

CUAI\_파이썬 머신러닝 완벽 가이드

---

## 02. 사이킷런으로 시작하는 머신러닝

*$\pi$ sun*

박소현 박은우 전찬웅 황인택

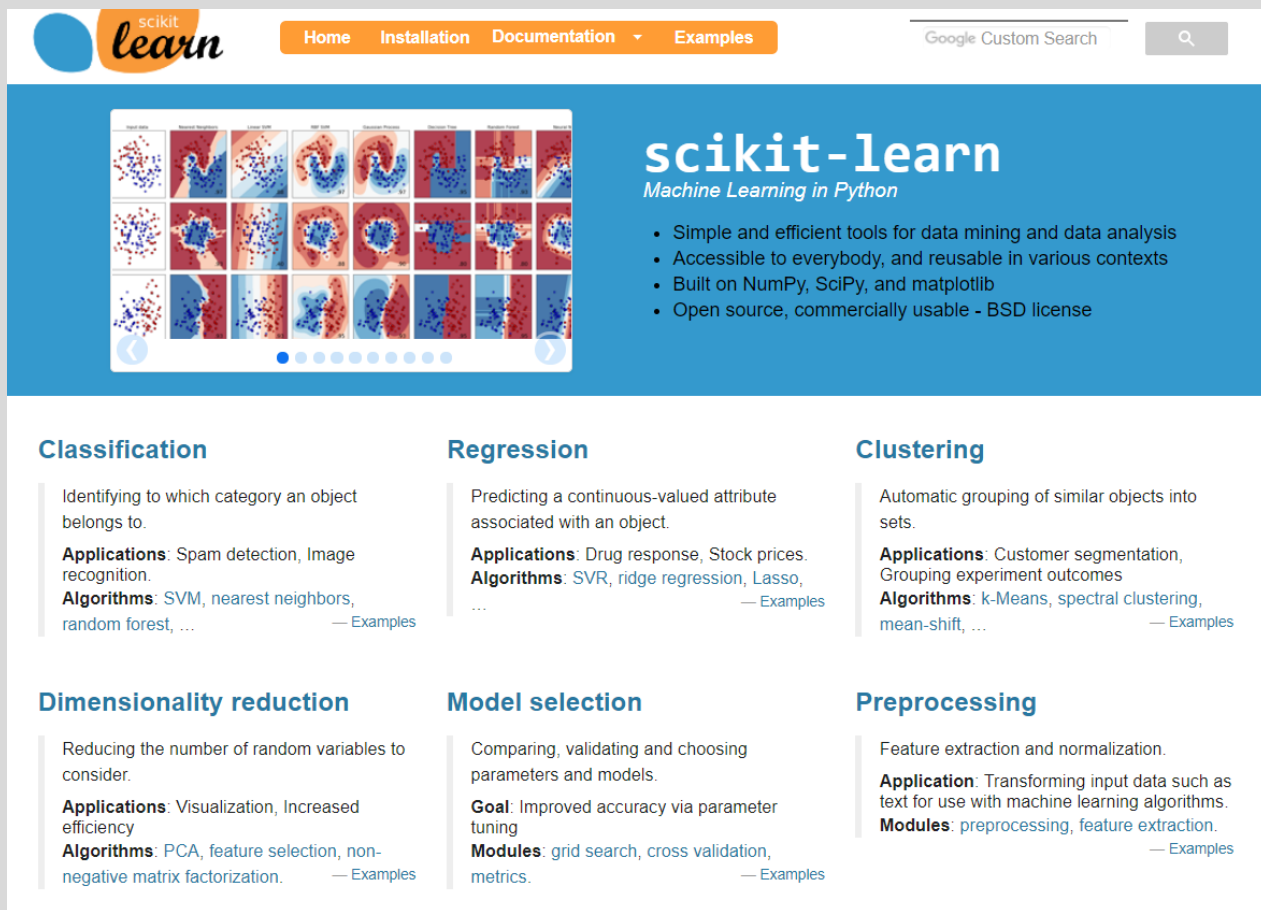
# CONTENTS

01. 사이킷런 소개와 특징

02. 첫 번째 머신러닝 만들어 보기  
- 붓꽃 품종 예측하기

03. 사이킷런의 기반 프레임워크  
익히기

04. Model Selection 모듈 소개



## 사이킷런이란?

- 파이썬 머신러닝 라이브러리 중 가장 많이 사용
- clustering, cross validation, datasets, dimensionality reduction 등을 포함

