



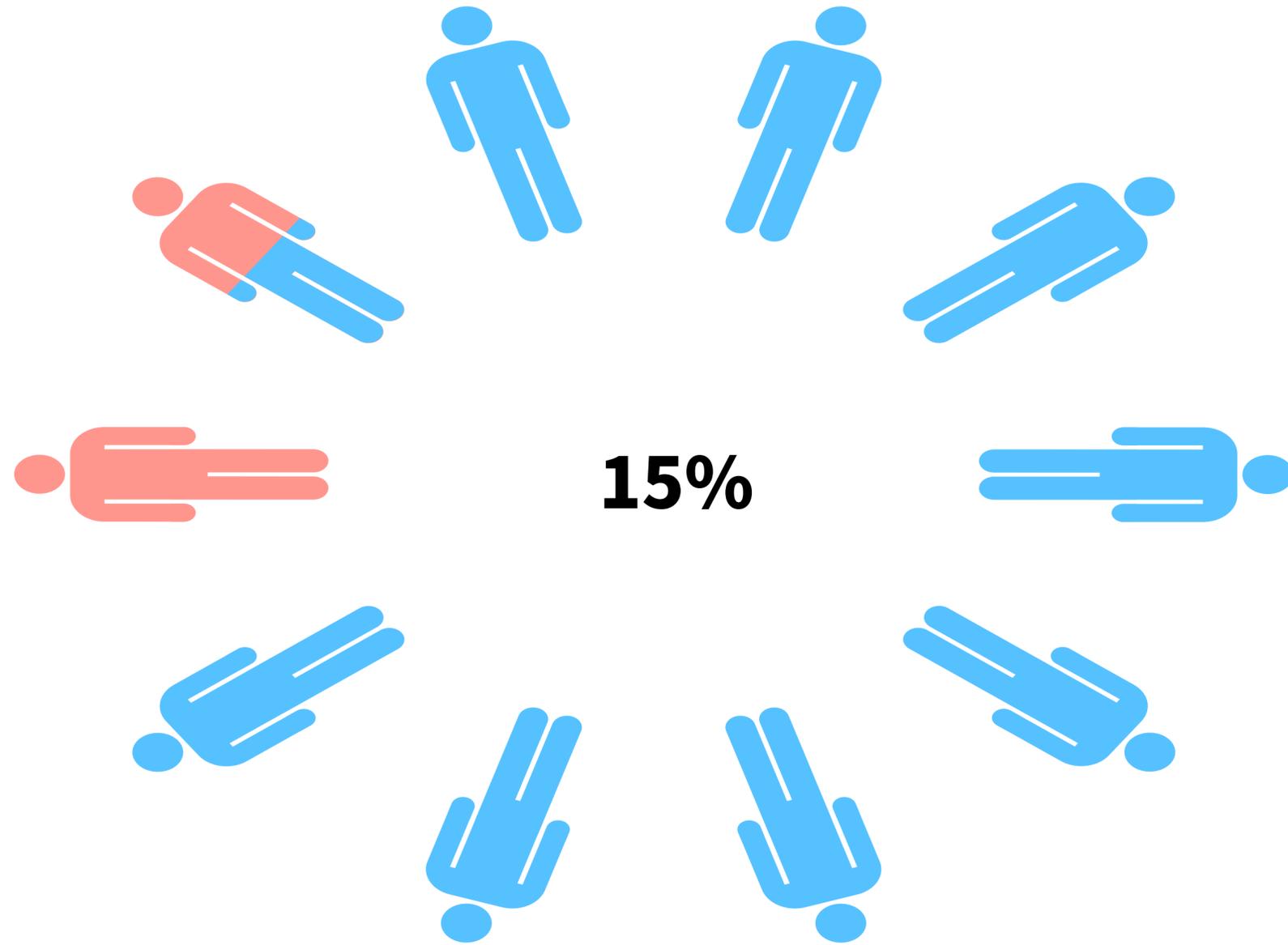
성인이미지 세계에서 살아남는 방법

오창화 [herschel.alway](#)

카카오

- 01 프로젝트의 목적**
- 02 이미지 ID 를 이용한 자동 처리**
- 03 색상 분포도를 이용한 이미지 분류**
- 04 CNN (딥러닝) 을 이용한 이미지 분류**
- 05 Ensemble**
- 06 고민중인 내용**
- 07 정리**

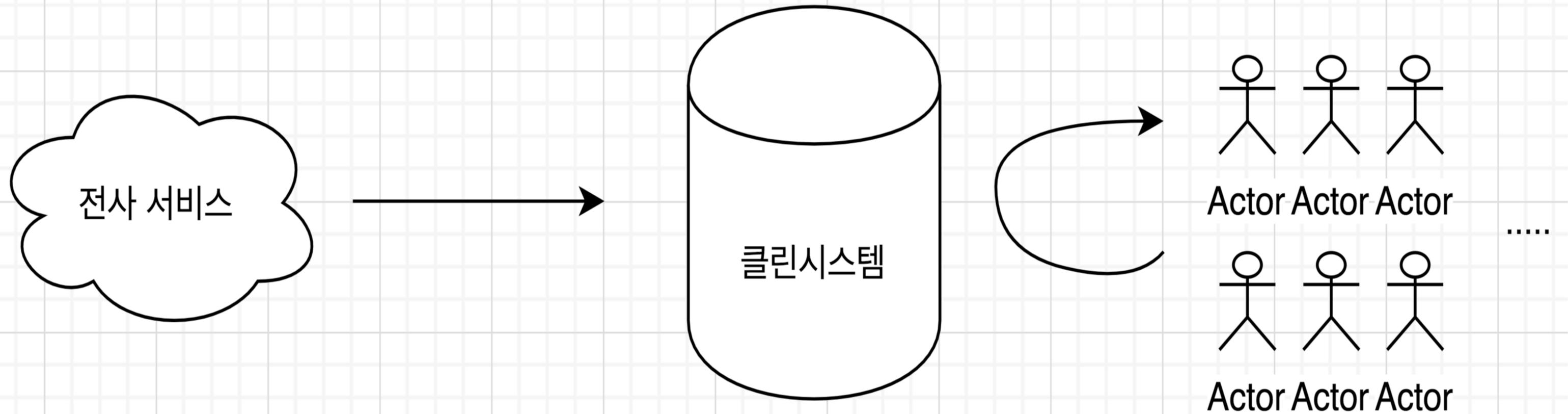
- 01 프로젝트의 목적**
 - 02 이미지 ID 를 이용한 자동 처리**
 - 03 색상 분포도를 이용한 이미지 분류**
 - 04 CNN (딥러닝) 을 이용한 이미지 분류**
 - 05 Ensemble**
 - 06 고민중인 내용**
 - 07 정리**
- => 딥알못 문과 서버 개발자의 딥러닝 분투기**



01 프로젝트의 목적

Why?

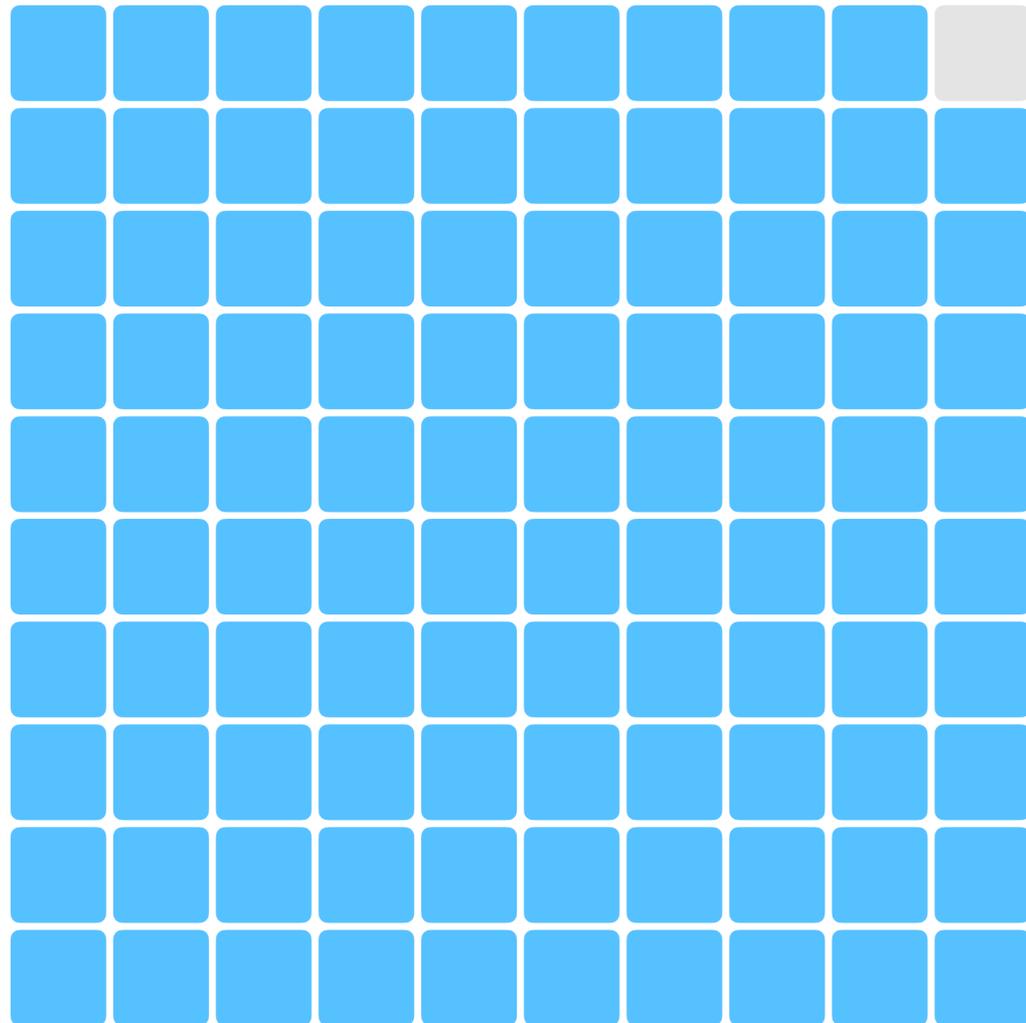
프로젝트의 목적



생산 콘텐츠 대비 리소스 부족

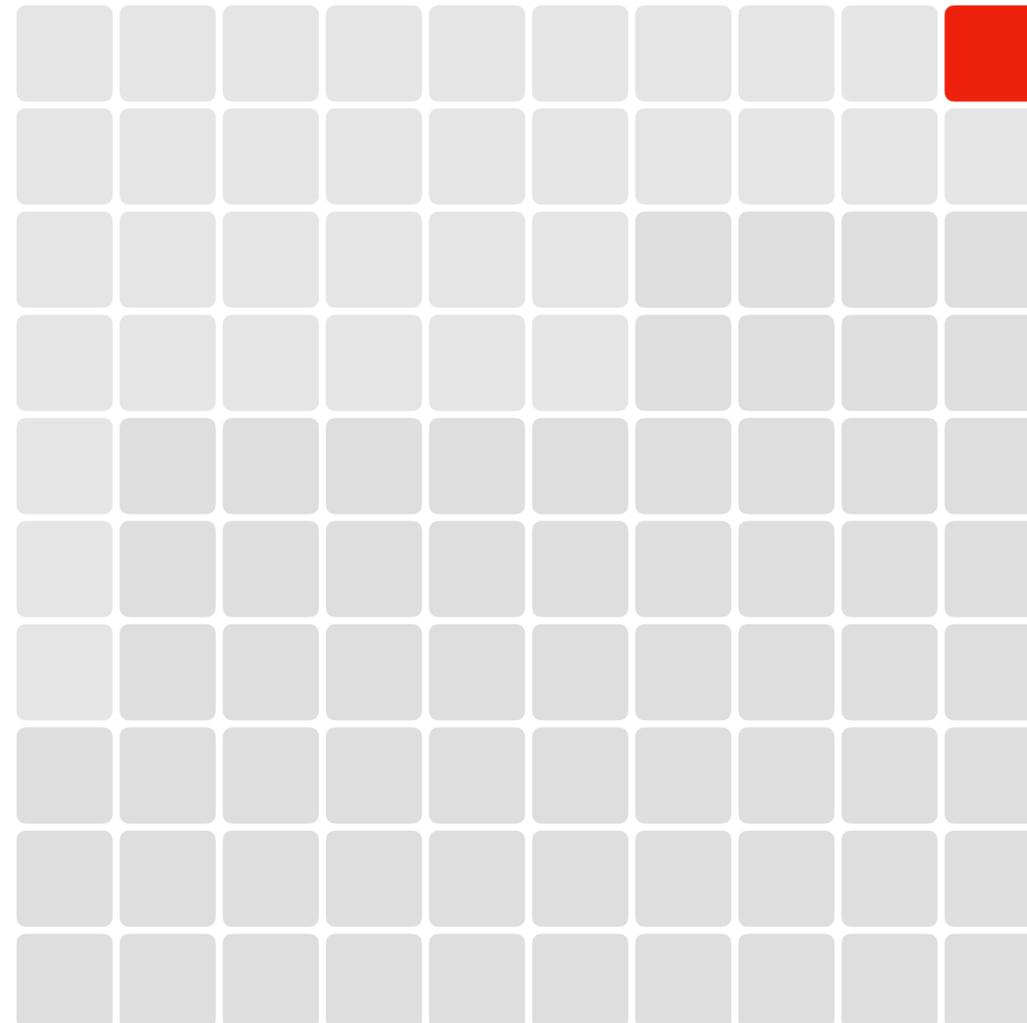
Why?

프로젝트의 목적



99%

정상



1%

규제

데이터의 1% 정도만
실제 규제 대상

Why?

