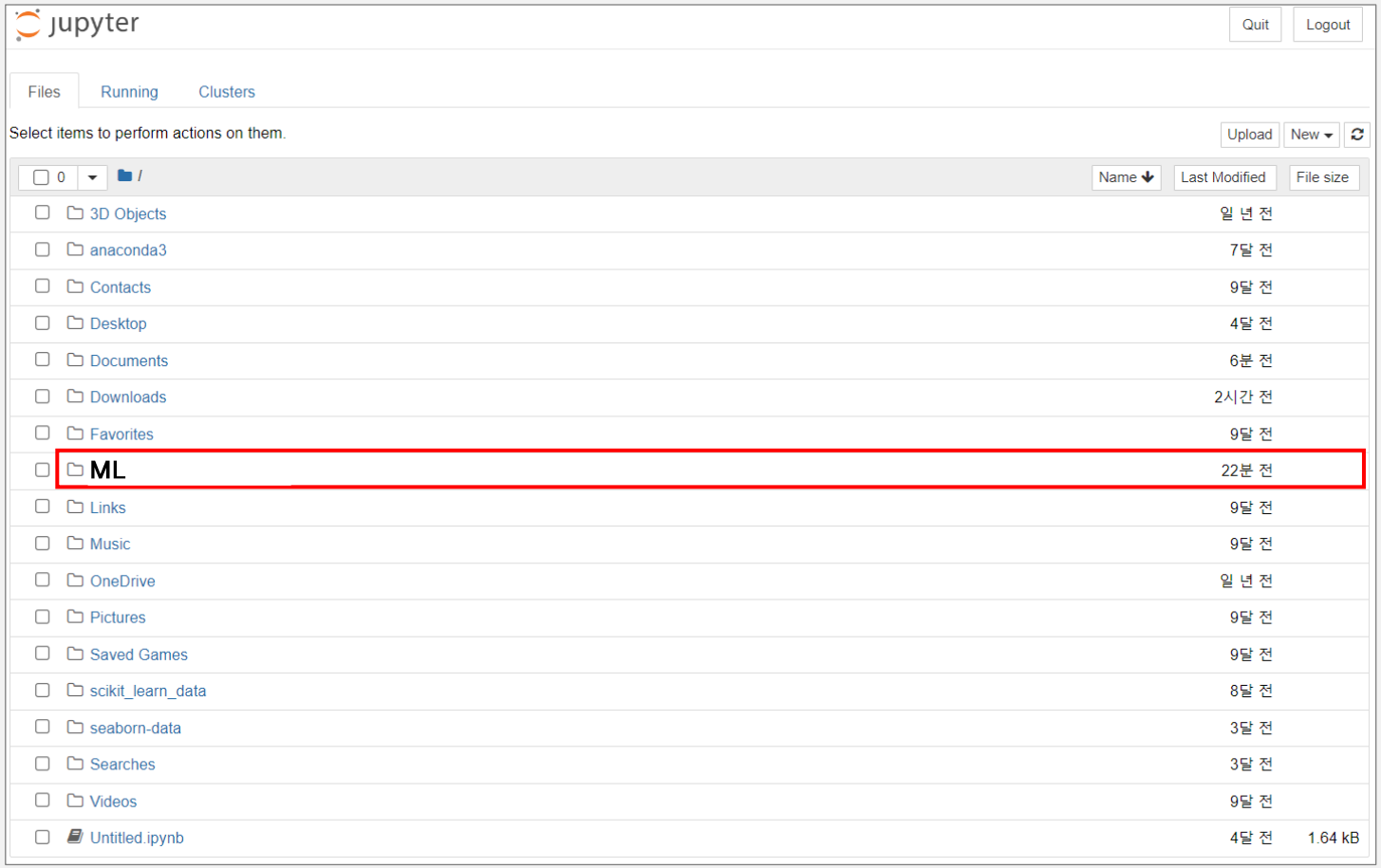

강원지역혁신플랫폼
기계학습
Machine Learning
Scikit-learn 모듈

학습목표

- Scikit-learn 모듈의 개념을 이해하고 구현할 수 있습니다.

