Deep Learning

Machine Learning

기계학습

첫번째 머신러닝 애플리케이션, 붓꽃 품종 분류기

Yoon Joong Kim

Department of Computer Engineering, Hanbat National University

yjkim@hanbat.ac.kr

참고도서 :"파이썬 라이브러리를 활용한 머신러닝"

Contents

1.1 기계학습이란?

- 1.1 기계학습(머신러닝Machine Learning)이란?
  - 데이터에서 지식을 추출하는 작업
  - 통계학, 인공지능,컴퓨터 과학
  - 응용 예
    - 웹사이트와 기기간의 머신러닝
      - 영화 추천,음식 주문, 쇼핑, 맞춤형 온라인 라디오 방송, 사진에서 친구 얼굴 찾아주기-
    - 필요한 구성 요소 새성
      - 페이스북, 아마존, 넷플릭스 같은 복잡한 웹사이트들은 여러 가지 머신러닝 모델을 사용
  - History
    - 규칙 기반 전문가 시스템rule-based expert system
      - 그 분야 전문가들의 지식에 기반하여 규칙을 결정(if-else)
      - 결정에 필요한 로직은 한 분야나 작업에 국한됩니다. 작업이 조금만 변경되더라도 전체 시스템을 다시 개발해야 할 수 있습니다
    - 얼굴인식(이미지에서 얼굴찾아 내기)
      - 2011전 불가능, 컴퓨터의 픽셀에 대한 규칙으로 불가능, 사람의 인식방식과 다르다. 머신러닝으로 해결됨

1.1 기계학습이란?(cont.)

- 머신러닝의 능력
  - 알려진 사례를 바탕으로 일반화된 모델을 만들어 의사 결정 프로세스를 자동화–지도학습
- 지도학습 : (입력, 기대되는 출력)을 제공하면 알고리즘이 주어진 입력에서 원하는 출력을 만드는 방법을 말한다.
  - 스펨메일 분류기
  - 예,
    - 우편번호 숫자판별기 : (편지봅투이미지,우편번호) 데이터셋 생성
    - 의료 영상 이미지에 기반한 종양 판단 : (의료 영상 데이터베이스, 의사의 진단결과)
- 비지도 학습unsupervised learning
  - 비지도 학습에서는 알고리즘에 입력은 주어지지만 출력은 제공되지 않습니다
  - 예,
    - 블로그 글의 주제 구분 : 쇼핑 사이트라면 부모, 독서광, 게이머 같은 그룹이 있을 수 있습니다.
    - 비정상적인 웹사이트 접근 탐지: 일상적이지 않은 접근 패턴
- 샘플과 특성
  - 하나의 개체 혹은 행을 샘플sample 또는 데이터 포인트data point라고 부릅니다. 그리고 샘플의 속성, 즉 열을 특성feature이라고 합니다.
    한샘플의 특성 : (고객이름, 나이, 성별, 계정 생성일, 온라인 쇼핑몰에서의 구매 빈도)
    데이터에는 기대되는 결과의 실마리가 어떤 형태로 든지 존재해야

1.2 문제와 데이터 이해하기

- 1.2 문제와 데이터 이해하기
  - 어떤 질문에 대한 답을 원하는가? 가지고 있는 데이터가 원하는 답을 줄 수 있는가?
  - 내 질문을 머신러닝의 문제로 가장 잘 기술하는 방법은 무엇인가?
  - 문제를 풀기에 충분한 데이터를 모았는가?
  - 내가 추출한 데이터의 특성은 무엇이며 좋은 예측을 만들어낼 수 있을 것인가?
  - 머신러닝 애플리케이션의 성과를 어떻게 측정할 수 있는가?
  - 머신러닝 솔루션이 다른 연구나 제품과 어떻게 협력할 수 있는가?