프로젝트의 목적

유해한 이미지를 최대한 빠른 시간에 규제 한다

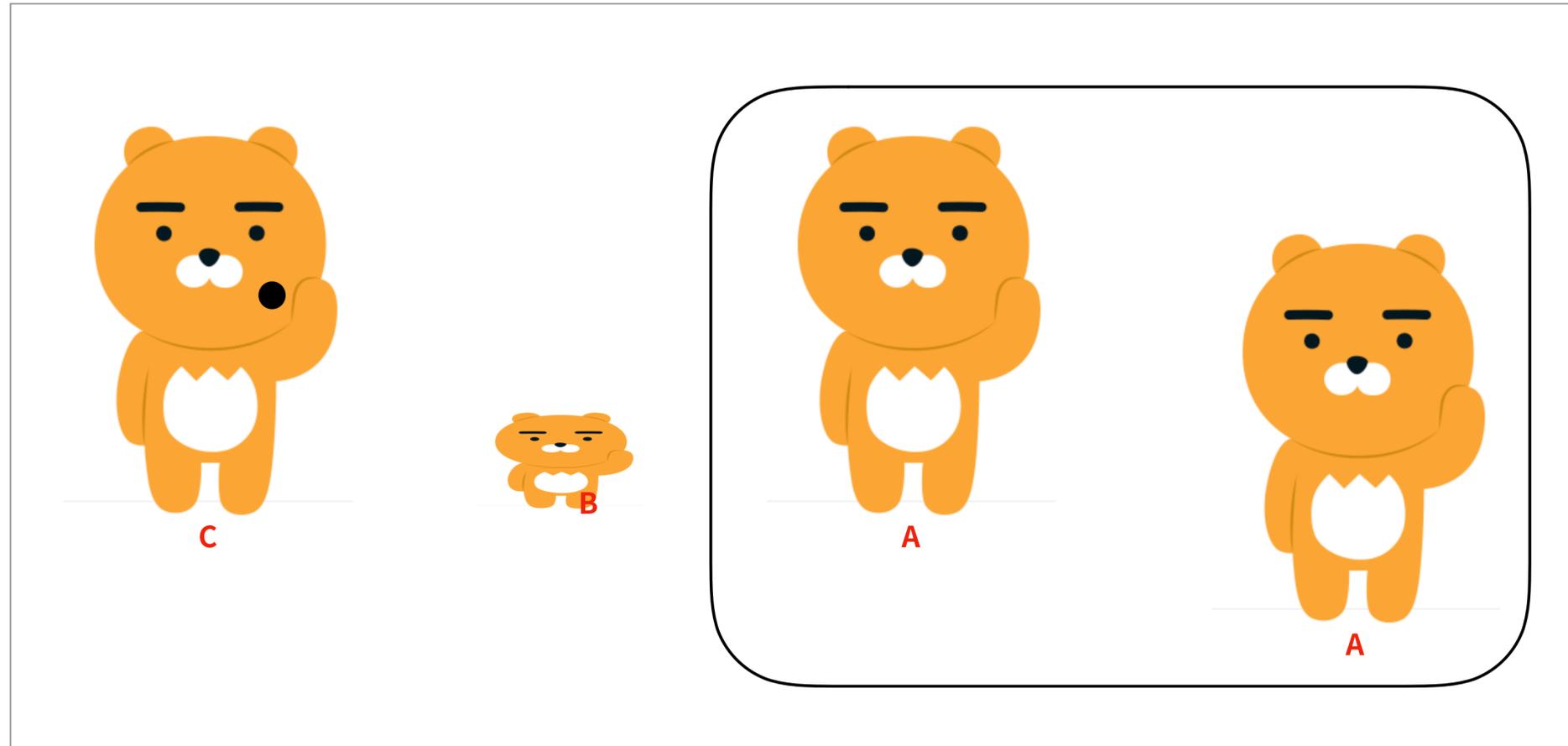
02 방법 1

이미지 ID 값을 이용한 자동 처리

방법 1

이미지 ID 를 이용한 자동 처리

이미지 바이너리 MD5 해시 ID

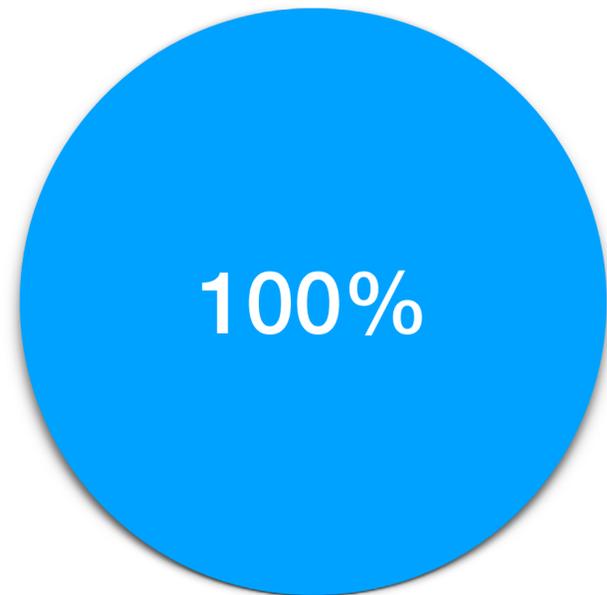


동일한 이미지는 동일한 처리 값을 가진다

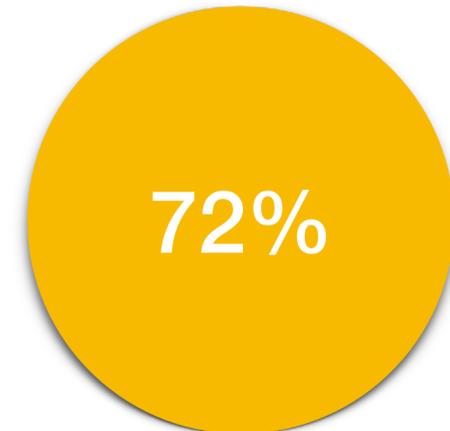
방법 1

이미지 ID 를 이용한 자동 처리

이미지 자동처리로 인한 검수량 감소



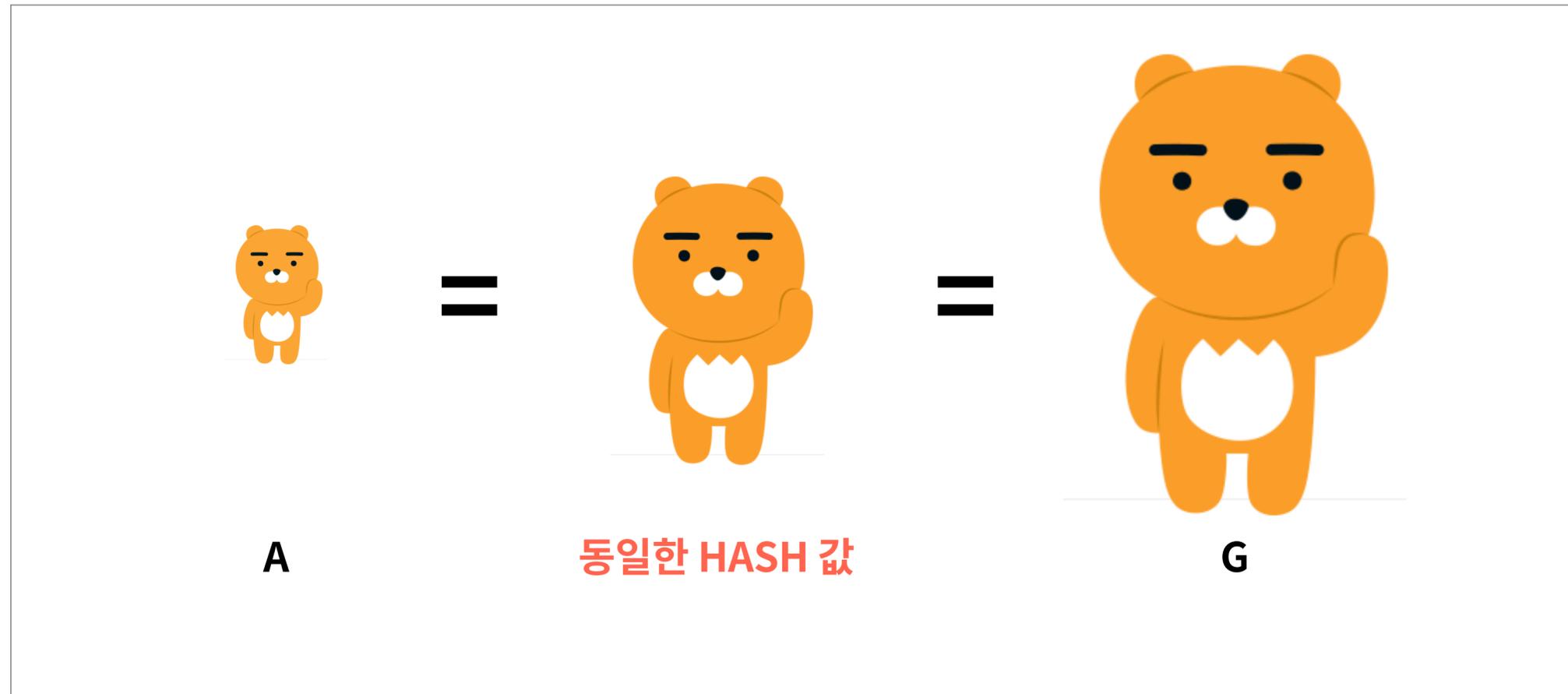
중복 이미지 제거



방법 1

이미지 ID 를 이용한 자동 처리

이미지의 크기 변화에 대한 대응

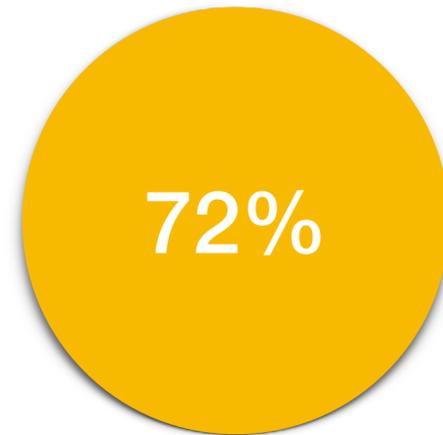


Resize 이미지도 **동일한 처리 값**을 가진다

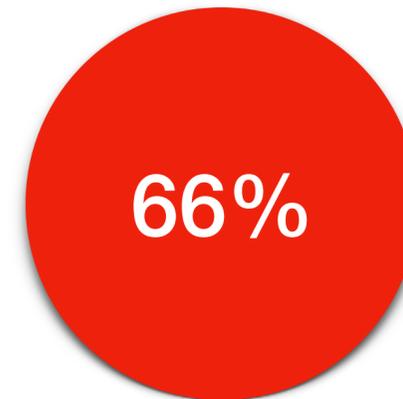
방법 1

이미지 ID 를 이용한 자동 처리

이미지 자동처리로 인한 검수량 감소



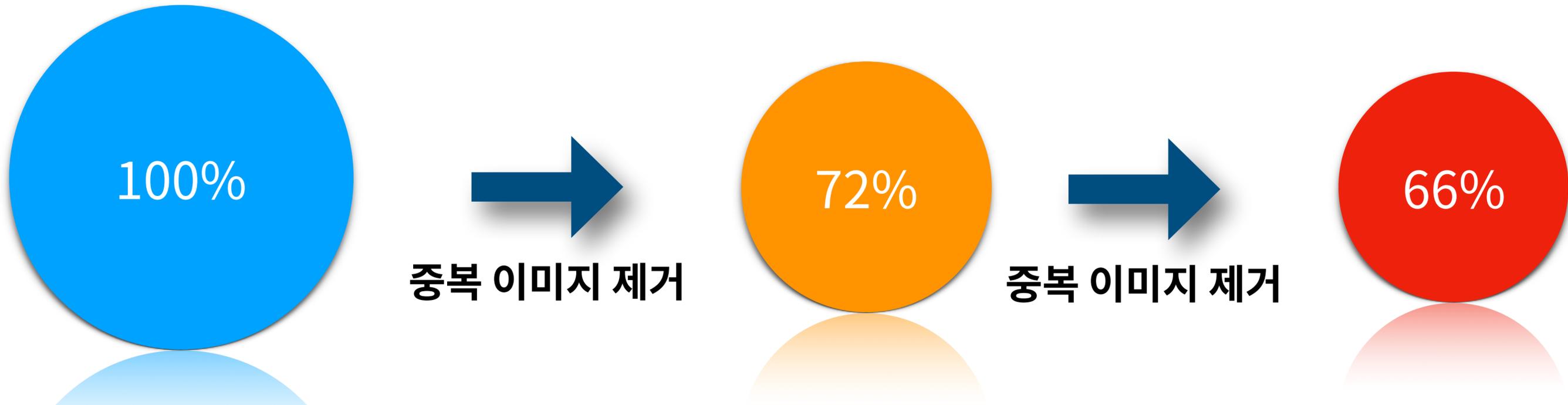
유사 이미지 제거



방법 1

이미지 ID 를 이용한 자동 처리

요약



03 방법 2

이미지의 색상 분포를 가지고 분류해보면 어떨까?

방법 2

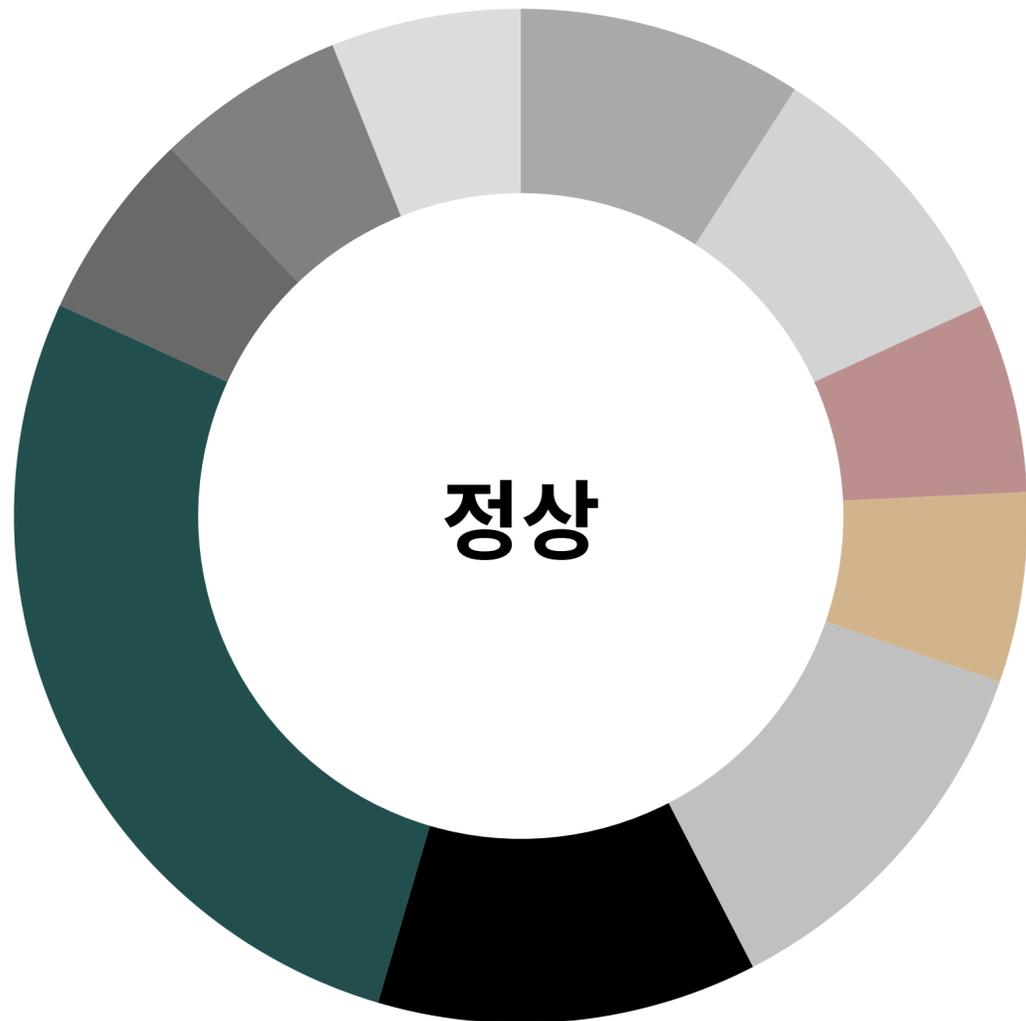
이미지의 색상 분포를 가지고 분류해보면 어떨까?

가설 - 성인 이미지의 색상 분포도를 표현하면 대부분이 특정 색 (피부색) 에 집중 될 것이다.

방법 2

이미지의 색상 분포를 가지고 분류해보면 어떨까?

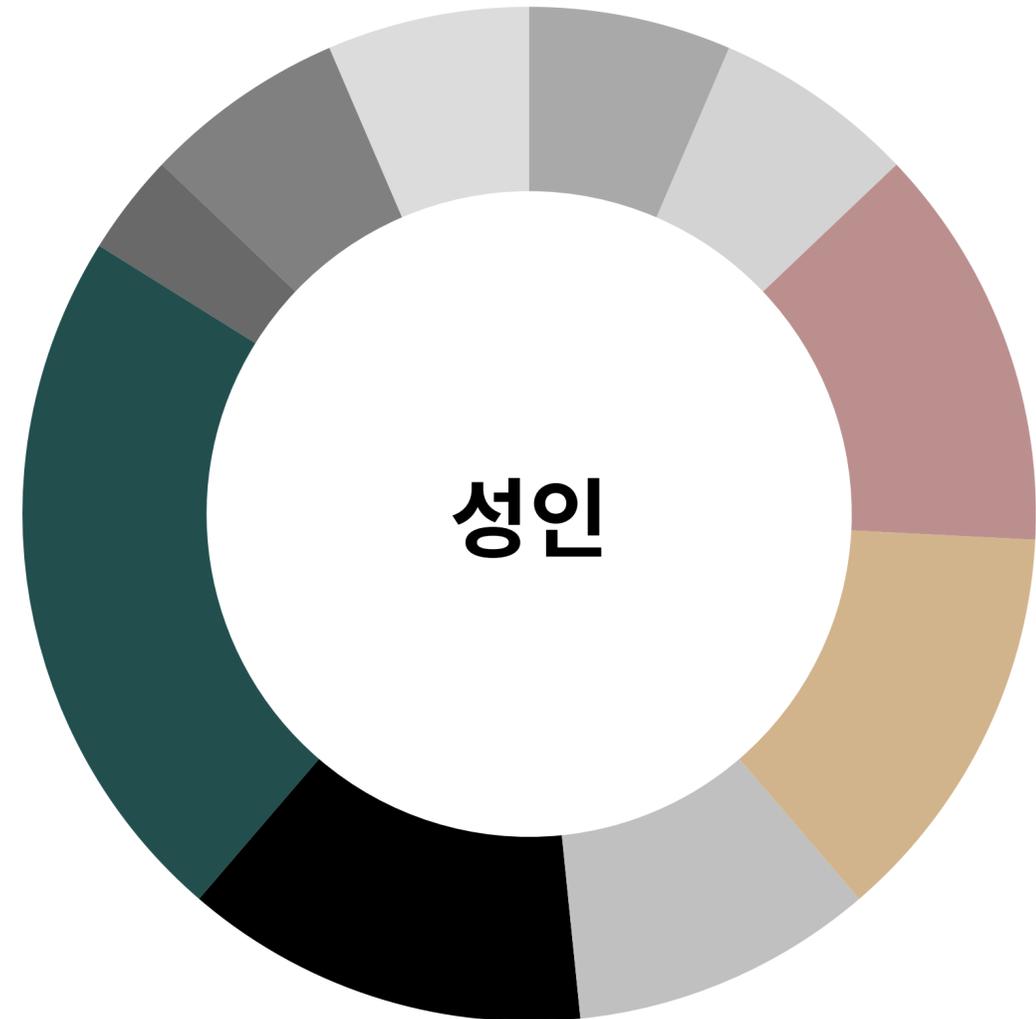
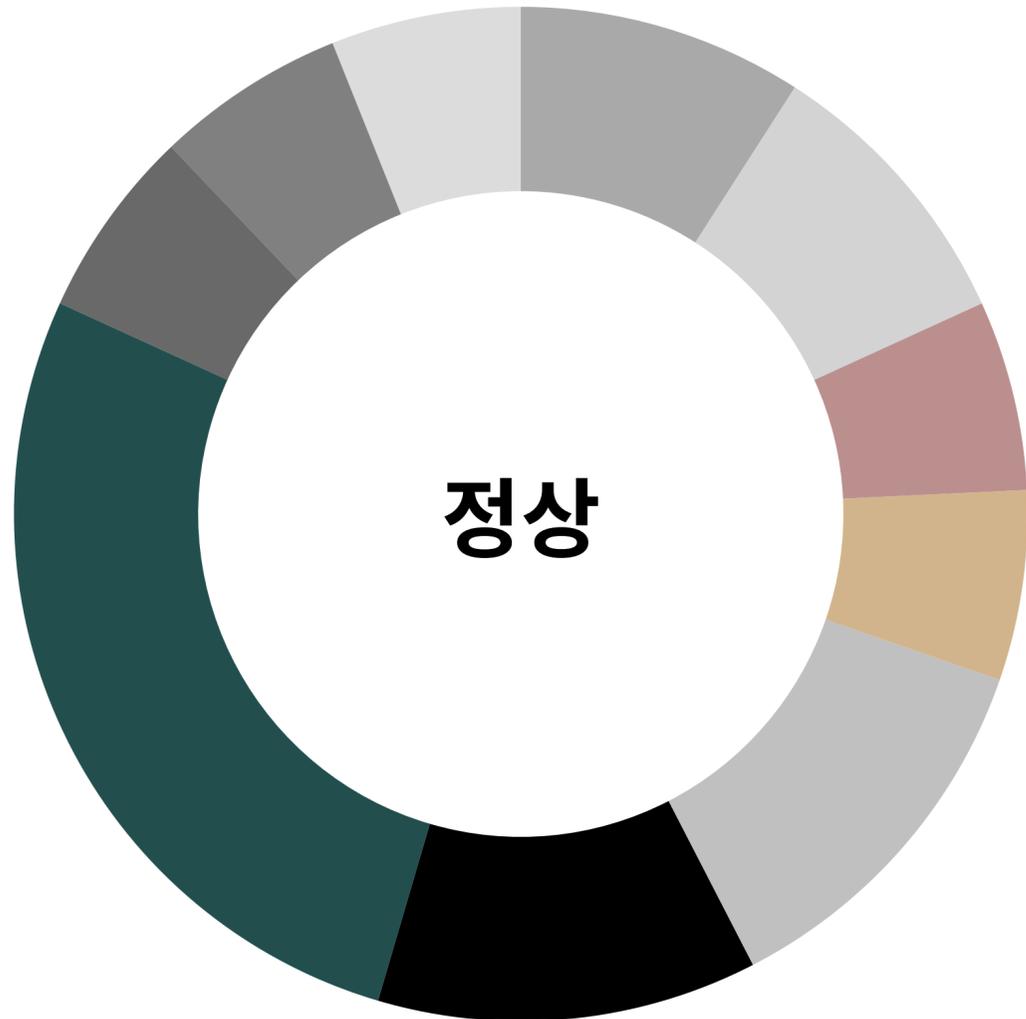
가설 - 성인 이미지의 색상 분포도를 표현하면 대부분이 특정 색 (피부색) 에 집중 될 것이다.



