K-최근접 이웃 (**K**-**N**earest **N**eighbor)

지도학습 (Supervised Learning)

- 정답을 주고 학습하는 것
- 학습 데이터로부터 생성한 모델을 이용하여 주어진 입력으로부터 출력을 예측
- 대표적인 기계학습 방법으로 분류 (Classification) 및 회귀 (Regression) 존재
- 분류는 미리 정의된 여러 클래스 레이블 중 하나를 예측하는 것 (Binary and Multiclass Classification)
- 회귀는 연속적인 숫자를 예측하는 것

K-최근접 이웃 (K-Nearest Neighbor, KNN)

- 기계학습에서 분류 문제를 해결하기 위해 주로 사용되는 알고리즘
- 분류에서 참조하게 될 가장 가까운 이웃 K개의 점을 결정하여 다수결 원칙으로 진행
- 데이터들이 분류 작업에 잘 반영될 수 있도록 전처리 과정 필수
- 사례 중심 학습 (Instanced Based Learnig ↔ Model Based Learning)
  - 학습 데이터로 판별 함수를 학습하는 대신 데이터를 메모리에 저장 (학습 = 저장)
  - 데이터 분포를 표현하기 위한 파라미터를 추정하지 않음
  - 모델을 별도로 구축하지 않음 (게으른 학습, Lazy Model)

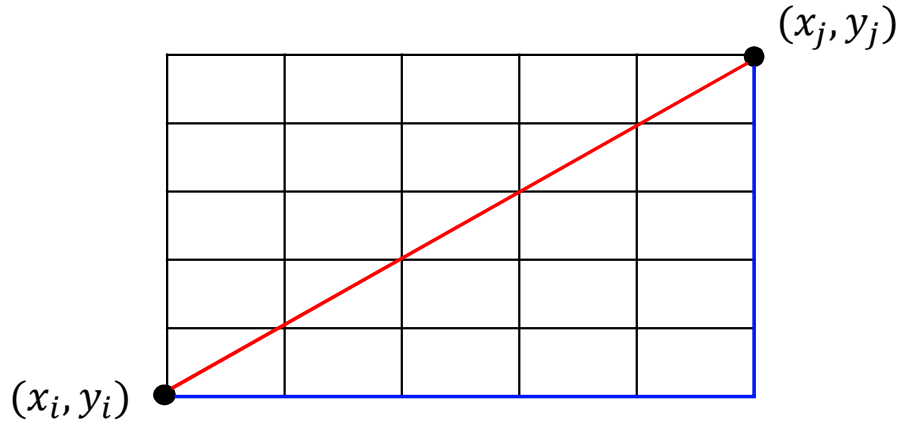
거리 기반 평가

- 유클리드 거리 (Euclidenan Distance) : 한 점과 다른 한 점 사이의 직선거리

$$d_{i,j} = \sqrt{(x_j - x_i)^2 + (y_j - y_i)^2}$$

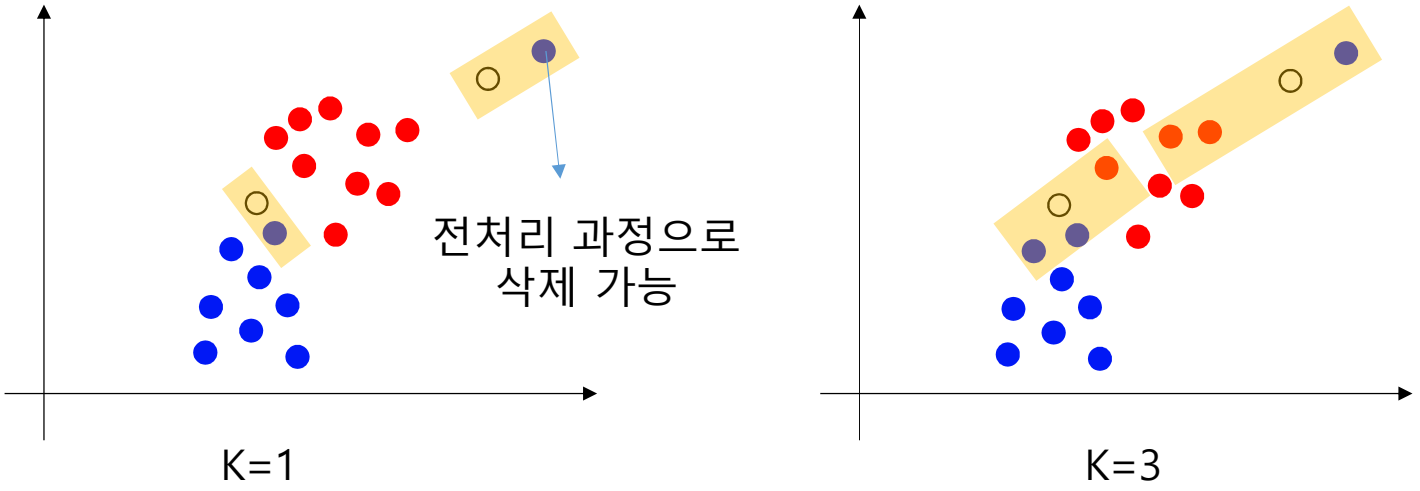
- 맨하탄 거리 (Manhattan Distance) : 맨하탄 거리의 블록을 걸을 때의 모습에 유래

