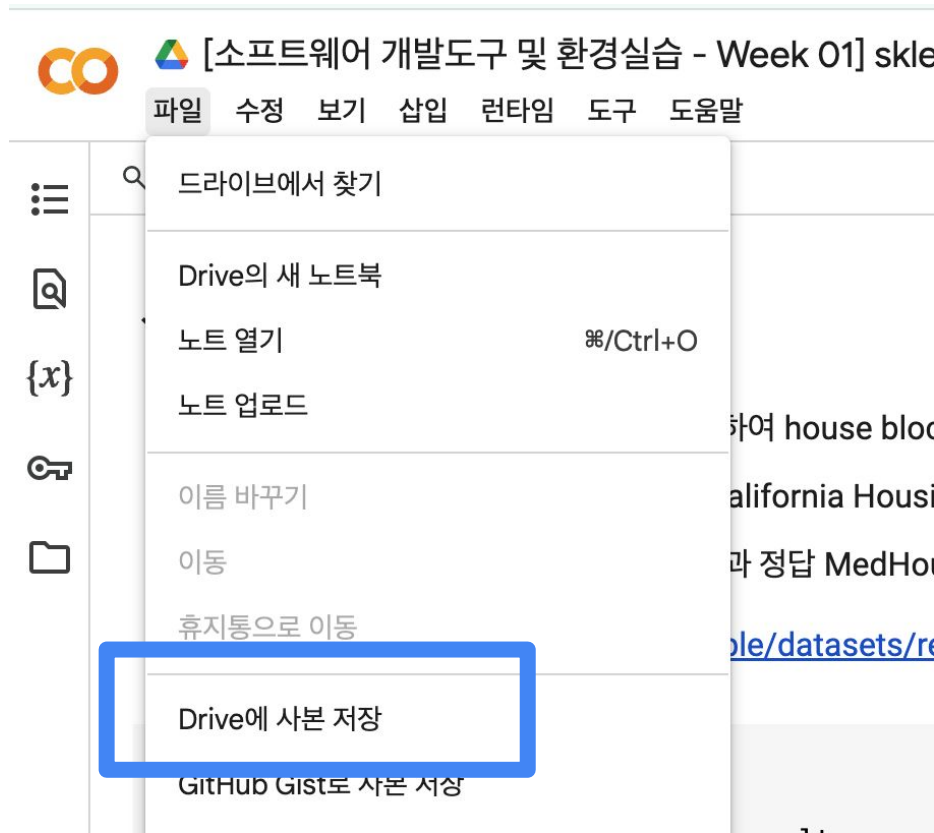
Jupyter Notebook

- <https://jupyter.org/>
- 코드 실행, 시각화, 문서 작성을 한 곳에서 할 수 있는 인터랙티브 웹 기반 개발



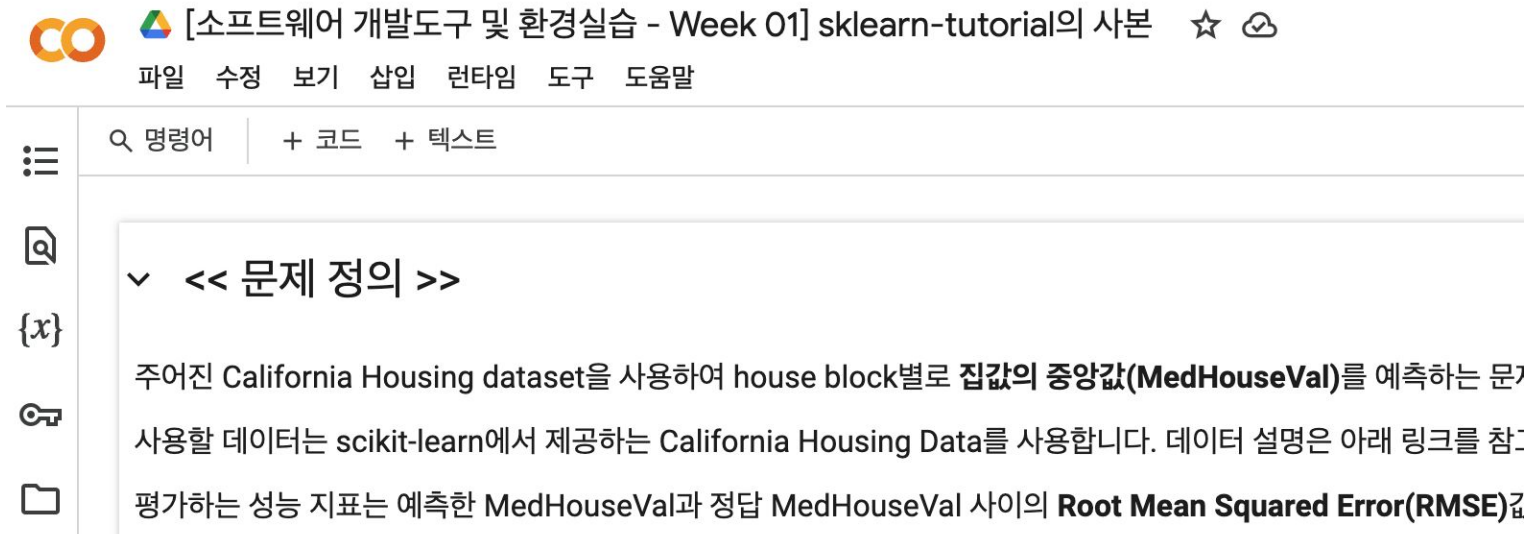
공유된 실습 사용 방법 - 저장

- [파일] > [Drive에 사본 저장]
 - [File] > [Save copy in Drive]



공유된 실습 사용 방법 - 저장

- `~**의 사본**` 이름으로 저장
- 이름을 더블 클릭해서 노트북 제목 변경 가능
- [파일] > [이동]을 통해 코랩 파일 이동 가능




The screenshot shows the Google Colab interface. At the top, there's a header with the Colab logo and the notebook title: "[소프트웨어 개발도구 및 환경실습 - Week 01] sklearn-tutorial의 사본". Below the title are icons for file, edit, view, insert, runtime, tools, and help. The left sidebar contains icons for a menu, a document, a variable $\{x\}$, a key, and a folder. The main content area shows a code cell with the following text:

```
✓ << 문제 정의 >>
```

주어진 California Housing dataset을 사용하여 house block별로 **집값의 중앙값(MedHouseVal)**를 예측하는 문제
사용할 데이터는 scikit-learn에서 제공하는 California Housing Data를 사용합니다. 데이터 설명은 아래 링크를 참고
평가하는 성능 지표는 예측한 MedHouseVal과 정답 MedHouseVal 사이의 **Root Mean Squared Error(RMSE)**값

Colab Notebooks 폴더에 사본이 저장됩니다

 드라이브

드라이브에서 검색

+ 신규

홈

내 드라이브

컴퓨터

중요 문서함
여기에 표시할 항목에 별표표시

공유 문서함

최근 문서함

스팸

휴지통

저장용량

내 드라이브 ▾

유형 ▾ 사람 ▾ 수정 날짜 ▾ 출처 ▾


이름 ↑

Colab Notebooks

내 드라이브 > Colab Notebooks ▾

유형 ▾ 사람 ▾ 수정 날짜 ▾ 출처 ▾

이름 ↑

 [소프트웨어 개발도구 및 환경실습 - Week 01] sklearn-tutorial의 사본

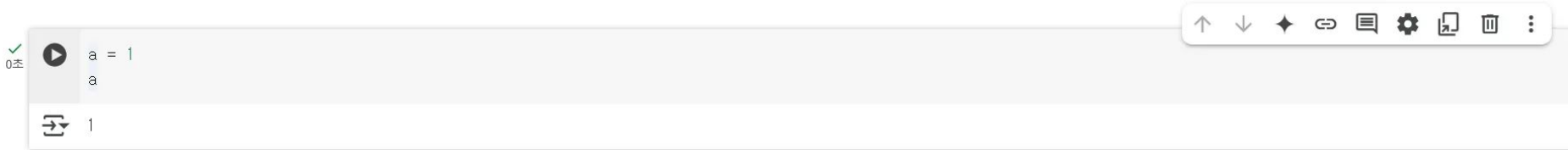
Google Drive에 파일 업로드하여 사용 -> 가능합니다

- 파일 업로드 -> 파일 클릭 -> Colab으로 이동

사이버캠퍼스(eclass.sogang.ac.kr)에 올라온 1주차 실습 자료를 다운받아 진행하시면 됩니다.