방법 2

이미지의 색상 분포를 가지고 분류해보면 어떨까?

가설 - 성인 이미지의 색상 분포도를 표현하면 대부분이 특정 색 (피부색) 에 집중 될 것이다.

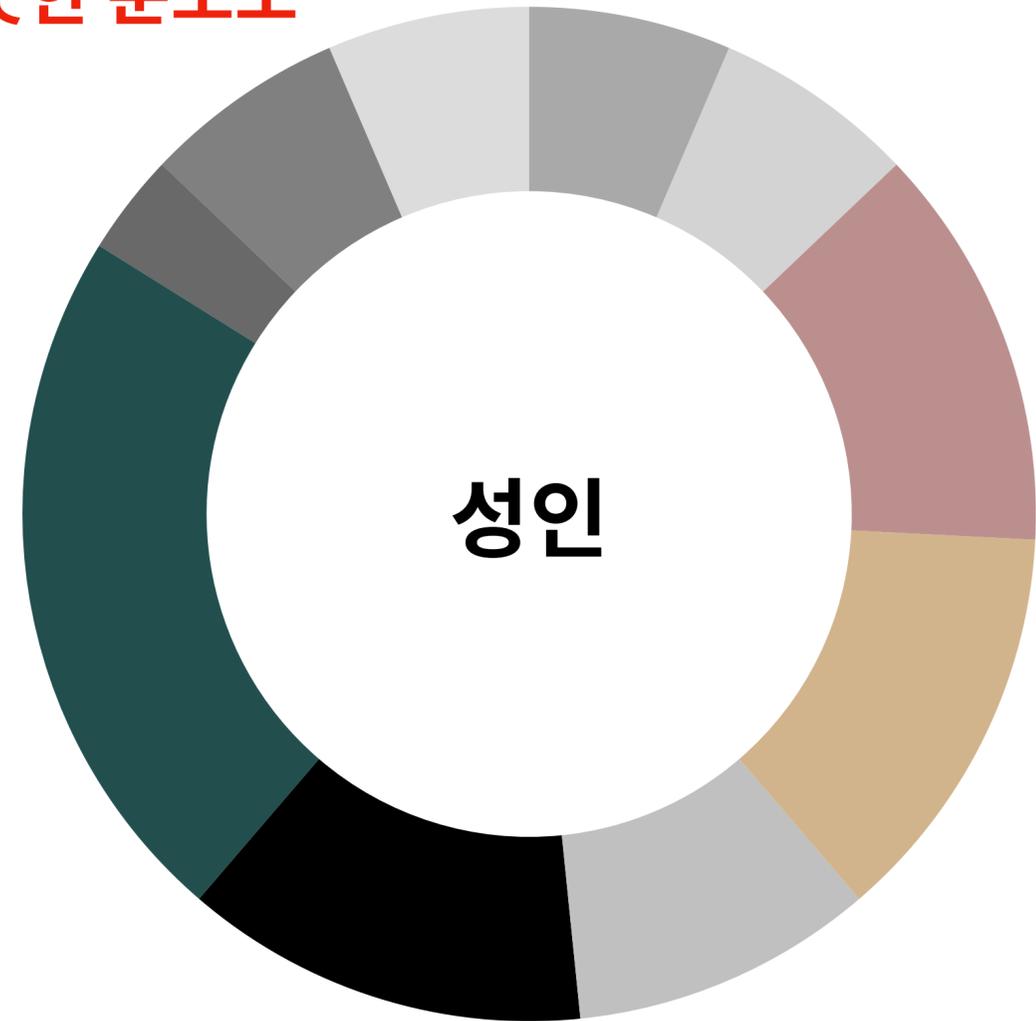
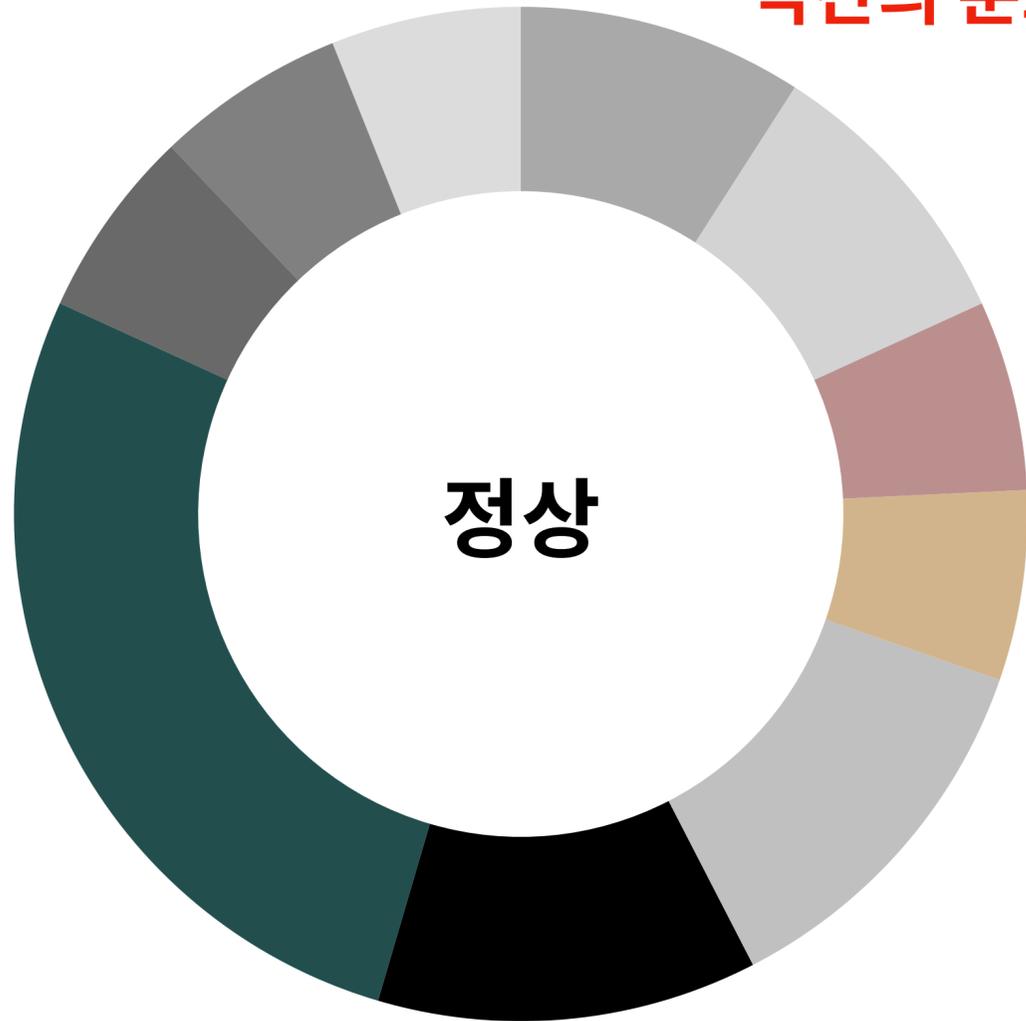


방법 2

이미지의 색상 분포를 가지고 분류해보면 어떨까?

~~가설 - 성인 이미지의 색상 분포도를 표현하면 대부분이 특정 색 (피부색) 에 집중 될 것이다.~~

약간의 분포 차이는 있으나 비슷한 분포도



방법 2

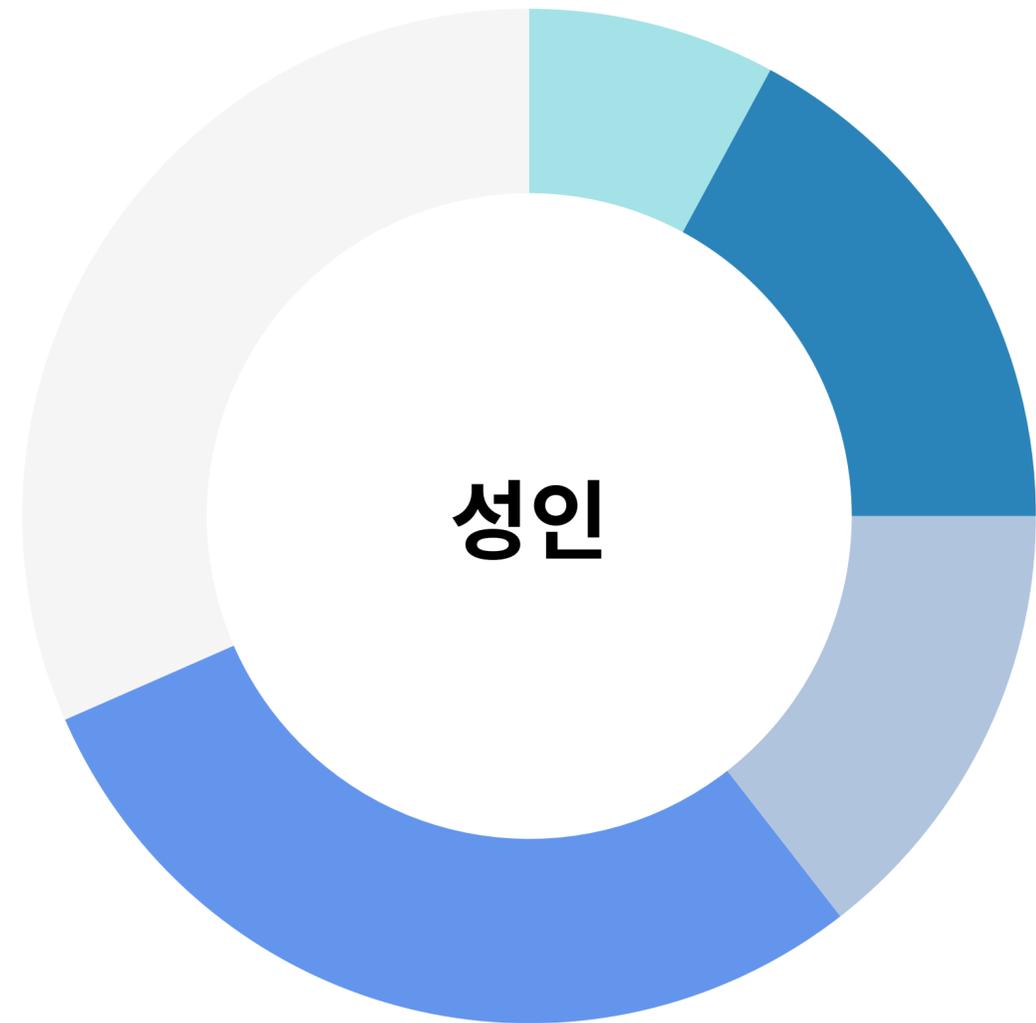
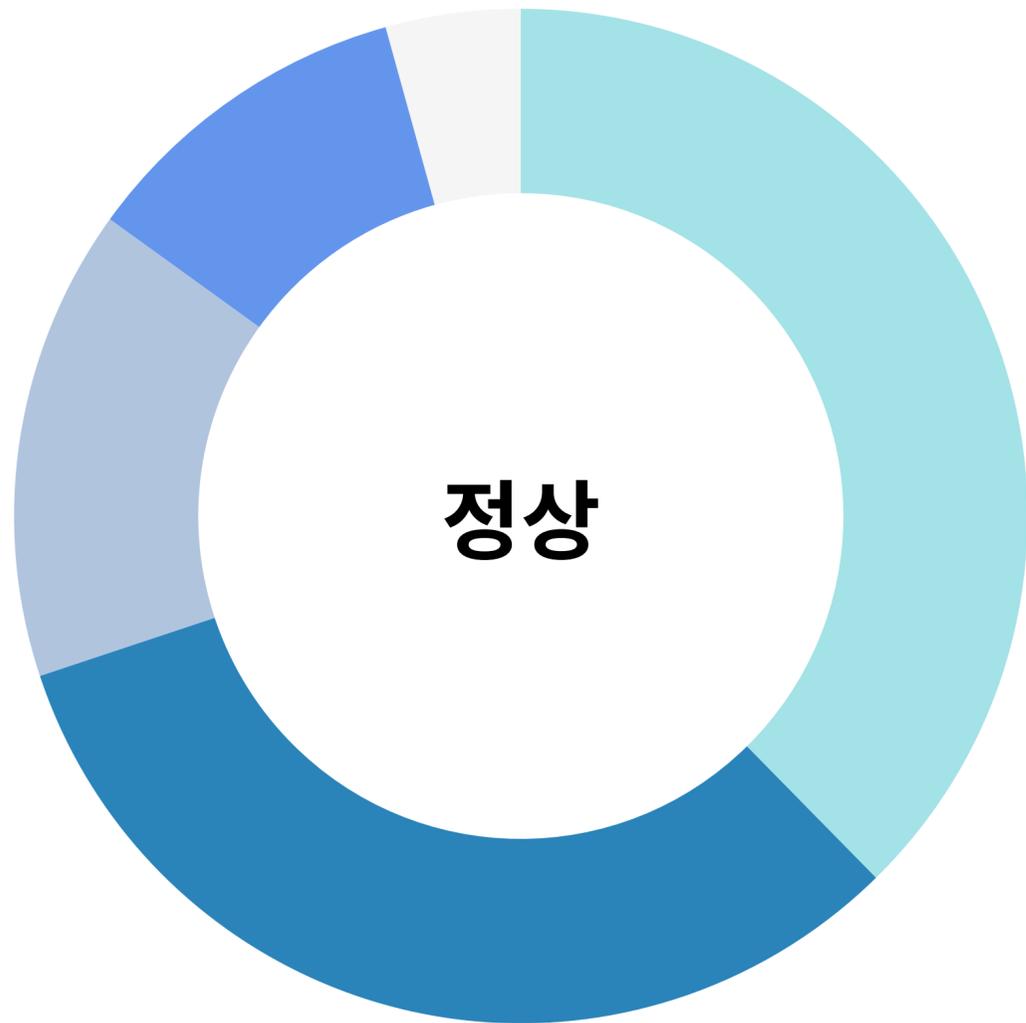
이미지의 색상 분포를 가지고 분류해보면 어떨까?