## 사이킷런의 특징

- 쉽고 가장 파이썬스러운 API 제공
- ML을 위한 다양한 알고리즘과 개발을 위한 편리한 프레임워크와 API 제공
- 오랜 기간 검증, 많은 환경에서 사용되는 성숙한 라이브러리

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

### 분류(Classification)

- 대표적 지도학습(supervised learning)
- 학습 데이터 세트(training data)와 테스트 데이터 세트(test data)로 분류
- 입력 데이터가 어떤 범주에 해당하는지 구분하는 것  
ex) 스팸 문자, 학점, 질병의 여부

### 코드 예제

#### 붓꽃 데이터 피쳐

- Sepal length
- Sepal width
- Petal length
- Petal width

#### 붓꽃 데이터 품종(레이블)

- Setosa(0)
- Vesicolor(1)
- Virginica(2)

#### 사용 모듈

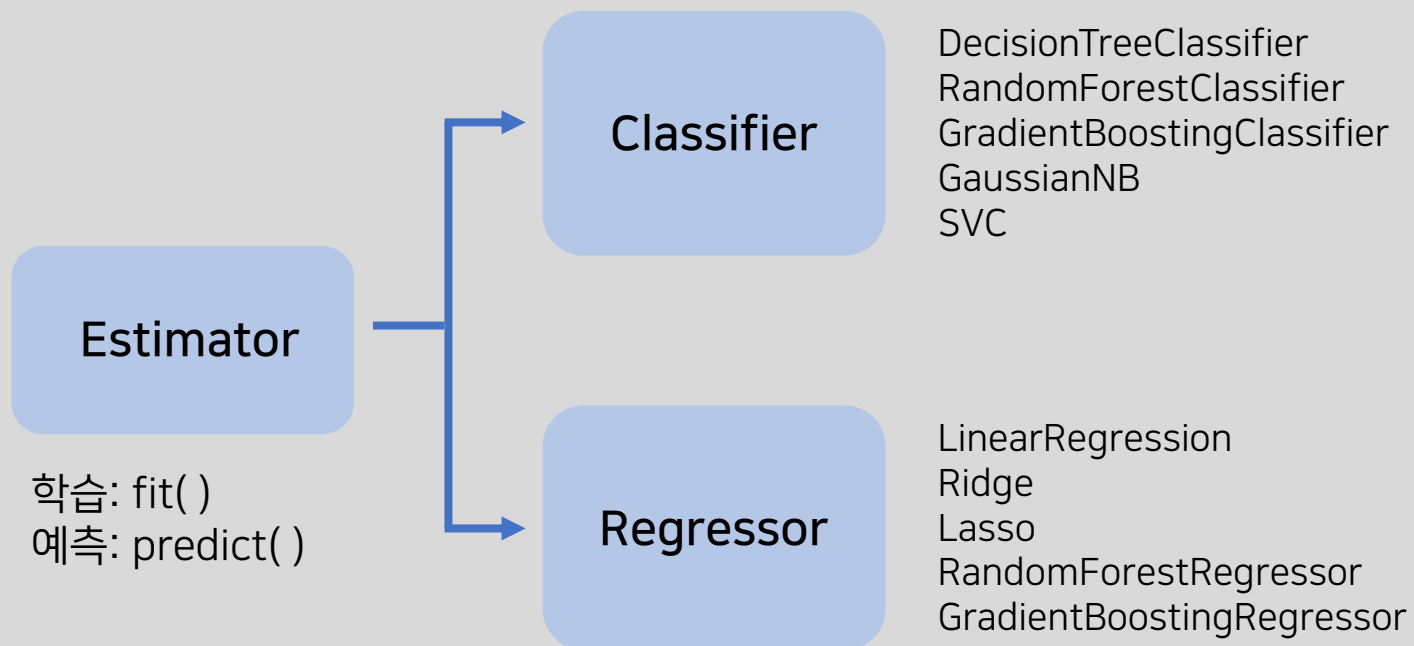
- sklearn.datasets -> load\_iris( )
- sklearn.tree -> DecisionTreeClassifier
- sklearn.model\_selection -> train\_test\_split( )

### 분류(Classification) 예측 프로세스

1. 데이터 세트 분리: 학습 데이터와 테스트 데이터로 분리
2. 모델 학습: 학습 데이터를 기반으로 ML 알고리즘을 적용해 모델을 학습
3. 예측 수행: 학습된 ML 모델을 이용해 테스트 데이터의 분류를 예측
4. 평가: 예측 결과값과 실제 결과값을 비교해 ML 모델 성능 평가

