

머신러닝 강의자료

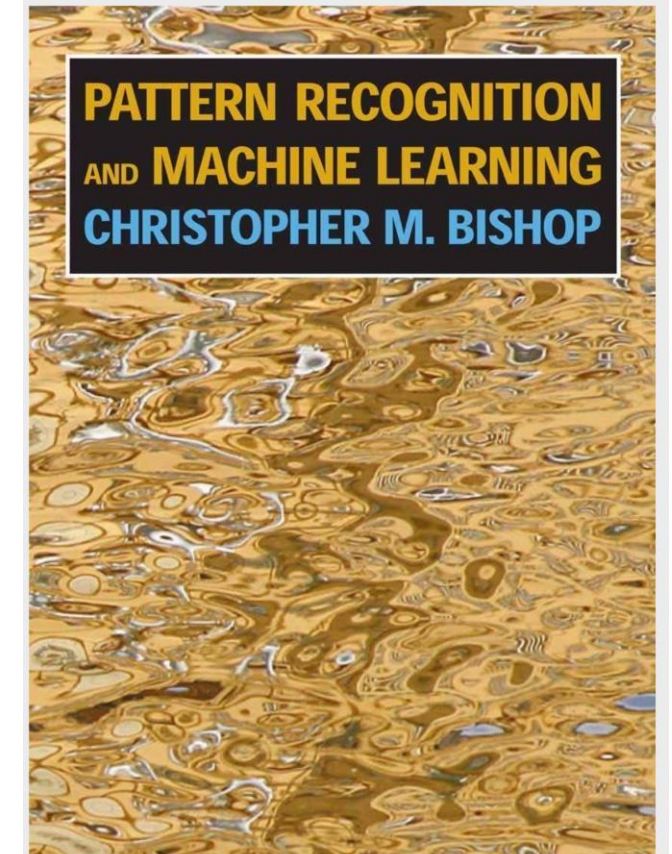
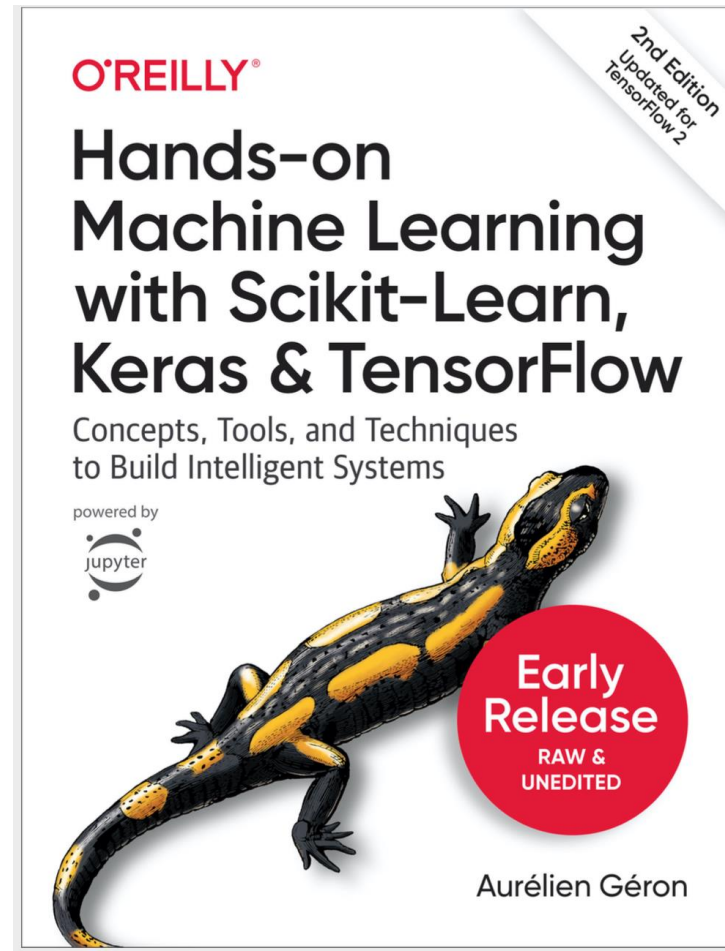
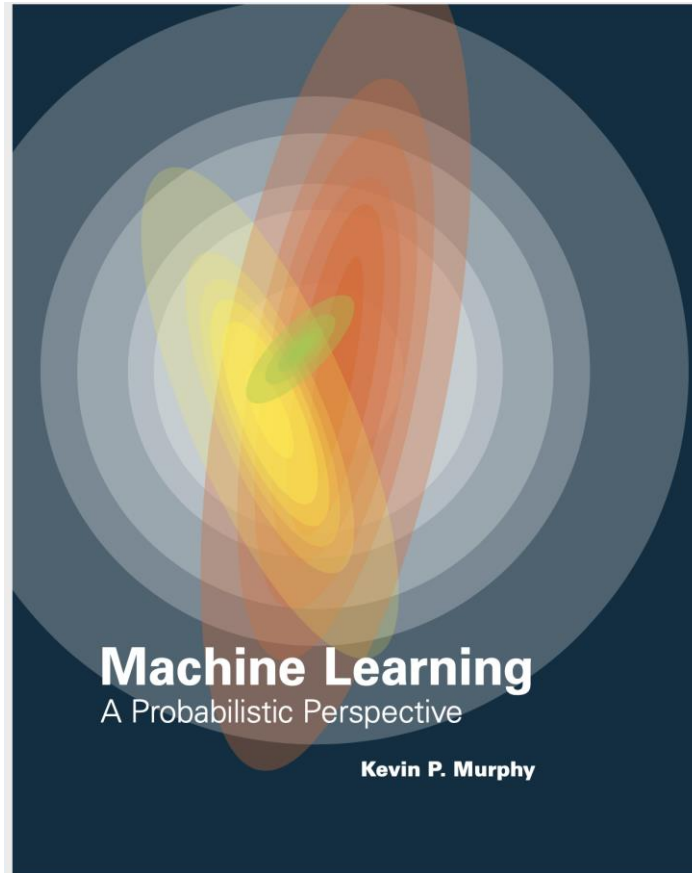
3강 - 분류(Classification)

닥터윌컨설팅 딥러닝 R&D 책임연구원
고려대학교 인공지능대학원 박사과정

류회성(Hoe Sung Ryu)



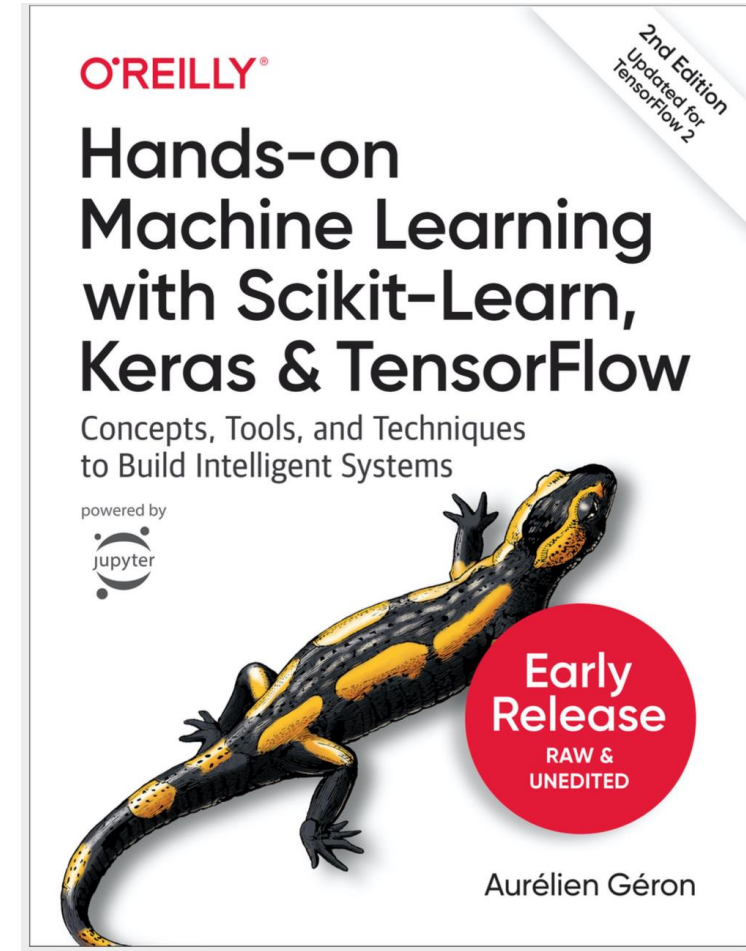
들어가기 앞서



강의내용

● 분류

- 분류 데이터셋
- 이진 분류기 훈련
- 성능 측정
- 다중 분류
- 에러 분석
- 다중 레이블 분류
- 다중 출력 분류



[지난 시간 복습]