가설 - 하늘+바다 풍경 이미지의 색상 분포도는 성인 이미지 색상 분포도와 다를 것 이다

방법 2

이미지의 색상 분포를 가지고 분류해보면 어떨까?

가설 - 하늘+바다 풍경 이미지의 색상 분포도는 성인 이미지 색상 분포도와 다를 것 이다

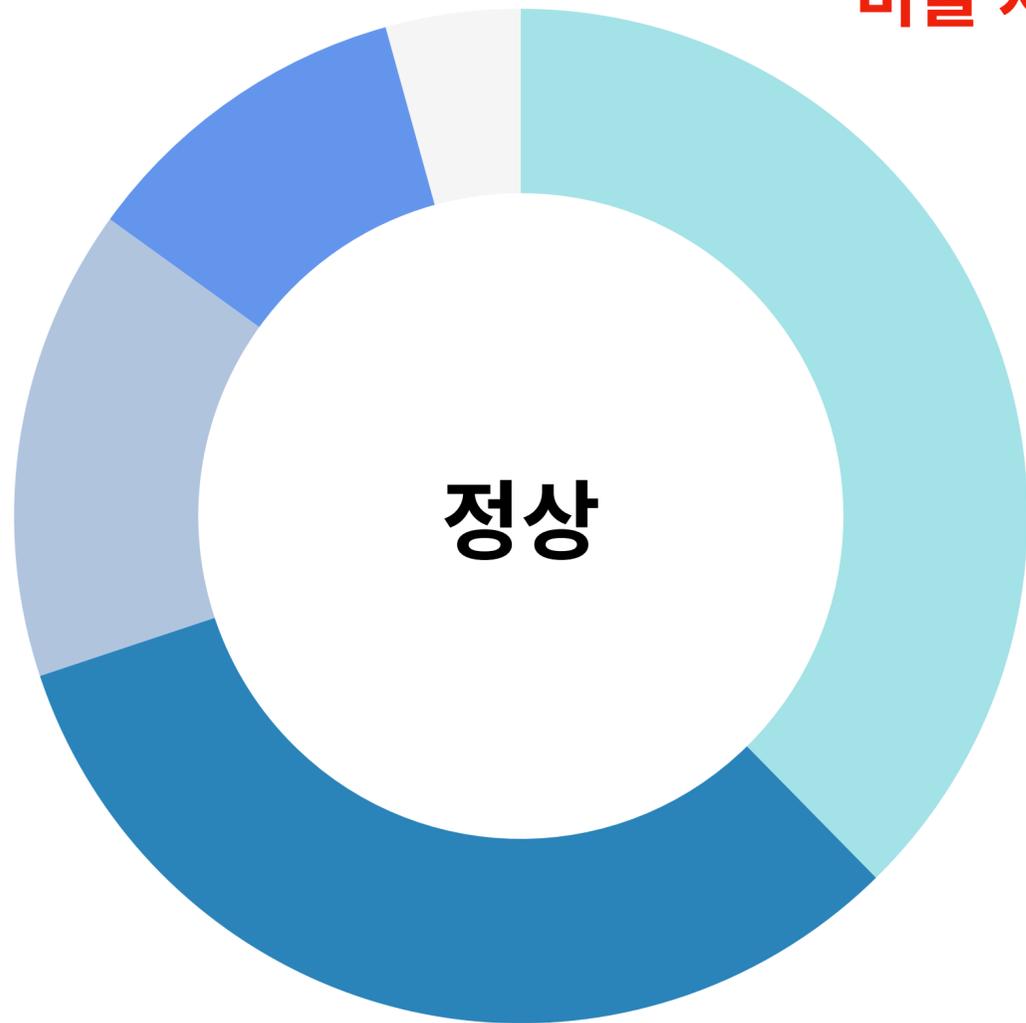


방법 2

이미지의 색상 분포를 가지고 분류해보면 어떨까?

~~가설 - 하늘+바다 풍경 이미지의 색상 분포도는 성인 이미지 색상 분포도와 다를 것이다~~

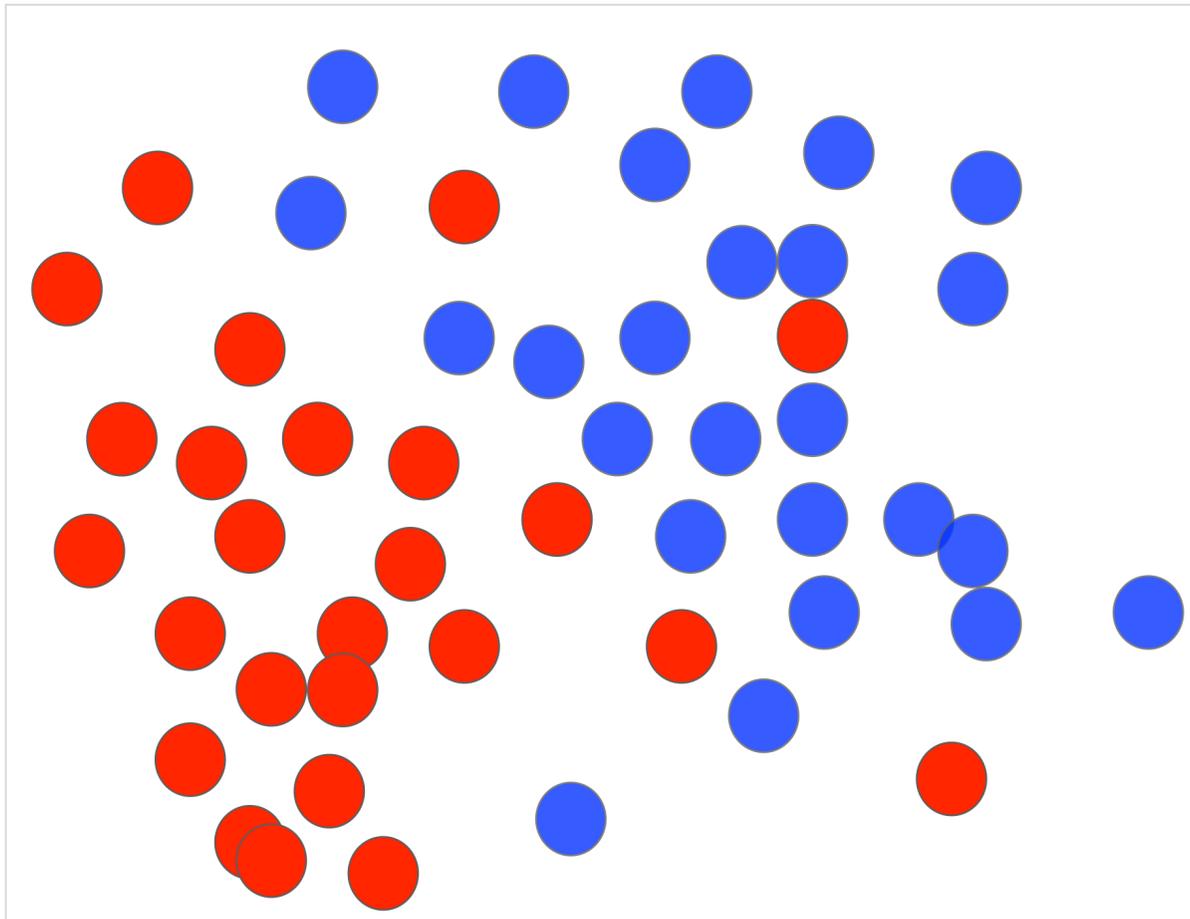
비율 차이는 있으나 비슷한 분포도



방법 2

이미지의 색상 분포를 가지고 분류해보면 어떨까?

머신러닝을 통해 분류해보자 - KNN 알고리즘



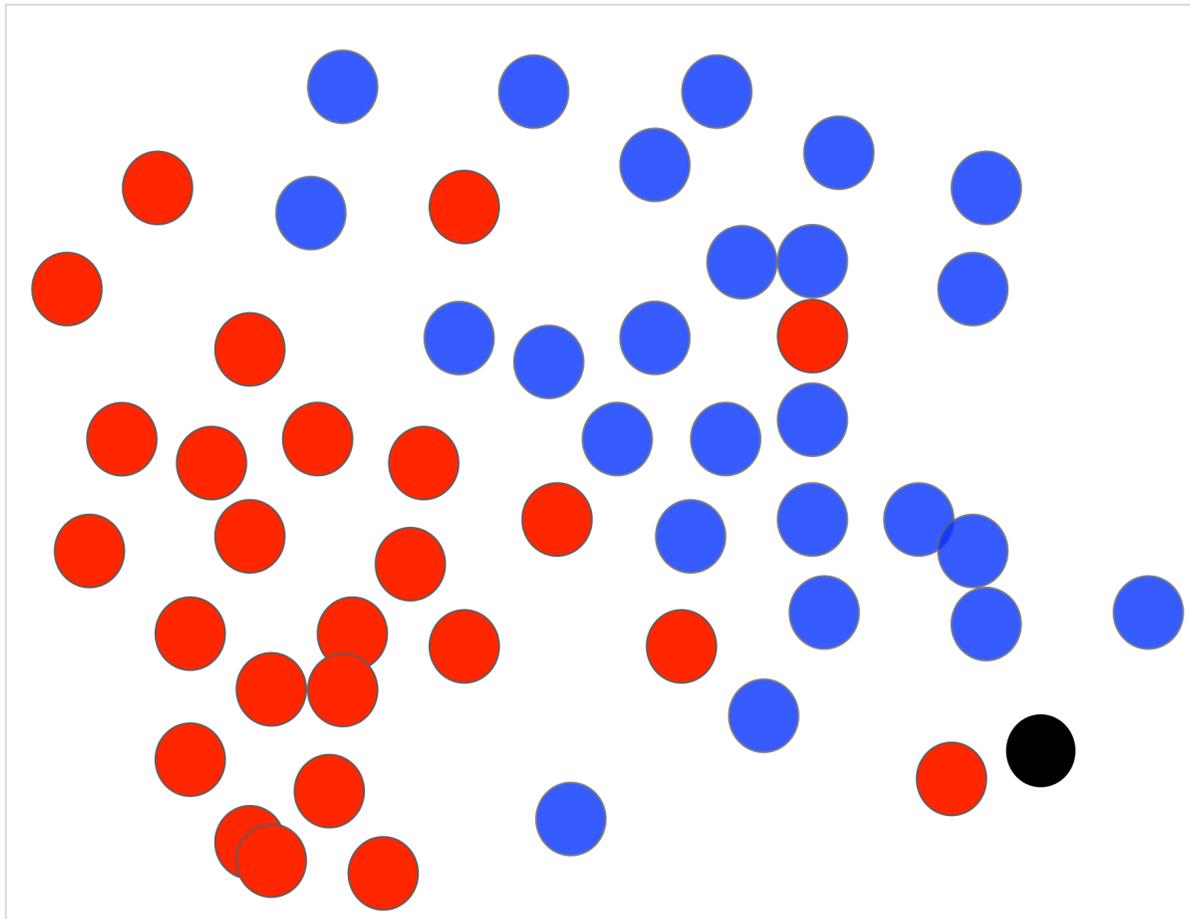
K- Nearest Neighbor

가장 가까운 거리에 있는 K 개의 값을 통한 분류

방법 2

이미지의 색상 분포를 가지고 분류해보면 어떨까?

머신러닝을 통해 분류해보자 - KNN 알고리즘



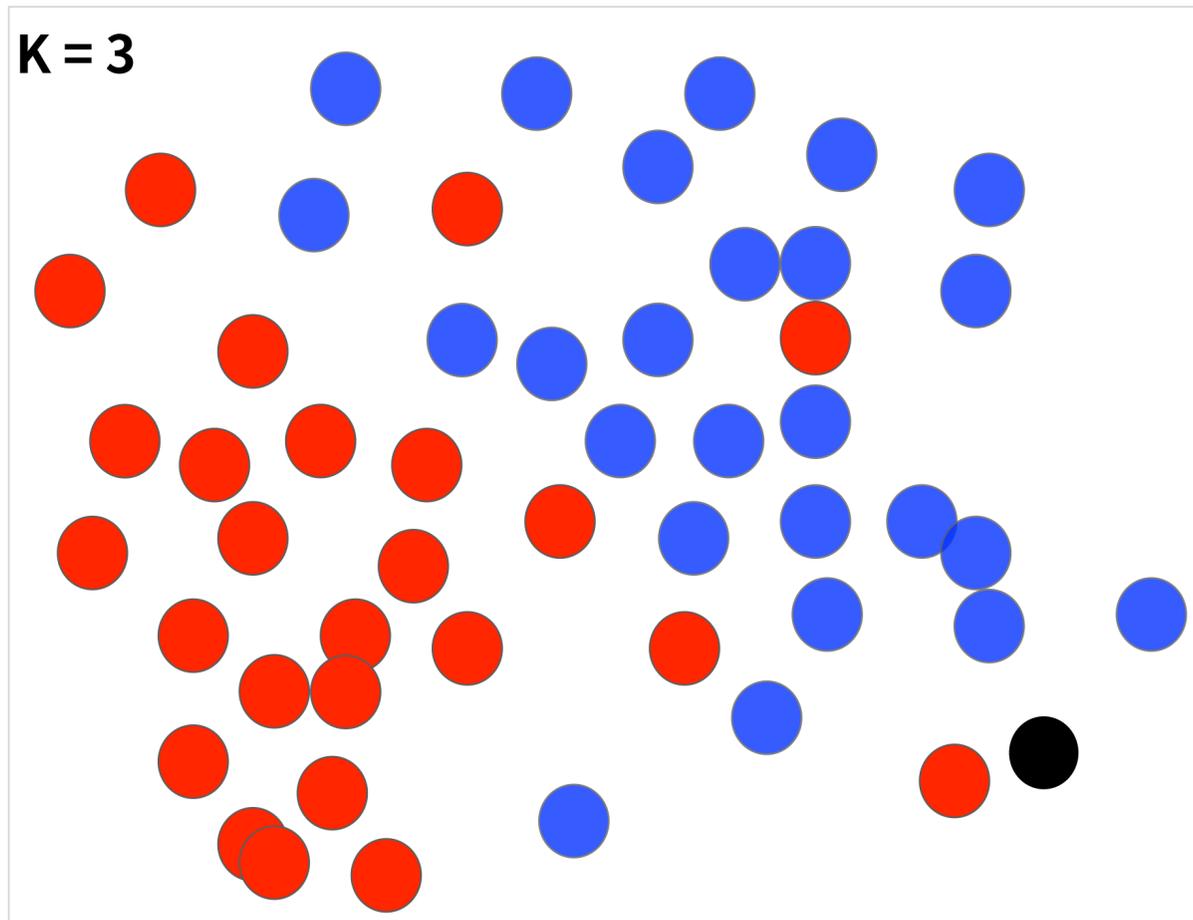
K- Nearest Neighbor

가장 가까운 거리에 있는 K 개의 값을 통한 분류

방법 2

이미지의 색상 분포를 가지고 분류해보면 어떨까?