## Estimator 이해 및 fit( ), predict( ) 메서드

- ML 모델 학습을 위해서 fit( )을, 학습된 모델의 예측을 위해서 predict( ) 메서드 제공
- 분류 알고리즘 구현 클래스: Classifier
- 회귀 알고리즘 구현 클래스: Regressor
- 지도학습: fit( ), predict( )
- 비지도학습: fit( ), transform( )



## 사이킷런의 주요 모듈

분류	모듈명	설명
예제 데이터	sklearn.datasets	사이킷런에 내장되어 예제로 제공하는 데이터 세트
피처 처리	sklearn.preprocessing	데이터 전처리에 필요한 다양한 가공 기능 제공
	sklearn.feature_selection	알고리즘에 큰 영향을 미치는 피처를 우선순위로 선택 작업을 수행하는 다양한 기능 제공
	sklearn.feature_extraction	텍스트 데이터나 이미지 데이터의 벡터화된 피처를 추출 하는데 사용됨
피처 처리&차원 축소	sklearn.decomposition	차원 축소와 관련한 알고리즘을 지원하는 모듈
데이터 분리, 검증&파라미터 튜닝	sklearn.model_selection	교차 검증을 위한 학습용/테스트용 분리, 그리드 서치 로 최적 파라미터 추출 등의 API제공
평가	sklearn.metrics	분류, 회귀, 클러스터링, 페어와이즈에 대한 다양한 성능 측정 방법 제공

## 사이킷런의 주요 모듈

분류	모듈명	설명
ML 알고리즘	sklearn.ensemble	앙상블 알고리즘 제공
	sklearn.linear_model	선형 회귀, 릿지, 라쏘 및 로지스틱 회귀 등 회귀 관련 알고리즘을 지원
	sklearn.naïve_bayes	나이브 베이즈 알고리즘 제공
	sklearn.neighbors	최근접 이웃 알고리즘 제공
	sklearn.svm	서포트 벡터 머신 알고리즘 제공
	sklearn.tree	의사 결정 트리 알고리즘 제공
	sklearn.cluster	비지도 클러스터링 알고리즘 제공
유틸리티	sklearn.pipeline	피처 처리 등의 변환과 ML 알고리즘 학습, 예측 등을 함께 묶어서 실행할 수 있는 유틸리티 제공

## 내장된 예제 데이터 세트

API명	설명
<code>datasets.load_boston( )</code>	회귀 용도/ 미국 보스턴의 집 피쳐들과 가격에 대한 데이터 세트
<code>datasets.load_breast_cancer( )</code>	분류 용도/ 위스콘신 유방암 피쳐들과 악성, 음성 레이블 데이터 세트
<code>datasets.load_diabetes( )</code>	회귀 용도/ 당뇨 데이터 세트
<code>datasets.load_digits( )</code>	분류 용도/ 0~9 숫자의 이미지 픽셀 데이터 세트
<code>datasets.load_iris( )</code>	분류 용도/ 붓꽃에 대한 피쳐를 가진 데이터 세트
<code>fetch_covtype( )</code>	회귀 용도/ 토지 조사 자료
<code>fetch_20newsgroups( )</code>	뉴스 그룹 텍스트 자료
<code>fetch_olivetti_faces( )</code>	얼굴 이미지 자료
<code>fetch_lfw_people( )</code>	얼굴 이미지 자료
<code>fetch_lfw_pairs( )</code>	얼굴 이미지 자료
<code>fetch_rcv1( )</code>	로이터 뉴스 말뭉치
<code>fetch_mldata( )</code>	ML 웹사이트에서 다운로드

## 코드 예제

사이킷런에 내장된 데이터 세트는 일반적으로 딕셔너리 형태로 되어 있음

- data : 피처의 데이터 세트
- target : 분류 시 레이블 값, 회귀일 때는 숫자 결과값 데이터 세트
- target\_names : 개별 레이블의 이름
- feature\_names : 피처의 이름
- DESCR : 데이터 세트에 대한 설명&각 피처의 설명