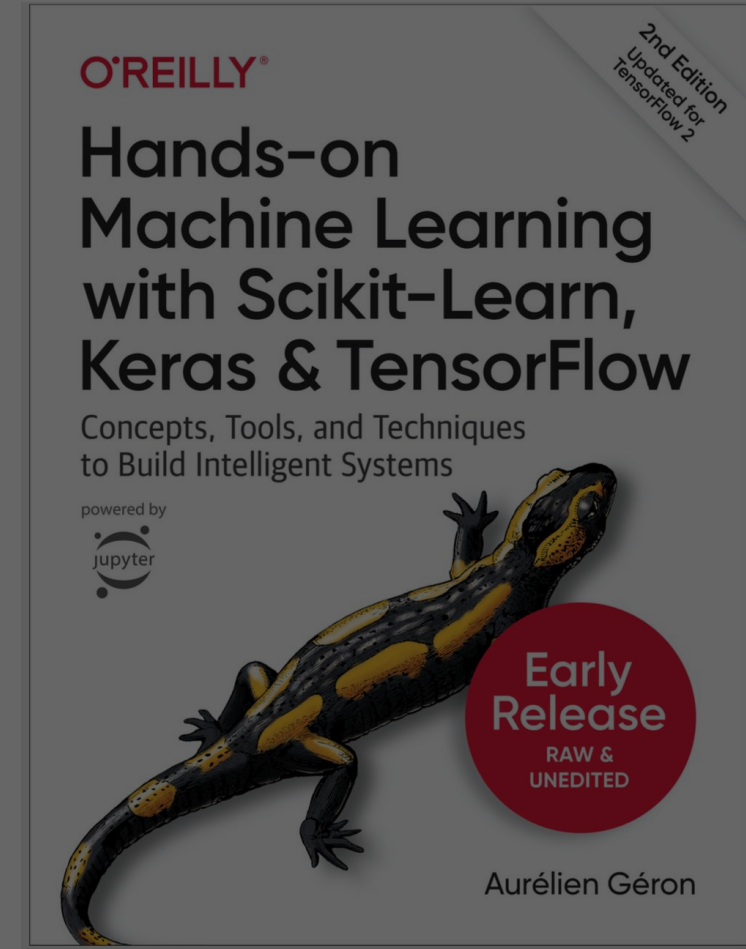
지난 시간 복습

● 머신러닝 개요


- 왜 머신러닝을 사용하는가?
- 머신러닝의 종류
- 머신러닝의 주요 도전 과제
- 테스트와 검증

● 머신러닝 파이프라인

- 데이터 가져오기
- 탐색적 데이터 분석
- 데이터 전처리
- 피처 엔지니어링
- 피처 선택
- 모델링
- 평가



머신러닝 파이프라인


[Install](#)
[User Guide](#)
[API](#)
[Examples](#)
[More](#)

scikit-learn

Machine Learning in Python

[Getting Started](#)
[Release Highlights for 0.23](#)
[GitHub](#)

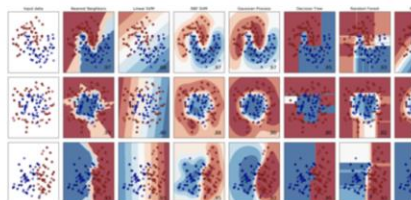
- Simple and efficient tools for predictive data analysis
- Accessible to everybody, and reusable in various contexts
- Built on NumPy, SciPy, and matplotlib
- Open source, commercially usable - BSD license

Classification

Identifying which category an object belongs to.

Applications: Spam detection, image recognition.

Algorithms: SVM, nearest neighbors, random forest, and more...



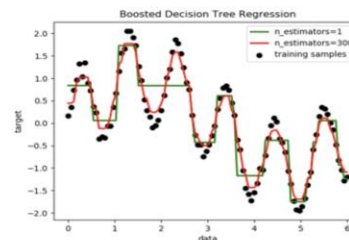
Examples

Regression

Predicting a continuous-valued attribute associated with an object.

Applications: Drug response, Stock prices.

Algorithms: SVR, nearest neighbors, random forest, and more...



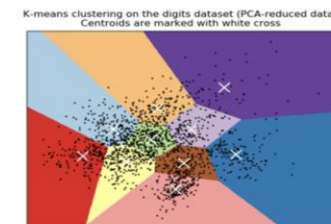
Examples

Clustering

Automatic grouping of similar objects into sets.

Applications: Customer segmentation, Grouping experiment outcomes

Algorithms: k-Means, spectral clustering, mean-shift, and more...



Examples

Dimensionality reduction

Reducing the number of random variables to consider.

Applications: Visualization, Increased efficiency

Model selection

Comparing, validating and choosing parameters and models.

Applications: Improved accuracy via parameter

Preprocessing

Feature extraction and normalization.

Applications: Transforming input data such as text for use with machine learning algorithms.

머신러닝 파이프라인

분류	모듈명	설명
예제 데이터	<code>sklearn.datasets</code>	Scikit-learn에 내장된 예제 데이터셋
피쳐 처리	<code>sklearn.preprocessing</code>	데이터 전처리에 필요한 다양한 기능 제공(인코딩, 정규화, 스케일링 등)
	<code>sklearn.feature_selection</code>	알고리즘에 영향을 미치는 특성을 우선순위로 선택 작업을 수행하는 다양한 기능 제공
	<code>sklearn.feature_extraction</code>	텍스트 데이터나 이미지 데이터의 벡터화된 피쳐를 추출하는데 사용됨.
	<code>sklearn.decomposition</code>	차원 축소와 관련된 알고리즘 지원 (PCA, NMF 등)
데이터 분리, 검증 & 파라미터 튜닝	<code>sklearn.model_selection</code>	교차 검증을 위한 훈련용/테스트용 데이터 분리, GridSearch 기능 등 제공
평가	<code>sklearn.metrics</code>	각종 머신 러닝 알고리즘(회귀, 분류, 클러스터링 등)의 다양한 성능 측정 방법 제공 (accuracy, precision, recall, ROC-AUC 곡선 등)
머신 러닝 알고리즘	<code>sklearn.ensemble</code>	앙상블 알고리즘 제공(Random Forest, AdaBoost, Gradient boosting, 등)
	<code>sklearn.linear_model</code>	선형회귀, 릿지, 라쏘 및 로지스틱 회귀 등 회귀 관련 알고리즘 지원
	<code>sklearn.naïve_bayes</code>	나이브 베이즈 알고리즘 제공. 가우시안 NB, 다항 분포 NB 등
	<code>sklearn.neighbors</code>	KNN 알고리즘 제공
	<code>sklearn.svm</code>	SVM 알고리즘 제공
	<code>sklearn.tree</code>	의사 결정 트리 알고리즘 제공
	<code>sklearn.cluster</code>	클러스터링 알고리즘 제공 (k-means, DBScan 등)
유틸리티 (지원 기능)	<code>sklearn.pipeline</code>	피쳐 처리 등의 변환과 ML 알고리즘 학습, 예측 등을 함께 묶어서 실행할 수 있는 유틸리티 제공

머신러닝 파이프라인 - 데이터셋 모듈

- 별도의 외부 웹사이트에서 데이터 세트를 내려 받을 필요 없이
- 예제로 활용 가능한 데이터셋 혹은 데이터 생성 기능 제공

<code>datasets.clear_data_home([data_home])</code>	Delete all the content of the data home cache.
<code>datasets.dump_svmlight_file(X, y, f[, ...])</code>	Dump the dataset in svmlight / libsvm file format.
<code>datasets.fetch_20newsgroups([data_home, ...])</code>	Load the filenames and data from the 20 newsgroups dataset (classification).
<code>datasets.fetch_20newsgroups_vectorized([...])</code>	Load the 20 newsgroups dataset and vectorize it into token counts (classification).
<code>datasets.fetch_california_housing([...])</code>	Load the California housing dataset (regression).
<code>datasets.fetch_covtype([data_home, ...])</code>	Load the covtype dataset (classification).
<code>datasets.fetch_kddcup99([subset, data_home, ...])</code>	Load the kddcup99 dataset (classification).
<code>datasets.fetch_lfw_pairs([subset, ...])</code>	Load the Labeled Faces in the Wild (LFW) pairs dataset (classification).
<code>datasets.fetch_lfw_people([data_home, ...])</code>	Load the Labeled Faces in the Wild (LFW) people dataset (classification).
<code>datasets.fetch_olivetti_faces([data_home, ...])</code>	Load the Olivetti faces data-set from AT&T (classification).
<code>datasets.fetch_openml([name, version, ...])</code>	Fetch dataset from openml by name or dataset id.
<code>datasets.fetch_rcv1([data_home, subset, ...])</code>	Load the RCV1 multilabel dataset (classification).
<code>datasets.fetch_species_distributions([...])</code>	Loader for species distribution dataset from Phillips et
<code>datasets.get_data_home([data_home])</code>	Return the path of the scikit-learn data dir.
<code>datasets.load_boston([return_X_y])</code>	Load and return the boston house-prices dataset (regression).
<code>datasets.load_breast_cancer([return_X_y])</code>	Load and return the breast cancer wisconsin dataset (classification).
<code>datasets.load_diabetes([return_X_y])</code>	Load and return the diabetes dataset (regression).
<code>datasets.load_digits([n_class, return_X_y])</code>	Load and return the digits dataset (classification).
<code>datasets.load_files(container_path[, ...])</code>	Load text files with categories as subfolder names.
<code>datasets.load_iris([return_X_y])</code>	Load and return the iris dataset (classification).
<code>datasets.load_linnerud([return_X_y])</code>	Load and return the linnerud dataset (multivariate regression).
<code>datasets.load_sample_image(image_name)</code>	Load the numpy array of a single sample image
<code>datasets.load_sample_images()</code>	Load sample images for image manipulation.
<code>datasets.load_svmlight_file(f[, n_features, ...])</code>	Load datasets in the svmlight / libsvm format into sparse CSR matrix
<code>datasets.load_svmlight_files(files[, ...])</code>	Load dataset from multiple files in SVMLight format
<code>datasets.load_wine([return_X_y])</code>	Load and return the wine dataset (classification).