1. 코드 셀과 텍스트 셀 / 1-1. 코드 셀

- 주피터 노트북은 여러 개의 셀을 포함할 수 있으며, 셀을 하나씩 실행할 수 있음
- 실행 방법
 - [실행 버튼] : 해당 셀 실행
 - [Ctrl+Enter] : 해당 셀 실행
 - [Shift+Enter] : 해당 셀을 실행하고 다음 셀로 넘어 감
 - [Alt+Enter] : 해당 셀을 실행하고 바로 밑에 새로운 셀 추가



Jupyter Notebook(IPython) 기반의 Google Colab

1. `display()` 함수로 Pandas 객체와 같은 데이터, 이미지, HTML 등 더 보기 좋게 출력 가능
2. 코드 실행 순서에 주의해야 한다
 - 아래에서 실행한 변수 값이 위에서 적용될 수 있다.
 - ``a = 10``을 실행하고, 위에서 ``print(a)``를 실행하면 10이 출력
 - [런타임] > [세션 다시 시작]으로 변수 값 초기화 가능

✓ [1] a = 3
초

✓ [2] print(a)
초

↻ 3

✓ [3] a = 10
초

✓ [1] a = 3
초

✓ [4] print(a)
초

↻ 10

✓ [3] a = 10
초

Jupyter Notebook(IPython) 기반의 Google Colab

3. ``A?`` 혹은 ``help(A)`` 으로 설명을 볼 수 있다.

- ``A?``: Google Colab/Jupyter Notebook/IPython 기본 기능
 - [Built-in magic commands — IPython 9.0.2 documentation](#)
- ``help(A)``: Python built-in function
 - [Built-in Functions — Python 3.13.2 documentation](#)

분할하세요.

(단, train : test 데이터의 비율은 8 : 2로 합니다.)

```
from sklearn.model_selection import train_test_split
train_test_split?
```

도움말

도움말 X

Signature: train_test_split(*arrays, test_size=None, train_size=None, random_state=None, shuffle=True, stratify=None)

Docstring:

Split arrays or matrices into random train and test subsets.

Quick utility that wraps input validation, ``next(ShuffleSplit().split(X, y))``, and application to input data into a single call for splitting (and optionally subsampling) data into a one-liner.

Read more in the :ref:`User Guide <cross_validation>`.

Parameters

*arrays : sequence of indexables with same length / shape[0]
Allowed inputs are lists, numpy arrays, scipy-sparse matrices or pandas dataframes.

test_size : float or int, default=None
If float, should be between 0.0 and 1.0 and represent the proportion of the dataset to include in the test split. If int, represents the absolute number of test samples. If None, the value is set to the complement of the train size. If ``train_size`` is also None, it will be set to 0.25.

```
from sklearn.model_selection import train_test_split
help(train_test_split)
```

Help on function train_test_split in module sklearn.model_selection._split:

train_test_split(*arrays, test_size=None, train_size=None, random_state=None, shuffle=True)
Split arrays or matrices into random train and test subsets.

Quick utility that wraps input validation, `next(ShuffleSplit().split(X, y))`, and application to input data into a single call for splitting (and optionally subsampling) data into a one-liner.

Read more in the :ref:`User Guide <cross_validation>`.

Parameters

***arrays** : sequence of indexables with same length / shape[0]

Allowed inputs are lists, numpy arrays, scipy-sparse matrices or pandas dataframes.

test_size : float or int, default=None

If float, should be between 0.0 and 1.0 and represent the proportion of the dataset to include in the test split. If int, represents the absolute number of test samples. If None, the value is set to the complement of the train size. If `train_size` is also None, it will be set to 0.25.

train_size : float or int, default=None

If float, should be between 0.0 and 1.0 and represent the proportion of the dataset to include in the train split. If int, represents the absolute number of train samples. If None, the value is automatically set to the complement of the test size.

random_state : int, RandomState instance or None, default=None

Controls the shuffling applied to the data before applying the split. Pass an int for reproducible output across multiple function calls. See :term:`Glossary <random_state>`.

shuffle : bool, default=True

Whether or not to shuffle the data before splitting. If shuffle=False then stratify must be None.