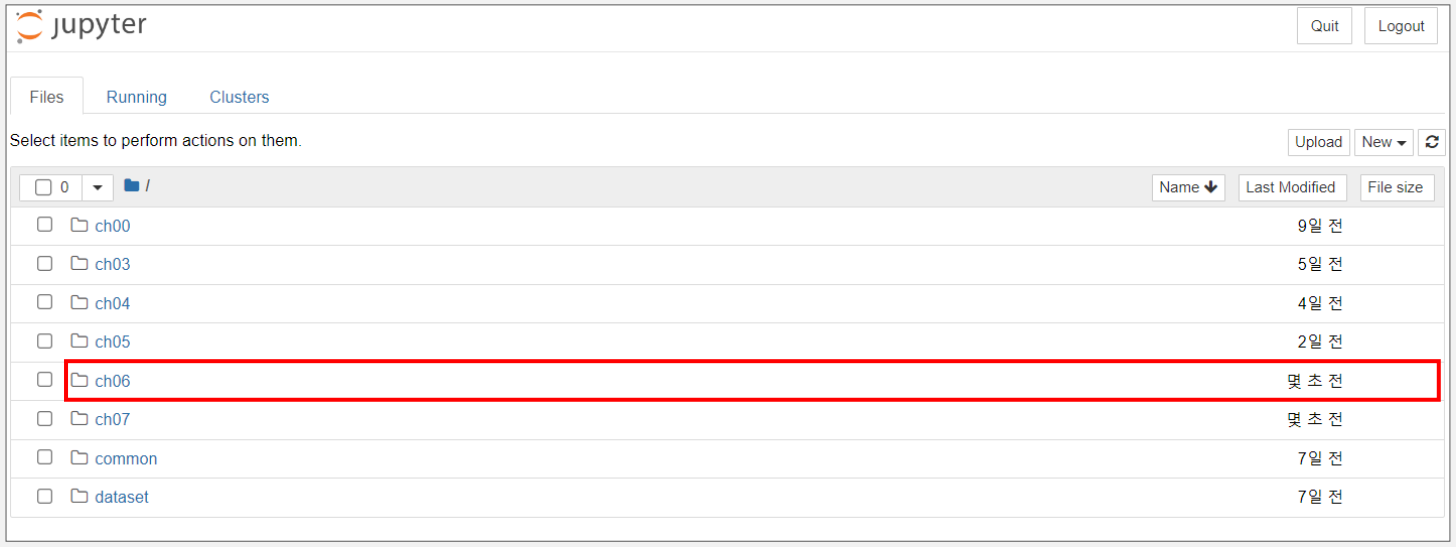
01 | ML 폴더

ML 폴더를 클릭하기



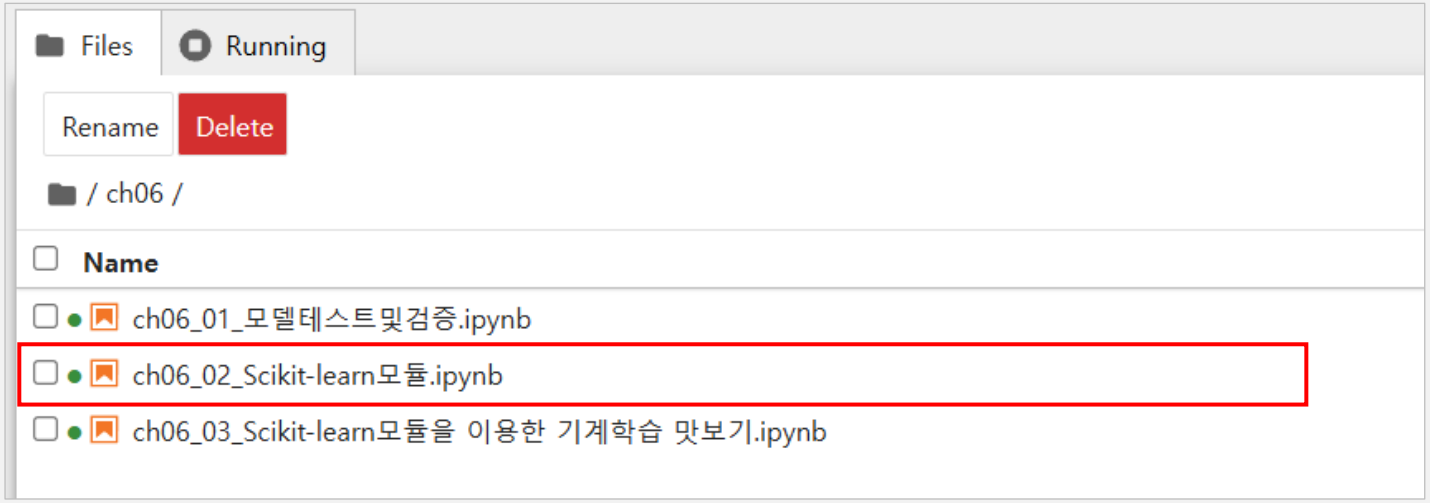
02 | ch06 폴더

ch06 폴더 클릭하기



03 | ch06_02_Scikit-learn모듈.ipynb

✦ ch06_02_Scikit-learn모듈.ipynb 파일 클릭하기



04 | Scikit-learn 모듈

Scikit-learn 개요

- 파이썬으로 구현된 가장 유명한 기계학습 오픈소스 라이브러리임
  - '사이킷런'이라고 부르기도 함
  - Scikit-learn은 오픈 소스로 공개되어 있으며, 개인, 비즈니스 관계없이 누구나 무료로 사용 가능함
  - 기계학습의 여러 가지 알고리즘 및 데이터 처리 기법을 쉽고 빠르게 적용해보고 최상의 결과를 얻을 수 있음
  - Scikit-learn은 현재도 활발하게 개발이 이루어지고 있음
  - 인터넷 상에서 정보를 찾기에도 쉬움
  - 샘플 데이터 셋이 부속되어 있음

04 | Scikit-learn 모듈

Scikit-learn의 특징은 다음과 같음

- 파사드 디자인 패턴을 적용하여 라이브러리 인터페이스 통일
- 다양한 기계학습 알고리즘, 모델 선택 및 데이터 전처리 기능 탑재
- Numpy를 기반으로 개발되어 속도 최적화

- 다른 라이브러리와의 호환성이 좋음
- GPU는 지원하지 않음

04 | Scikit-learn 모듈

- Scikit-learn은 고수준의 API로 직관적이고 사용하기 쉬운 인터페이스가 특징임
  - estimator, fit, predict, transform 네가지 개념만 익히면 금방 적응이 가능함

05 | Scikit-learn 데이터셋

Scikit-learn 데이터셋

- Scikit-learn의 서브패키지인 sklearn.datasets는 실습을 위한 샘플용 데이터셋을 제공함
