

이웃회귀

# 회귀(Regression)

- 연속적인 숫자를 예측하는 것
- 변수 간의 관련성을 분석하여, 측정된 데이터로부터 예측 수행
- 독립변수로부터 수학적식을 통하여 종속변수의 값을 추정

이 세가지가 꼭어지고 어느게결  
이냐고 물어볼때의 순서

월	평균기온	선풍기 판매수	난방기 판매수
봄	12	80	50
여름	23	100	5
가을	13	50	70
겨울	1	?	?

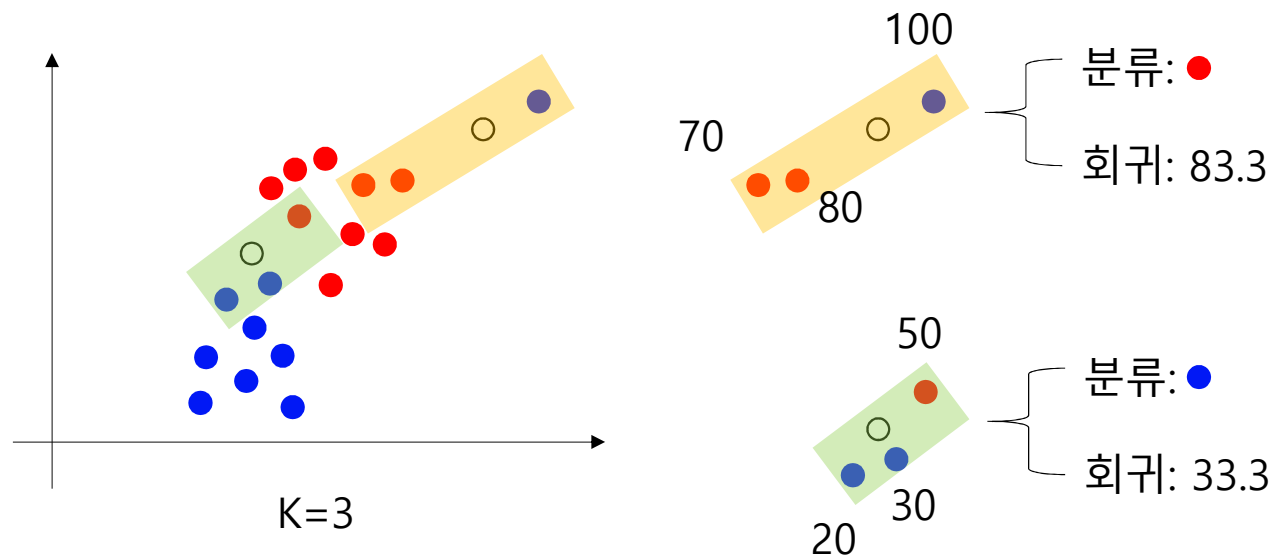
독립변수

종속변수

회귀

# KNN 이웃회귀

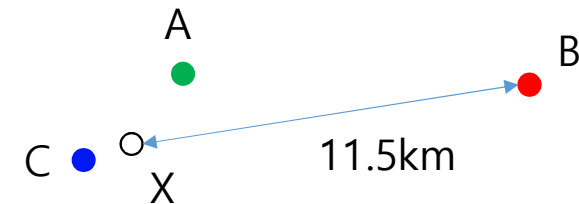
- KNN 분류와 유사하지만, 클래스를 결정하는 것이 아닌 임의의 값을 예측하는 문제
- 가장 가까운 이웃 K개의 점을 결정하여, 그들의 값들로부터 예측값을 계산 (ex, 평균)



# 단순회귀, 가중회귀

- 단순회귀: 가장 가까운 이웃들의 값들을 단순히 평균하여 추정하는 방식
- 가중회귀: 각 이웃이 얼마나 가까이 있는지에 따라 가중 평균을 구해 거리가 가까울수록 데이터가 더 유사할 것이라고 보고 가중치를 부여하는 방식

영화	리뷰점수	거리(km)
A	5.0	3.2
B	6.8	11.5
C	9.0	1.1



X의 리뷰점수?