Jupyter Notebook(IPython) 기반의 Google Colab

그 외...

- ``!`` 을 사용해 Linux 터미널 명령어 사용 가능 (e.g. `!ls`)
- Magic Commands (`%`를 활용한 명령어) 제공 (e.g. ``%pwd``, ``%cd 폴더명``)
 - <https://ipython.readthedocs.io/en/stable/interactive/magics.html>

[오늘의 실습 문제]

scikit-learn 라이브러리를 이용하여
California Housing Dataset으로
캘리포니아 각 지역의 주택 중위 가격 예측하기

실습에 앞서...

- 세상이 좋아져서 코드를 작성할 때 답 얻기 쉬운 것을 저도 잘 압니다....



실습에 앞서...

- 하지만 그 친구는 항상 정답을 줄 수 있을까요?
 - 참말로?
 - 그 친구가 배운 데이터가 내가 필요한 기능보다 과거의 데이터라면?
 - 혹은 내가 다루는 분야의 문서가 없거나 적어서 배우지 못했다면?
 - 내가 일하는 곳이 코ㅇㅇㅇ, 챗ㅇㅇㅇ 등의 도구를 쓸 수 없다고 한다면?



예차니즘 🌟🎄🎅

@yechanism_

Q: 조선왕조실록에 기록 된 세종대왕의 맥북프로 던짐 사건에 대해 알려줘



조선왕조실록에 기록 된 세종대왕의 맥북프로 던짐 사건에 대해 알려줘



세종대왕의 맥북프로 던짐 사건은 역사서적인 조선왕조실록에 기록된 일화로, 15세기 조선시대 세종대왕이 새로 개발한 훈민정음(한글)의 초고를 작성하던 중, 문서 작성 중단에 대한 담당자에게 분노하여 맥북프로와 함께 그를 방으로 던진 사건입니다.

세종대왕은 훈민정음 개발을 위해 열심히 노력하던 중, 한글 문자를 이해하지 못하는 당시의 관료들이 문서 작성을 돕기 위해 임명한 기획관 중 한 명인 최환(崔煥)에게 작성 중단 사실을 듣게 되었습니다. 세종대왕은 이에 분노하여 최환을 부르고, 맥북프로를 함께 들고 그를 방으로 불러들이며 화를 내었습니다. 이후, 세종대왕은 훈민정음의 초고 작성을 다른 담당자에게 넘겨주게 되었습니다.

이 사건은 조선시대의 통치자로서 세종대왕이 어떻게 민중문화를 발전시키기 위해 노력했는지를 보여주는 일화로 유명합니다. 또한, 세종대왕의 열정과 업무처리에 대한 철저한 태도를 보여주는 사례로도 언급되고 있습니다.

이번 기회에 Documentation과 친해지자!

하던대로 하셔도 상관 없으나... 조금씩이라도 친해져보는 것도 좋습니다.

사실 세상의 모든 Docs가 친절하진 않은 아니지만 많은 경우에는 도움이 될 수 있습니다.

- 어떤 함수를 쓰고 싶은데 뭘 넣어할지 모르겠어요!
 - 문서 윗부분(Overview, Signature) 확인
 - 더 알고 싶으면 Parameters(매개변수) 섹션 확인
- 그 함수의 출력값이 궁금해요.
 - Returns(반환 값) 섹션 확인
- 종종 Examples도 있습니다

e.g.) `numpy.array`

<https://numpy.org/doc/stable/reference/generated/numpy.array.html>



NumPy

User Guide

API reference

Building from source

Development

Release notes

Learn

More

`numpy.empty`

`numpy.empty_like`

`numpy.eye`

`numpy.identity`

`numpy.ones`

`numpy.ones_like`

`numpy.zeros`

`numpy.zeros_like`

`numpy.full`

`numpy.full_like`

`numpy.array`

`numpy.asarray`

`numpy.asanyarray`

`numpy.ascontiguousarray`

`numpy.asmatrix`

`numpy.astype`

`numpy.copy`

`numpy.frombuffer`

`numpy.from_dlpack`

`numpy.fromfile`

`numpy.fromfunction`

🏠 > NumPy reference > Routines and objects by topic > Array creation routines
> `numpy.array`

`numpy.array`

`numpy.array(object, dtype=None, *, copy=True, order='K', subok=False, ndmin=0, like=None)`

Create an array.