- 1.3 기계학습용 패키지 in python
  - sklearn, scikit-learn, [link]
    - Machine Learning in Python
    - Simple and efficient tools for data mining and data analysis
    - Accessible to everybody, and reusable in various contexts
    - Built on NumPy, SciPy, and matplotlib
    - Open source, commercially usable – BSD license
    - Classification, Regression, Clustering, Dimension Reduction, Model selection, Preprocessing
  - pandas [link]
    - In computer programming, **pandas** is a software library written for the Python programming language for **data manipulation and analysis**.
  - mglearn [link]
    - helper functions to create figures and datasets.

```
from sklearn.datasets import load_iris
from sklearn.model_selection import train_test_split
import matplotlib.pyplot as plt

import pandas as pd
import mglearn
```

2.첫번째 머신러닝 애플리케이션, 붓꽃 품종 분류기(cont.)

- 2.1 문제의 분석
  - 붓꽃의 품종 분류기 개발

  - 붓꽃의 품종
    - Versicolor
    - Virginica
    - Setosa

출처 : http://mirlab.org/의 아이리스 항목

  - 붓꽃의 특징
    - 구분 지울 수 있는 특징?
    - 꽃받침과 꽃잎의 두께와 길이

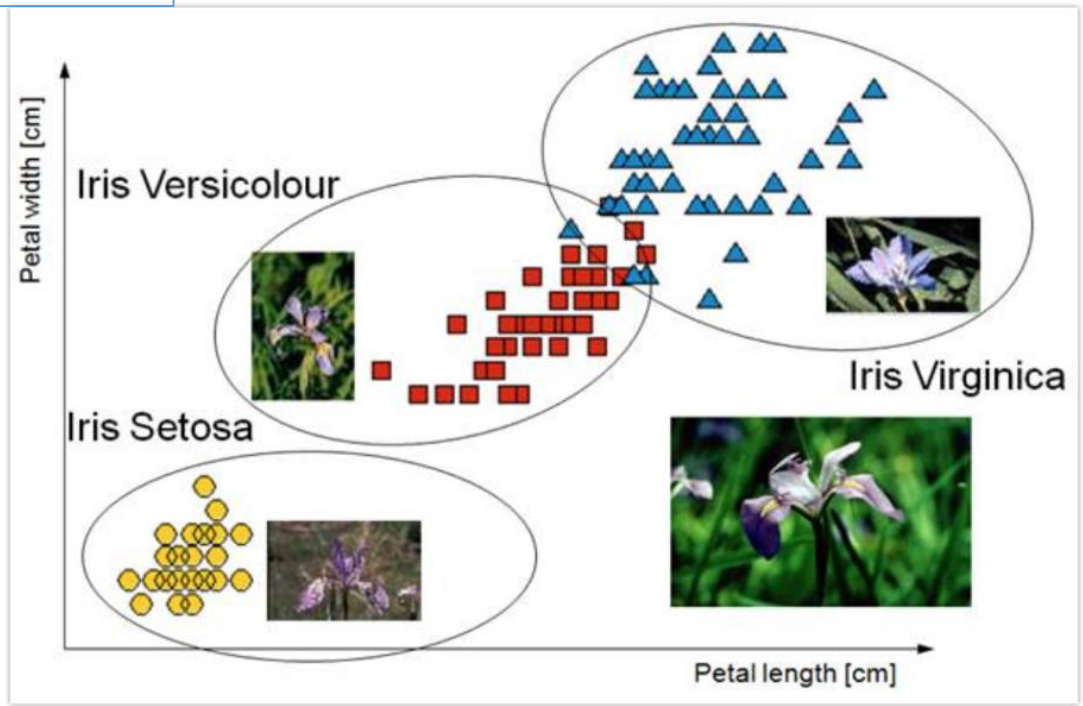
출처 : http://mirlab.org/의 아이리스 항목

2.1 문제의 분석(cont.)

- 2.1 문제의 분석(cont.)
  - 특징데이터 샘플의 특징

특징벡터
['sepal length (cm)', 'sepal width (cm)', 'petal length (cm)', 'petal width (cm)']
[꽃받침길이 꽃받침두께 꽃잎길이 꽃잎두께]

  - 특징과 분류 가능성 분석
    - 산점도



출처 : http://articles.concreteinteractive.com/

  - 데이터 셋의 예

특징벡터				target	꽃품종
[5.1	3.8	1.6	0.2]	[0]	Versicolour
[7.7	3.0	6.1	2.3]	[2]	Virginica
[6.9	3.1	4.9	1.5]	[1]	Setosa