$$d_{i,j} = |x_j - x_i| + |y_j - y_i|$$



분류 과정

- 다수결 방식 : 이웃 데이터들 중 빈도 기준 제일 많은 클래스를 새 데이터의 클래스로 분류
- 가중합 방식 : 가까운 이웃 정보에 좀 더 가중치를 부여하여 분류



KNN 장단점

- 학습 데이터 수가 많은 경우 매우 효율적이고, 학습 데이터 내 노이즈 영향을 크게 받지 않음
- 매우 간단한 방법이지만 높은 성능 (문제에 따라 성능 다름)
- 데이터 특성에 맞게 최적의 이웃 수 K 및 거리 기반 평가 방식 결정 필요
- 새로운 관측치와 학습 데이터 각각의 거리를 측정해야하므로 복잡도가 높음

sklearn.neighbors.KNeighborsClassifier

```
>>> X = [[0], [1], [2], [3]]
>>> y = [0, 0, 1, 1]
>>> from sklearn.neighbors import KNeighborsClassifier
>>> neigh = KNeighborsClassifier(n_neighbors=3)
>>> neigh.fit(X, y)
KNeighborsClassifier(...)
>>> print(neigh.predict([[1.1]]))
[0]
>>> print(neigh.predict_proba([[0.9]]))
[[0.666... 0.333...]]
```

```
[ ]  X = [[0, 0], [1, 1], [2, 1.5], [3, 3]]
     y = [0, 0, 1, 1]
     from sklearn.neighbors import KNeighborsClassifier
     neigh = KNeighborsClassifier(n_neighbors = 1)
     neigh.fit(X, y)

     KNeighborsClassifier(n_neighbors=1)
```

```
[ ]  print(neigh.predict([[1.1, 1.1], [1.5, 1.5], [1.9, 1.9]]))

     [0 1 1]
```

```
[ ]  print(neigh.predict_proba([[1.1, 1.1], [1.5, 1.5], [1.9, 1.9]]))

     [[1. 0.]
      [0. 1.]
      [0. 1.]]
```

https://scikit-learn.org/stable/modules/generated/sklearn.neighbors.KNeighborsClassifier.html?highlight=kneighborsclassifier#sklearn.neighbors.KNeighborsClassifier

파일 입출력

```
H,W,T
130.1,30.7,1
120.5,29.2,1
127.3,25.4,1
122.9,23.0,1
126.0,25.8,1
152.8,49.9,0
155.9,46.2,0
158.5,60.0,0
156.6,62.2,0
150.1,49.4,0
```

health.csv

```
health = []
with open("data/health.csv", "r") as file:
    lines = file.readlines()[1:]
    for line in lines:
        h, w, t = line.strip().split(",")
        health.append([float(h), float(w)])
print(health)
```

```
health = []
with open("data/health.csv", "r") as file:
    lines = file.readlines()[1:]
    for line in lines:
        health.append(list(map(float, line.strip().split(","))))
print(health)
```

```
health = [list(map(float, i.strip().split(','))) for i in open('data/health.csv').readlines()[1:]]
print(health)
```

파이썬 코드

```
import pandas as pd
health = pd.read_csv("data/health.csv")
print(health.values)
```

```
[[130.1  30.7   1. ]
 [120.5  29.2   1. ]
 [127.3  25.4   1. ]
 [122.9  23.    1. ]
 [126.   25.8   1. ]
 [152.8  49.9   0. ]
 [155.9  46.2   0. ]
 [158.5  60.    0. ]
 [156.6  62.2   0. ]
 [150.1  49.4   0. ]]
```

```
[[130.1, 30.7], [120.5, 29.2], [127.3, 25.4], [122.9, 23.0], [126.0, 25.8], [152.8, 49.9], [155.9, 46.2], [158.5, 60.0], [156.6, 62.2], [150.1, 49.4]]
```