Parameters:

`object` : *array_like*

An array, any object exposing the array interface, an object whose `__array__` method returns an array, or any (nested) sequence. If object is a scalar, a 0-dimensional array containing object is returned.

`dtype` : *data-type, optional*

The desired data-type for the array. If not given, NumPy will try to use a default `dtype` that can represent the values (by applying promotion rules when necessary.)

`copy` : *bool, optional*

If `True` (default), then the array data is copied. If `None`, a copy will

Function Signature = func_name + parameters + default values + return type



```
numpy.array(  
    object,  
    dtype=None,  
    *,  
    copy=True,  
    order='K',  
    subok=False,  
    ndmin=0,  
    like=None  
)
```

object, dtype: 위치 인자 (positional argument)로 전달 가능

`*` 이후의 인자들: 반드시 키워드 인자 (keyword argument)로 사용 (e.g. **copy=False**)



```
import numpy as np
```

```
arr = np.array([1, 2, 3], dtype=float, copy=False, ndmin=2) # OK
```

```
arr = np.array([1, 2, 3], float, False, 'C') # 오류 발생
```



```
-----  
TypeError                                Traceback (most recent call last)  
<ipython-input-11-009725b40ee2> in <cell line: 0>()  
      3 arr = np.array([1, 2, 3], dtype=float, copy=False, ndmin=2) # OK  
      4  
----> 5 arr = np.array([1, 2, 3], float, False, 'C') # 오류 발생
```

TypeError: array() takes from 1 to 2 positional arguments but 4 were given

Parameters

어떤 기능을 하는지 적혀있음

가능하면 무슨 타입인지도 작성되어 있음

optional인지도 써줌

Parameters:

object : *array_like*

An array, any object exposing the array interface, an object whose `__array__` method returns an array, or any (nested) sequence. If object is a scalar, a 0-dimensional array containing object is returned.

dtype : *data-type, optional*

The desired data-type for the array. If not given, NumPy will try to use a default `dtype` that can represent the values (by applying promotion rules when necessary.)

copy : *bool, optional*

If `True` (default), then the array data is copied. If `None`, a copy will only be made if `__array__` returns a copy, if obj is a nested sequence, or if a copy is needed to satisfy any of the other requirements (`dtype`, `order`, etc.). Note that any copy of the data is shallow, i.e., for arrays with object dtype, the new array will point to the same objects. See Examples for `ndarray.copy`. For `False` it raises a `ValueError` if a copy cannot be avoided. Default: `True`.

Returns

어떻게 나오는지 알려준다
(ndarray의 형식으로 나온다는 글)

가끔씩 같이 보면 좋은 항목도 표시해줍니다.

Returns:

out : *ndarray*

An array object satisfying the specified requirements.

See also

[`empty_like`](#)

Return an empty array with shape and type of input.

[`ones_like`](#)

Return an array of ones with shape and type of input.

[`zeros_like`](#)

Return an array of zeros with shape and type of input.

[`full_like`](#)

Return a new array with shape of input filled with value.

[`empty`](#)

Return a new uninitialized array.

[`ones`](#)

Return a new array setting values to one.

[`zeros`](#)

Return a new array setting values to zero.

[`full`](#)

Return a new array of given shape filled with value.

[`copy`](#)

Return an array copy of the given object.

Examples

정말 도움이 될 때가 많습니다.