데이터의 크기가 커서 패
키지에 저장되어 있는 것
이 아니라 인터넷에서 다
운로드 받는 함수
(fetch_XXX)

분류/회귀용

샘플 데이터 (load_XXX)

머신러닝 파이프라인 - 전처리 모듈

- 데이터 품질을 향상 시키기 위한 다양한 기능 제공
- 인코딩(Encoding)
 - 범주형 feature를 수치형으로 변환
- 스케일링(scaling)
 - feature 들의 값 범위를 일정한 수준으로 맞추는 기능

분류	모듈명	설명
데이터 인코딩	LabelEncoder	레이블 인코딩 (카테고리 피처를 코드형 숫자값으로 변환) (e.g. 성별 피처의 값인 Male, Female을 각각 0과 1로 변환)
	OneHotEncoder	특성(피처)의 모든 값들을 새로운 특성으로 추가해 해당 값에 해당하는 칼럼에 만 1을 표시, 나머지는 0으로 표시하는 방식
피처 스케일링 /정규화	StandardScaler	피처의 값들이 가우시안 분포(평균 = 0, 분산 = 1)를 따르도록 변환
	MinMaxScaler	피처의 값들을 0과 1사이의 값으로 변환

[분류 데이터셋]

분류 데이터셋


- 붓꽃 데이터셋
- 유방암 데이터셋
- MNIST 데이터셋
- Fashion MNIST 데이터셋
- :

<code>datasets.clear_data_home([data_home])</code>	Delete all the content of the data home cache.
<code>datasets.dump_svmlight_file(X, y, f[, ...])</code>	Dump the dataset in svmlight / libsvm file format.
<code>datasets.fetch_20newsgroups([data_home, ...])</code>	Load the filenames and data from the 20 newsgroups dataset (classification).
<code>datasets.fetch_20newsgroups_vectorized([...])</code>	Load the 20 newsgroups dataset and vectorize it into token counts (classification).
<code>datasets.fetch_california_housing([...])</code>	Load the California housing dataset (regression).
<code>datasets.fetch_covtype([data_home, ...])</code>	Load the covtype dataset (classification).
<code>datasets.fetch_kddcup99([subset, data_home, ...])</code>	Load the kddcup99 dataset (classification).
<code>datasets.fetch_lfw_pairs([subset, ...])</code>	Load the Labeled Faces in the Wild (LFW) pairs dataset (classification).
<code>datasets.fetch_lfw_people([data_home, ...])</code>	Load the Labeled Faces in the Wild (LFW) people dataset (classification).
<code>datasets.fetch_olivetti_faces([data_home, ...])</code>	Load the Olivetti faces data-set from AT&T (classification).
<code>datasets.fetch_openml([name, version, ...])</code>	Fetch dataset from openml by name or dataset id.
<code>datasets.fetch_rcv1([data_home, subset, ...])</code>	Load the RCV1 multilabel dataset (classification).
<code>datasets.fetch_species_distributions([...])</code>	Loader for species distribution dataset from Phillips et
<code>datasets.get_data_home([data_home])</code>	Return the path of the scikit-learn data dir.
<code>datasets.load_boston([return_X_y])</code>	Load and return the boston house-prices dataset (regression).
<code>datasets.load_breast_cancer([return_X_y])</code>	Load and return the breast cancer wisconsin dataset (classification).
<code>datasets.load_diabetes([return_X_y])</code>	Load and return the diabetes dataset (regression).
<code>datasets.load_digits([n_class, return_X_y])</code>	Load and return the digits dataset (classification).
<code>datasets.load_files(container_path[, ...])</code>	Load text files with categories as subfolder names.
<code>datasets.load_iris([return_X_y])</code>	Load and return the iris dataset (classification).
<code>datasets.load_linnerud([return_X_y])</code>	Load and return the linnerud dataset (multivariate regression).
<code>datasets.load_sample_image(image_name)</code>	Load the numpy array of a single sample image
<code>datasets.load_sample_images()</code>	Load sample images for image manipulation.
<code>datasets.load_svmlight_file([f[, n_features, ...]])</code>	Load datasets in the svmlight / libsvm format into sparse CSR matrix
<code>datasets.load_svmlight_files(files[, ...])</code>	Load dataset from multiple files in SVMlight format
<code>datasets.load_wine([return_X_y])</code>	Load and return the wine dataset (classification).

데이터의 크기가 커서 패
키지에 저장되어 있는 것
이 아니라 인터넷에서 다
운로드 받는 함수
(fetch_XXX)

분류/회귀용
샘플 데이터 (load_XXX)

붓꽃 데이터셋


[Install](#)
[User Guide](#)
[API](#)
[Examples](#)
[More ▾](#)

[Prev](#)
[Up](#)
[Next](#)

scikit-learn 0.23.2
[Other versions](#)

Please [cite us](#) if you use the software.

[sklearn.datasets.load_iris](#)
[Examples using sklearn.datasets.load_iris](#)

sklearn.datasets.load_iris

```
sklearn.datasets.load_iris(*, return_X_y=False, as_frame=False)
```

[\[source\]](#)

Load and return the iris dataset (classification).

The iris dataset is a classic and very easy multi-class classification dataset.

Classes	3
Samples per class	50
Samples total	150
Dimensionality	4
Features	real, positive

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

Parameters:

return_X_y : bool, default=False.
 If True, returns (data, target) instead of a Bunch object. See below for more information about the data and target object.
New in version 0.18.

as_frame : bool, default=False
 If True, the data is a pandas DataFrame including columns with appropriate dtypes (numeric). The target is a pandas DataFrame or Series depending on the number of target columns. If return_X_y is True, then (data, target) will be pandas DataFrames or Series as described below.
New in version 0.23.

Returns:

data : Bunch
 Dictionary-like object, with the following attributes.

data : (ndarray, dataframe) of shape (150, 4)
 The data matrix. If as_frame=True, data will be a pandas DataFrame.

target: (ndarray, Series) of shape (150,)
 The classification target. If as_frame=True, target will be a pandas Series.

붓꽃 데이터셋

```
In [5]: 1 iris_df = pd.DataFrame(data=iris.data, columns=iris.feature_names)
        2 iris_df['label'] = iris.target
        3 iris_df
```


executed in 27ms, finished 00:00:12 2020-08-07

Out [5]:

	sepal length (cm)	sepal width (cm)	petal length (cm)	petal width (cm)	label
0	5.1	3.5	1.4	0.2	0
1	4.9	3.0	1.4	0.2	0
2	4.7	3.2	1.3	0.2	0
3	4.6	3.1	1.5	0.2	0
4	5.0	3.6	1.4	0.2	0
...
145	6.7	3.0	5.2	2.3	2
146	6.3	2.5	5.0	1.9	2
147	6.5	3.0	5.2	2.0	2
148	6.2	3.4	5.4	2.3	2
149	5.9	3.0	5.1	1.8	2

150 rows × 5 columns

유방암 데이터셋


[Install](#) [User Guide](#) [API](#) [Examples](#) [More ▾](#)

[Prev](#) [Up](#) [Next](#)

scikit-learn 0.23.2
[Other versions](#)

Please [cite us](#) if you use the software.

[sklearn.datasets.load_breast_cancer](#)
[Examples using sklearn.datasets.load_breast_cancer](#)

sklearn.datasets.load_breast_cancer

```
sklearn.datasets.load_breast_cancer(*, return_X_y=False, as_frame=False)
```

[\[source\]](#)

Load and return the breast cancer wisconsin dataset (classification).

The breast cancer dataset is a classic and very easy binary classification dataset.

Classes	2
Samples per class	212(M),357(B)
Samples total	569
Dimensionality	30
Features	real, positive

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

Parameters:

return_X_y : bool, default=False
 If True, returns (data, target) instead of a Bunch object. See below for more information about the data and target object.
New in version 0.18.

as_frame : bool, default=False
 If True, the data is a pandas DataFrame including columns with appropriate dtypes (numeric). The target is a pandas DataFrame or Series depending on the number of target columns. If return_X_y is True, then (data, target) will be pandas DataFrames or Series as described below.
New in version 0.23.

Returns:

data : Bunch
 Dictionary-like object, with the following attributes.

data : (ndarray, dataframe) of shape (569, 30)
 The data matrix. If as_frame=True, data will be a pandas DataFrame.

target: (ndarray, Series) of shape (569,)
 The classification target. If as_frame=True, target will be a pandas Series.

유방암 데이터셋

In [12]:

```
1 data_bc = load_breast_cancer()
2 data = pd.DataFrame(data=data_bc.data, columns=data_bc.feature_names)
3 data['label'] = data_bc.target
4 data
```

executed in 37ms, finished 00:02:58 2020-08-07

Out [12]:

mean ness	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	label
7760	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	0
7864	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	0
5990	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	0
8390	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	0
3280	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	0
...
1590	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	0
0340	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	0
0230	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	0
7700	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	0
4362	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	1

MNIST



- MNIST (Modified National Institute of Standards and Technology database)
 - 손으로 쓴 숫자들로 이루어진 대형 데이터베이스
 - 데이터베이스는 또한 기계 학습 분야의 트레이닝 및 테스트에 널리 사용
 - 정규화 및 스케일링 처리가 완료된 깔끔한 데이터셋