머신러닝을 통해 분류해보자 - KNN 알고리즘



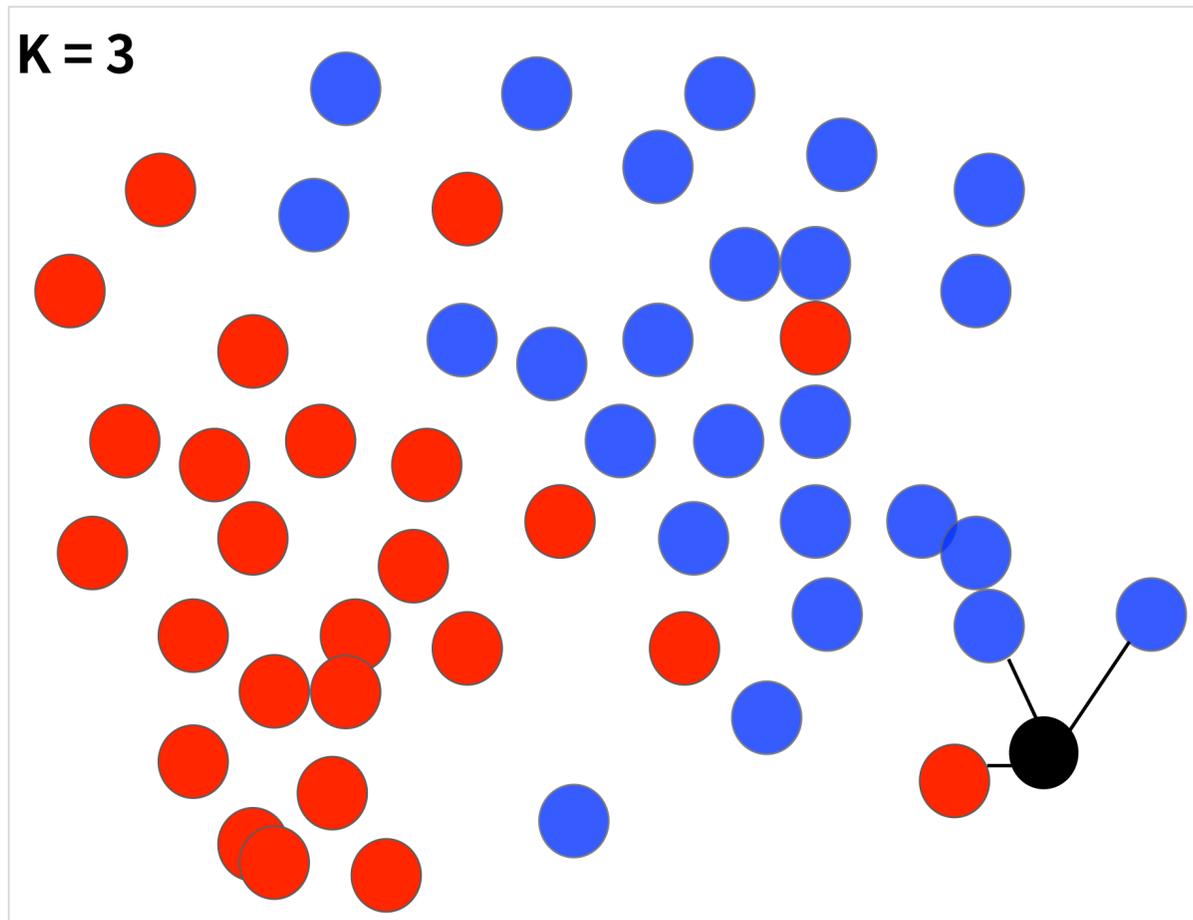
K- Nearest Neighbor

가장 가까운 거리에 있는 K 개의 값을 통한 분류

방법 2

이미지의 색상 분포를 가지고 분류해보면 어떨까?

머신러닝을 통해 분류해보자 - KNN 알고리즘



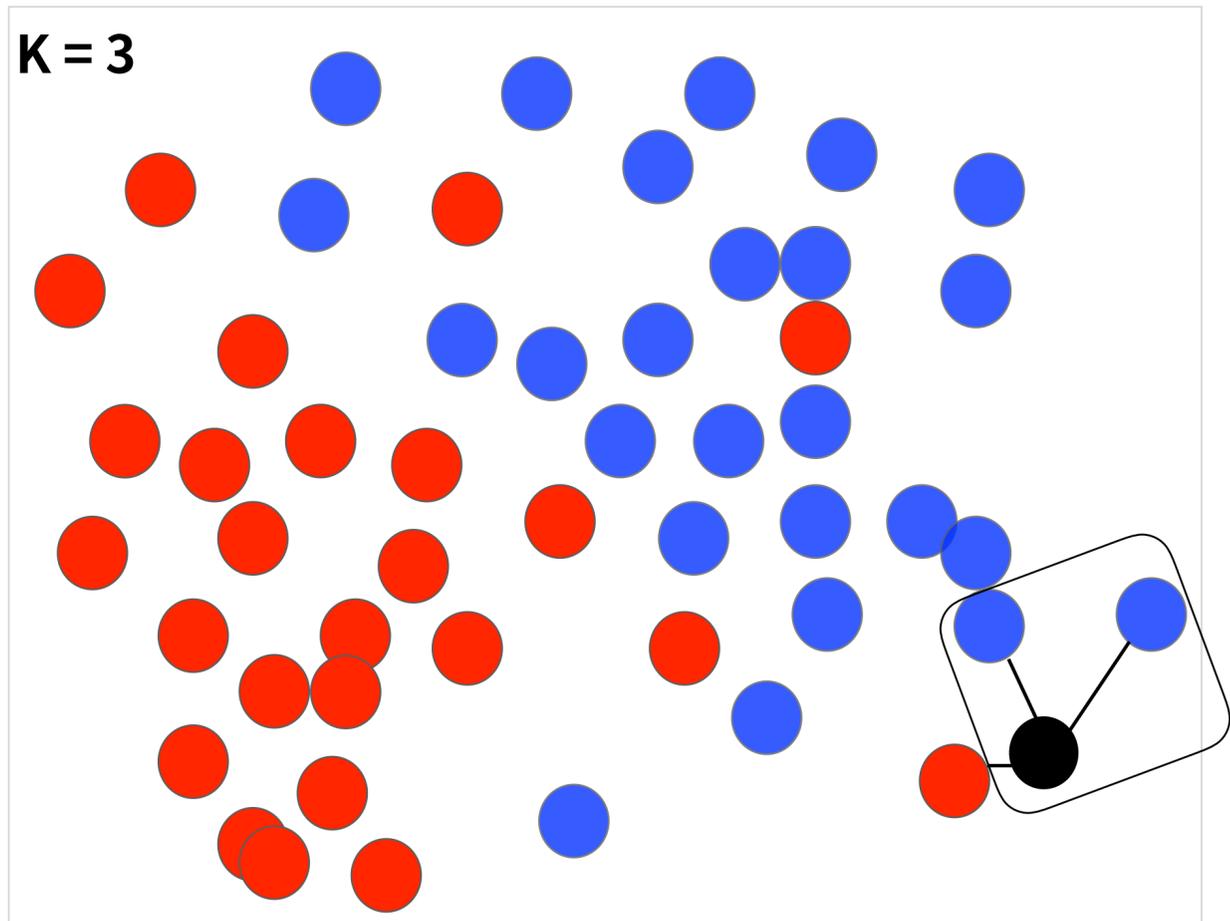
K- Nearest Neighbor

가장 가까운 거리에 있는 K 개의 값을 통한 분류

방법 2

이미지의 색상 분포를 가지고 분류해보면 어떨까?

머신러닝을 통해 분류해보자 - KNN 알고리즘



K- Nearest Neighbor

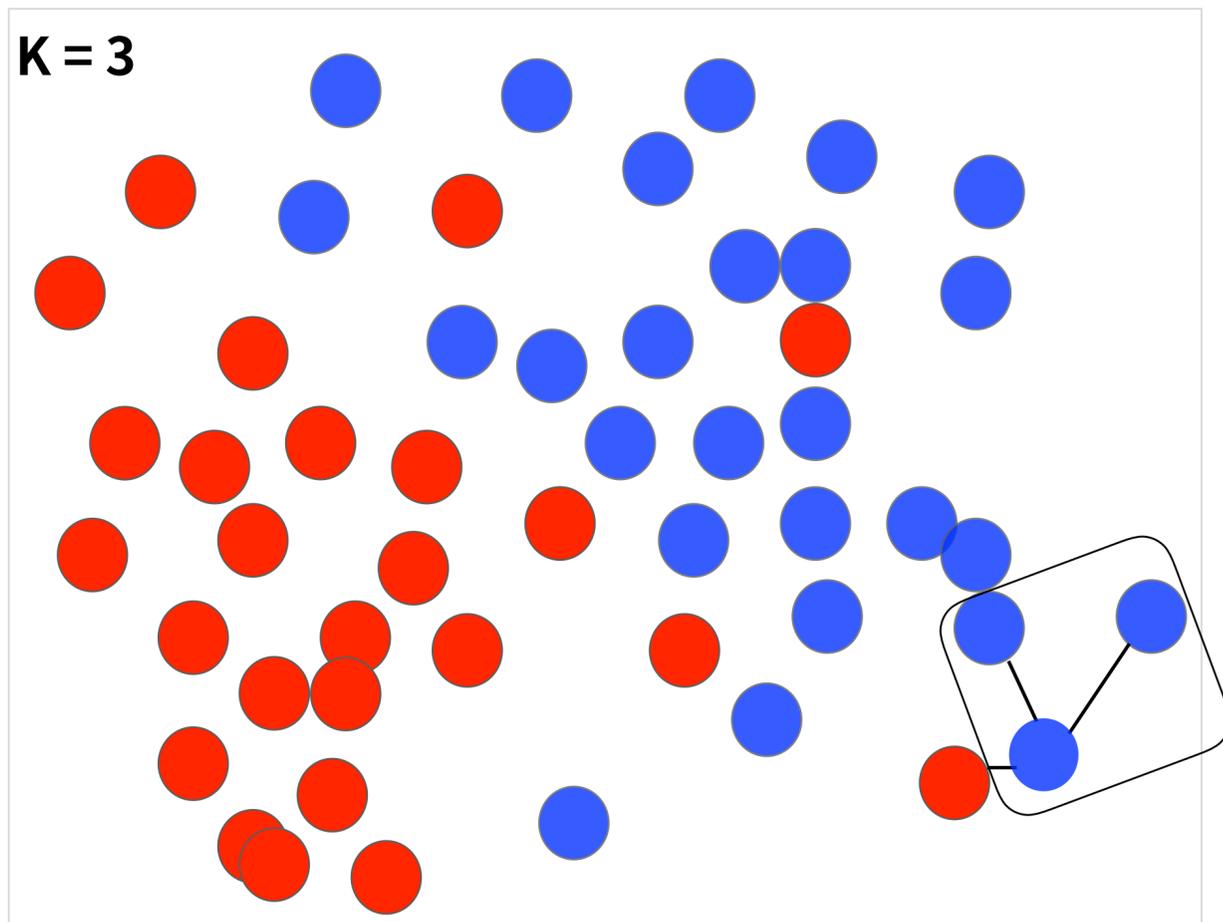
가장 가까운 거리에 있는 K 개의 값을 통한 분류

다수결로 결정함

방법 2

이미지의 색상 분포를 가지고 분류해보면 어떨까?

머신러닝을 통해 분류해보자 - KNN 알고리즘



K- Nearest Neighbor

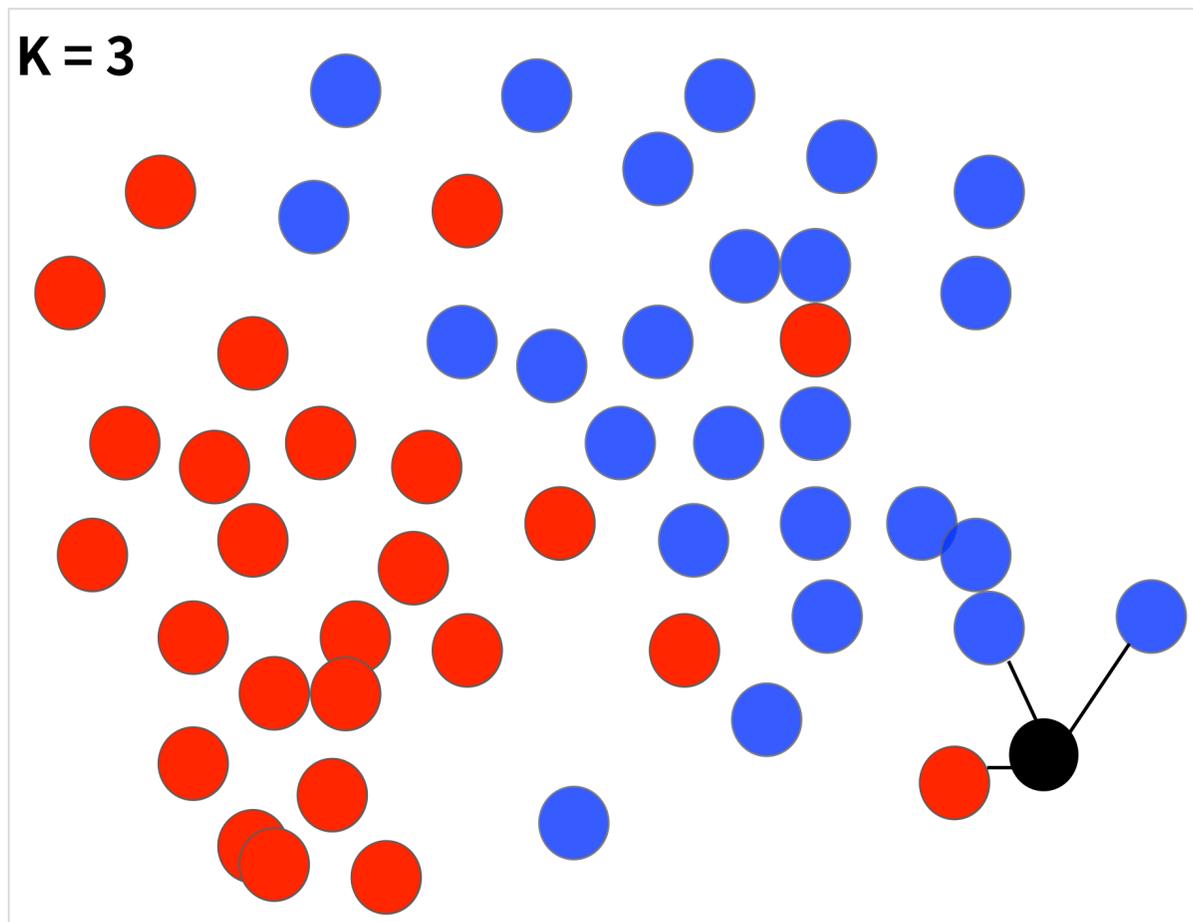
가장 가까운 거리에 있는 K 개의 값을 통한 분류

다수결로 결정함

방법 2

이미지의 색상 분포를 가지고 분류해보면 어떨까?

머신러닝을 통해 분류해보자 - KNN 알고리즘



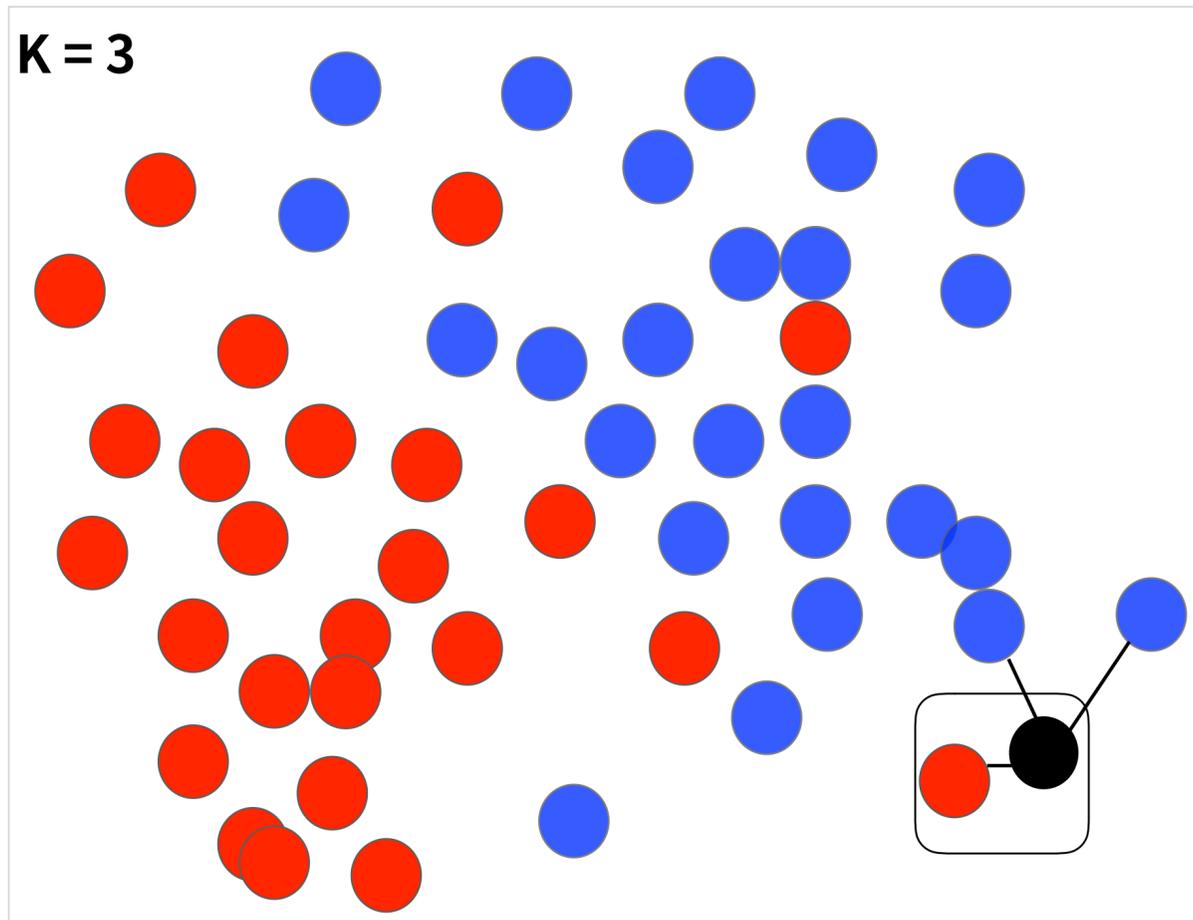
K- Nearest Neighbor

가장 가까운 거리에 있는 K 개의 값을 통한 분류

방법 2

이미지의 색상 분포를 가지고 분류해보면 어떨까?