2.2 데이터 셋 분석

- 2.2 데이터 셋 분석
  - 2.2.1 데이터의 적재 및 훈련/평가세트 생성

```
# 데이터세트 적재
iris_dataset = load_iris()

# 훈련 셋(75%)과 평가셋 생성
X_train,X_test,y_train,y_test=train_test_split(
        iris_dataset['data'],iris_dataset['target'],random_state=0)
print('X_train:{} X_test:{} y_train:{}  y_test:{} ‘,
        X_train.shape,X_test.shape,y_train.shape,y_test.shape)
```

```
from sklearn.datasets import load_iris
from sklearn.model_selection import train_test_split
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
import mglearn
```

2.2.2 데이터 셋 분석-기본정보

- 2.2.2 Iris 데이터 세트 분석 - 기본정보

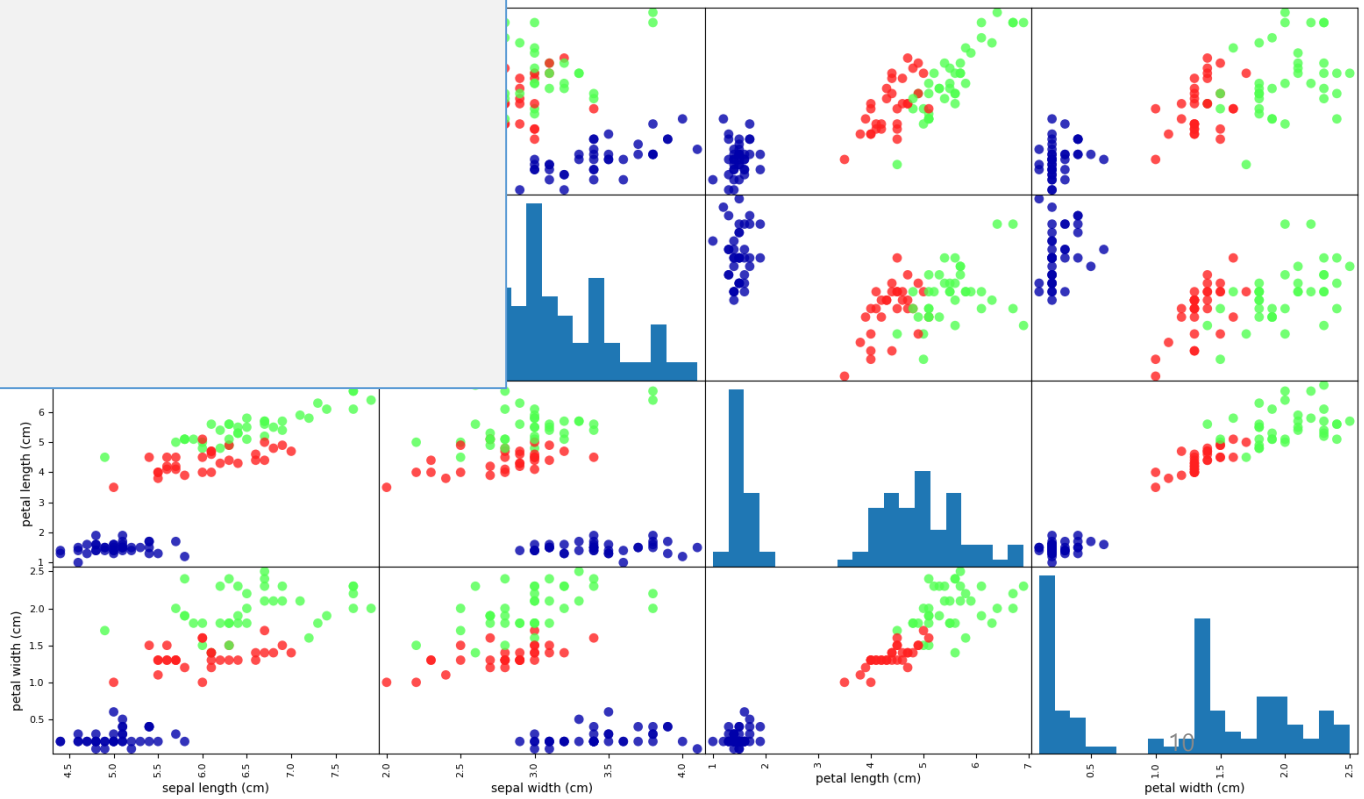
```
#데이터 셋 기본정보분석
print("iris_dataset의 키: \n{}".format(iris_dataset.keys()))
        #['data', 'target', 'target_names', 'DESCR', 'feature_names', 'filename']

print(iris_dataset['DESCR'][:193] + "\n...")
        #

print("data의 크기: {}".format(iris_dataset['data'].shape))
        #(150,4)

print("데이터 특성의 이름: \n{}".format(iris_dataset['feature_names']))
        #['sepal length (cm)', 'sepal width (cm)', 'petal length (cm)', 'petal width (cm)']

print("target의 크기: {}".format(iris_dataset['target'].shape))
