

# **k Nearest Neighbor**

PinkLAB Edu

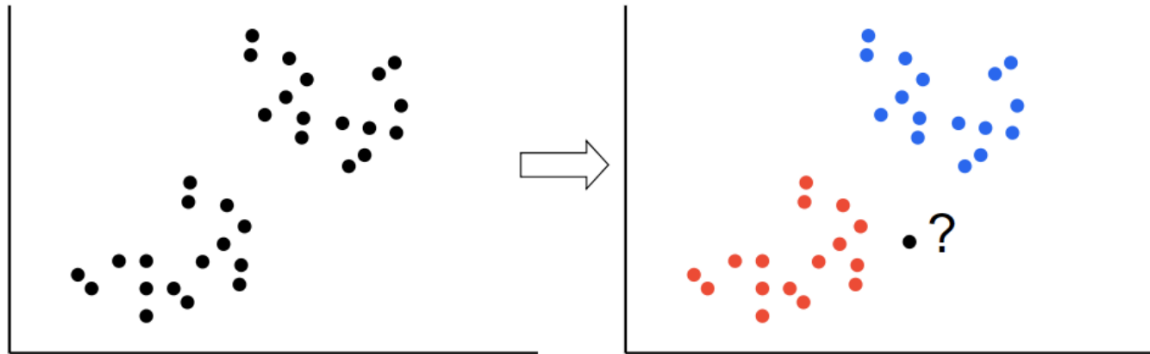
10 November, 2025

# Table of Contents

1 k Nearest Neighbor .....	3
1.1 kNN이란 .....	3
1.2 좀더 상세히 .....	3
1.3 k=5로 설정하면 5번째까지 가까운 데이터.....	4
1.4 k값에 따라 결과값이 바뀔 수 있다 .....	5
1.5 거리를 계산하는 것? - 유클리드 기하.....	5
1.6 단위에 따라 바뀔 수도 있다 - 표준화 필요.....	6
1.7 장단점 .....	6
2 연습.....	7
2.1 iris 데이터 .....	7
2.2 kNN 학습 .....	7
2.3 accuracy .....	7
2.4 간단한 성과 .....	8