$$\frac{5.0 + 6.8 + 9.0}{1 + 1 + 1} = 6.9$$

단순회귀

$$\frac{\frac{5.0}{3.2} + \frac{6.8}{11.5} + \frac{9.0}{1.1}}{\frac{1}{3.2} + \frac{1}{11.5} + \frac{1}{1.1}} = 7.9$$

가중회귀

# 데이터셋

```
import pandas as pd
data3 = pd.read_csv("data/health.csv")
h = data3[["H"]]
w = data3[["W"]]
# print("DataFrame", x.shape, y.shape)
# print(x); print("\n"); print(y)
# h = data3["H"]
# w = data3["W"]
# print("Series", x.shape, y.shape)
```

```
DataFrame (10, 1) (10, 1)
```

```
Series (10,) (10,)
```

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

제거

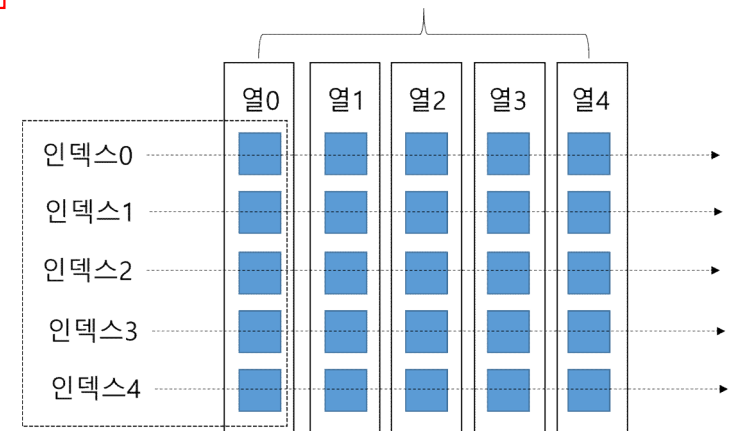
어린이

청소년

	H
0	130.1
1	120.5
2	127.3
3	122.9
4	126.0
5	152.8
6	155.9
7	158.5
8	156.6
9	150.1

	W
0	30.7
1	29.2
2	25.4
3	23.0
4	25.8
5	49.9
6	46.2
7	60.0
8	62.2
9	49.4

데이터프레임

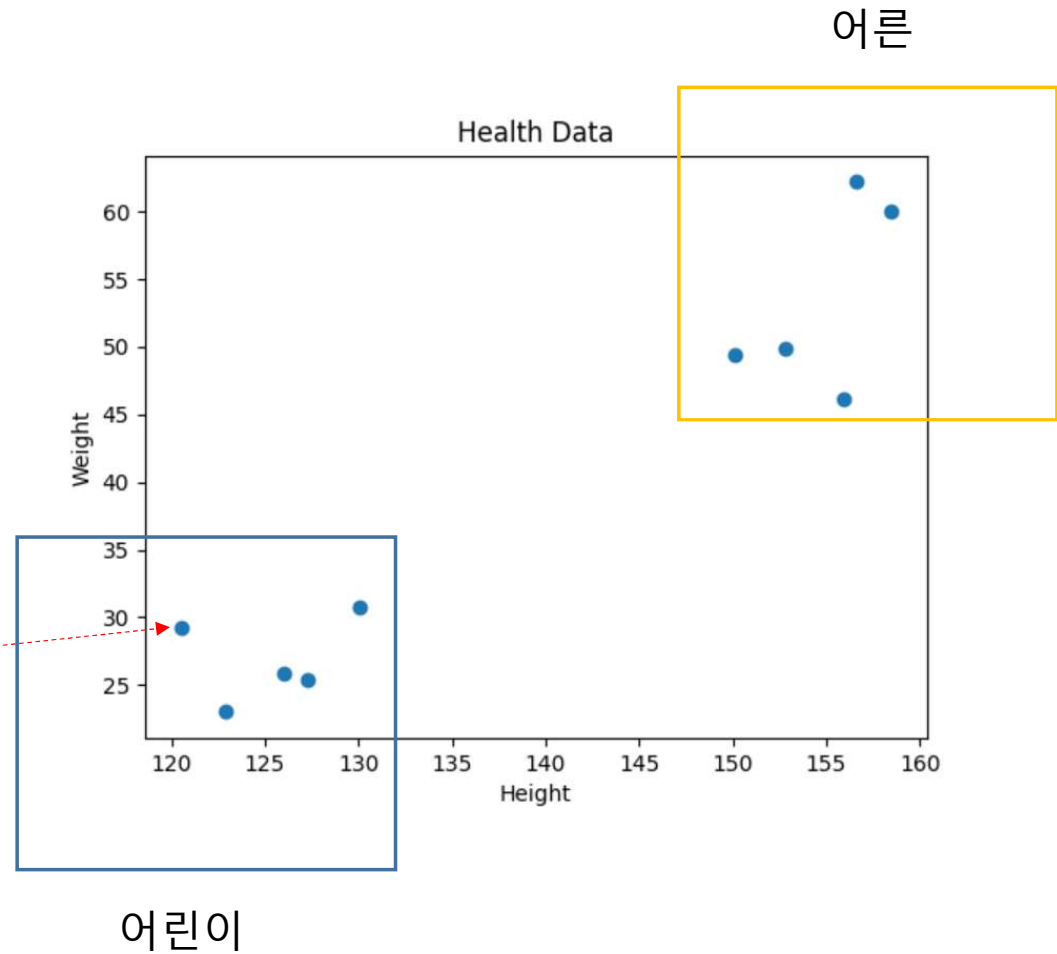


시리즈

# 데이터 시각화

```
import matplotlib.pyplot as plt
plt.scatter(h, w)
plt.xlabel("Height")
plt.ylabel("Weight")
plt.title("Health Data")
plt.show()
```

	H		W
0	130.1	0	30.7
1	120.5	1	29.2
2	127.3	2	25.4
3	122.9	3	23.0
4	126.0	4	25.8
5	152.8	5	49.9
6	155.9	6	46.2
7	158.5	7	60.0
8	156.6	8	62.2
9	150.1	9	49.4



# sklearn.neighbors.KNeighborsRegressor

```