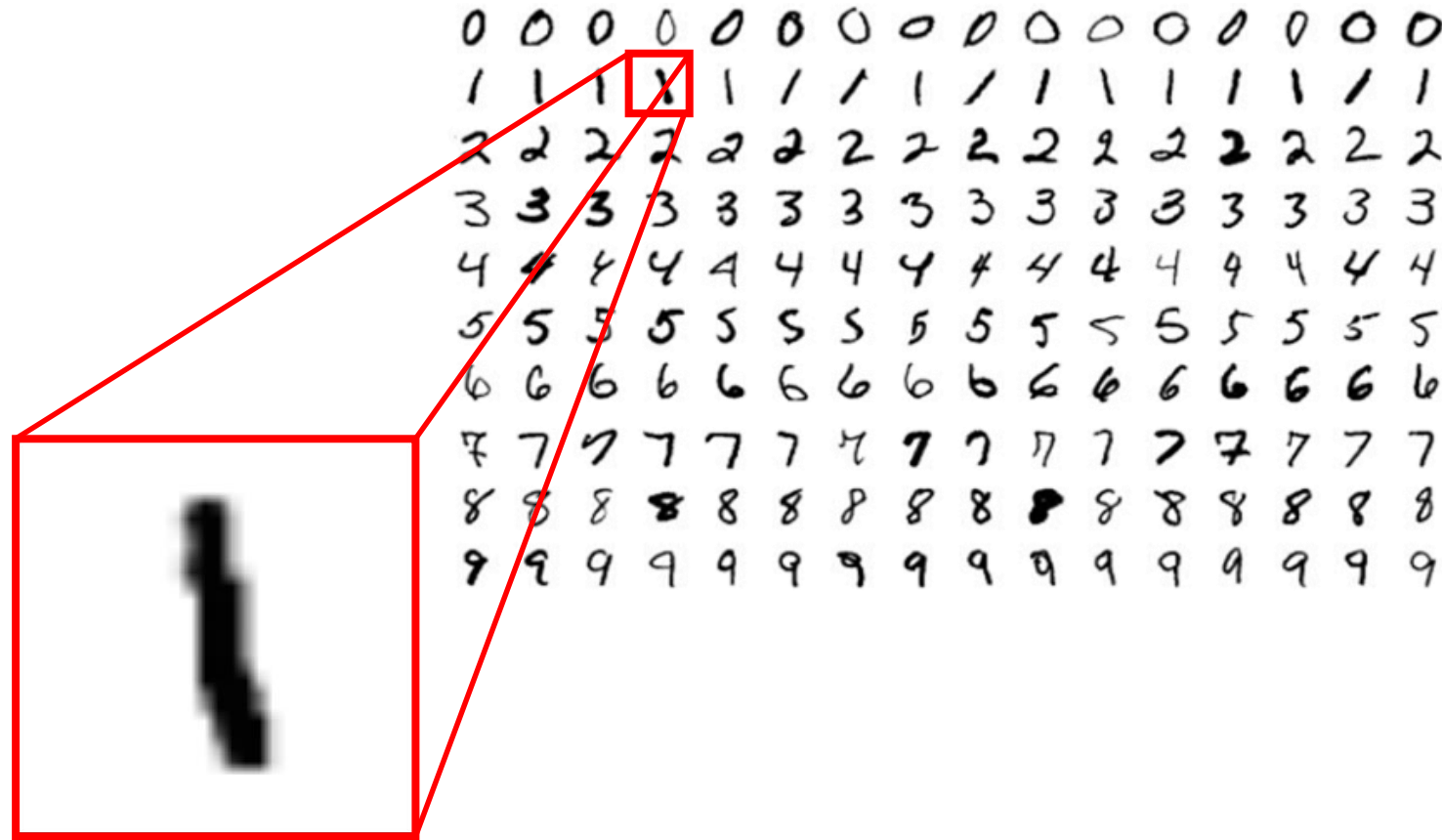
MNIST



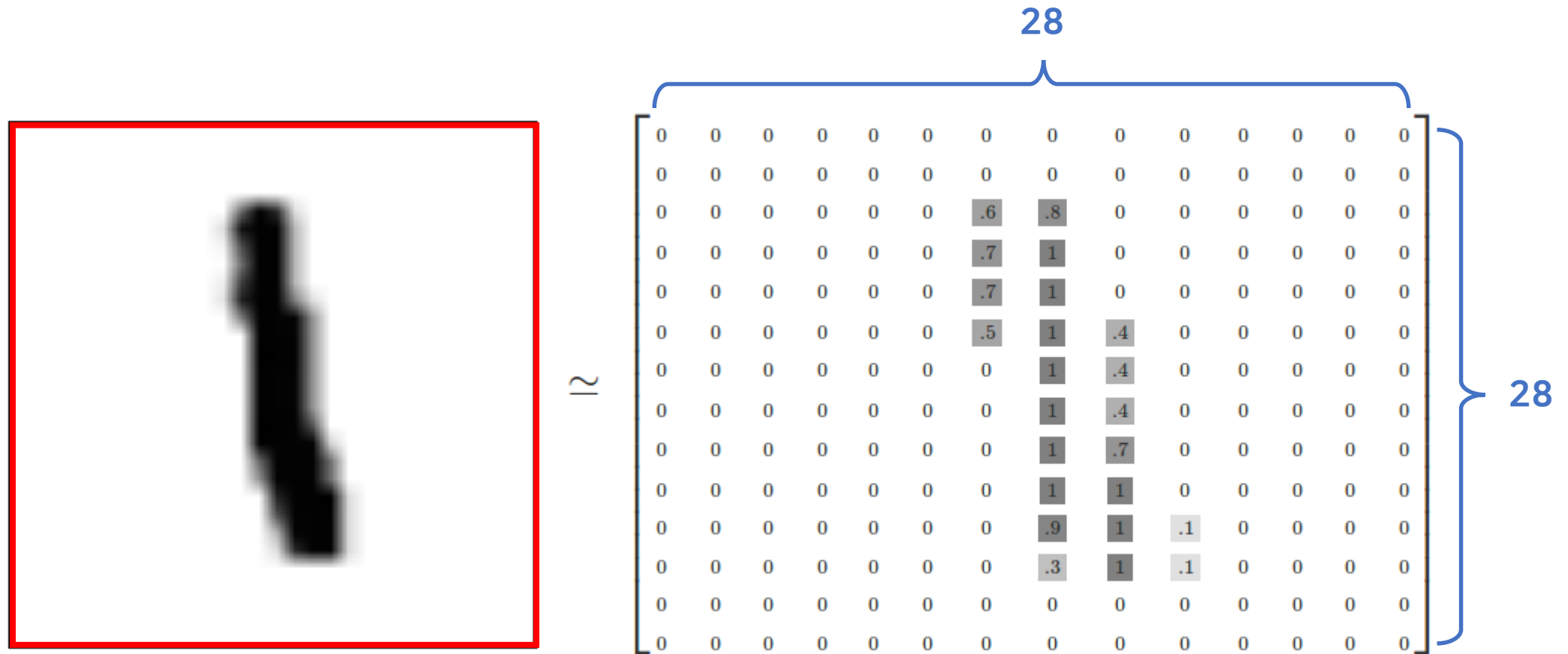
- MNIST (Modified National Institute of Standards and Technology database)