  - 샘플용 데이터셋의 접근 방법은 다음과 같음
    - 기본적으로 Scikit-learn 패키지 안에 내장되어 있는 형태
      - load 명령으로 import
    - 인터넷에서 다운로드하여 사용하는 형태
      - fetch 명령으로 import
    - 새로운 데이터셋을 생성시켜 사용하는 형태
      - make 명령으로 생성

06 | Scikit-learn 모듈

load 계열: scikit-learn 패키지에 포함된 데이터

✦ load 계열 데이터셋 종류는 다음과 같음

데이터 함수	내용	구분
load_boston	1978년 보스톤 집 가격	regression
load_iris	붓꽃(iris) 유형	classification
load_diabetes	442명의 당뇨병 환자의 데이터	regression
load_digits	0~9까지의 숫자 필기 이미지 데이터	classification
load_linnerud	운동 능력 데이터	regression
load_wine	와인 등급 데이터	classification
load_breast_cancer	위스콘신 유방암 진단 데이터	classification

06 | Scikit-learn 모듈

- fetch 계열: 인터넷에서 다운로드하여 실행되는 대량의 데이터
  - fetch 계열 데이터셋 종류는 다음과 같음
    - fetch_california_housing: 캘리포니아 집값(회귀 분석용)
    - fetch_covtype: 토지 조사(회귀 분석용)
    - fetch_20newsgroups: 뉴스 그룹 텍스트 자료
    - fetch_livetti_faces: 얼굴 이미지
    - fetch_lfw_people: 유명인 얼굴 이미지
    - fetch_lfw_pairs: 유명인 얼굴 이미지
    - fetch_kddcup99: Kddcup 99 TCP dump
    - fetch_rcv1: 로이터 뉴스 말뭉치