## 학습/테스트 데이터 세트 분리 - train\_test\_split( )

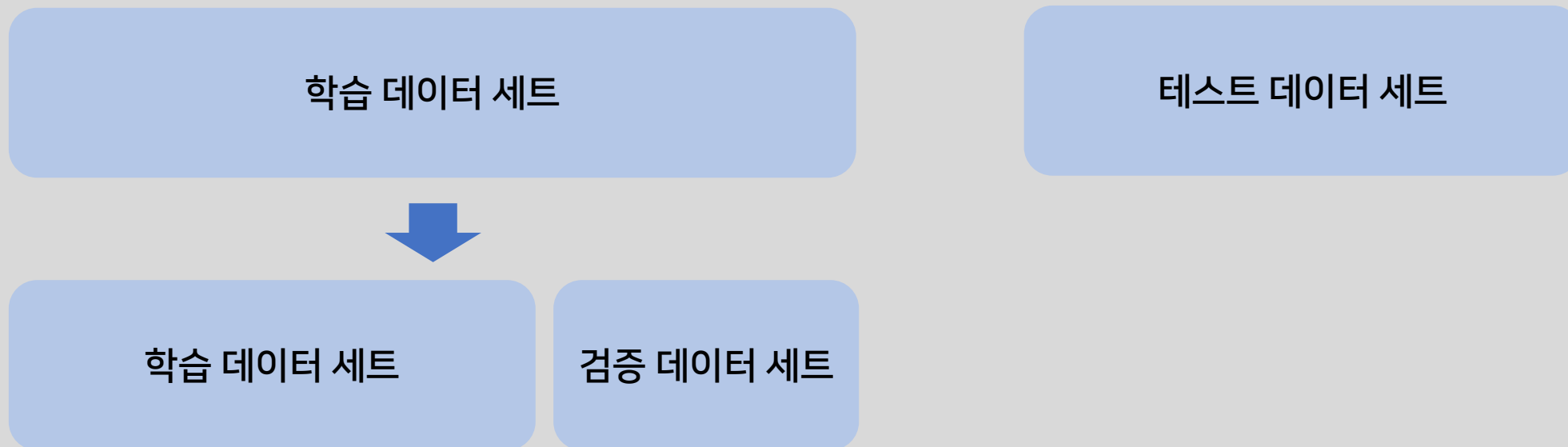
train\_test\_split(피쳐 데이터, 레이블 데이터, test\_size, train\_size, shuffle, random\_state)

x\_train, x\_test, y\_train, y\_test 반환

- test\_size : 테스트 데이터 세트 크기 설정. 디폴트는 25%
- train\_size : 학습용 데이터 세트 크기 설정. 잘 사용x
- shuffle : 데이터 분리 전 미리 섞을지 결정. 디폴트는 True
- random\_state : seed와 같은 역할. 지정하지 않으면 수행할 때마다 다른 학습/테스트 데이터 생성

## 교차 검증(Cross Validation)

충분한 독립적인 테스트용 데이터를 구할 수 없거나 과적합이 발생했을 때 교차 검증 사용  
데이터 편중 발생 시 별도의 여러 세트로 구성된 학습 데이터/검증 데이터에서 학습/평가 수행



## K 폴드 교차 검증

가장 보편적으로 사용되는 교차 검증 기법. K개의 데이터 폴드 세트를 만들어 K번만큼 각 폴드 세트에 학습과 검증 평가를 반복적으로 수행.