        #(150,)

print("타깃의 이름: {}".format(iris_dataset['target_names']))
        #['setosa' 'versicolor' 'virginica']

#data 와 target 출력
xy=[list(xx)+[yy] for xx,yy in zip(iris_dataset['data'],iris_dataset['target'])]
print(pd.DataFrame(xy,columns=iris_dataset.feature_names+['target'']))

print('X_train:{} X_test:{} y_train:{}  y_test:{}',
        X_train.shape,X_test.shape,y_train.shape,y_test.shape)
```

```
iris_dataset의 키:
dict_keys(['data', 'target', 'target_names', 'DESCR', 'feature_names', 'filename'])
.. _iris_dataset:

Iris plants dataset
--------------------

**Data Set Characteristics:**

    :Number of Instances: 150 (50 in each of three classes)
    :Number of Attributes: 4 numeric, pre
...
data의 크기: (150, 4)
데이터 특성의 이름:
['sepal length (cm)', 'sepal width (cm)', 'petal length (cm)', 'petal width (cm)']
target의 크기: (150,)
타깃의 이름: ['setosa' 'versicolor' 'virginica']
     sepal length (cm)  sepal width (cm)  petal length (cm)  petal width (cm)  target
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 x 5 columns]
X_train:{} X_test:{} y_train:{}  y_test:{}  (112, 4) (38, 4) (112,) (38,)
```