06 | Scikit-learn 모듈

- Make 계열: 새로운 데이터셋을 생성시켜 사용하는 형태
  - make 계열 데이터셋 종류는 다음과 같음
    - make_regression(): regression용 데이터 생성
    - make_classification(): classification용 데이터 생성
    - make_blobs(): clustering용 데이터 생성

07 | Scikit-learn 데이터셋 객체

Sciket-learn 데이터셋 객체

- 예를 들어 load_wine()함수를 이용해 와인 데이터를 쉽게 로드할 수 있으며, 로드된 데이터 셋의 결과는 sklearn.utils.Bunch 클래스 형태로 저장됨
  - sklearn.utils.Bunch 객체는 몇 가지의 key를 제공하며 이를 통해 데이터의 정보를 쉽게 확인할 수 있음
    - data: 독립 변수의 ndarray 배열 형태
    - target: 종속 변수의 ndarray 배열 형태
    - feature_names: 독립 변수 이름의 리스트 형태
    - target_names: 종속 변수 이름의 리스트 형태
    - DESCR: 데이터에 대한 설명
    - filename: 데이터가 저장된 로컬 주소

07 | Scikit-learn 데이터셋 객체

- 유방암 진단 데이터 셋(breast cancer dataset)
  - 유방암 진단 데이터 셋은 **유방암 진단 사진**으로부터 **측정한 종양의 특징값**을 사용하여 종양이 **양성**(benign)인지 **악성**(malignant)인지를 **판별**함
    - 유방암 진단 데이터 셋은 다음과 같이 구성되어 있음
      - 관측치 569개
      - 30개의 독립변수, 1개의 종속 변수로 구성

07 | Scikit-learn 데이터셋 객체

다음은 유방암 진단 데이터셋을 읽어오는 코드이다.

- sklearn.utils.Bunch 객체는 몇 가지의 key를 제공하며 이를 통해 데이터의 정보를 쉽게 확인할 수 있음

```
cancer = load_breast_cancer()
print(cancer.keys())
```

```
dict_keys(['data', 'target', 'frame', 'target_names', 'DESCR', 'feature_names', 'filename', 'data_module'])
```

07 | Scikit-learn 데이터셋 객체

다음은 유방암 진단 데이터셋 객체 cancer에서 종속 변수 레이블과 독립 변수 이름을 출력하는 코드이다.

- 종속 변수의 레이블은 malignant(악성), benign(양성)인 것을 볼 수 있음

```
# 종속 변수 레이블, 독립 변수 이름 출력
print(cancer.target_names)
print(cancer.feature_names)
```

```
['malignant' 'benign']
['mean radius' 'mean texture' 'mean perimeter' 'mean area'
 'mean smoothness' 'mean compactness' 'mean concavity'
 'mean concave points' 'mean symmetry' 'mean fractal dimension'
 'radius error' 'texture error' 'perimeter error' 'area error'
 'smoothness error' 'compactness error' 'concavity error'
 'concave points error' 'symmetry error' 'fractal dimension error'
 'worst radius' 'worst texture' 'worst perimeter' 'worst area'
 'worst smoothness' 'worst compactness' 'worst concavity'
 'worst concave points' 'worst symmetry' 'worst fractal dimension']
```

07 | Scikit-learn 데이터셋 객체

다음은 유방암 진단 데이터셋을 데이터프레임으로 변환하는 코드이다.

