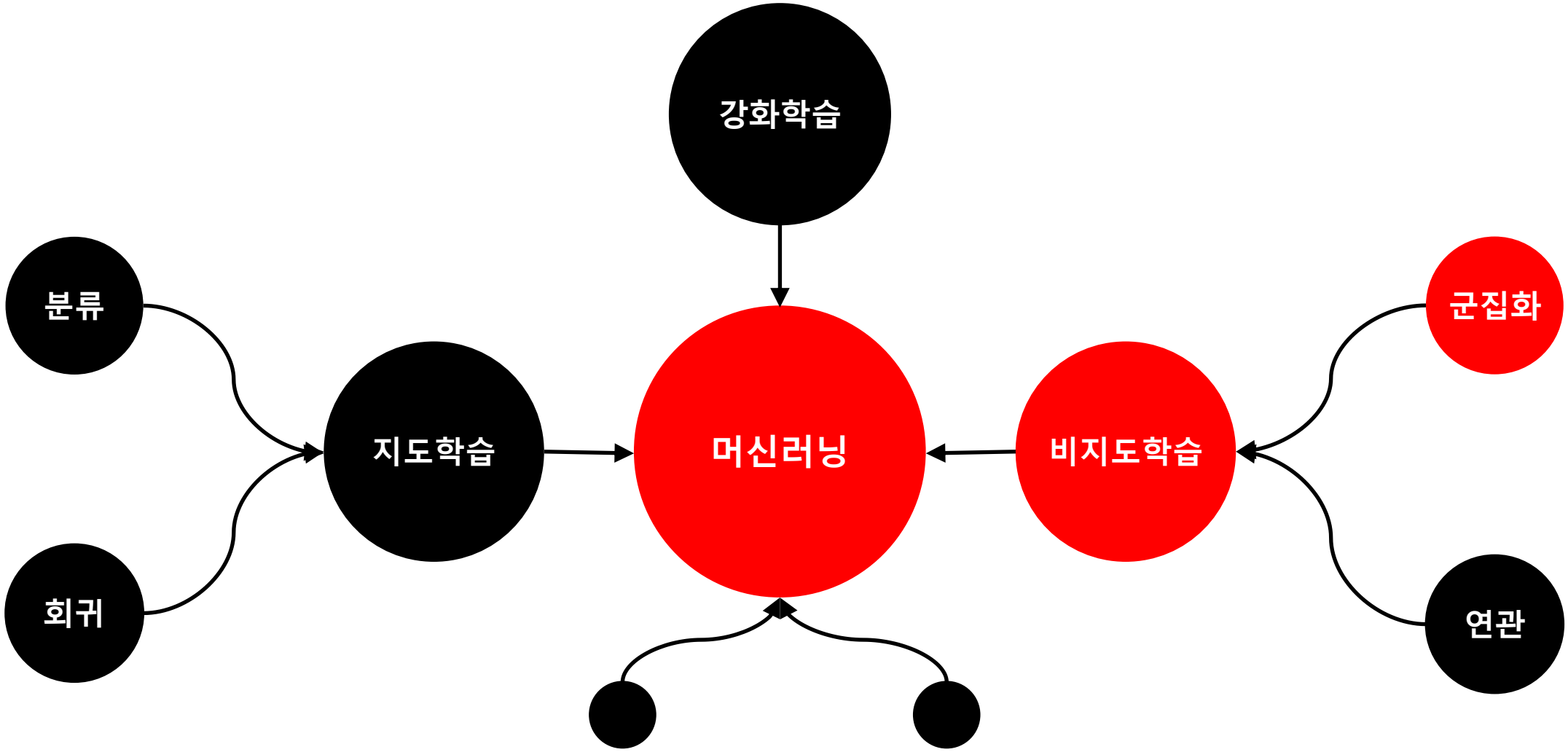
8장_인프런 댓글 분석(군집화, KMeans)

오늘의 모델 : 군집화



How Much Information Does the Machine Need to Predict?

Y LeCun

- "Pure" Reinforcement Learning (cherry)
  - The machine predicts a scalar reward given once in a while.
  - **A few bits for some samples**
- Supervised Learning (icing)
  - The machine predicts a category or a few numbers for each input
  - Predicting human-supplied data
  - **10→10,000 bits per sample**
- Unsupervised/Predictive Learning (cake)
  - The machine predicts any part of its input for any observed part.
  - Predicts future frames in videos
  - **Millions of bits per sample**

- (Yes, I know, this picture is slightly offensive to RL folks. But I'll make it up)

만약 지능이 케이크라면 비지도 학습은 케이크의 본체다.