머신러닝을 통해 분류해보자 - KNN 알고리즘



K- Nearest Neighbor

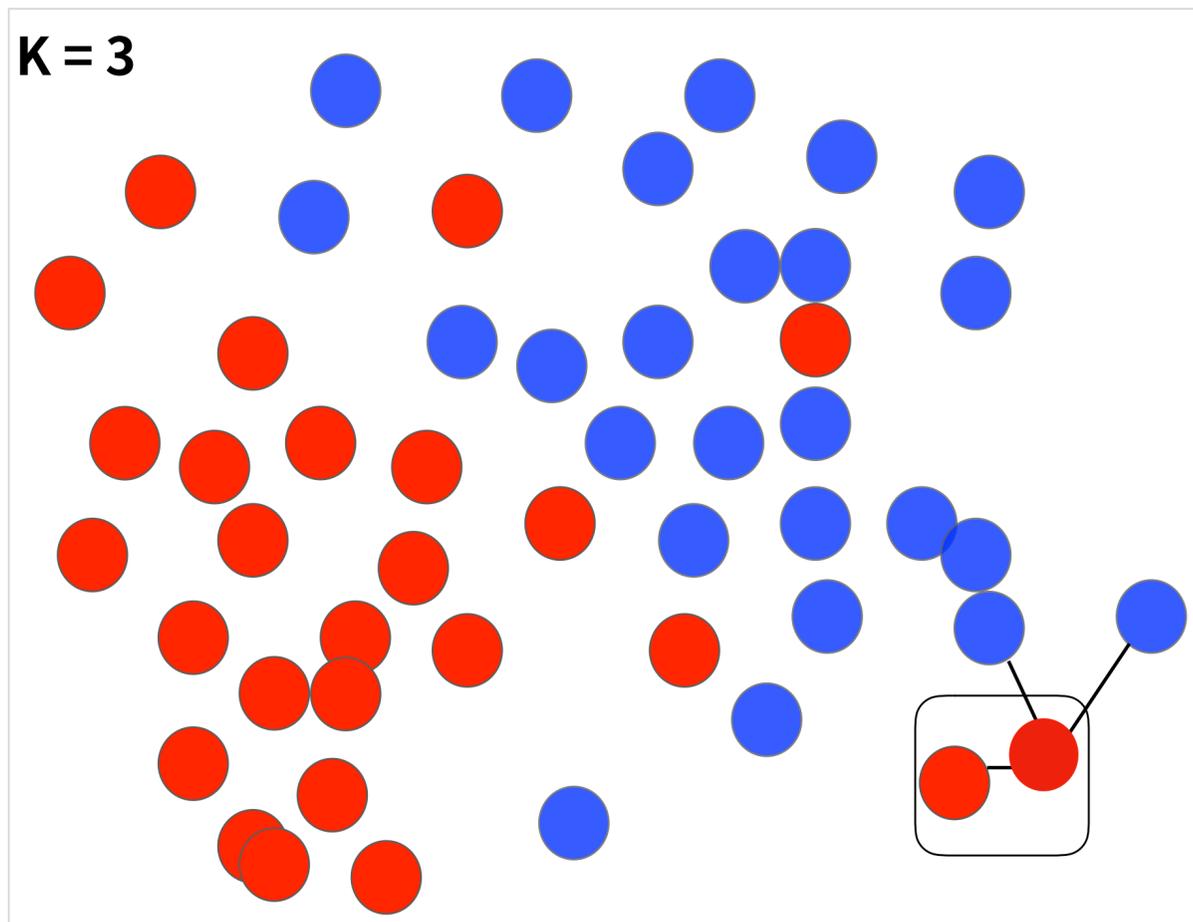
가장 가까운 거리에 있는 K 개의 값을 통한 분류

거리의 가중합 으로 결정함

방법 2

이미지의 색상 분포를 가지고 분류해보면 어떨까?

머신러닝을 통해 분류해보자 - KNN 알고리즘



K- Nearest Neighbor

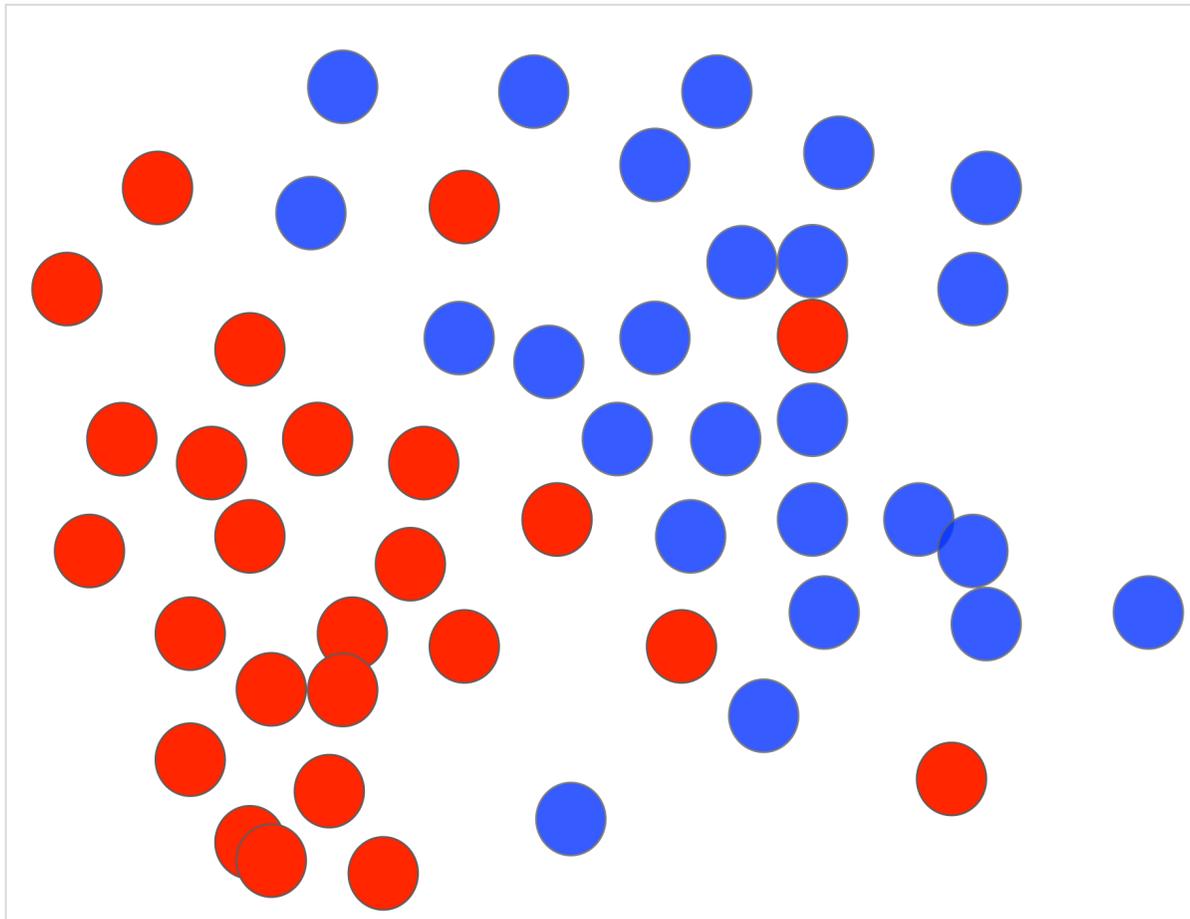
가장 가까운 거리에 있는 K 개의 값을 통한 분류

거리의 가중합 으로 결정함

방법 2

이미지의 색상 분포를 가지고 분류해보면 어떨까?

KNN 알고리즘 - 장점



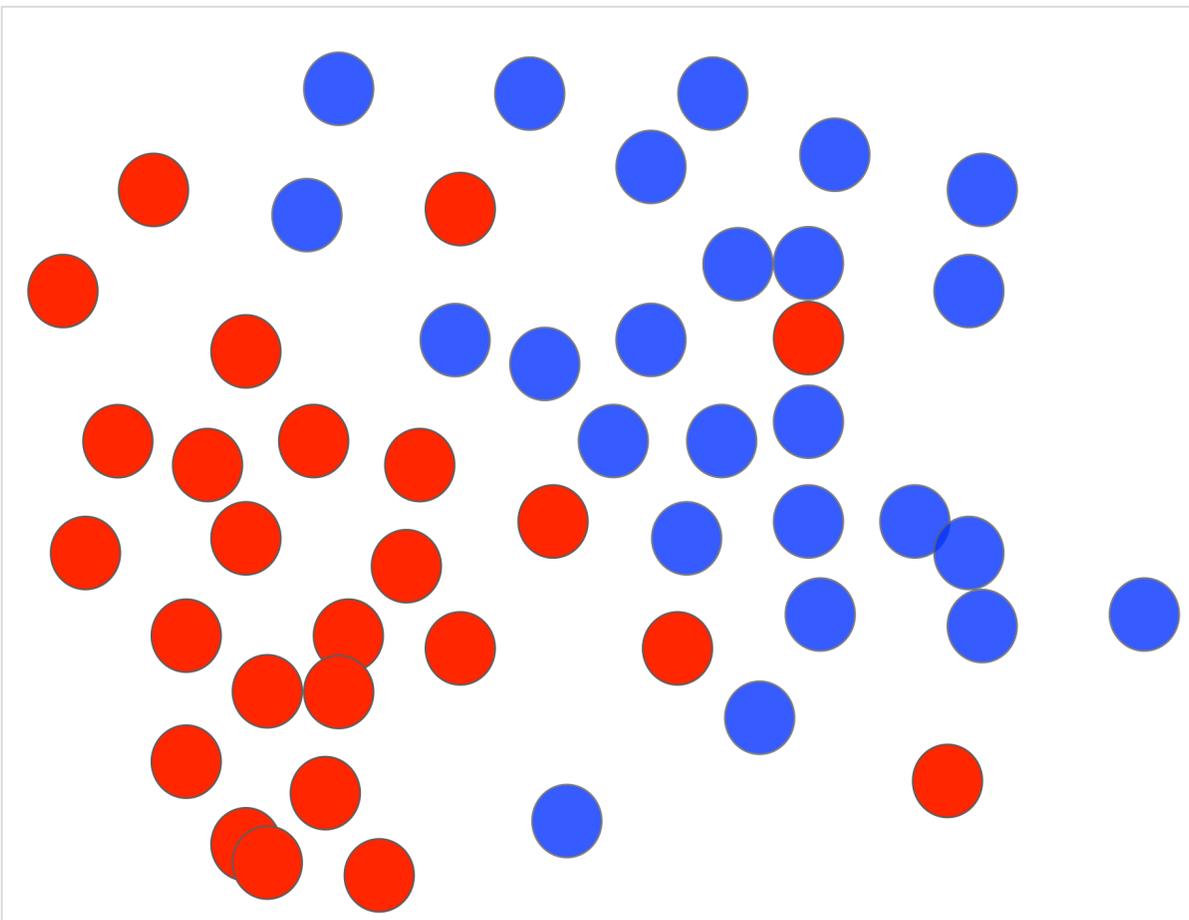
특별한 학습 과정이 필요 없다

데이터 노이즈에 영향이 적다

방법 2

이미지의 색상 분포를 가지고 분류해보면 어떨까?

KNN 알고리즘 - 단점



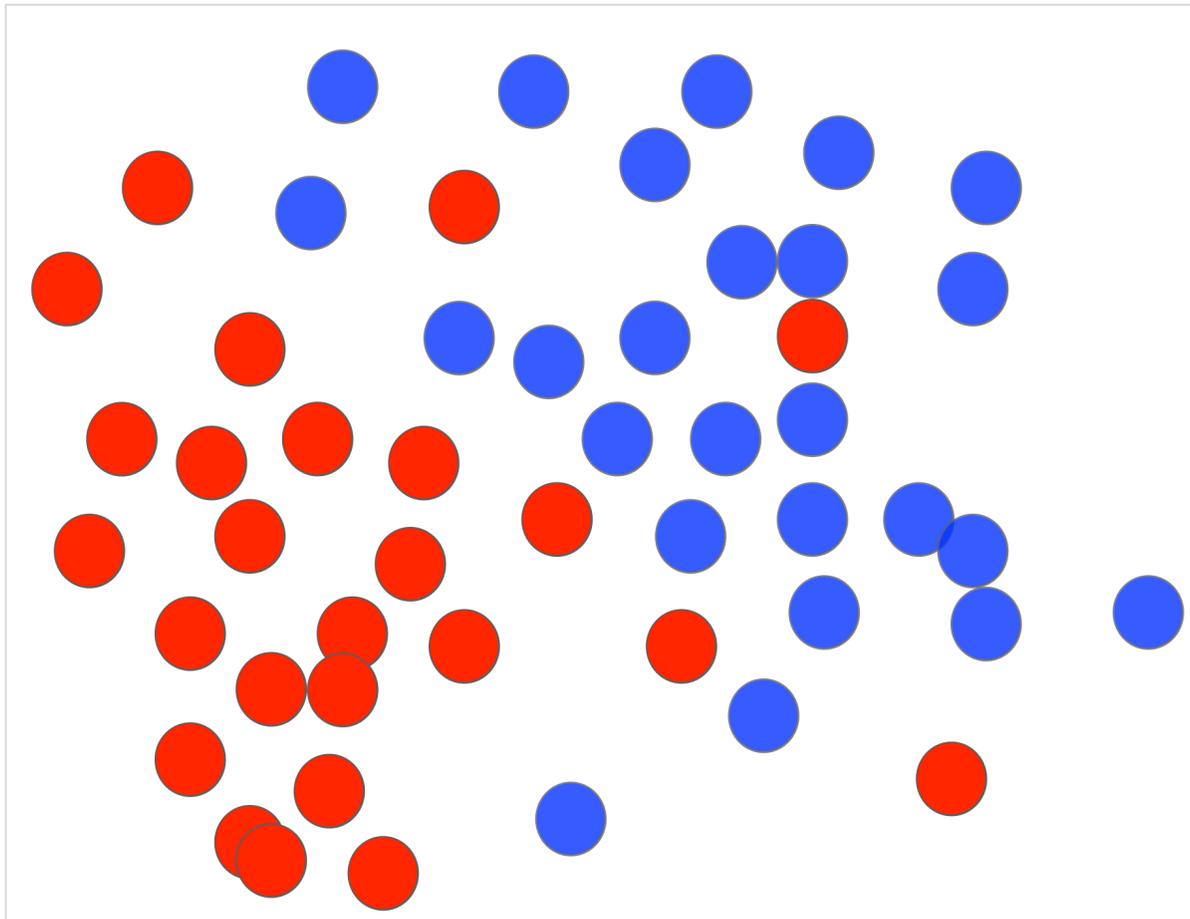
새로운 데이터 마다 거리값을 전부 계산해야함

고차원 데이터인 이미지에서 거리값은 ...?

방법 2

이미지의 색상 분포를 가지고 분류해보면 어떨까?