- 데이터 형상은 (569, 31)인 것을 볼 수 있음

```
cancer_feature = pd.DataFrame(cancer.data, columns=cancer.feature_names)  # 독립변수
cancer_target = pd.Series(cancer.target, dtype="category")                # 종속변수
cancer_target = cancer_target.cat.rename_categories(cancer.target_names)  # 종속변수 속성의 값을
target_names으로 변경  df_cancer = pd.concat([cancer_feature, cancer_target], axis=1)  # 열로 병합
df_cancer.rename({0:"target"}, axis=1, inplace=True)          # target: 양성(benign), 악성(malignant)
print(df_cancer.shape)                                        # (569, 31)
df_cancer
```

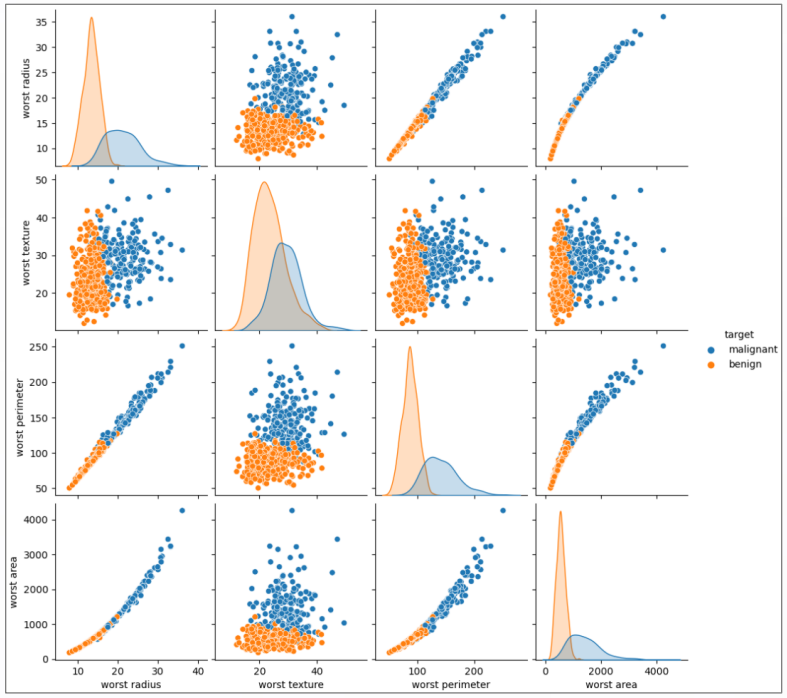
mean concavity	mean concave points	mean symmetry	mean fractal dimension	...	worst texture	worst perimeter	worst area	worst smoothness	worst compactness	worst concavity	worst concave points	worst symmetry	worst fractal dimension	target
0.30010	0.14710	0.2419	0.07871	...	17.33	184.60	2019.0	0.16220	0.66560	0.7119	0.2654	0.4601	0.11890	malignant
0.08690	0.07017	0.1812	0.05667	...	23.41	158.80	1956.0	0.12380	0.18660	0.2416	0.1860	0.2750	0.08902	malignant
0.19740	0.12790	0.2069	0.05999	...	25.53	152.50	1709.0	0.14440	0.42450	0.4504	0.2430	0.3613	0.08758	malignant
0.24140	0.10520	0.2597	0.09744	...	26.50	98.87	567.7	0.20980	0.86630	0.6869	0.2575	0.6638	0.17300	malignant
0.19800	0.10430	0.1809	0.05883	...	16.67	152.20	1575.0	0.13740	0.20500	0.4000	0.1625	0.2364	0.07678	malignant
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
0.24390	0.13890	0.1726	0.05623	...	26.40	166.10	2027.0	0.14100	0.21130	0.4107	0.2216	0.2060	0.07115	malignant
0.14400	0.09791	0.1752	0.05533	...	38.25	155.00	1731.0	0.11660	0.19220	0.3215	0.1628	0.2572	0.06637	malignant
0.09251	0.05302	0.1590	0.05648	...	34.12	126.70	1124.0	0.11390	0.30940	0.3403	0.1418	0.2218	0.07820	malignant
0.35140	0.15200	0.2397	0.07016	...	39.42	184.60	1821.0	0.16500	0.86810	0.9387	0.2650	0.4087	0.12400	malignant
0.00000	0.00000	0.1587	0.05884	...	30.37	59.16	268.6	0.08996	0.06444	0.0000	0.0000	0.2871	0.07039	benign

07 | Scikit-learn 데이터셋 객체

다음은 유방암 진단 데이터셋의 일부 특징값으로 분포와 상관관계 및 산점도를 그리는 코드이다.