지도학습은 케이크 본체의 겉에 발린 크림(icing)이고,

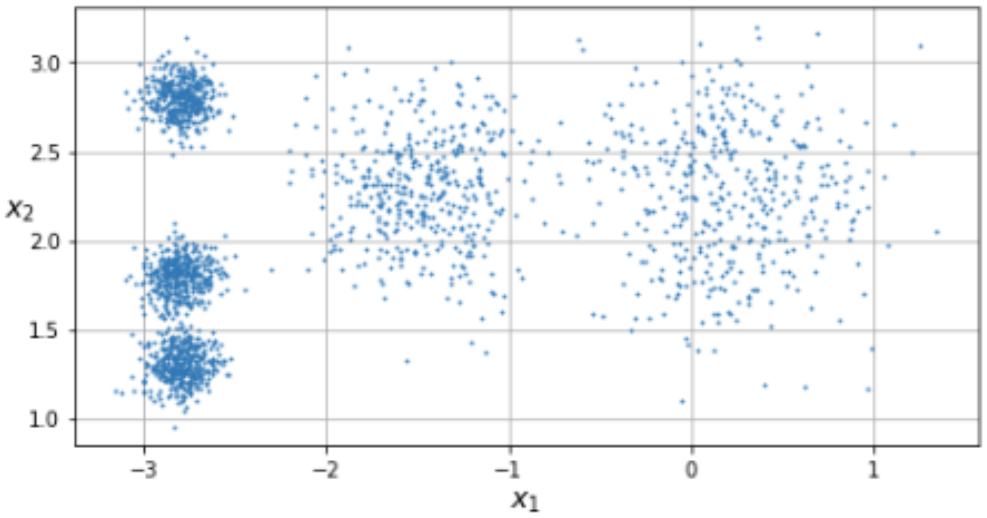
강화학습은 케이크 위의 체리다.

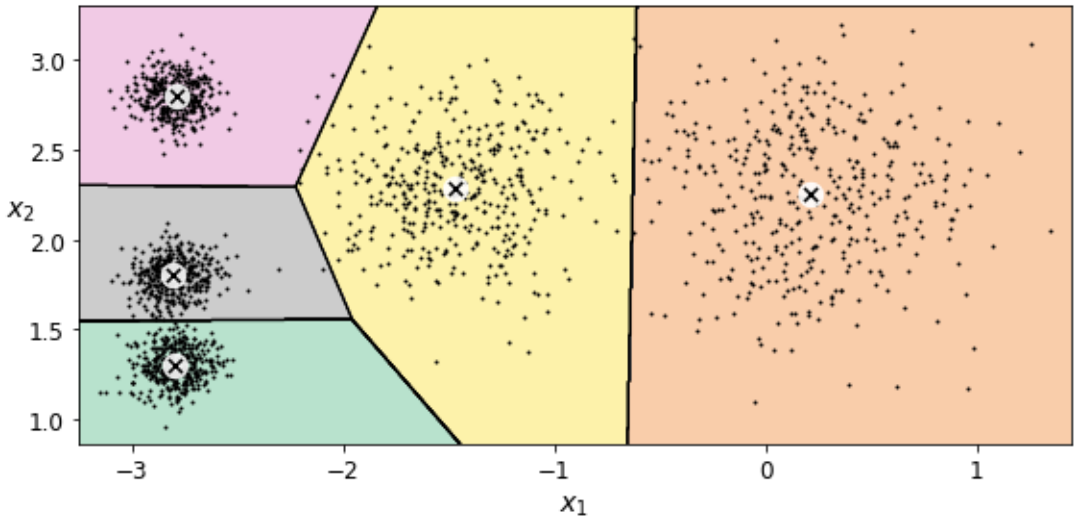
우리는 크림과 체리를 어떻게 만드는지 안다.

그러나 케이크를 만드는 법은 모른다.