KNN 알고리즘 - 단점



새로운 데이터 마다 거리값을 전부 계산해야함

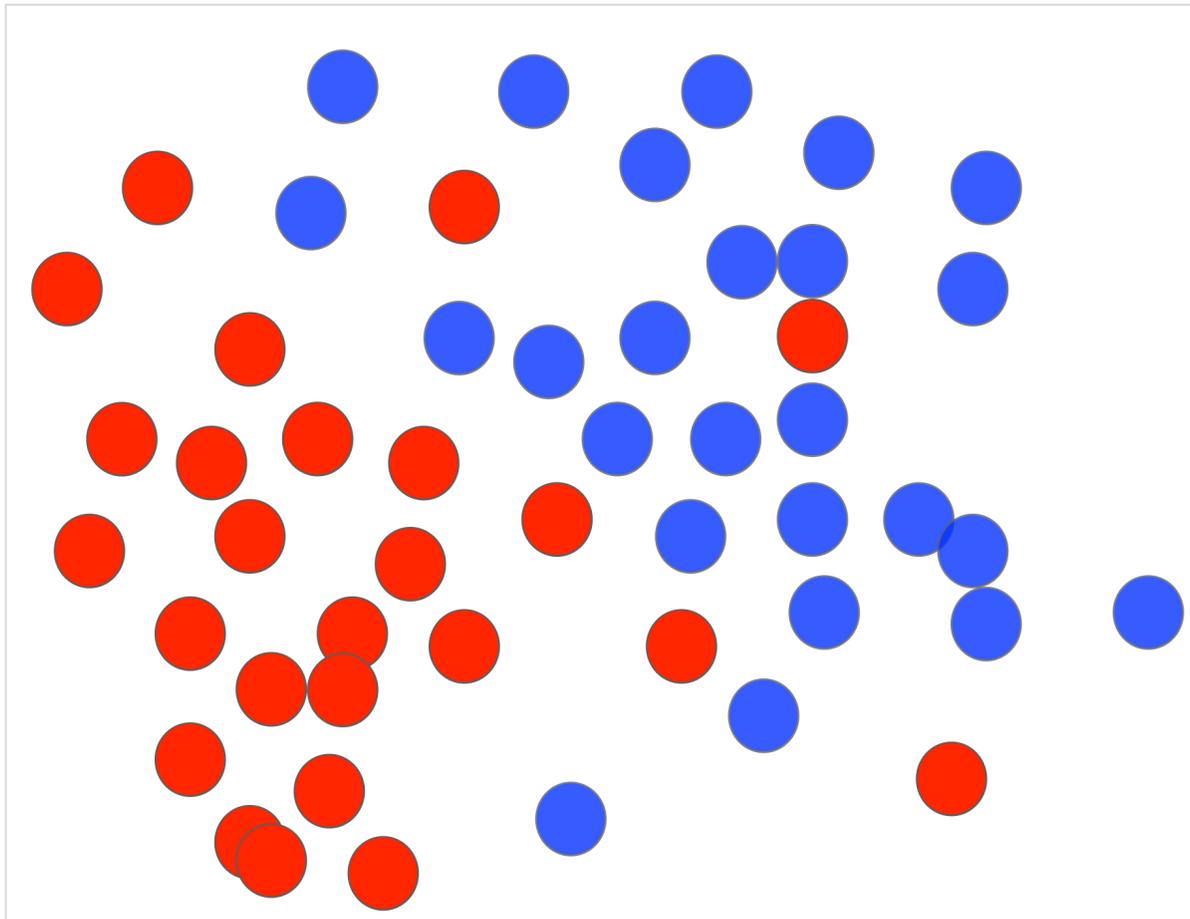
=> 5시간

고차원 데이터인 이미지에서 거리값은 ...?

방법 2

이미지의 색상 분포를 가지고 분류해보면 어떨까?

KNN 알고리즘 - 속도 문제 개선



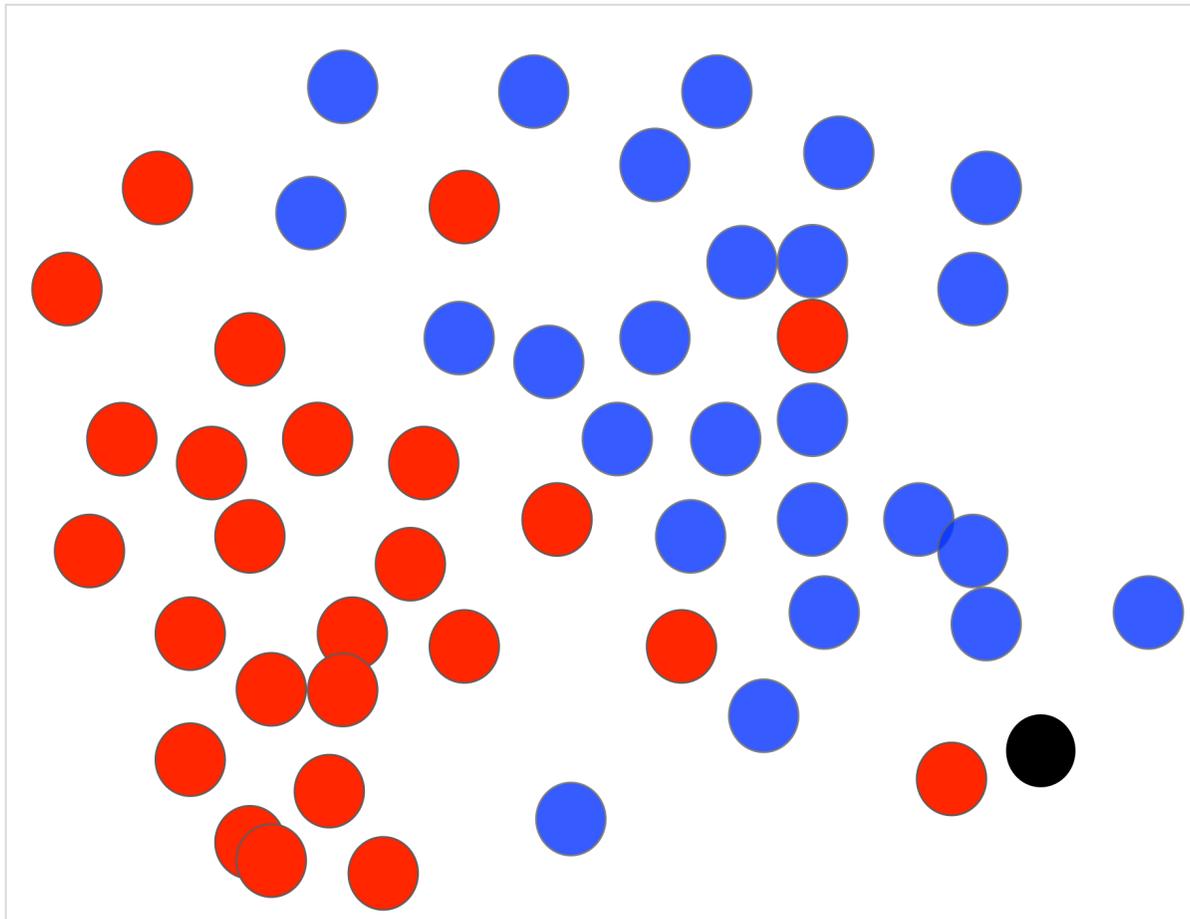
Approximate k Nearest Neighbor

가장 가까운 이웃 N 개 찾기 -> 근사한 이웃 N 개 찾기

방법 2

이미지의 색상 분포를 가지고 분류해보면 어떨까?

KNN 알고리즘 - 속도 문제 개선



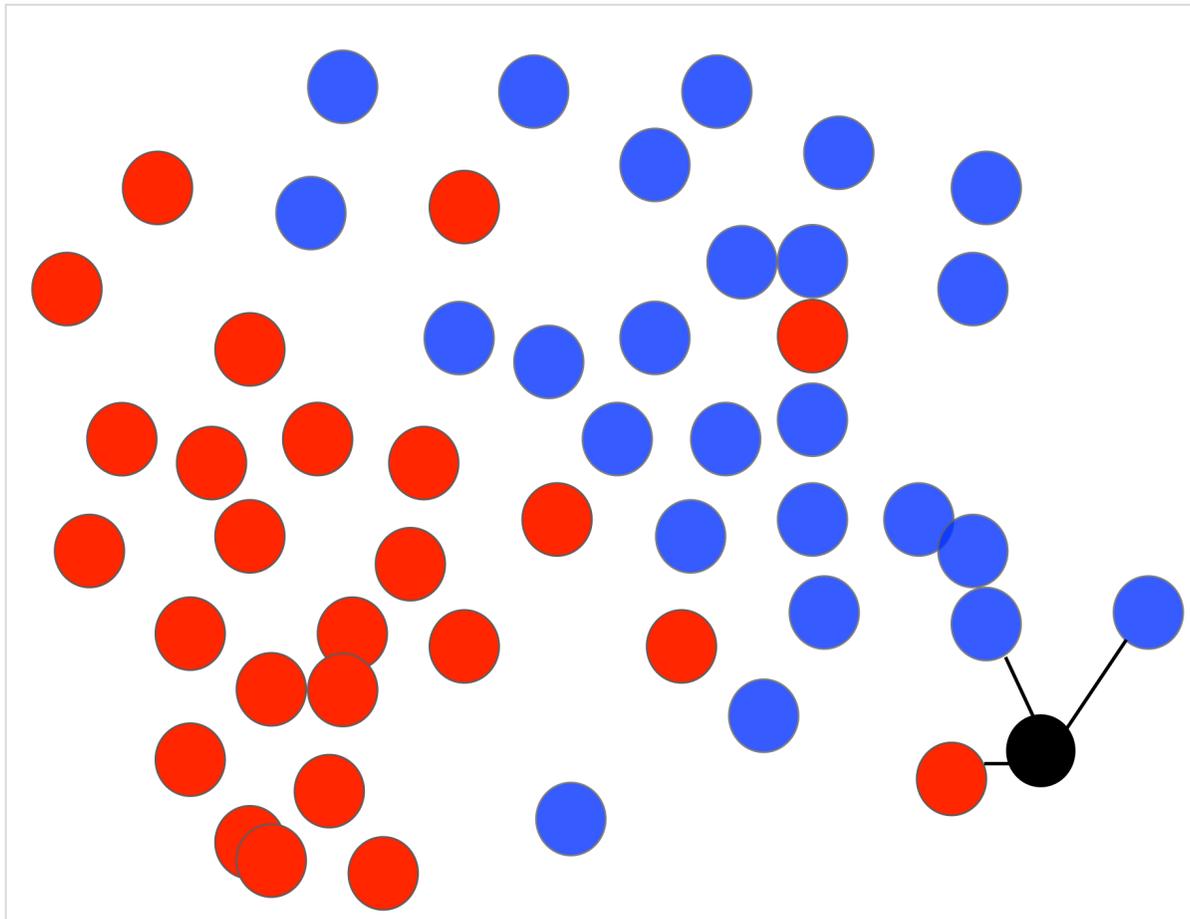
Approximate k Nearest Neighbor

가장 가까운 이웃 N 개 찾기 -> 근사한 이웃 N 개 찾기

방법 2

이미지의 색상 분포를 가지고 분류해보면 어떨까?

KNN 알고리즘 - 속도 문제 개선



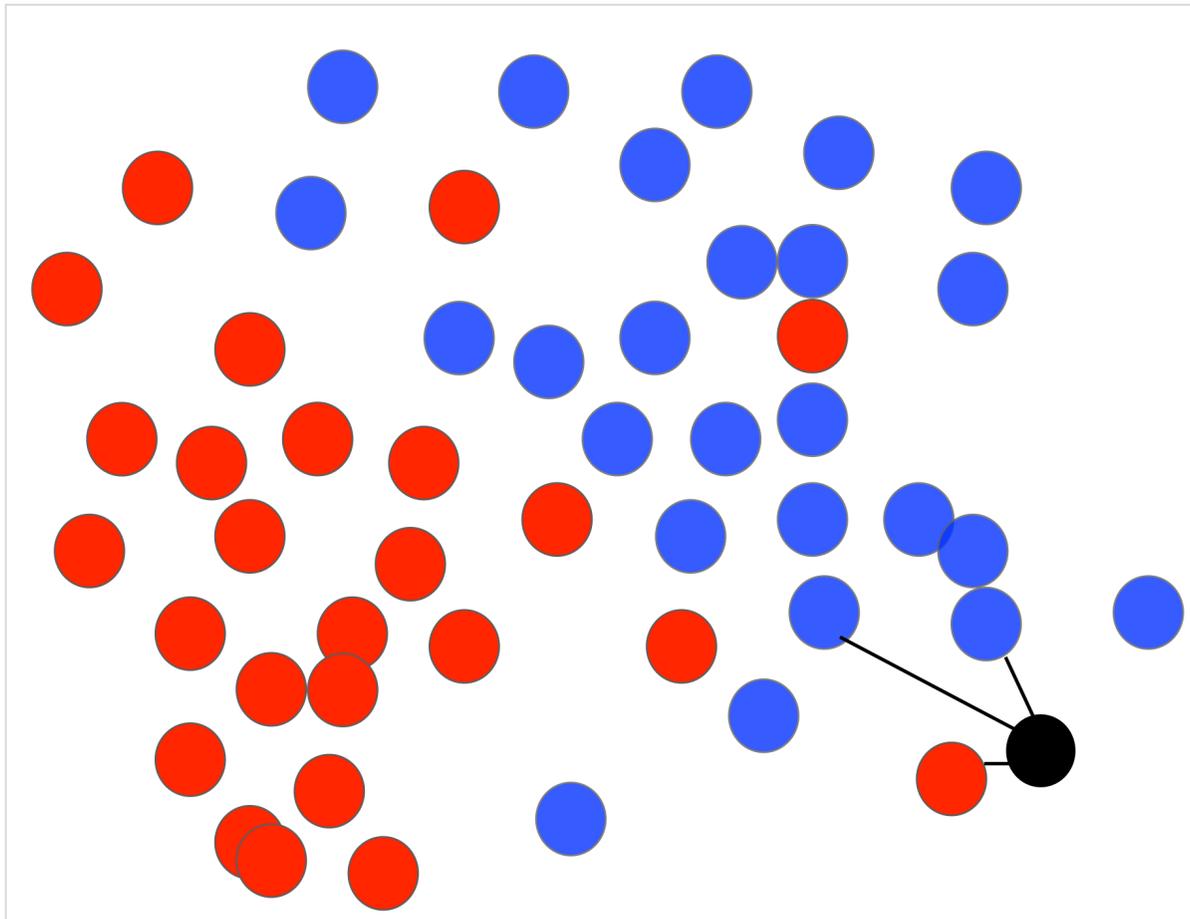
Approximate k Nearest Neighbor

가장 가까운 이웃 N 개 찾기 -> 근사한 이웃 N 개 찾기

방법 2

이미지의 색상 분포를 가지고 분류해보면 어떨까?

KNN 알고리즘 - 속도 문제 개선



Approximate k Nearest Neighbor

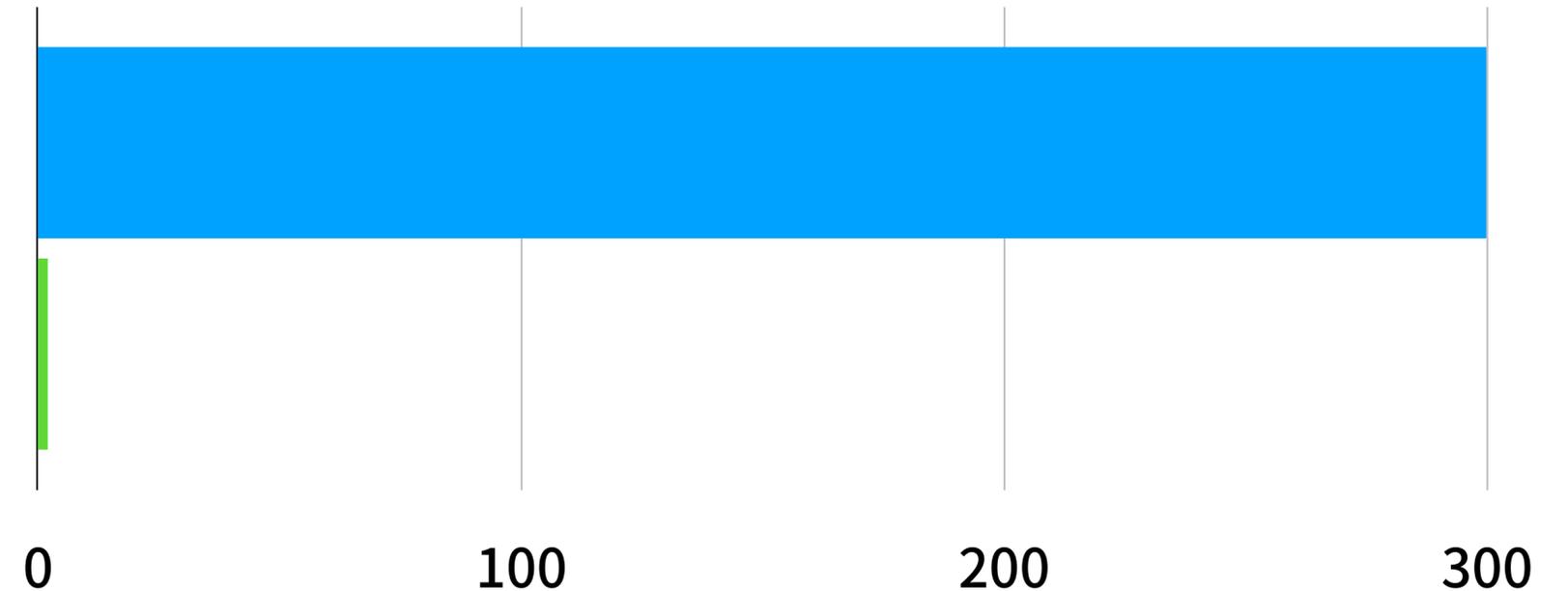
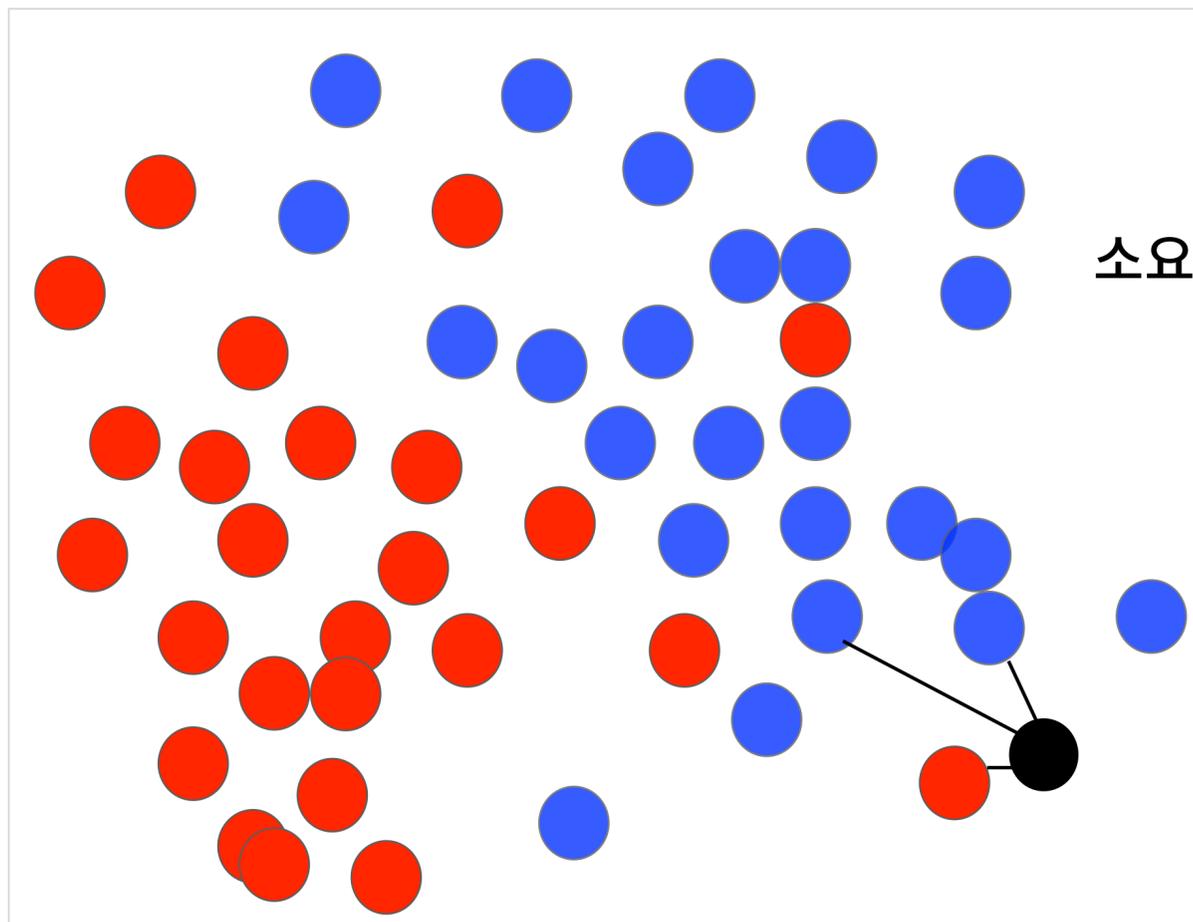
가장 가까운 이웃 N 개 찾기 -> 근사한 이웃 N 개 찾기