class sklearn.neighbors.KNeighborsRegressor(n_neighbors=5, *, weights='uniform', algorithm='auto', leaf_size=30, p=2, metric='minkowski', metric_params=None, n_jobs=None)
```

[\[source\]](#)

Regression based on k-nearest neighbors.

→ 가중치

→ 유클리드 거리

```
>>> 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...]]
```

y: 클래스

```
>>> X = [[0], [1], [2], [3]]
>>> y = [0, 0, 1, 1]
>>> from sklearn.neighbors import KNeighborsRegressor
>>> neigh = KNeighborsRegressor(n_neighbors=2)
>>> neigh.fit(X, y)
KNeighborsRegressor(...)
>>> print(neigh.predict([[1.5]]))
[0.5]
```

y: 값

# 예측

```
from sklearn.model_selection import train_test_split
train_data, test_data, train_target, test_target = train_test_split(
    h, w, test_size = 0.2, random_state = 42)

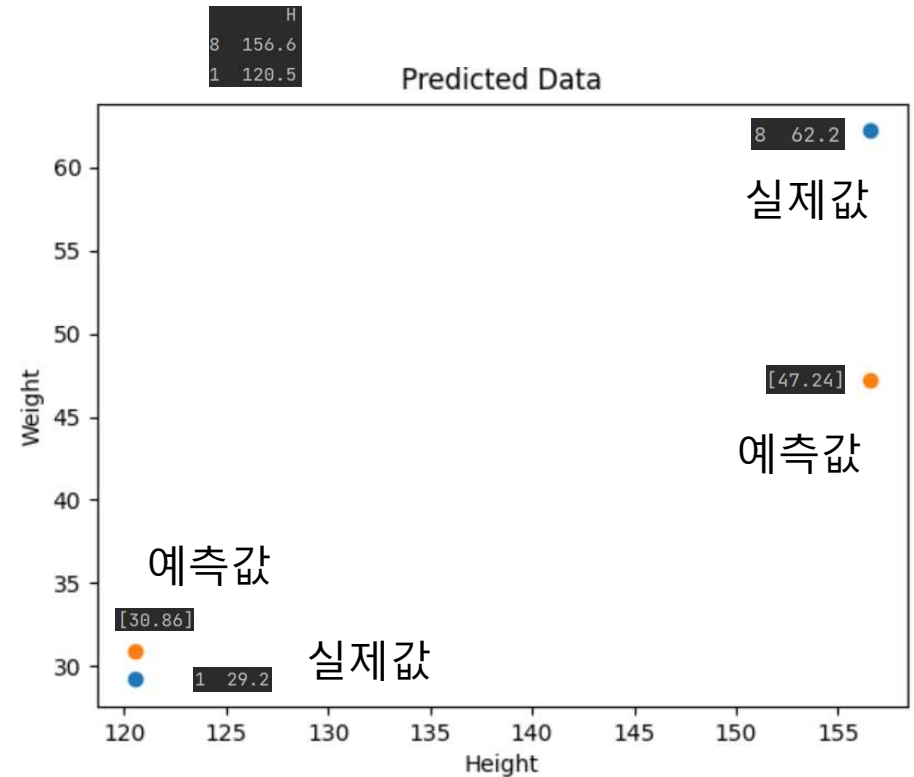
from sklearn.neighbors import KNeighborsRegressor
knr = KNeighborsRegressor()
knr.fit(train_data, train_target)
print("Train-Eval:", knr.score(train_data, train_target))
print("Test-Eval :", knr.score(test_data, test_target))
```

```
Train-Eval: 0.7913472691260918
Test-Eval : 0.5839169880624423
```

```
test_pred = knr.predict(test_data)
# print(test_data)
# print(test_target)
# print(test_pred)
import matplotlib.pyplot as plt
```