실행결과

어린이와 청소년 구분

```python
w = []; h = []; t = []
with open("data/health.csv", "r") as file:
    lines = file.readlines()[1:]
    for line in lines:
        a, b, c = line.strip().split(",")
        h.append(float(a)) # 키
        w.append(float(b)) # 몸무게
        t.append(int(c))   # 어린이/청소년
data = [[x, y] for x, y in zip(h, w)]                # 리스트 생성 [키, 몸무게]
# data3 = [[x, y, z] for x, y, z in zip(h, w, t)]    # 리스트 생성 [키, 몸무게, 정답]
# ch_h = [x for x, y, z in data3 if z == 1]          # 어린이 키
# ch_w = [y for x, y, z in data3 if z == 1]          # 어린이 몸무게
# ad_h = [x for x, y, z in data3 if z == 0]          # 청소년 키
# ad_w = [y for x, y, z in data3 if z == 0]          # 청소년 몸무게
```

```python
from sklearn.neighbors import KNeighborsClassifier
neighbor = 3 # int(input("how many points?"))
kn = KNeighborsClassifier(n_neighbors = neighbor, p = 2)
kn.fit(data, t)
print("Eval:", kn.score(data, t))

# import random
# test_h = random.randrange(120, 160)
# test_w = random.randrange(20, 60)
test_h = 150; test_w = 29
print("Test:", test_h, test_w, "=>", kn.predict([[test_h, test_w]]))
print("Prob:", kn.predict_proba([[test_h, test_w]]))
```

```
Eval: 1.0
Test: 150 29 => [0]
Prob: [[0.66666667 0.33333333]]
```

실행결과

health.csv

```
H,W,T
130.1,30.7,1
120.5,29.2,1
127.3,25.4,1
122.9,23.0,1
126.0,25.8,1
152.8,49.9,0
155.9,46.2,0
158.5,60.0,0
156.6,62.2,0
150.1,49.4,0
```

어린이

청소년

pandas.read_csv()

```
In [1]: import pandas as pd

In [2]: from io import StringIO

In [3]: data = "col1,col2,col3\na,b,1\na,b,2\nc,d,3"

In [4]: pd.read_csv(StringIO(data))
Out[4]:
  col1 col2  col3
0    a    b     1
1    a    b     2
2    c    d     3
```

문자열=파일

```
w = []; h = []; t = []
with open("data/health.csv", "r") as file:
    lines = file.readlines()[1:]
    for line in lines:
        a, b, c = line.strip().split(",")
        h.append(float(a))  # 키
        w.append(float(b))  # 몸무게
        t.append(int(c))    # 어린이/청소년
data = [[x, y] for x, y in zip(h, w)]            # 리스트 생성 [키, 몸무게]
```

```
from sklearn.neighbors import KNeighborsClassifier
neighbor = 3 # int(input("how many points?"))
kn = KNeighborsClassifier(n_neighbors = neighbor, p = 2)
kn.fit(data, t)
print("Eval:", kn.score(data, t))
```

```
test_h = 150; test_w = 29
print("Test:", test_h, test_w, "=>", kn.predict([[test_h, test_w]]))
print("Prob:", kn.predict_proba([[test_h, test_w]]))
```

```
import pandas as pd
data3 = pd.read_csv("data/health.csv")
data = data3[["H", "W"]]
t = data3["T"]
```

```
from sklearn.neighbors import KNeighborsClassifier
neighbor = 3 # int(input("how many points?"))
kn = KNeighborsClassifier(n_neighbors = neighbor, p = 2)
kn.fit(data, t)
print("Eval:", kn.score(data, t))
```

```
test_h = 150; test_w = 29
test = pd.DataFrame([[test_h, test_w]], columns = ["H", "W"])
print("Test:", test_w, test_h, "=>", kn.predict(test))
print("Prob:", kn.predict_proba(test))
```

https://pandas.pydata.org/docs/user_guide/io.html#csv-text-files

참고자료

- 지능기전공학부 최유경 교수님 자료, https://github.com/sejongresearch/2021.MachineLearning
- 코랩(Colab), https://colab.research.google.com/
- 파이썬(Python), https://www.python.org/doc/
- 사이킷런(sckit-learn), https://scikit-learn.org/stable/index.html
- 판다스(pandas), https://pandas.pydata.org/
- 맷플롯립(matplotlib), https://matplotlib.org/
- 씨본(seaborn), https://seaborn.pydata.org/
- 캐글(Kaggle), https://www.kaggle.com/
- 넘파이(numpy), https://numpy.org/doc/stable/
- 스택오퍼플러우(stackoverflow), https://stackoverflow.com/