- 전체 데이터셋: 70,000개
- 학습 데이터셋: 60,000개
- 평가 데이터셋: 10,000개

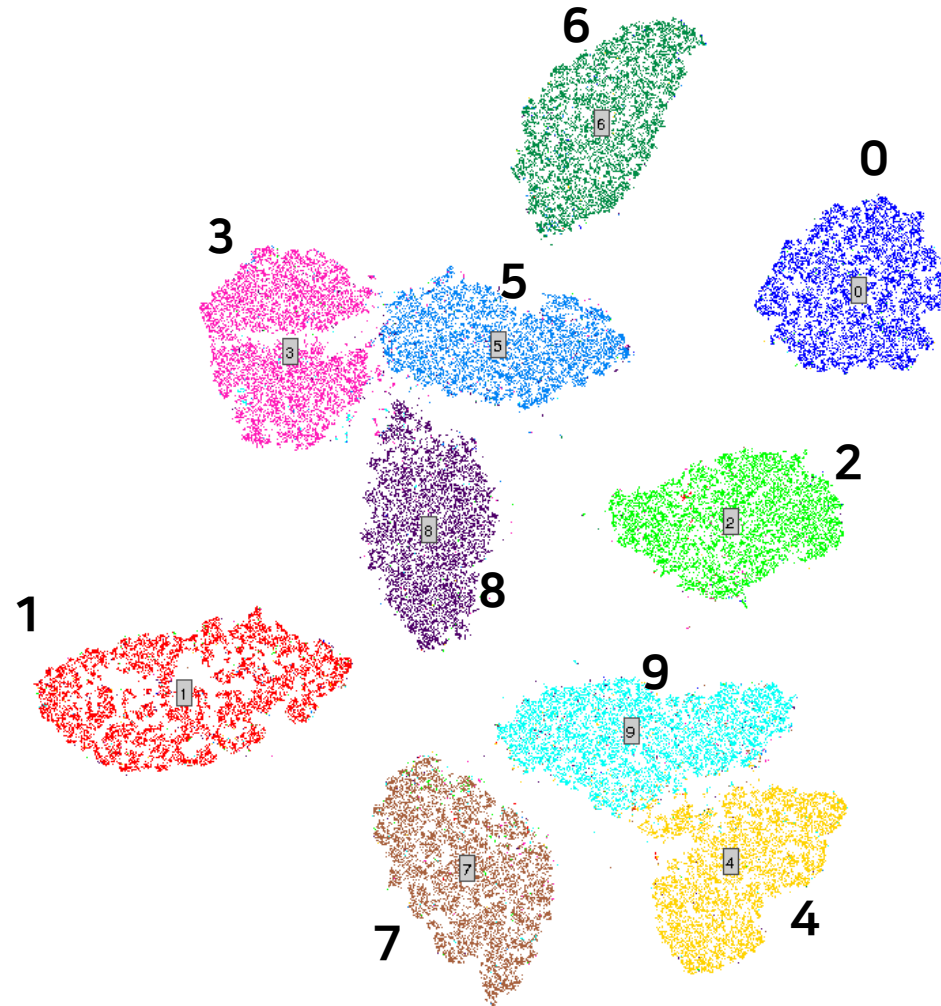
컴퓨터에서 이미지를 읽는 과정 - MNIST



컴퓨터에서 이미지를 읽는 과정 - MNIST



MNIST



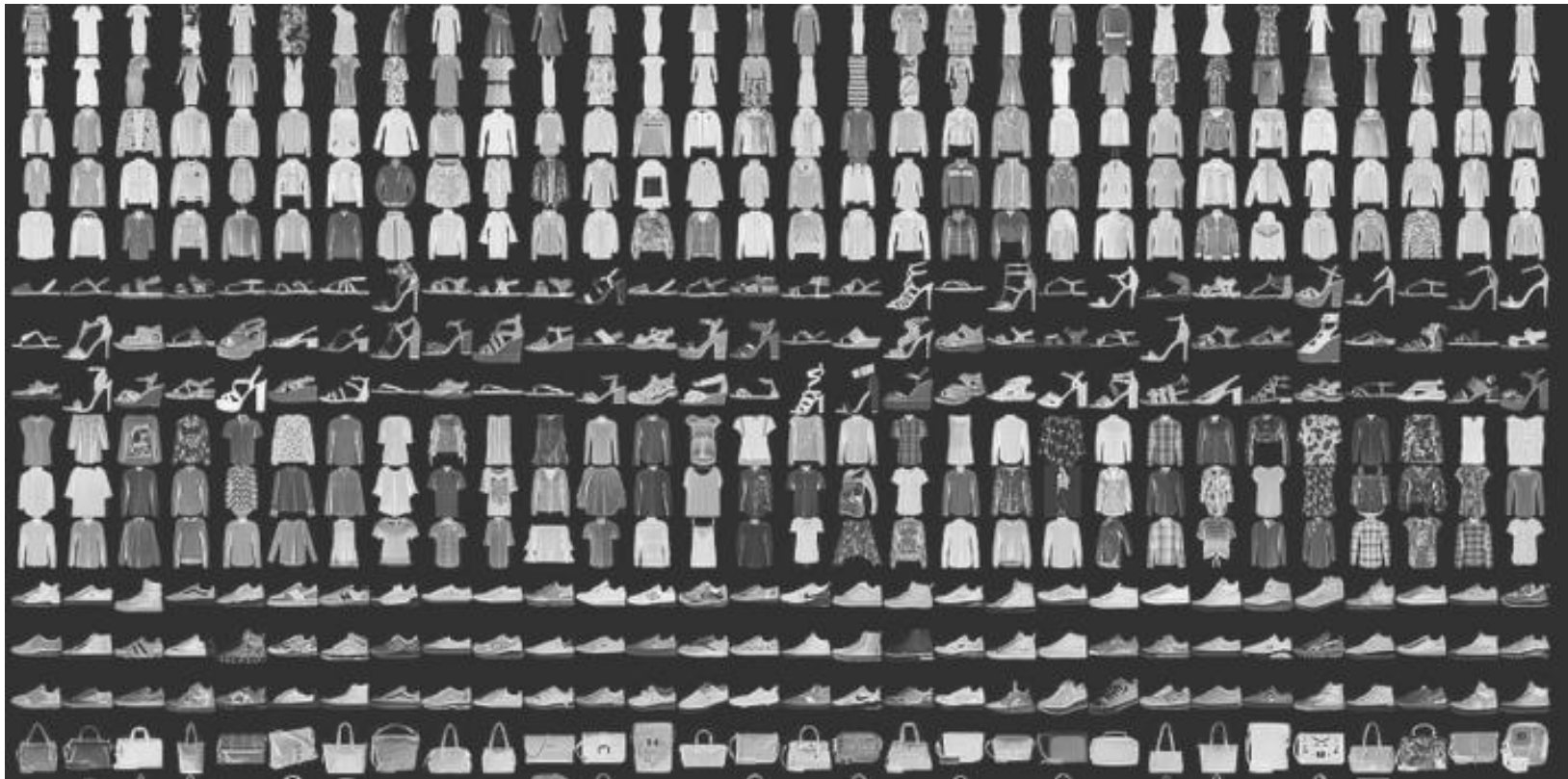
MNIST

```
>>> from sklearn.datasets import fetch_openml
>>> mnist = fetch_openml('mnist_784', version=1)
>>> mnist.keys()
dict_keys(['data', 'target', 'feature_names', 'DESCR', 'details',
           'categories', 'url'])
```

```
>>> X, y = mnist["data"], mnist["target"]
>>> X.shape
(70000, 784)
>>> y.shape
(70000,)
```

```
X_train, X_test, y_train, y_test = X[:60000], X[60000:], y[:60000], y[60000:]
```

Fashion-MNIST



Kannda-MNIST

೦	೧	೨	೩	೪	೫	೬	೭	೮	೯	೦೦
ಒಂದು	ಎರಡು	ಮೂರು	ನಾಲ್ಕು	ಐದು	ಆರು	ಏಳು	ಎಂಟು	ಒಂಬತ್ತು	ಹತ್ತು	
omdu	eraḍu	mūru	nāḷku	aidu	āru	ēḷu	eṁṭu	oṁbattu	hattu	
1	2	3	4	5	6	7	8	9	10	

[이진 분류기 훈련]

이진 분류



이진 분류

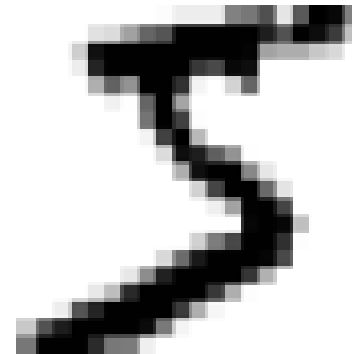


이진 분류

```
y_train_5 = (y_train == 5) # 5는 True고, 다른 숫자는 모두 False  
y_test_5 = (y_test == 5)
```

```
from sklearn.linear_model import SGDClassifier  
  
sgd_clf = SGDClassifier(random_state=42)  
sgd_clf.fit(X_train, y_train_5)
```

```
>>> sgd_clf.predict([some_digit])  
array([ True])
```



Quiz1

- Q1. 머신러닝 파이프라인
 - A1.
- Q2. 분류는 지도학습이다. (O / X)
 - A2.
- Q3. MNIST 데이터셋의 형상(shape)은 얼마인가?
 - A3.



(1/3)

[분류에서의 성능 측정]

오차행렬

● 오차행렬을 만들기 위해선

- 실제 값과 비교할 수 있도록 먼저 예측 값을 만들어야 함

```
>>> from sklearn.metrics import confusion_matrix
>>> confusion_matrix(y_train_5, y_train_pred)
array([[53057, 1522],
       [ 1325, 4096]])
```

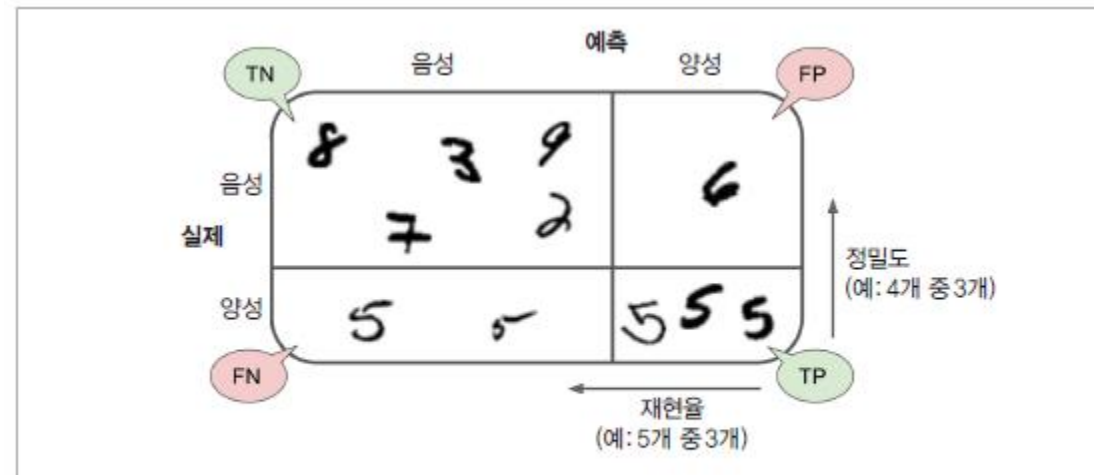


그림 3-2 이 오차 행렬 그림은 진짜 음성 샘플(왼쪽 위), 거짓 양성(오른쪽 위), 거짓 음성(왼쪽 아래), 진짜 양성(오른쪽 아래)를 보여줍니다.

이진 분류에서의 성능 측정

		실제 정답	
		True	False
		True	False
분류 결과	True	True Positive	False Positive
	False	False Negative	True Negative

$$accuracy = \frac{TP + TN}{TP + FN + TN + FP}$$

이진 분류에서의 성능 측정 - Accuracy

		실제 정답	
		True	False
		True	False
분류 결과	True	True Positive	False Positive
	False	False Negative	True Negative

$$accuracy = \frac{TP + TN}{TP + FN + TN + FP}$$

Accuracy만으로 부족한가?

● 예시: 암 검사를 Accuracy로?

- 암 검사를 위해서 Accuracy로 판단할까?
- 아무래도 암 환자는 일반인에 비해서 확연하게 적기 때문에 Accuracy는 높음
 - 1000명 중 1명 꼴로 암에 걸린다고 할 때, 단순히 생각해도 암 걸릴 확률을 $\frac{1}{1000}$ 이라고 할 수 있음
 - 따라서 임의의 환자가 암인지 아닌지 맞출 Accuracy는 99.9%라고 할 수 있음
 - 왜냐하면 눈 감고 짚어도 높은 확률로 정상인임
 - 즉, 여기 있는 여러분 누구라도 암인지 아닌지 검사할 수 있는 Accuracy 99% 이상이라고 말할 수 있음
 - 과연 이러한 지표로 암 검사를 하면 확실할까? 즉, 환자에게 신뢰를 줄 수 있을까?

● Accuracy를 보완하기 위해 나온 다른 지표들

- Recall
- Precision
- F1 Score

이진 분류에서의 성능 측정 - Precision

		실제 정답	
		True	False
		True	False
분류 결과	True	True Positive	False Positive
	False	False Negative	True Negative

$$precision = \frac{TP}{TP + FP}$$

$$recall = \frac{TP}{TP + FN}$$

$$F1 = \frac{2 \times precision \times recall}{precision + recall}$$

$$accuracy = \frac{TP + TN}{TP + FN + TN + FP}$$

$$specificity = \frac{TN}{TN + FP}$$

이진 분류에서의 성능 측정 - Recall

		실제 정답	
		True	False
		True	False
분류 결과	True	True Positive	False Positive
	False	False Negative	True Negative

$$precision = \frac{TP}{TP + FP}$$

$$recall = \frac{TP}{TP + FN}$$

$$F1 = \frac{2 \times precision \times recall}{precision + recall}$$

$$accuracy = \frac{TP + TN}{TP + FN + TN + FP}$$

$$specificity = \frac{TN}{TN + FP}$$

이진 분류에서의 성능 측정 - Precision과 Recall

```
>>> from sklearn.metrics import precision_score, recall_score
>>> precision_score(y_train_5, y_train_pred) # = 4096 / (4096 + 1522)
0.7290850836596654

>>> recall_score(y_train_5, y_train_pred) # = 4096 / (4096 + 1325)
0.7555801512636044
```

Precision / Recall Trade-off

- 두 마리 토끼를 모두 잡을 순 없다!
 - 적절한 임계값을 선택해야함
 - Precision을 높이면 Recall이 내려가고
 - Recall을 높이려면 Precision이 내려간다

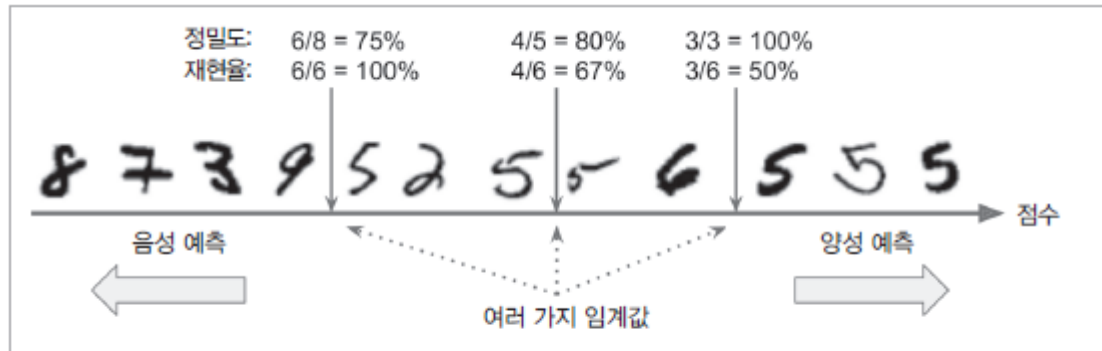


그림 3-3 이 정밀도/재현율 트레이드오프 이미지는 분류기가 만든 점수 순으로 나열되어 있습니다. 선택한 결정 임계값 위의 것을 양성으로 판단합니다. 임계값이 높을수록 재현율은 낮아지고 반대로 (보통) 정밀도는 높아집니다.