2.2.3 데이터 셋 분석-산점도

- 2.2.3 Iris 데이터 세트 분석 - 산점도
  - 학습시킬 데이터의 산점도를 분석하여 분류의 가능성을 분석한다.

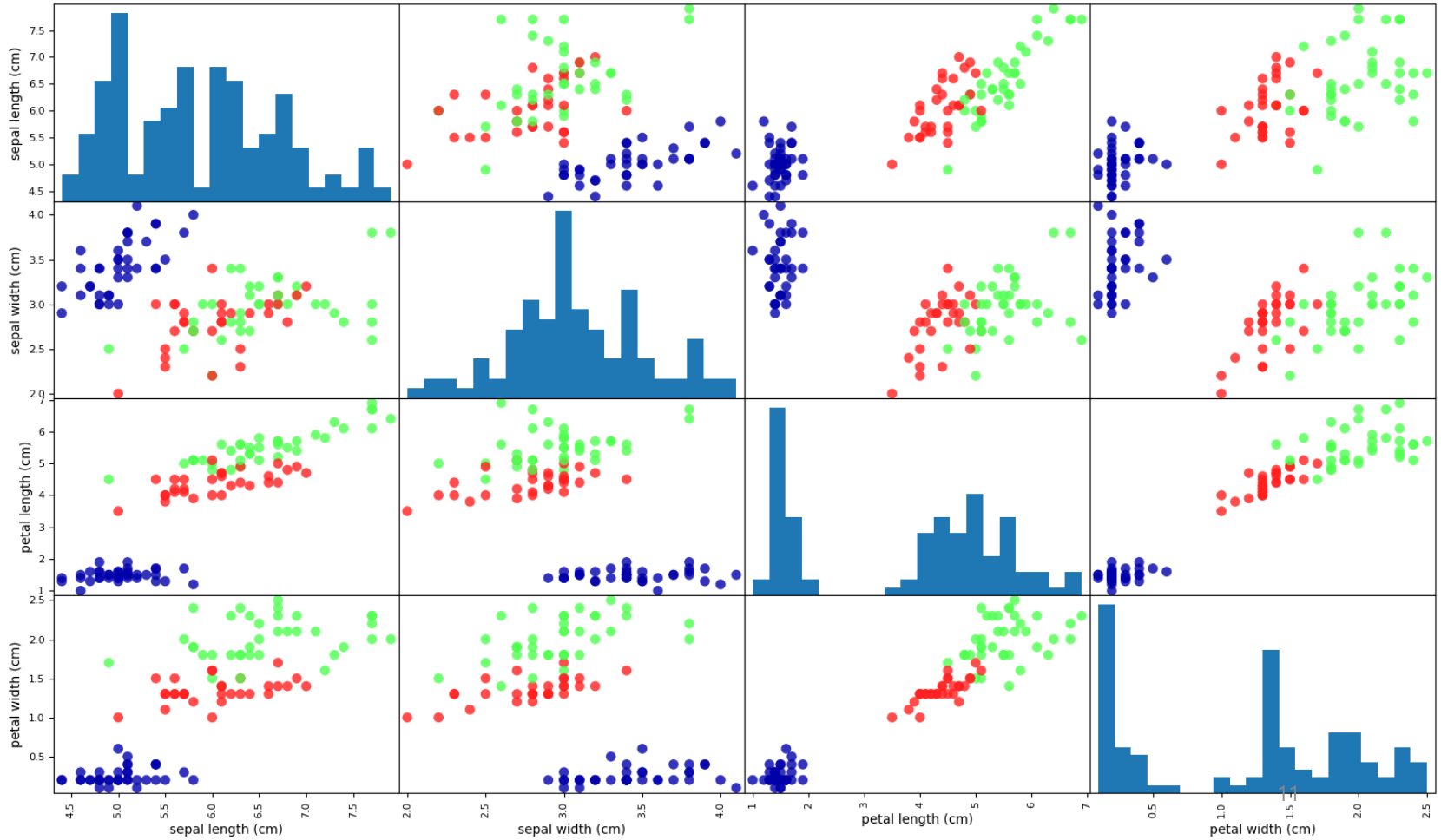
```
# X_train의 산점도 분석
# 판다스의 데이터프레임을 사용해 y_train에 따라 색으로 구분된 산점도 행렬을 만듭니다.
iris_dataframe = pd.DataFrame(              # DataFrame 형으로 변환
        X_train,                            # 분석대상 데이터 셋
        columns=iris_dataset.feature_names) # 데이터 셋의 열이름
pd.plotting.scatter_matrix(    # 산점도 작성
        iris_dataframe,        #대상 데이터셋
         c=y_train),           #점의 색상
        figsize=(15, 15),      #그림의 크기 인치
        marker=' o',           #점의 모양
        hist_kwds={'bins': 20}, #히스토그림의 바의 개수
        alpha=.8,              # 투명도
        cmap=mglearn.cm3)      # 색상셋
plt.show()                     # 그리기 시작
```



2.2.3 데이터 셋 분석-산점도

- 2.2.3 Iris 데이터 세트 분석 - 산점도
  - 학습시킬 데이터의 산점도를 분석하여 분류의 가능성을 분석한다.

```
# X_train의 산점도 분석
# 판다스의 데이터프레임을 사용해 y_trai
iris_dataframe = pd.DataFrame(                #
        X_train,                              #
        columns=iris_dataset.feature_names) #
pd.plotting.scatter_matrix(    # 산점도 작성
        iris_dataframe,        #대상 데이터
         c=y_train),           #점의 색상
        figsize=(15, 15),      #그림의 크기
        marker=' o',           #점의 모양
        hist_kwds={'bins': 20}, #히스토그림
        alpha=.8,              # 투명도
        cmap=mglearn.cm3)      # 색상셋
plt.show()                     # 그리기 시작
```