- 독립 변수: “worst radius”, “worst texture”, “worst perimeter”, “worst area”

```
sns.pairplot(vars=["worst radius", "worst texture", "worst perimeter", "worst area"],
             hue="target", data=df_cancer)
plt.show()
```



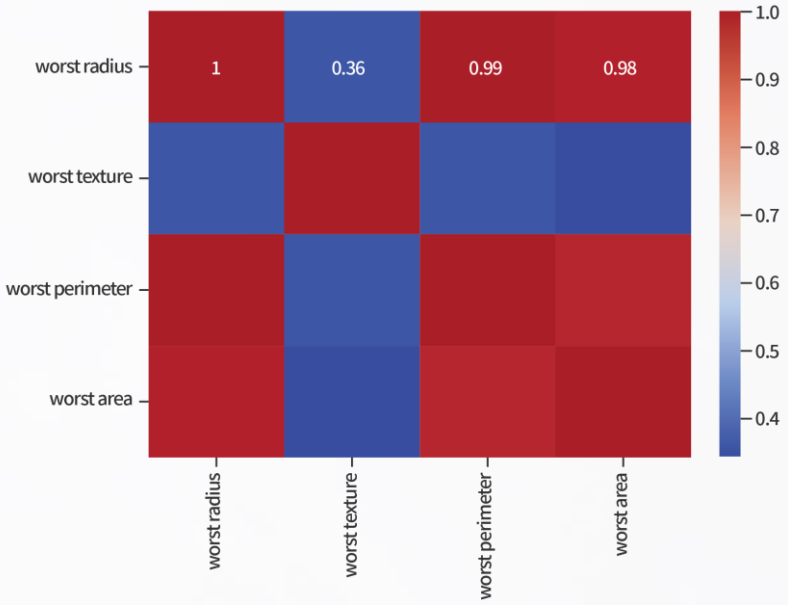
07 | Scikit-learn 데이터셋 객체

다음은 유방암 진단 데이터셋의 일부 특징값으로 상관계수와 히트맵을 그리는 코드이다.

- 독립 변수: “worst radius”, “worst texture”, “worst perimeter”, “worst area”

```
print(df_cancer.loc[:,["worst radius", "worst texture", "worst perimeter", "worst area"]].corr())
sns.heatmap(df_cancer.loc[:,["worst radius", "worst texture", "worst perimeter", "worst area"]].corr(),
            cmap="coolwarm", annot=True, annot_kws={"fontsize":8})
plt.tight_layout(); plt.show()
```

	worst radius	worst texture	worst perimeter	worst area
worst radius	1.000000	0.359921	0.993708	0.984015
worst texture	0.359921	1.000000	0.365098	0.345842
worst perimeter	0.993708	0.365098	1.000000	0.977578
worst area	0.984015	0.345842	0.977578	1.000000



07 | Scikit-learn 데이터셋 객체

다음은 유방암 진단 데이터셋에서 계층별 랜덤(StratifiedShuffleSplit)하게 훈련 데이터와 테스트 데이터를 7:3 비율로 추출하는 코드이다.

- 실행결과 훈련 데이터와 테스트 데이터를 7:3 비율로 분리하고, 계층별 비율도 7:3으로 분리된 것을 볼 수 있음

```
sfld = StratifiedShuffleSplit(n_splits=1, test_size=0.3, random_state=0)
cv_accuracy=[]        # CV별 정확도 저장
n_iter = 0            # 반복횟수
for train_index, test_index in sfld.split(df_cancer.iloc[:,:-1],df_cancer['target']):
    print(train_index.shape, test_index.shape)
# kfold.split()으로 반환된 인덱스를 이용해 학습용, 검증용 테스트 데이터 추출
X_train = cancer_feature.iloc[train_index]
X_test = cancer_feature.iloc[test_index]
y_train = df_cancer['target'].iloc[train_index]
y_test = df_cancer['target'].iloc[test_index]
n_iter += 1                           # 반복횟수
label_train = df_cancer['target'].iloc[train_index]
label_test = df_cancer['target'].iloc[test_index]   print("n_iter=",n_iter,"\n",count_frequency(label_train),
count_frequency(label_test))
print('-------------------------------------------------------------------------------')
```

```
(398,) (171,)
n_iter= 1
 [('benign', 250), ('malignant', 148)] [('benign', 107), ('malignant', 64)]
--------------------------------------------------------------------------------
```