구현

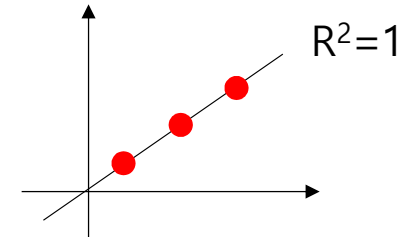
```
plt.show()
```





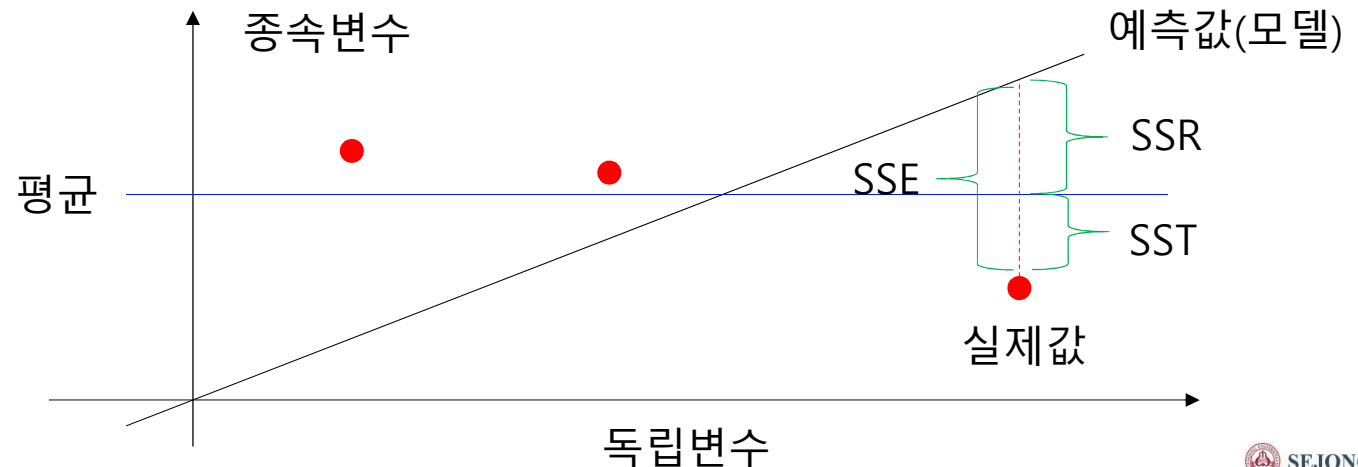
# 결정계수(Coefficient of Determination)

- 회귀 문제에서의 성능 측정 도구 (결정계수= $R^2$ ,  $0 \leq R^2 \leq 1$ )
- 독립변수로부터 설명 가능한 종속변수의 변동 비율
  - SST(Total Sum of Squares): 총 변동 (예측값에 대한 편차 제곱합)
  - SSE(Error Sum of Squares): 직선으로 설명 불가능한 변동 (예측값 오차에 대한 제곱합)
  - SSR(Regression Sum of Squares): 직선으로 설명 가능한 변동 (예측값과 평균값의 차이에 대한 제곱합)