# 1 k Nearest Neighbor

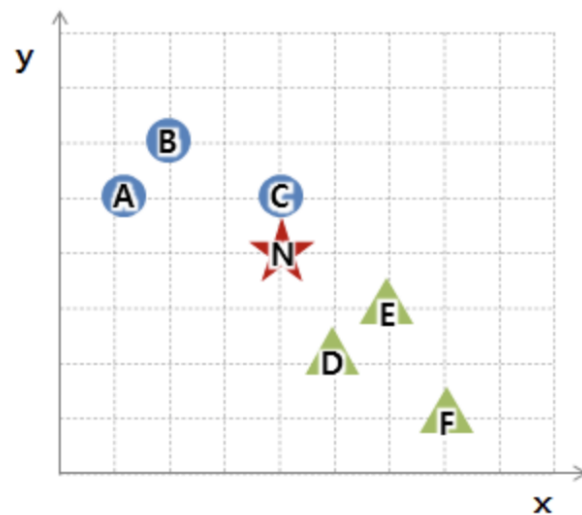
## 1.1 kNN이란



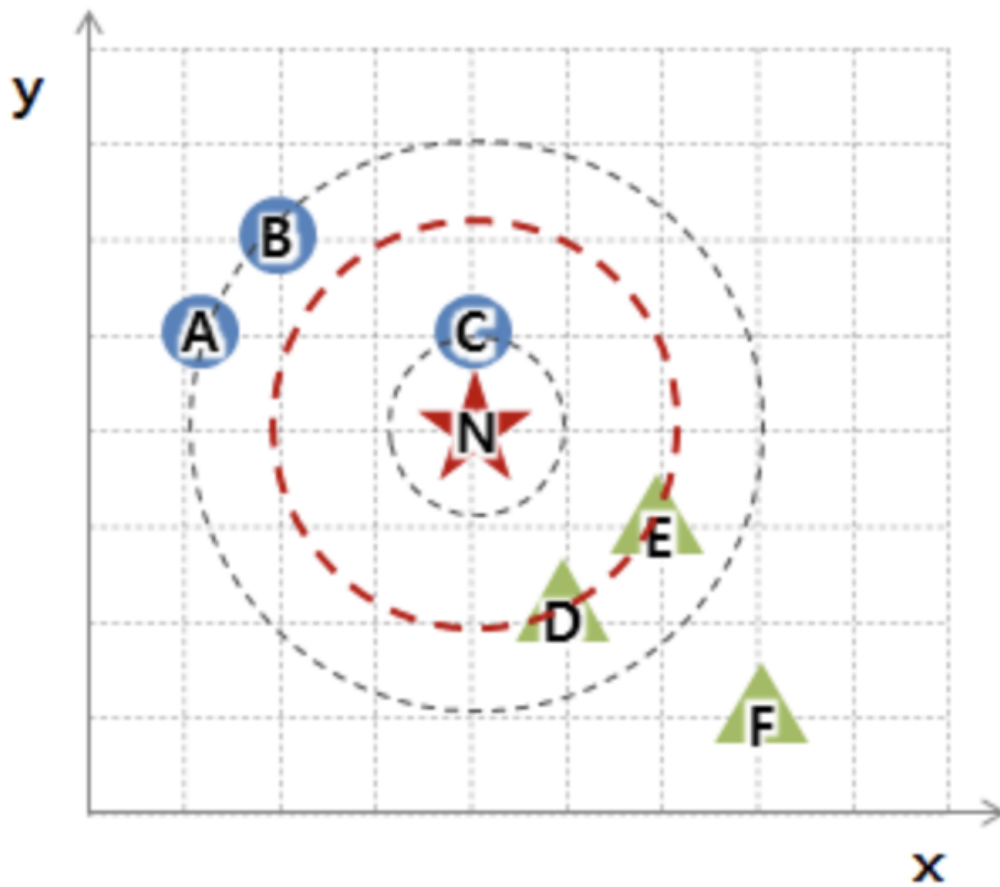
- 새로운 데이터가 있을 때, 기존 데이터의 그룹 중 어떤 그룹에 속하는지를 분류하는 문제
- k는 몇 번째 가까운 데이터까지 볼 것인가를 정하는 수치

## 1.2 좀더 상세히

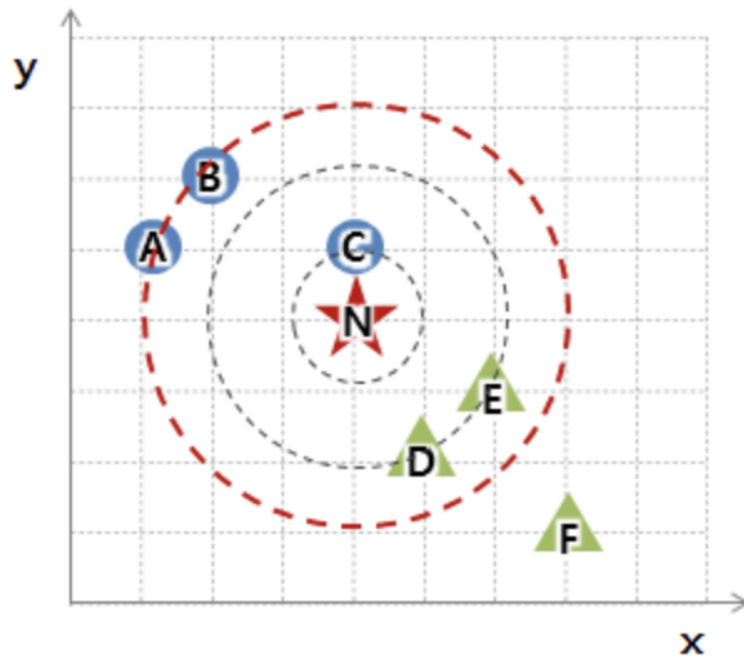
데이터	x좌표	y좌표	그룹
A	1	5	●
B	2	6	●
C	4	5	●
D	5	2	▲
E	6	3	▲
F	7	1	▲
N	4	4	?