Examples

```
>>> import numpy as np
>>> np.array([1, 2, 3])
array([1, 2, 3])
```

Upcasting:

```
>>> np.array([1, 2, 3.0])
array([ 1.,  2.,  3.])
```

More than one dimension:

```
>>> np.array([[1, 2], [3, 4]])
array([[1, 2],
       [3, 4]])
```

사용 데이터셋: California Housing Dataset

[7.2. Real world datasets — scikit-learn 1.6.1 documentation](#)

- 8 features, 1 target

7.2.7. California Housing dataset

Data Set Characteristics:

Number of Instances:	20640
Number of Attributes:	8 numeric, predictive attributes and the target
Attribute Information:	<ul style="list-style-type: none">• MedInc median income in block group• HouseAge median house age in block group• AveRooms average number of rooms per household• AveBedrms average number of bedrooms per household• Population block group population• AveOccup average number of household members• Latitude block group latitude• Longitude block group longitude

Missing Attribute Values: None

사용 데이터셋: California Housing Dataset

[7.2. Real world datasets — scikit-learn 1.6.1 documentation](#)

The target variable is the median house value for California districts, expressed in hundreds of thousands of dollars (\$100,000).

∴ MedHouseVal 예측해야 함 (Target)

실습 과정

1. 데이터와 라이브러리 불러오기
 - Pandas에서 제공하는 데이터 형식으로 변환 필요
2. 살펴보기(원본 데이터, 기본적인 정보 확인) -> Q1, Q2
3. 데이터셋을 train / test로 나누기
 - 데이터를 나누기 위해서 `feature_vector`, `target_value` 나누기
4. train 데이터로 LinearRegression 모델 학습 진행
5. test 데이터도 사용해 모델 성능 평가

1. 데이터와 라이브러리 불러오기

- 일단 실행시키면 됩니다

- [NumPy](#)
- [pandas 2.2.3 documentation](#)
- [matplotlib.pyplot — Matplotlib 3.5.3 documentation](#)
- [Seaborn](#)
- [GitHub - tqdm/tqdm](#)
- [Scikit-learn](#)

```
1 import numpy as np
2 import pandas as pd
3 import matplotlib.pyplot as plt
4 import seaborn as sns
5
6 import os
7 import sys
8 import gc
9
10 from tqdm.auto import tqdm
11
12 import sklearn
```

Q1

A. 데이터를 불러올 때

`fetch_california_housing()` 함수 안에 어떤 인자를 넣는다면 Pandas의 DataFrame 형태로 다룰 수 있습니다.

B. `fetch_california_housing()` 함수가 뭘 반환하는지 문서 등에서 살펴보면 좋습니다.



	MedInc	HouseAge	AveRooms	AveBedrms	Population	AveOccup	Latitude	Longitude	MedHouseVal
0	8.3252	41.0	6.984127	1.023810	322.0	2.555556	37.88	-122.23	4.526
1	8.3014	21.0	6.238137	0.971880	2401.0	2.109842	37.86	-122.22	3.585
2	7.2574	52.0	8.288136	1.073446	496.0	2.802260	37.85	-122.24	3.521
3	5.6431	52.0	5.817352	1.073059	558.0	2.547945	37.85	-122.25	3.413
4	3.8462	52.0	6.281853	1.081081	565.0	2.181467	37.85	-122.25	3.422

...
20635	1.5603	25.0	5.045455	1.133333	845.0	2.560606	39.48	-121.09	0.781
20636	2.5568	18.0	6.114035	1.315789	356.0	3.122807	39.49	-121.21	0.771
20637	1.7000	17.0	5.205543	1.120092	1007.0	2.325635	39.43	-121.22	0.923
20638	1.8672	18.0	5.329513	1.171920	741.0	2.123209	39.43	-121.32	0.847
20639	2.3886	16.0	5.254717	1.162264	1387.0	2.616981	39.37	-121.24	0.894

20640 rows × 9 columns

```
.. _california_housing_dataset:
```

California Housing dataset

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

:Number of Instances: 20640

:Number of Attributes: 8 numeric, predictive attributes and the target

:Attribute Information:

- MedInc median income in block group
- HouseAge median house age in block group
- AveRooms average number of rooms per household
- AveBedrms average number of bedrooms per household
- Population block group population
- AveOccup average number of household members
- Latitude block group latitude
- Longitude block group longitude

:Missing Attribute Values: None

This dataset was obtained from the StatLib repository.
https://www.dcc.fc.up.pt/~ltorgo/Regression/cal_housing.html

The target variable is the median house value for California districts, expressed in hundreds of thousands of dollars (\$100,000).