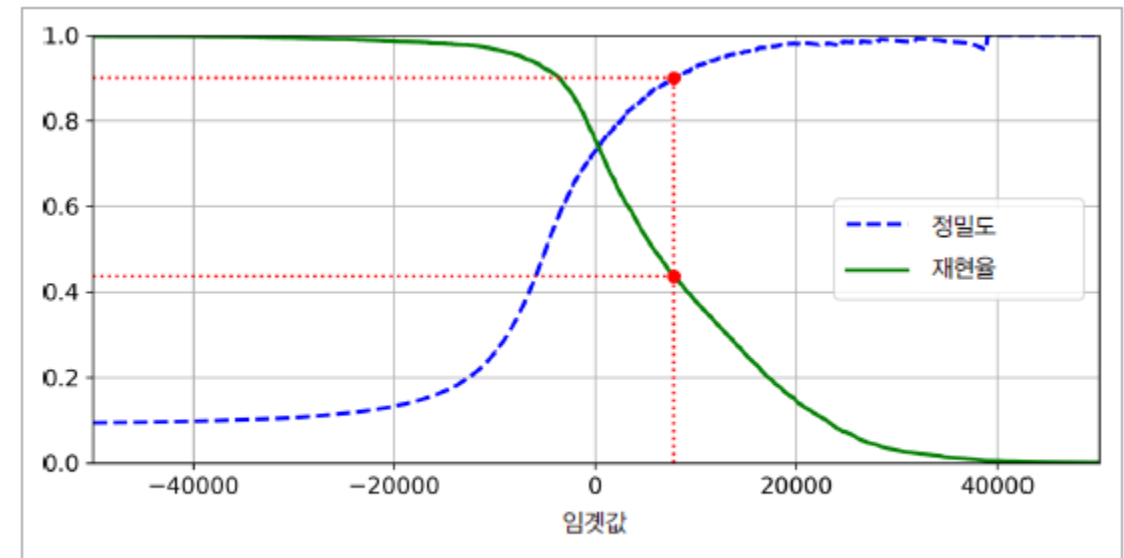
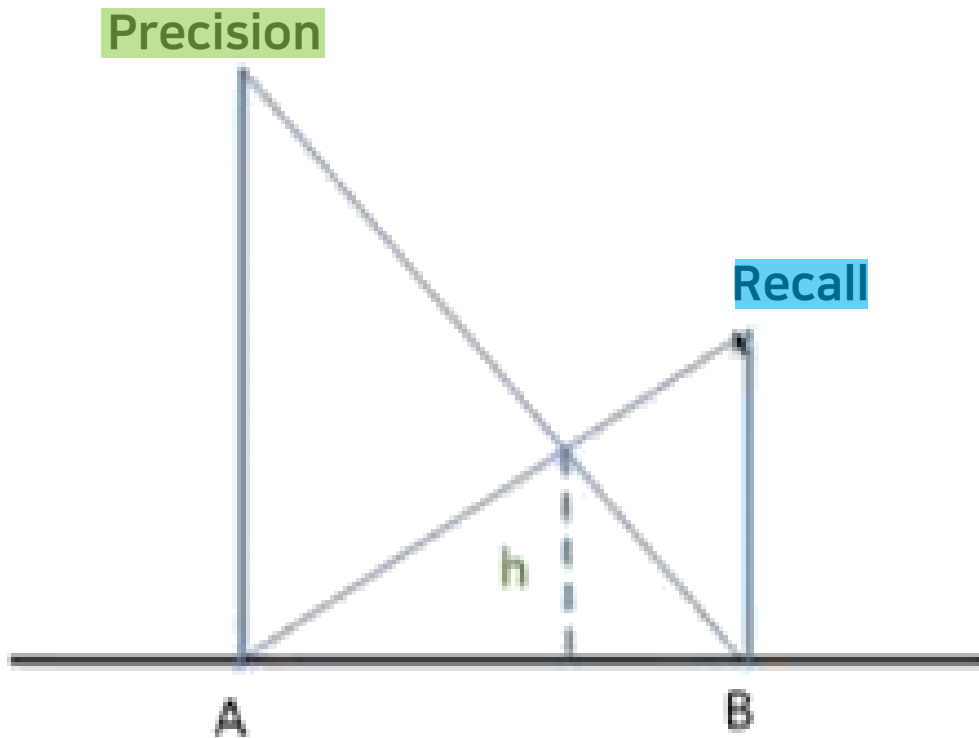


그림 3-4 결정 임계값에 대한 정밀도와 재현율

이진 분류에서의 성능 측정 - F1 Score



$$\text{precision} = \frac{TP}{TP + FP}$$

$$\text{recall} = \frac{TP}{TP + FN}$$

$$F1 = \frac{2 \times \text{precision} \times \text{recall}}{\text{precision} + \text{recall}}$$

$$\text{accuracy} = \frac{TP + TN}{TP + FN + TN + FP}$$

$$\text{specificity} = \frac{TN}{TN + FP}$$

```
>>> from sklearn.metrics import f1_score
>>> f1_score(y_train_5, y_train_pred)
0.7420962043663375
```

이진 분류에서의 성능 측정 - ROC와 AUC

● 수신기 조작 특성(Receiver Operating Characteristic, ROC)

- 거짓 양성 비율(FPR)에 대한 진짜 양성 비율(TPR)의 곡선

$$- FPR = \frac{FP}{FP+TN} = \frac{FP+TN-TN}{FP+TN} = 1 - \frac{TN}{FP+TN} = 1 - TNR$$

● 곡선 아래의 면적(Area Under the Curve, AUC)

- 완벽하게 분류한 ROC의 AUC의 값은 1

- 완전 랜덤한 ROC의 AUC의 값은 0.5

```
from sklearn.metrics import roc_curve
```

```
fpr, tpr, thresholds = roc_curve(y_train_5, y_scores)
```

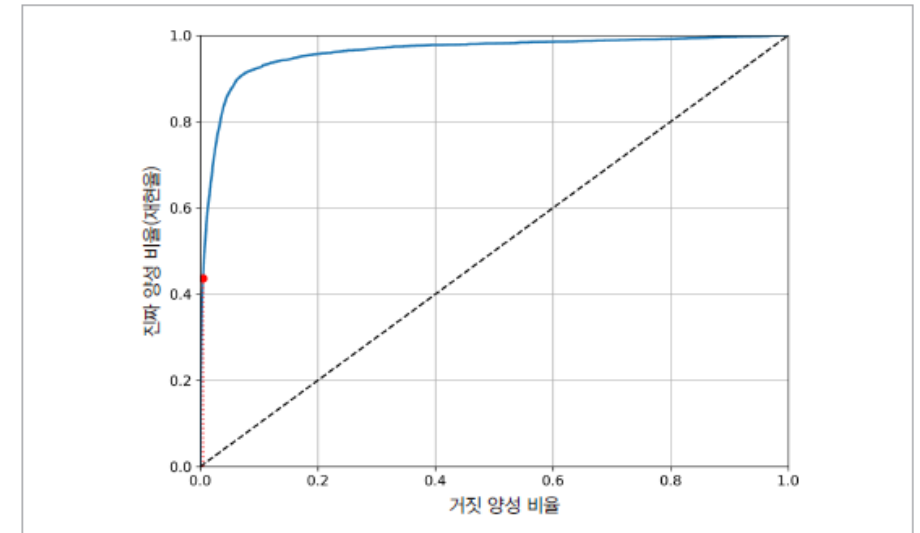


그림 3-6 모든 가능한 임계값에서 진짜 양성 비율에 대한 거짓 양성 비율을 나타낸 ROC 곡선. 붉은 점이 선택한 비율의 지점입니다(43.68%의 재현율).

Quiz2

- Q1. 정확도(Accuracy)의 정의는 무엇인가?
 - A1.
- Q2. 재현율(Recall)의 정의는 무엇인가?
 - A2.
- Q3. 정밀도(Precision)의 정의는 무엇인가?
 - A3.
- Q4. F1 스코어의 정의는 무엇인가?
 - A4.