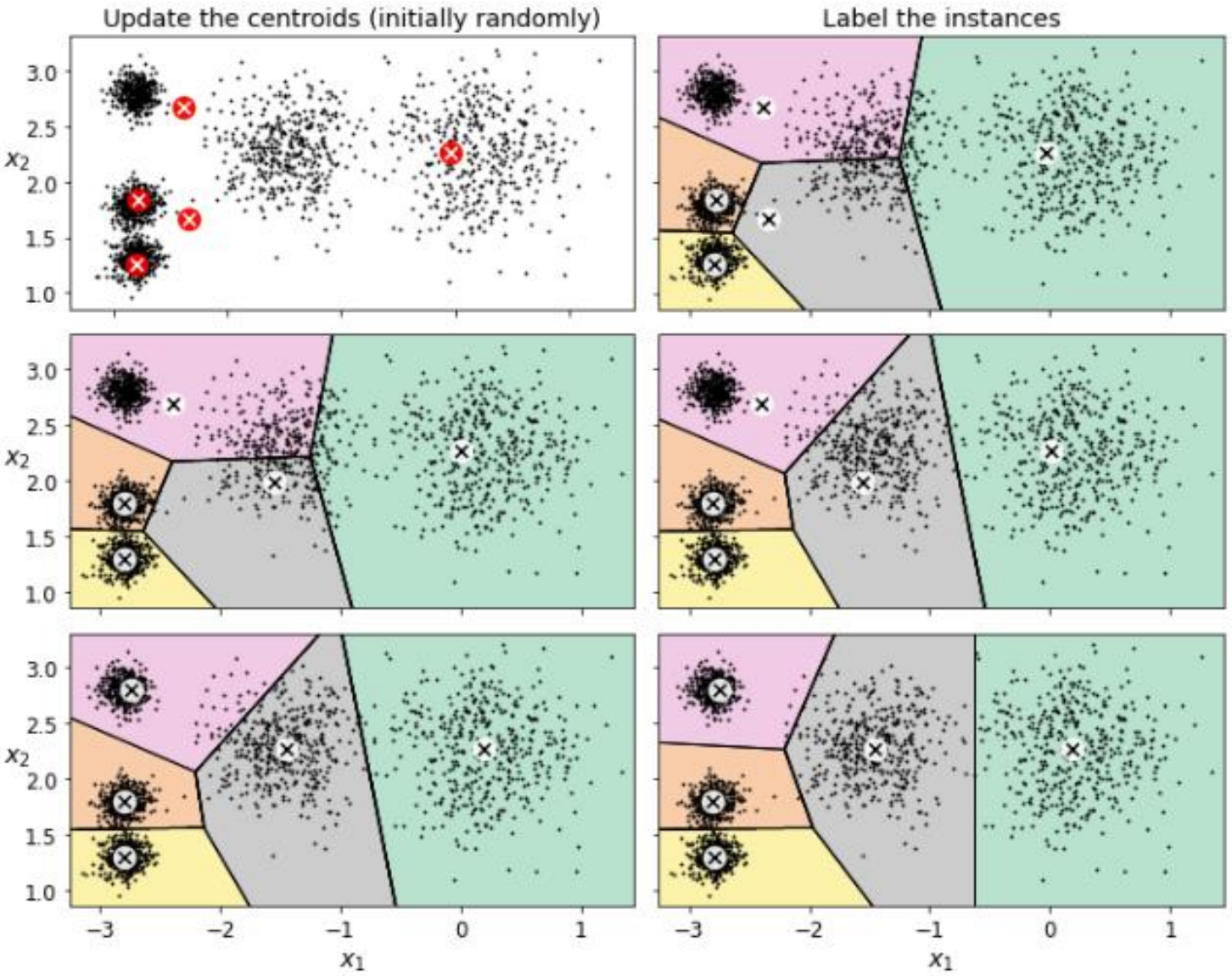
-얀 르쿤, 'NIPS(신경정보처리시스템 학회), 2016

오늘의 모델 : 군집화(Kmeans)





오늘의 모델 : 군집화(Kmeans)