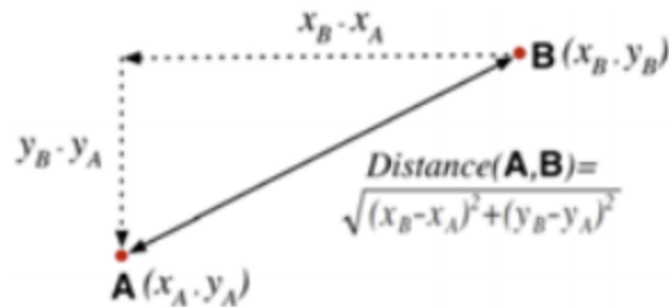
### 1.3 k=5로 설정하면 5번째까지 가까운 데이터



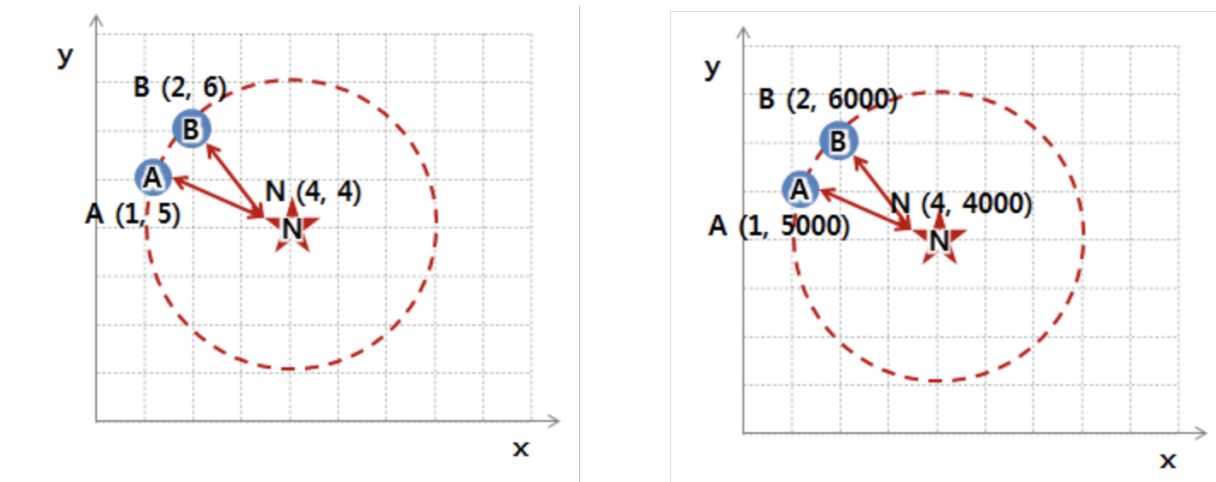
## 1.4 k값에 따라 결과값이 바뀔 수 있다



## 1.5 거리를 계산하는 것? - 유클리드 기하



## 1.6 단위에 따라 바뀔 수도 있다 - 표준화 필요



## 1.7 장단점

- 실시간 예측을 위한 학습이 필요치 않다.
- 결국 속도가 빨라진다.
- 고차원 데이터에는 적합하지 않다.

## 2 연습

### 2.1 iris 데이터

```
from sklearn.datasets import load_iris

iris = load_iris()
```

Python

```
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(
    iris.data, iris.target, test_size=0.2, random_state=13, stratify=iris.target
)
```

Python

### 2.2 kNN 학습

```
from sklearn.neighbors import KNeighborsClassifier

knn = KNeighborsClassifier(n_neighbors=5)
knn.fit(X_train, y_train)
```

Python

```
▼ KNeighborsClassifier
KNeighborsClassifier()
```

### 2.3 accuracy

```
from sklearn.metrics import accuracy_score

pred = knn.predict(X_test)
print(accuracy_score(y_test, pred))
```

Python

```
0.9666666666666667
```

## 2.4 간단한 성과

```
from sklearn.metrics import classification_report, confusion_matrix

print(confusion_matrix(y_test, pred))
print(classification_report(y_test, pred))
```

✓ 0.0s

Python

```
[[10  0  0]
 [ 0  9  1]
 [ 0  0 10]]
```

	precision	recall	f1-score	support
0	1.00	1.00	1.00	10
1	1.00	0.90	0.95	10
2	0.91	1.00	0.95	10
accuracy			0.97	30
macro avg	0.97	0.97	0.97	30
weighted avg	0.97	0.97	0.97	30