This dataset was derived from the 1990 U.S. census, using one row per census block group. A block group is the smallest geographical unit for which the U.S.

Q2

1. 위에서 구한 data의 크기를 출력

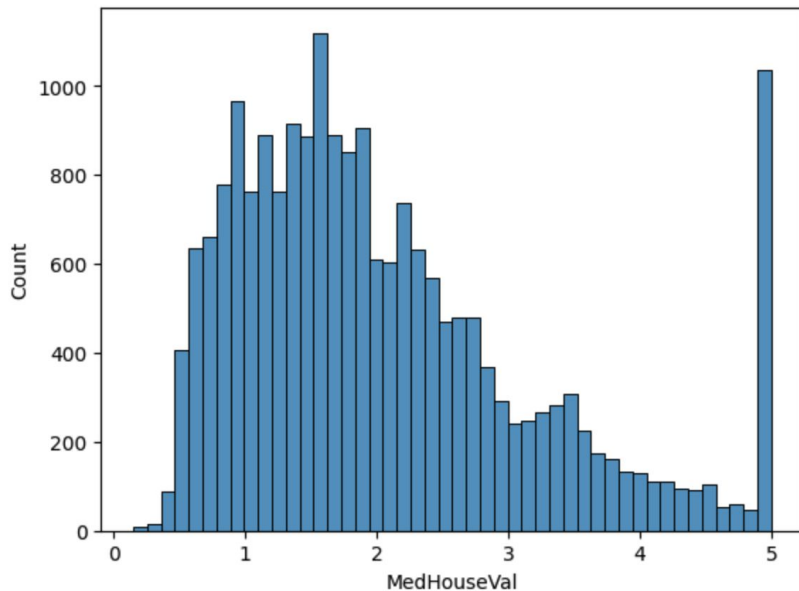
data의 타입은 'pandas.core.frame.DataFrame' 형식으로 되어있습니다.

2. 그래프 그리는 라이브러리를 사용하여

MedHouseVal의 histogram을 출력해보세요.

관련 라이브러리는 이미 import 되어있습니다

(20640, 9)
<Axes: xlabel='MedHouseVal', ylabel='Count'>



Q3

A. DataFrame형식의 data에서 feature vector와 target value을 찾아야 합니다.

B. train_test_split() 함수의 return값으로 다음의 sanity check가 진행됩니다

```
10 # sanity check: (16512, 8) (4128, 8) (16512,) (4128,)
11 print(X_train.shape, X_test.shape, y_train.shape, y_test.shape)
```

⇒ (16512, 8) (4128, 8) (16512,) (4128,)

Q4

LinearRegression 모델을 불러옵니다.

import된 라이브러리에 속한 해당 모델은 여러가지 함수도 포함한 class로 되어있습니다.



▼ LinearRegression ⓘ ?

LinearRegression()

Q5

`train_test_split()`에서 `test_size`와 `random_state`를 고정했기 때문에 항상 같은 train/test dataset이 만들어 집니다.

-> LinearRegression 모델은 같은 데이터셋을 학습하면 같은 모델이 만들어집니다.

-> Q5의 값은 아래처럼 나옵니다.

> 평가하는 성능 지표는 예측한 `MedHouseVal`과 정답 `MedHouseVal` 사이의 **Root Mean Squared Error(RMSE)**값으로 정의합니다.

훈련한 모델을 이용해서 진행하면 됩니다.



`sklearn`과 `numpy`를 이용한 RMSE값 계산은 단 하나의 함수를 이용할 필요는 없습니다.

```
➤ Train RMSE : 0.7197
   Test RMSE : 0.7456
```

도움이 필요하거나 실습 완료하면 손 들어주세요.

감사합니다! 🙏

실습 관련

1. 실습 후 검사 말고 **`1_class_[학번].ipynb`** 파일을 사이버캠퍼스에 제출하면 됩니다
-> 그러고 집 가면 됩니다!  
 - a. 굳이 코랩에서 안해도 됩니다. **`ipynb`** 형식만 지켜주세요
2. 예비 레포트 / 결과 레포트 / 과제 집가서 하세요
 - a. 저도 집에 가고 싶어요.....