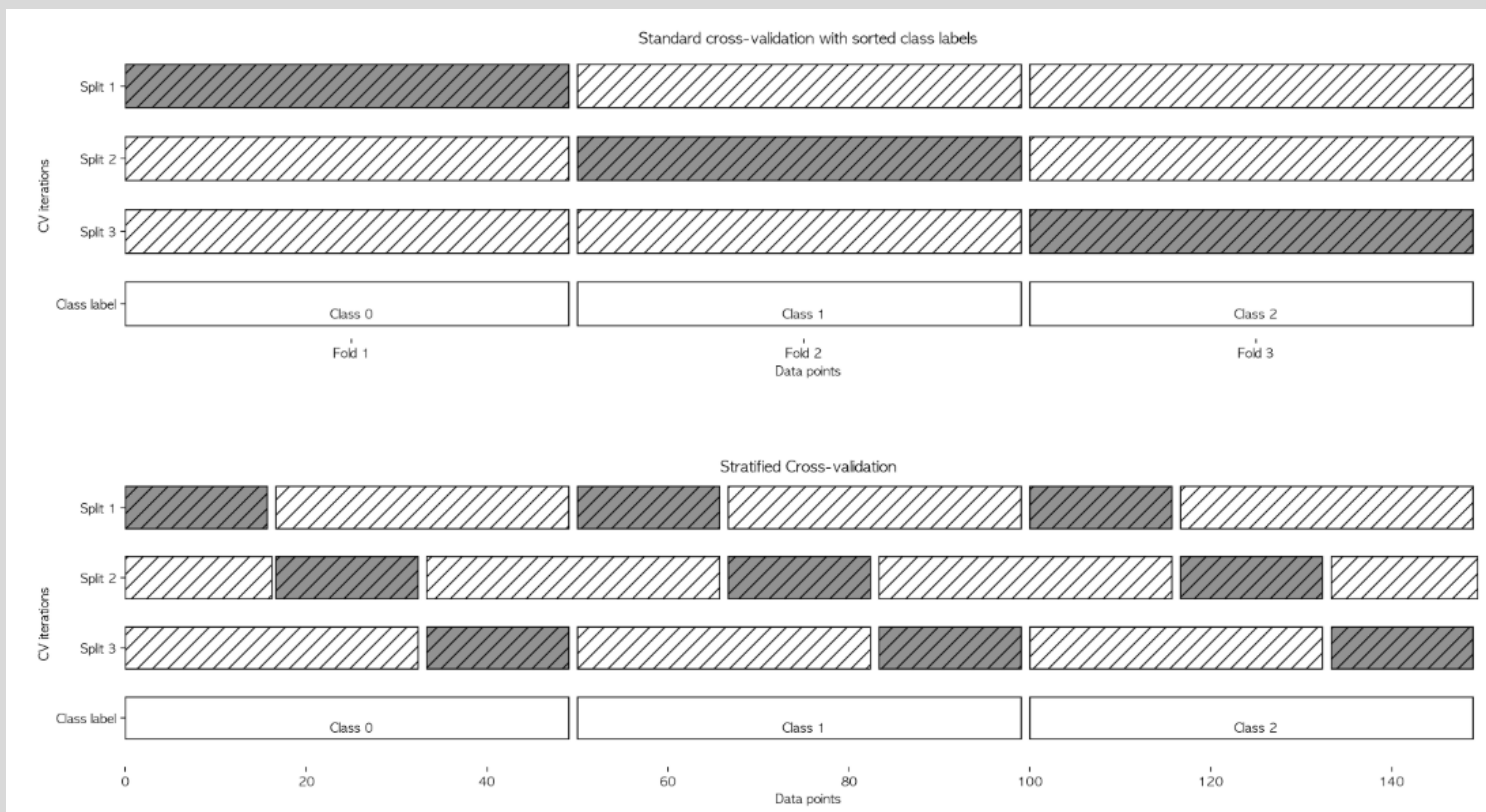


## Stratified K 폴드

불균형한 분포를 가진 레이블 데이터 집합을 위한 K 폴드 방식

(→ 특정 레이블 값이 특이하게 많거나 매우 적어서 값의 분포가 한쪽으로 치우침)

K 폴드가 레이블 데이터 집합이 원본 데이터 집합의 레이블 분포를 학습 및 테스트 세트에 제대로 분배하지 못하는 경우의 문제 해결



# split( ) 메서드에 인자로 피쳐 데이터 세트뿐만 아니라 레이블 데이터 세트도 반드시 필요

# 회귀에서는 stratified K 폴드가 지원x

교차 검증을 보다 간편하게 - `cross_val_score()`

1. 폴드 세트를 설정
2. for 루프에서 반복으로 학습 및 테스트 데이터의 인덱스를 추출
3. 반복적으로 학습과 예측을 수행하고 예측 성능 반환

⇒ `cross_val_score()`로 한 번에 해결!

```
cross_val_score(estimator, X, y=None, scoring=None, cv=None, n_jobs=1, verbose=0,  
fit_params=None, pre_dispatch='2*n_jobs')
```

- estimator, X(피쳐 데이터 세트), y(레이블 데이터 세트), scoring(예측 성능 평가 지표), cv(교차 검증 폴드 수)가 주요 파라미터

GridSearchCV – 교차 검증과 최적 하이퍼 파라미터 튜닝을 한 번에

촘촘하게 파라미터를 입력하면서 테스트를 하는 방식

Classifier나 Regressor와 같은 알고리즘에 사용되는 하이퍼 파라미터를 순차적으로 입력

⇒ 최적의 파라미터 도출 방안 제공

그러나, 순차적으로 파라미터를 테스트하므로 수행시간이 상대적으로 오래 걸림

- estimator : classifier, regressor, pipeline이 사용될 수 있음
- param\_grid : key+리스트 값을 가지는 딕셔너리가 주어짐. Estimator의 튜닝을 위해 파라미터명과 사용될 여러 파라미터 값을 지정함
- scoring : 예측 성능을 측정할 평가 방법을 지정
- cv : 교차 검증을 위해 분할되는 학습/테스트 세트의 개수를 지정
- refit : 디폴트는 True이며 이 때 가장 최적의 하이퍼 파라미터를 찾은 뒤 입력된 estimator 객체를 해당 하이퍼 파라미터로 재학습시킴

**Q&A**

**Thank you**