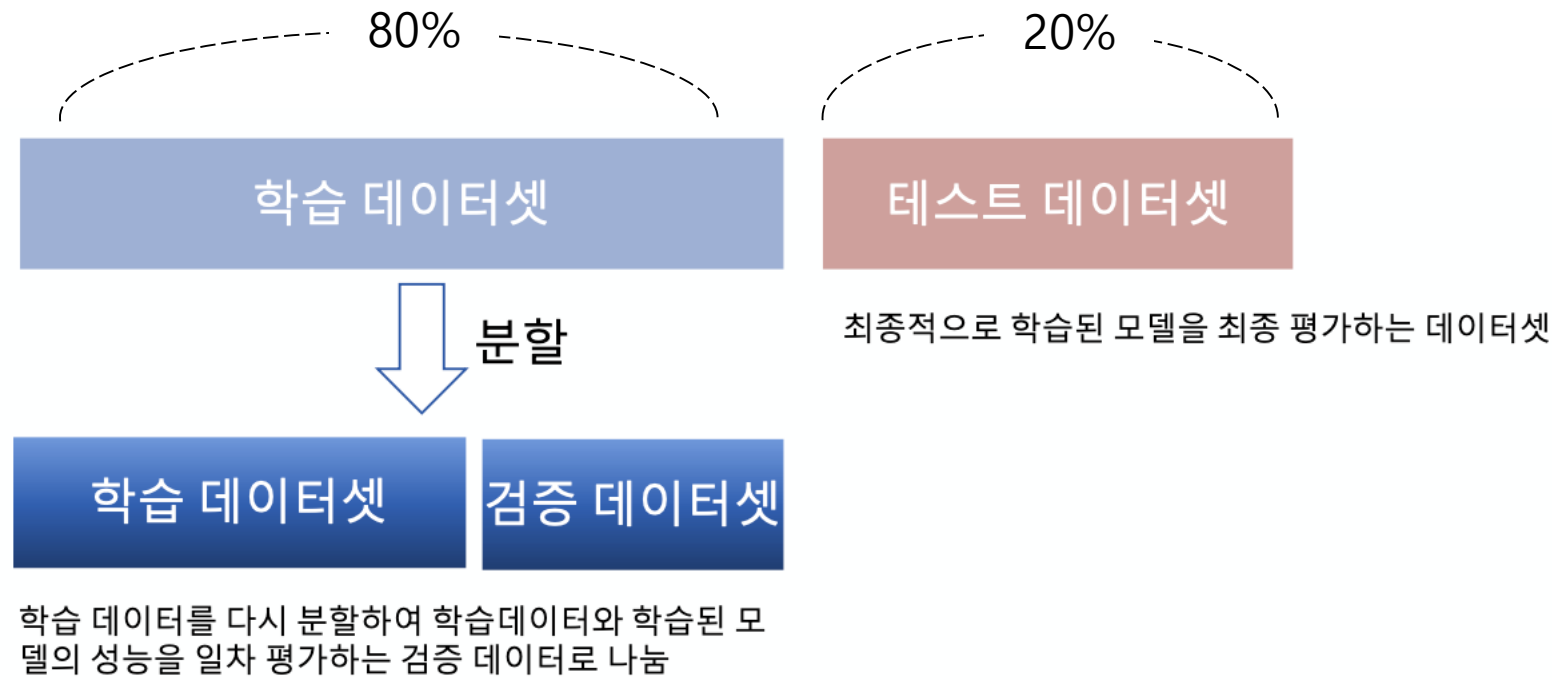


(2/3)

[교차 검증]

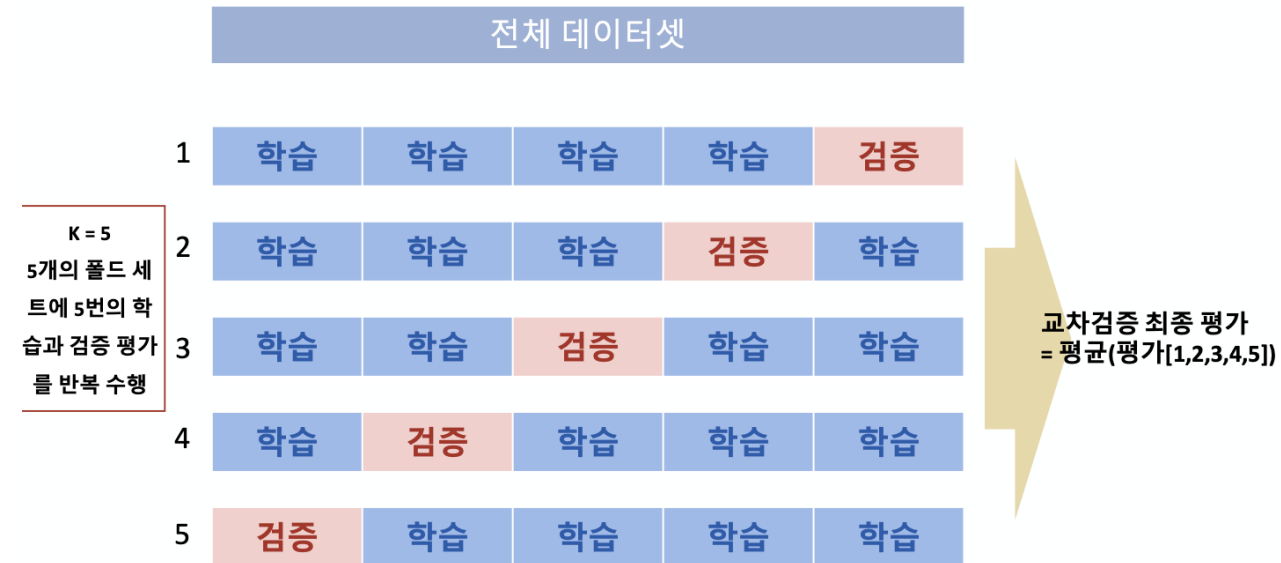
학습, 검증, 평가 데이터셋

- 주어진 데이터에 과적합이 되는 걸 방지함
- 데이터셋이 충분할 때, 성능 평가를 위해선
 - 학습 데이터와 검증 데이터셋과 테스트 데이터셋으로 나뉘어야 한다



교차 검증

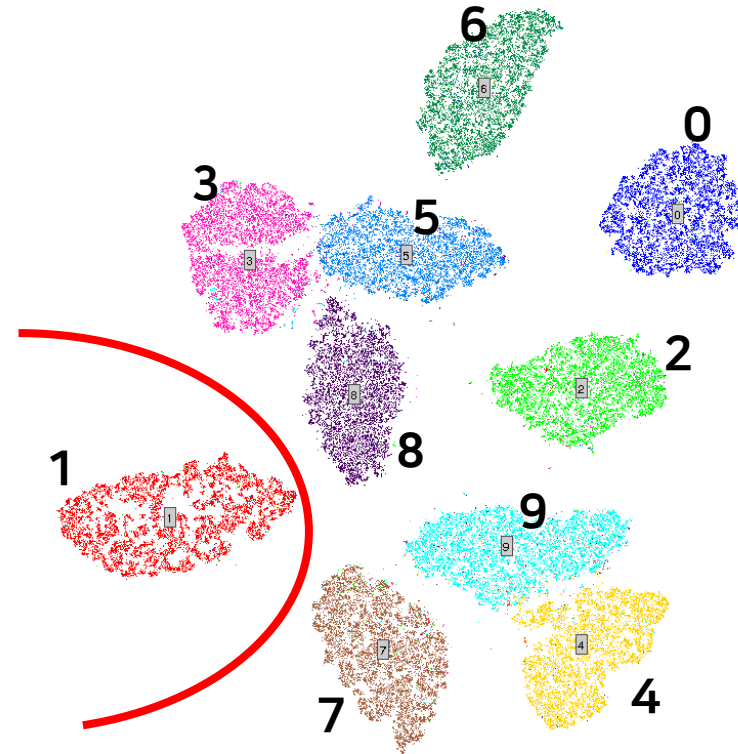
- 학습 데이터가 부족할 때는 어떻게 해야할까?
 - 데이터가 부족하면 성능에 대한 신뢰도가 낮아짐
- K-교차검증
 - 데이터를 k개로 나누어 학습
- Stratified K - 교차검증
 - 데이터를 k개로 나눌 때, 각 클래스 별로 고르게 분포하도록 나눔



[다중 분류]

다중 분류

- 둘 이상의 클래스를 구별하는 작업
- OvA and OvO
 - OvA (One-Versus-All)
 - 1 vs (2,3, ..., 9)
 - 2 vs (1,3, ..., 9)
 - ...
 - OvO (One-Versus-One)
 - 1 vs 2
 - 1 vs 3
 - ...
 - 8 vs 9



다중 분류

- 다중 분류란?

- 둘 이상의 클래스를 구별하는 작업

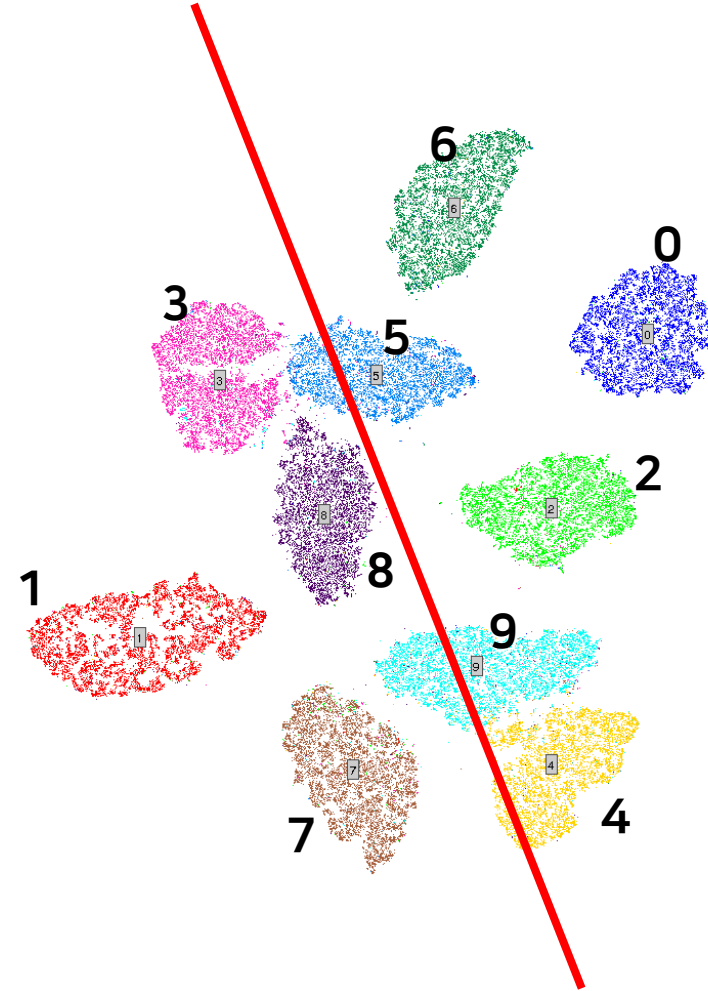
- OvA and OvO

- OvA (One-Versus-All)

- 1 vs (2,3, ..., 9)
- 2 vs (1,3, ..., 9)
- ...