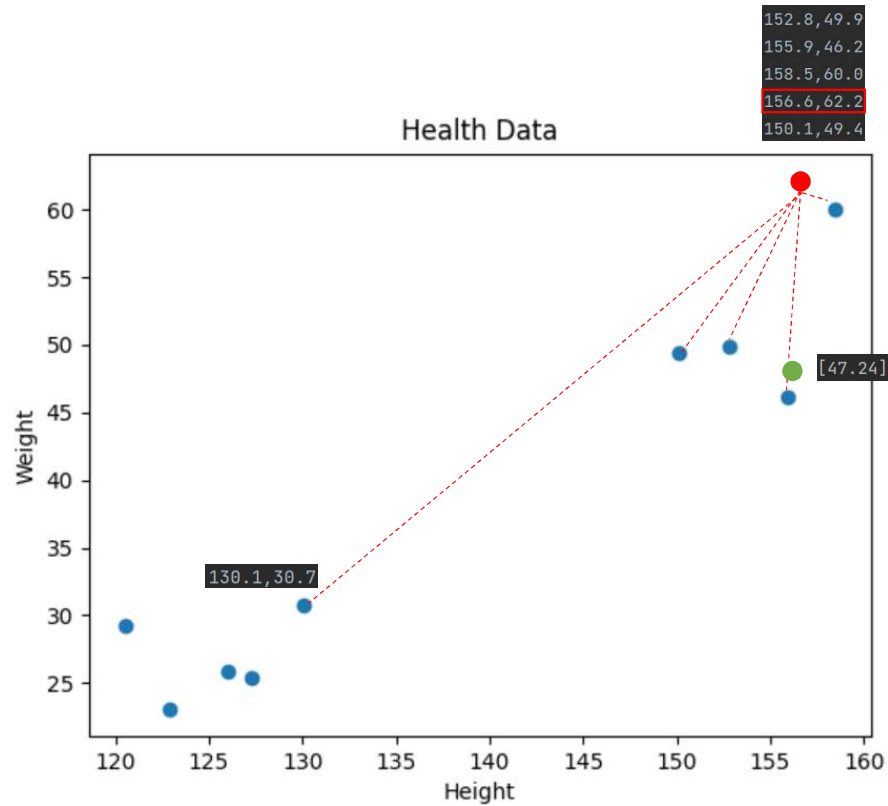


$$R^2 = \frac{SSR}{SST} = 1 - \frac{SSE}{SST}$$

$$= 1 - \frac{\sum(\text{정답} - \text{예측})^2}{\sum(\text{정답} - \text{평균})^2}$$



# 결과분석

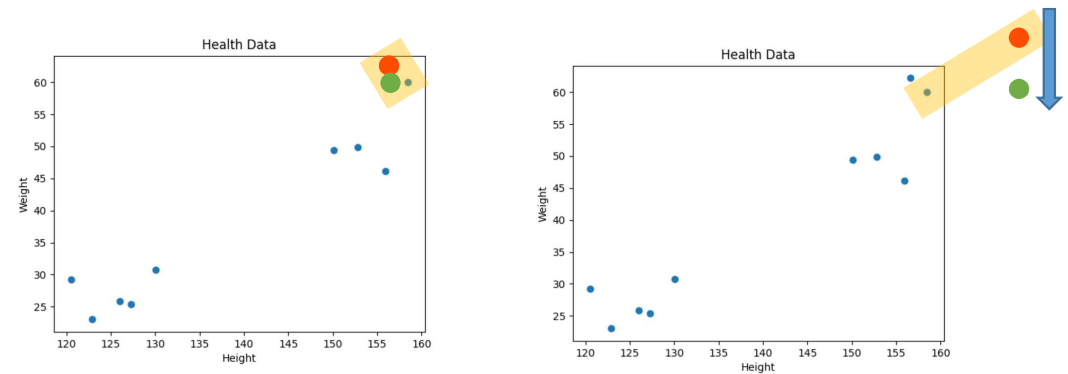


```
class sklearn.neighbors.KNeighborsRegressor(n_neighbors=5, *, weights='uniform', algorithm='auto', leaf_size=30, p=2,
metric='minkowski', metric_params=None, n_jobs=None)
```