2.3 K-최근접 이웃 알고리즘 모델

- 2.3 K-최근접 이웃 알고리즘 모델
  - KNeighbhorsClassifier모델생성,학습 및 품종예측, 예측정밀도 계산

```
# k-Nearest Neighbors classifier
#k-최근접 이웃 알고리즘에서 주어지는 데이터에서 가장 가까운 이웃 가장 가까운 'k개'의 이웃을 찾는다

# k-Nearest Neighbors classifier 모델의 정의
print('\n\n KNeighborsClassifier 모델 생성 및 평가')
from sklearn.neighbors import KNeighborsClassifier

knn = KNeighborsClassifier(n_neighbors=1)       #모델 정의(생성)
knn.fit(X_train, y_train)                       #모델 학습

X_new = np.array([[5, 2.9, 1, 0.2]])            #평가용 특징 설정
prediction = knn.predict(X_new)                 #모델로 예측,
print("예측: {}=>{}".format(
      X_new,prediction))                        #예측 값 출력 [0]
print("예측값의 품종이름: {}".format(
iris_dataset['target_names'][prediction]))      #예측 값의 품종이름

# X_test 데이터 전체를 예측하고 y_test와 비교하여 예측 정확도를 계산한다.
y_pred = knn.predict(X_test)                    #테스트셋 전체의 예측
print("테스트 세트에 대한 예측값:\n {}".format(
      y_pred))                                  #예측 배열을 출력
print("테스트 세트의 정확도: {:.2f}".format(
      np.mean(y_pred == y_test)))               #테스트셋의 인식 정확도 계산
```

```
 KNeighborsClassifier 모델 생성 및 평가
예측: [[5.  2.9 1.  0.2]]=>[0]
예측값의 품종이름: ['setosa']
테스트 세트에 대한 예측값:
 [2 1 0 2 0 2 0 1 1 1 2 1 1 1 1 0 1 1 0 0 2 1 0 0 2 0 0 1 1 0 2 1 0 2 2 1 0
 2]
테스트 세트의 정확도: 0.97
Press any key to continue . . .
```

```
# k-Nearest Neighbors classifier 모델의 정의
from sklearn.neighbors import KNeighborsClassifier
knn = KNeighborsClassifier(n_neighbors=1)   #모델 생성
knn.fit(X_train, y_train)                   #모델 학습
y_hat = knn.predict([[5, 2.9, 1, 0.2]])     #모델로 예측,
acc   =knn.score(X_test, y_test)            #모델로 평가
```

2.3 K-최근접 이웃 알고리즘 모델

- 2.3 K-최근접 이웃 알고리즘 모델
  - 요약
    - 데이터셋 적재
      - iris_dataset = load_iris()
    - 훈련 및 평가용 데이터셋 생성
      - 데이터셋을 0.75비율로 훈련 및 평가용 셋으로 분리
      - X_train,X_test,y_train,y_test=train_test_split(iris_dataset['data'],iris_dataset['target'],random_state=0)
    - KNeighbhorsClassifier모델 knn생성
      - K 최근이웃 알고리즘의 모델을 생성
      - knn = KNeighborsClassifier(n_neighbors=1)
    - KNeighborClassifier 학습
      - 학습 셋의 샘플과 레이블(기대 값) 셋으로 학습
      - knn.fit(X_train, y_train)
    - 샘플에 대한 예측
      - 학습된 모델로 임의의 샘플의 예측, 샘플의 shape는 학습용 셋의 shape과 같아야 한다.
      - y_hat = knn.predict([[5, 2.9, 1, 0.2]])
    - 정확도 계산
      - 평가셋 전체의 룬류 정확도
      - acc = knn.score(X_test, y_test)

```
from sklearn.datasets import load_iris
from sklearn.model_selection import train_test_split
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
import mglearn
```

```
iris_dataset = load_iris()                          # 데이터세트 적개
X_train,X_test,y_train,y_test=train_test_split(                  # 0.75비율로 훈#련 및
        iris_dataset['data'],iris_dataset['target'],random_state=0) # 평가용 데이터셋 생성

knn = KNeighborsClassifier(n_neighbors=1)#모델 생성
knn.fit(X_train, y_train)                           #모델 학습

y_hat = knn.predict([[5, 2.9, 1, 0.2]])            #모델로 예측,
acc    = knn.score(X_test, y_test)                  #모델로 평가
```