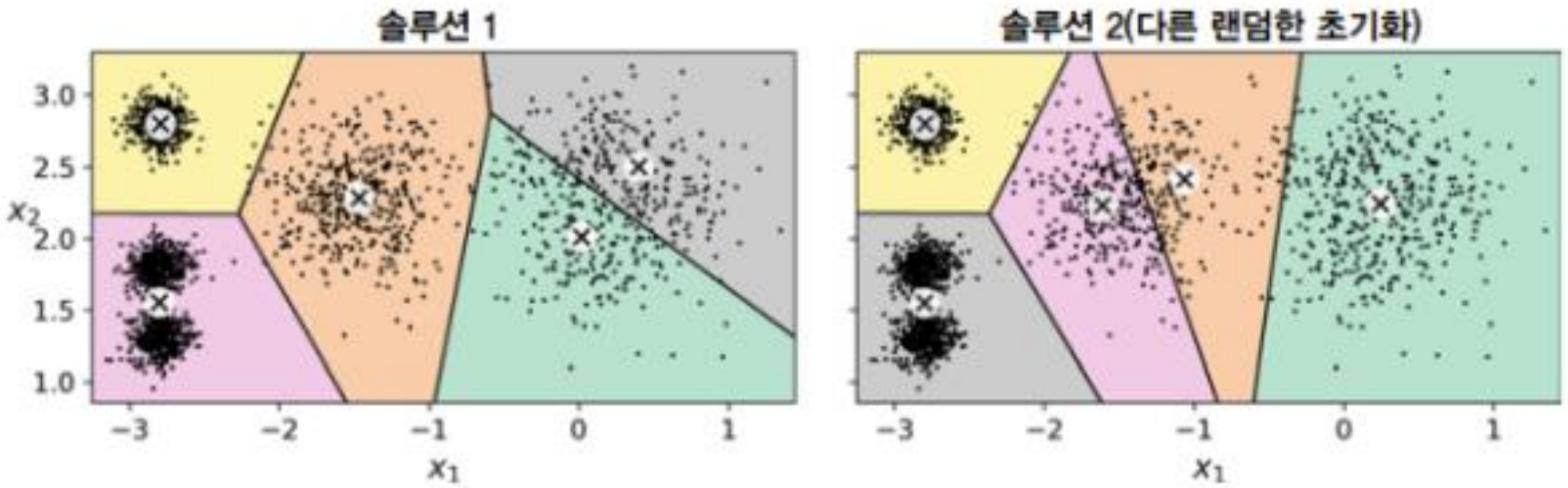


1. 센트로이드를 랜덤하게 초기화
2. 샘플에 레이블 할당
3. 센트로이드 업데이트
4. 샘플에 다시 레이블 할당
5. 최적의 클러스터 찾기

오늘의 모델 : 군집화(Kmeans)

- 센트로이드 초기화 하는 방법

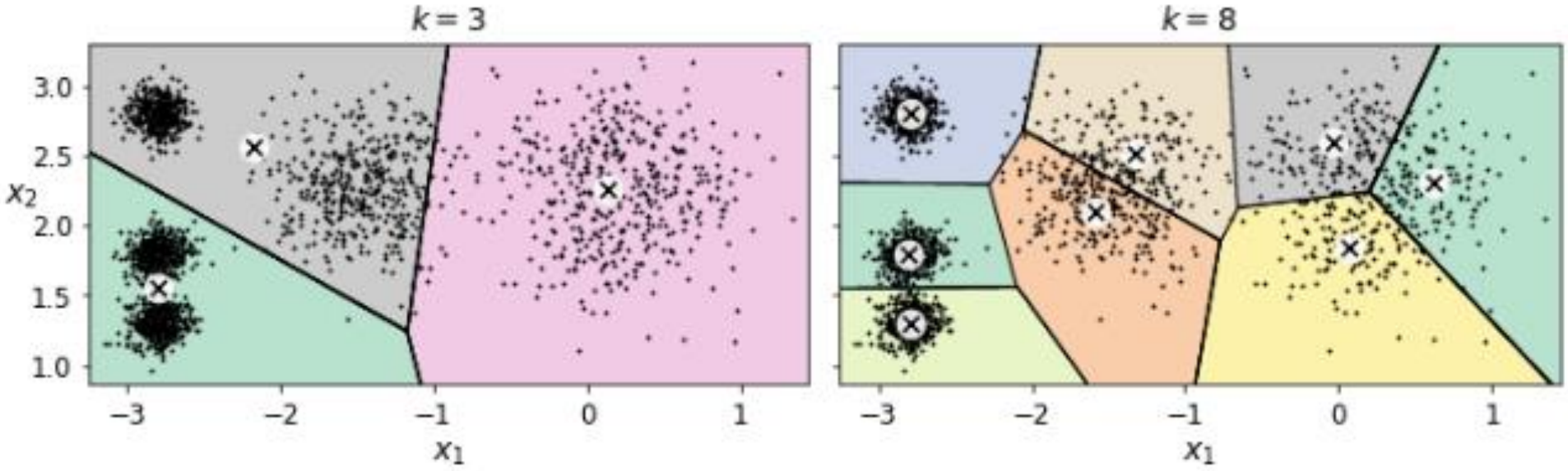
  이너셔(inertia) : 센트로이드를 초기화 하는 모델. 각 샘플과 가장 가까운 센트로이드 사이의 제곱 거리의 합.