- OvO (One-Versus-One)

- 1 vs 2
- 1 vs 3
- ...
- 8 vs 9

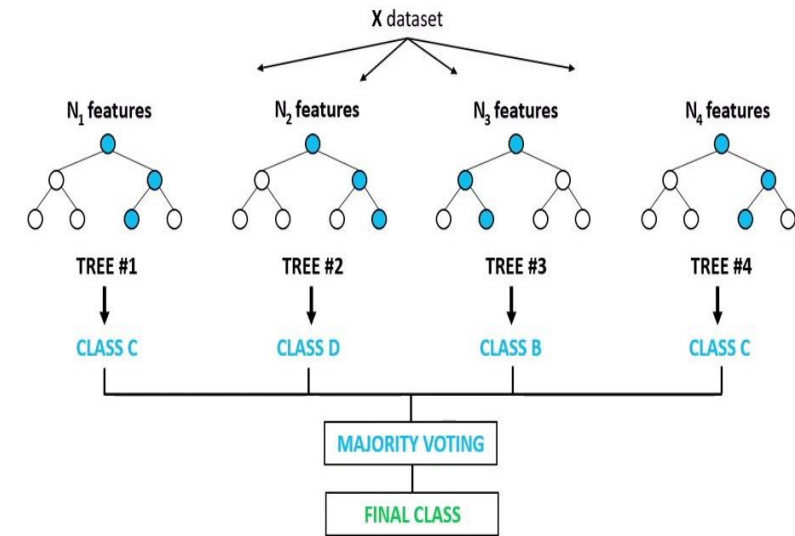
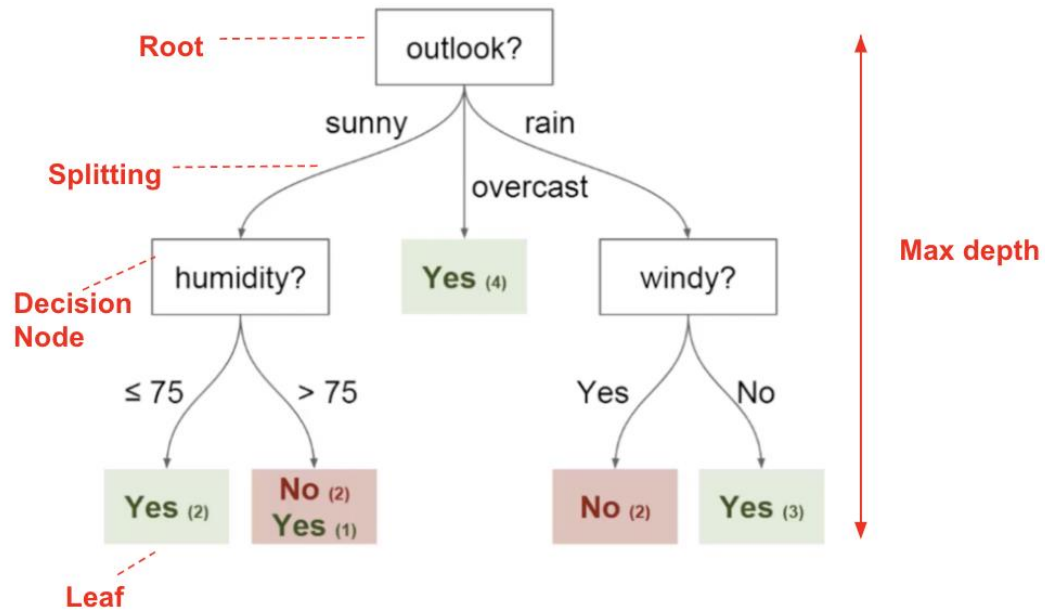


다중 분류

```
>>> from sklearn.multiclass import OneVsRestClassifier
>>> ovr_clf = OneVsRestClassifier(SVC())
>>> ovr_clf.fit(X_train, y_train)
>>> ovr_clf.predict([some_digit])
array([5], dtype=uint8)
>>> len(ovr_clf.estimators_)
10
```

```
>>> sgd_clf.fit(X_train, y_train)
>>> sgd_clf.predict([some_digit])
array([5], dtype=uint8)
```

다중 분류



[에러 분석]

에러 분석

- 모델의 성능을 향상시킬 수 있는 방법 → 에러분석
 - 오차행렬을 통해서 확인해보자

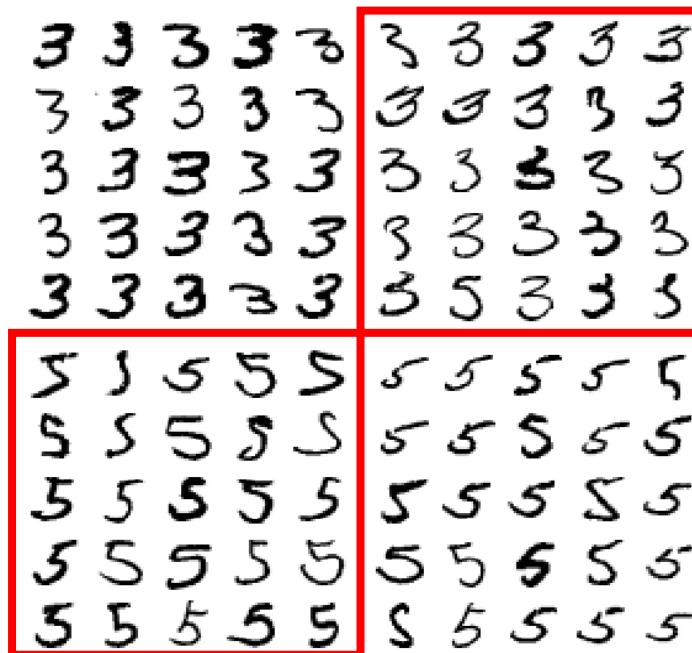
```
>>> y_train_pred = cross_val_predict(sgd_clf, X_train_scaled, y_train, cv=3)
>>> conf_mx = confusion_matrix(y_train, y_train_pred)
>>> conf_mx
array([[5578,   0,  22,   7,   8,  45,  35,   5, 222,   1],
       [   0, 6410,  35,  26,   4,  44,   4,   8, 198,  13],
       [ 28,  27, 5232, 100,  74,  27,  68,  37, 354,  11],
       [ 23,  18, 115, 5254,   2, 209,  26,  38, 373,  73],
       [ 11,  14,  45,  12, 5219,  11,  33,  26, 299, 172],
       [ 26,  16,  31, 173,  54, 4484,  76,  14, 482,  65],
       [ 31,  17,  45,   2,  42,  98, 5556,   3, 123,   1],
       [ 20,  10,  53,  27,  50,  13,   3, 5696, 173, 220],
       [ 17,  64,  47,  91,   3, 125,  24,  11, 5421,  48],
       [ 24,  18,  29,  67, 116,  39,   1, 174, 329, 5152]])
```


에러 분석

```
In [69]: cl_a, cl_b = 3, 5
X_aa = X_train[(y_train == cl_a) & (y_train_pred == cl_a)]
X_ab = X_train[(y_train == cl_a) & (y_train_pred == cl_b)]
X_ba = X_train[(y_train == cl_b) & (y_train_pred == cl_a)]
X_bb = X_train[(y_train == cl_b) & (y_train_pred == cl_b)]

plt.figure(figsize=(8,8))
plt.subplot(221); plot_digits(X_aa[:25], images_per_row=5)
plt.subplot(222); plot_digits(X_ab[:25], images_per_row=5)
plt.subplot(223); plot_digits(X_ba[:25], images_per_row=5)
plt.subplot(224); plot_digits(X_bb[:25], images_per_row=5)
save_fig("error_analysis_digits_plot")
plt.show()
```

그림 저장: error_analysis_digits_plot



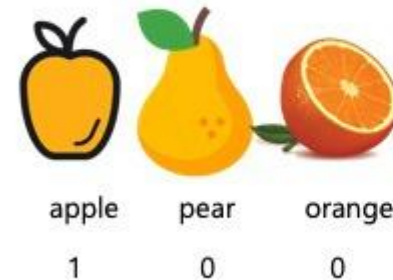
[다중 레이블 분류]

다중 레이블 분류

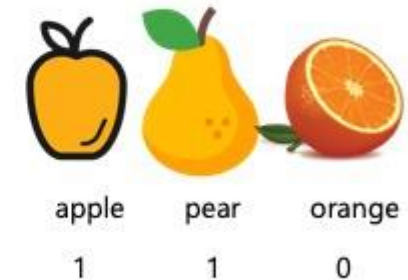
● 다중 레이블 분류와 다중 분류는 다르다

- 다중 분류는 **여러 개의 클래스**를 분류하는 것
 - 오늘 저녁 식사 뭐 먹을지 예측
 - 김치찌개, 부대찌개, 햄버거 등
- 다중 레이블 분류는 **여러 개의 레이블**을 분류하는 것
 - 오늘 점심, 저녁 식사 뭐 먹을지 예측
 - 점심
 - 김치찌개, 부대찌개
 - 저녁
 - 햄버거, 피자, 치킨










Multi-Class



Multi-Label



다중 레이블 분류

	Multi-Class	Multi-Label
C = 3		
  	<p>Samples</p> <div>    </div> <p>Labels (t)</p> <div> [0 0 1] [1 0 0] [0 1 0] </div>	<p>Samples</p> <div>    </div> <p>Labels (t)</p> <div> [1 0 1] [0 1 0] [1 1 1] </div>

Quiz3

- Q1. OvA 와 OvO가 차이점은 무엇인가?
 - A1.
- Q2. 다중분류와 다중클래스 분류의 차이점은 무엇인가?
 - A2.
- Q3. 교차 검증이란 무엇이며 왜 하는 걸까요?
 - A3.



(3/3)

[실습]

MNIST로 이진 분류 및 다중 분류를 해봅시다!

- 다음 시간에 -