08 | Scikit-learn 모듈: estimate, fit, predict, transform

estimator

- 학습 데이터에 기반해 모델을 적합시키고 새로운 데이터의 어떤 특성을 추론할 수 있는 객체를 estimator라고 지칭함
  - estimator는 fit 메서드를 가지고 있음
    - scikit-learn이 제공하는 모든 기계학습 알고리즘은 estimator이며 클래스로 구현되어 있음
      - 모델의 성능을 검증하는 cross_val_score()함수나 하이퍼파라미터 튜닝을 지원하는 GridSearchCV 같은 경우 이 estimator를 인자로 받아 내부에서 fit을 실행함

08 | Scikit-learn 모듈: estimate, fit, predict, transform

- 다음은 의사결정나무 모델을 생성하는 DecisionTreeClassifier클래스 Estimator객체 및 하이퍼파라미터를 설정하는 코드이다.
  - 아래와 같이 의사결정나무 모델의 estimator객체 model이 생성됨

```
model = DecisionTreeClassifier(criterion='entropy')
model
```

08 | Scikit-learn 모듈: estimate, fit, predict, transform

fit

- 인스턴스화된 estimator에서 fit 메서드를 이용해서 학습시킴
  - 지도학습 알고리즘은 학습 데이터와 레이블 데이터를 함께 인자로 전달함
  - 비지도학습 알고리즘은 학습 데이터만 인자로 전달함

08 | Scikit-learn 모듈: estimate, fit, predict, transform

- 다음은 DecisionTreeClassifier 클래스 Estimator 객체 model의 fit 메서드로 학습을 수행하는 코드이다.
  - 아래와 같이 의사결정나무 모델은 지도 학습이므로 fit 메서드로 학습시킬 때 훈련 데이터와 레이블 데이터가 함께 전달된 것을 볼 수 있음

```
model.fit(X_train, y_train)          # fit 메서드를 이용한 학습
```

08 | Scikit-learn 모듈: estimate, fit, predict, transform

predict

- 입력 데이터에 대한 모델의 예측 결과를 반환함

- 다음은 predict 메서드로 모델의 예측 결과를 반환하는 코드이다.
  - 아래와 같이 모델의 예측 결과가 반환 것을 볼 수 있음
    - 모델의 예측 결과 양성(benign)이 101개, 악성(malignant)이 70개인 것을 볼 수 있음

```
y_pred = model.predict(X_test)
print(count_frequency(y_pred))
print(y_pred[:5])
```

```
[('benign', 101), ('malignant', 70)]
['benign' 'benign' 'benign' 'malignant' 'benign']
```

08 | Scikit-learn 모듈: estimate, fit, predict, transform

transform

- 피처를 처리하는 기능은 transform메서드로 실행되고 결과를 반환함
  - 훈련 데이터(train data)로 부터 학습하고, 테스트 데이터에 적용하기 위해 transform()메서드를 사용함

08 | Scikit-learn 모듈: estimate, fit, predict, transform

- 다음은 표준화 함수 StandardScaler객체를 이용해 transform()메서드로 훈련 데이터를 학습하고, 테스트 데이터를 변환하는 코드이다.
  - 즉, 훈련 데이터에서 평균과 분산 값을 학습함

```
# 전처리 - 스케일링 적용
scaler = StandardScaler()                    # 객체 생성
scaler.fit(X_train)                          # 학습
X_train_ss = scaler.transform(X_train)  # 변환
X_test_ss = scaler.transform(X_test)    # 변환
```

08 | Scikit-learn 모듈: estimate, fit, predict, transform

- fit과 transform을 하나로 결합한 fit_transform()메서드를 제공함
  - 여기서 명심할 것은 fit_transform()메서드는 훈련 데이터(train data)에서만 사용됨
    - 즉, test data에는 사용하지 않음
  - 우리가 만든 모델은 훈련 데이터에 있는 평균과 분산을 학습하게 됨
    - 이렇게 학습된 Scaler()의 파라미터(parameter)는 테스트 데이터를 스케일(scale)하는데 사용됨
    - 다시 말해, 훈련 데이터로 학습된 Scaler()의 파라미터를 통해 테스트 데이터의 변수 값들이 스케일 되는 것임