운 나쁜 초기화때문에 만들어진 최적이 아닌 솔루션

오늘의 모델 : 군집화(Kmeans)

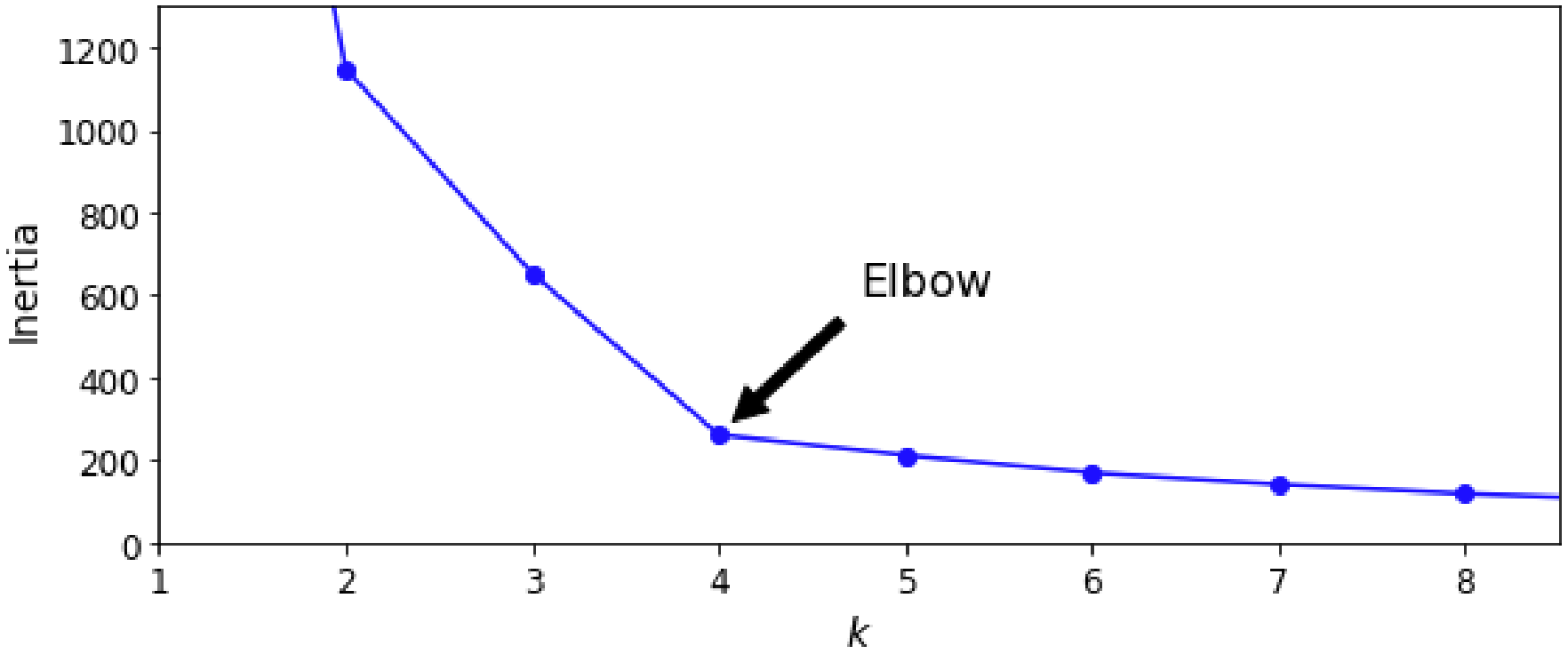
- 최적의 클러스터 개수 찾기



클러스터의 개수가 너무 작으면 별개의 클러스터를 합치고, 너무 크면 하나의 클러스터가 여러 개로 나뉨

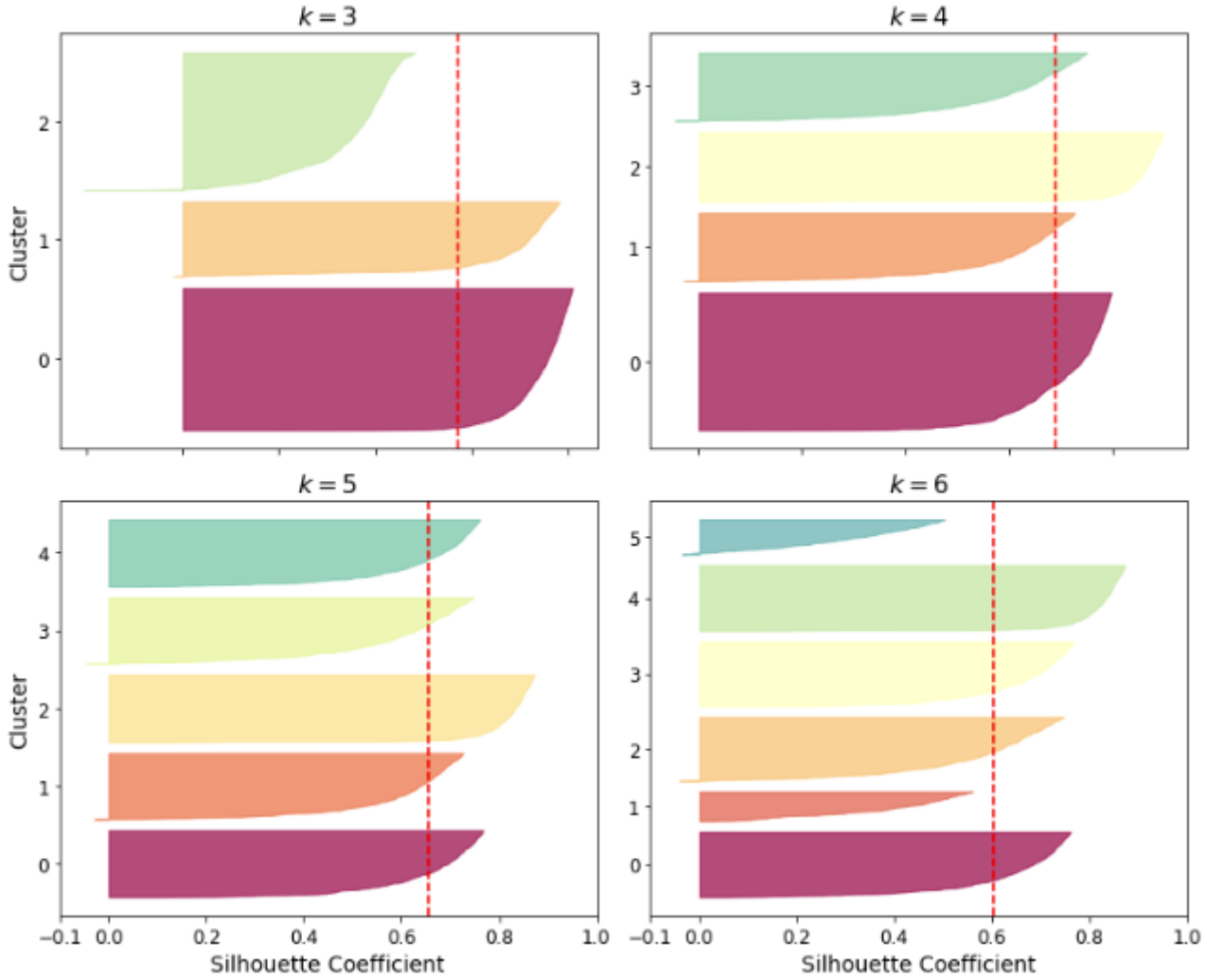
오늘의 모델 : 군집화(Kmeans)

- 최적의 클러스터 개수 찾기 : 엘보우



오늘의 모델 : 군집화(Kmeans)

- 최적의 클러스터 개수 찾기 : 실루엣 점수



클러스터 내부의 평균 거리

가장 가까운 클러스터의 샘플까지 평균거리 이용

우리의 코드 중 살펴볼 부분

```
from sklearn.cluster import KMeans
from tqdm import trange
inertia = []

start = 10
end = 70

# 적절한 클러스터의 갯수를 알기 위해 inertia 값을 구함
# trange 를 통해 시작과 끝 값을 지정해 주면 진행 정도를 알 수 있습니다.
# 학습을 할 때는 feature_tfidf 값을 사용합니다.
# cls.inertia_ 값을 inertia 리스트에 저장합니다.
for i in trange(start, end):
    kmeans = KMeans(n_clusters=i, random_state=42)
    kmeans.fit(feature_tfidf)
    inertia.append(kmeans.inertia_)
```

우리의 코드 중 살펴볼 부분

```
n_clusters = 50
kmeans = KMeans(n_clusters=n_clusters, random_state=42)
kmeans.fit(feature_tfidf)
prediction = kmeans.predict(feature_tfidf)
df["cluster"] = prediction
```

오늘의 모델 : 군집화(Kmeans)

https://experiments.withgoogle.com/ai/drum-machine/view/

끝.