[source]

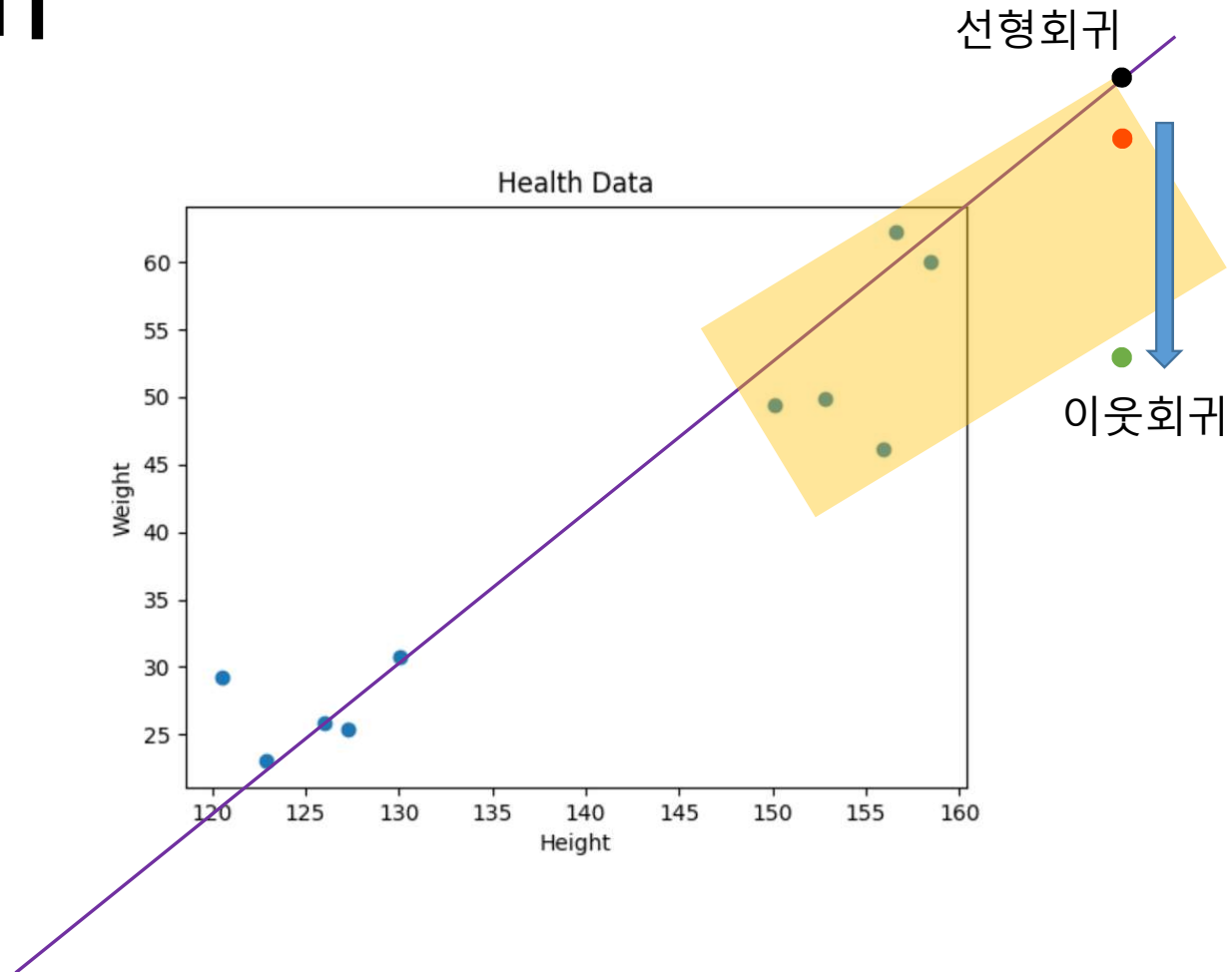
Regression based on k-nearest neighbors.

$$Pred = \frac{49.9 + 46.2 + 60.0 + 49.4 + 30.7}{5} = 47.24$$



K=1 경우, 문제점

# 선형회귀



# 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/>