08 | Scikit-learn 모듈: estimate, fit, predict, transform

- 다음은 표준화 함수 StandardScaler 객체를 이용해 fit_transform()메서드로 훈련 데이터를 학습 및 변환하는 코드이다.
  - 즉, 훈련 데이터에서 평균과 분산 값을 학습해서 훈련 데이터를 변환함

```
# 전처리 - 스케일링 적용
X_train_ss = StandardScaler().fit_transform(X_train)
```

08 | Scikit-learn 모듈: estimate, fit, predict, transform

- 기계학습의 메커니즘을 이해한다면 fit_transform()메서드와 transform()메서드의 차이를 보다 잘 이해할 수 있을 것임
  - 우리는 훈련 데이터를 통해 데이터의 패턴을 학습함
    - 테스트 데이터를 통해 처음 보는 데이터에 대해서도 일반화된 성능을 얻길 원함
  - 모델링을 할 때도 훈련 데이터로 모델의 파라미터를 학습시킴
    - 테스트 데이터에 대해서는 훈련 데이터로 학습된 모델의 성능을 측정하길 원하는 것임

08 | Scikit-learn 모듈: estimate, fit, predict, transform

- fit_transform()메서드는 학습을 위한 훈련 데이터에 사용됨
  - 테스트 데이터에서는 transform()메서드만 사용해줘야 함
  - 만약, 테스트 데이터에 대해서도 fit_transform()메서드를 사용하게 된다면 모델은 테스트 데이터에 대해서도 학습을 하게 되는 꼴이 됨

09 | Scikit-learn 주요 모듈

Scikit-learn 주요 모듈

분류	모듈명	내장 기능
예제 데이터	sklearn.datasets	• 연습용 데이터셋
피처 처리	sklearn.preprocessing	• 전처리 관련 기법(원핫 인코딩, 정규화, 스케일링 등)
	sklearn.feature_selection	• 모델에 중요한 영향을 미치는 피처를 탐색 및 선택하는 기법
	sklearn.feature_extraction	• 원시 데이터로부터 피처를 추출하는 기능 • 이미지에 대한 피처 추출은 하위 모듈 image, 텍스트 데이터의 피처 추출은 하위 모듈 text에 지원 API가 있음
차원 축소	sklearn.decomposition	• 차원 축소 관련 알고리즘 계열(PCA, NMF, Truncated SVD 등)
검증, 하이퍼 파라미터 튜닝, 데이터 분리	sklearn.model_selection	• 검증, 하이퍼 파라미터 튜닝, 데이터 분리 등(cross_validate, GridSearchCV, train_test_split, learning_curve 등)
모델 평가	sklearn.metrics	• 모델의 성능을 측정 및 평가하는 기법(accuracy, precision, recall, ROC curve 등)
기계학습 알고리즘	sklearn.ensemble	• 앙상블 알고리즘 계열(랜덤 포레스트, 에이다 부스트, 배깅 등)
	sklearn.linear_model	• 선형 알고리즘 계열(선형회귀, 로지스틱 회귀, SGD 등)
	sklearn.naïve_bayes	• 나이브 베이즈 알고리즘 계열(베르누이 NB, 가우시안 NB, 다항분포 NB 등)

09 | Scikit-learn 주요 모듈

분류	모듈명	내장 기능
기계학습 알고리즘	sklearn.neighbors	• 최근접 이웃 알고리즘 계열(K-NN 등)
	sklearn.svm	• Support Vector Machine 계열 알고리즘
	sklearn.tree	• 의사 결정 나무 계열 알고리즘
	sklearn.cluster	• 비지도 학습(클러스터링) 알고리즘(k-Means, DBSCAN 등)
유틸리티	sklearn.pipeline	• 피처 처리 등의 변환과 기계학습 알고리즘 등을 연쇄적으로 실행하는 기능