방법 2

이미지의 색상 분포를 가지고 분류해보면 어떨까?

KNN 알고리즘 - 속도 문제 개선

■ kNN ■ Approximate



Approximate k Nearest Neighbor

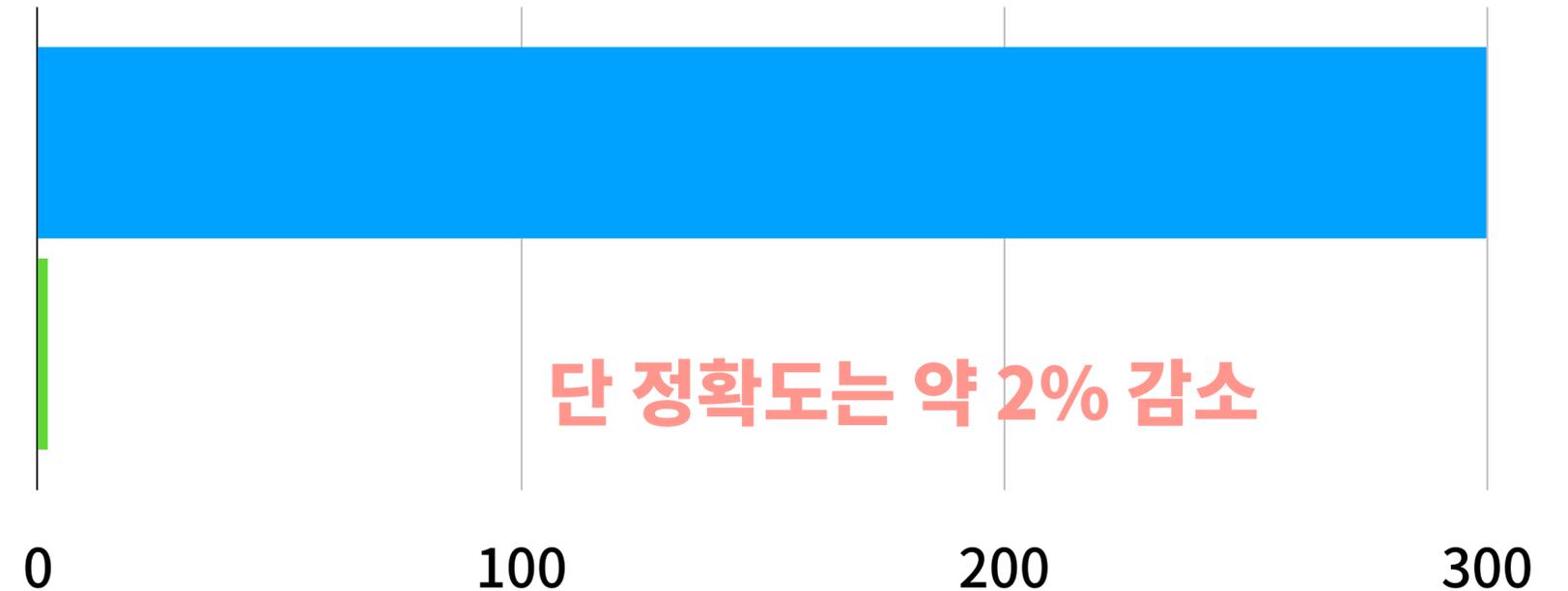
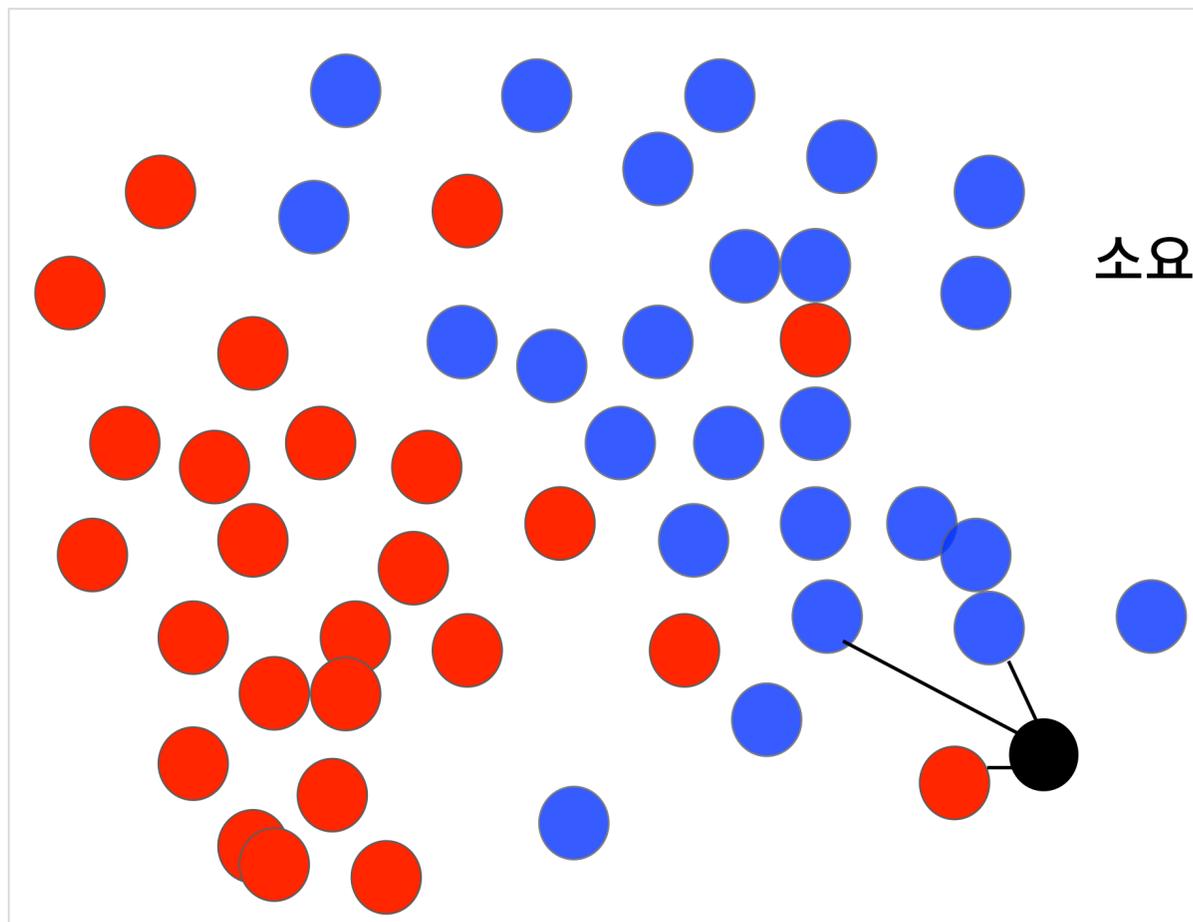
가장 가까운 이웃 N 개 찾기 -> 근사한 이웃 N 개 찾기

방법 2

이미지의 색상 분포를 가지고 분류해보면 어떨까?

KNN 알고리즘 - 속도 문제 개선

■ kNN ■ Approximate



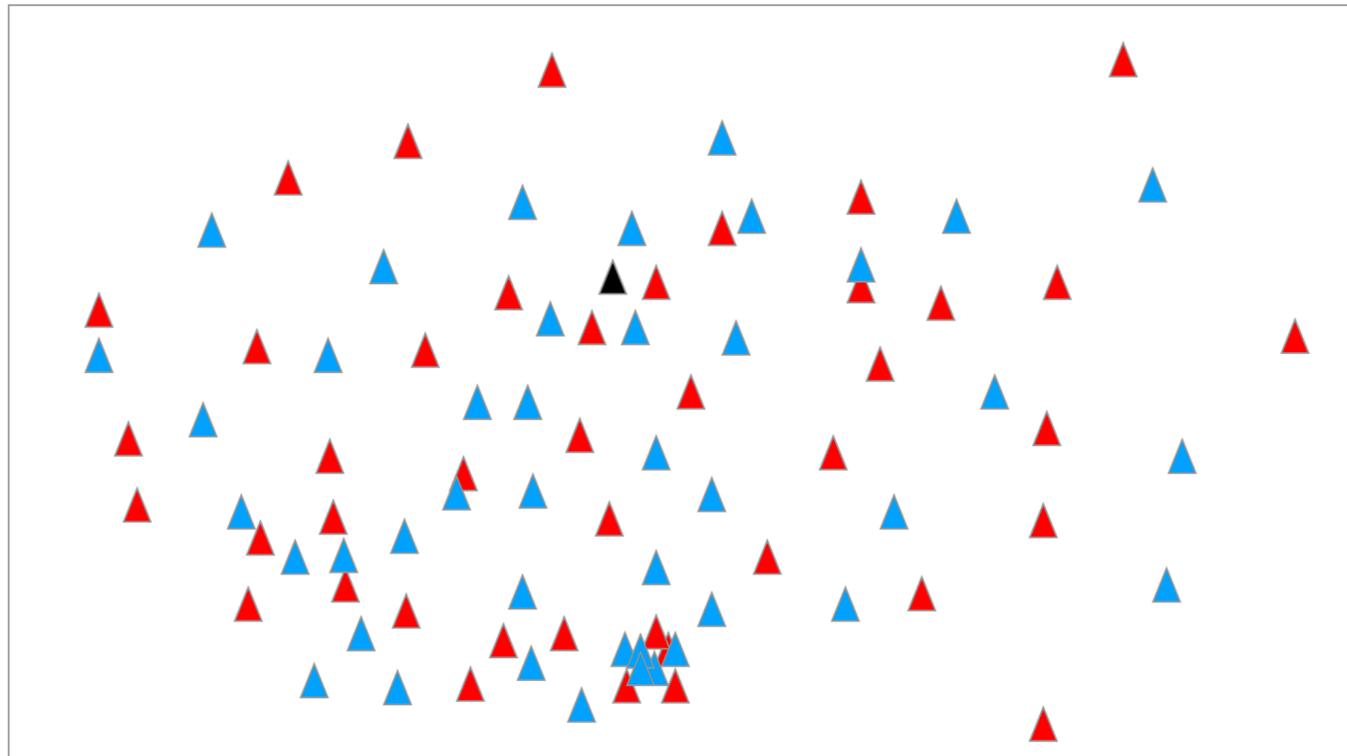
Approximate k Nearest Neighbor

가장 가까운 이웃 N 개 찾기 -> 근사한 이웃 N 개 찾기

방법 2

이미지의 색상 분포를 가지고 분류해보면 어떨까?

KNN 알고리즘 - 결과



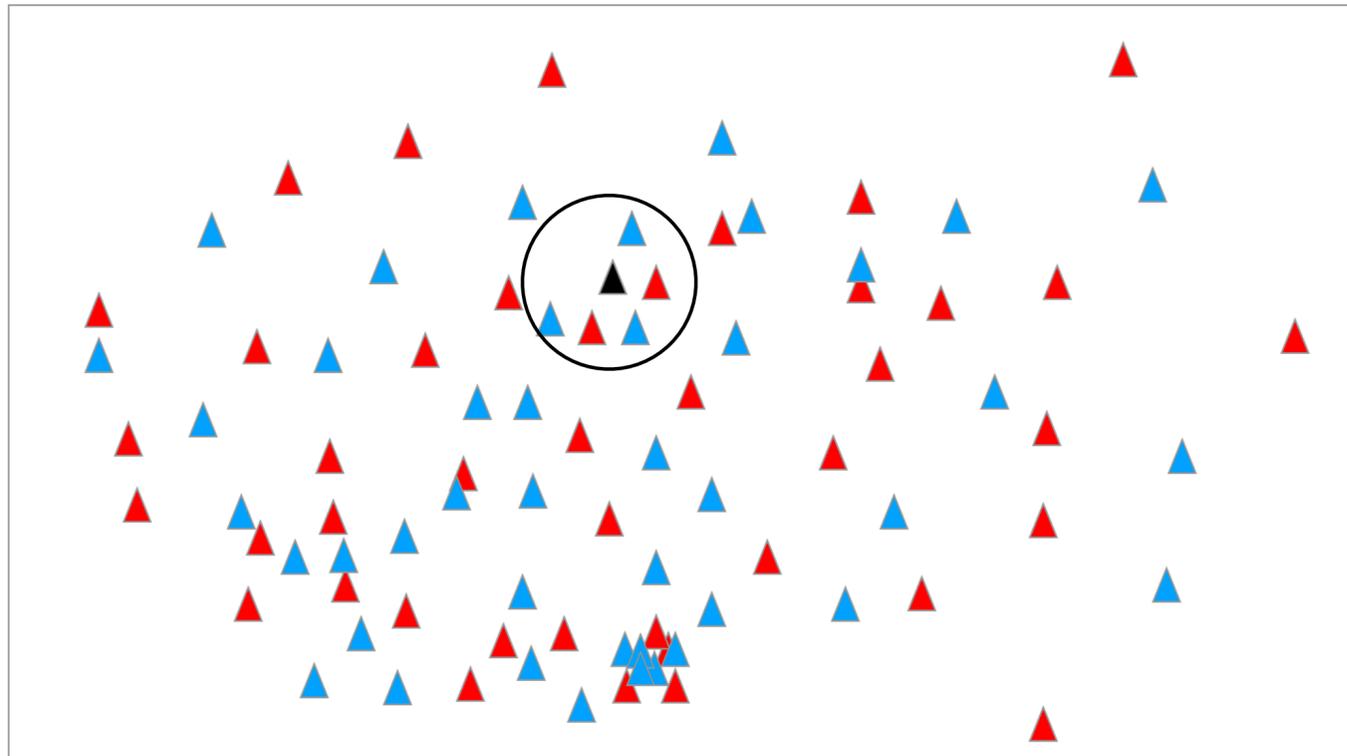
이해를 돕기 위해서 간략화 함

이미지가 군집화 되지 않았다

방법 2

이미지의 색상 분포를 가지고 분류해보면 어떨까?

KNN 알고리즘 - 결과



이해를 돕기 위해서 간략화 함

이미지가 군집화 되지 않았다

방법 2

이미지의 색상 분포를 가지고 분류해보면 어떨까?

KNN 알고리즘 - 결과

23%

방법 2

이미지의 색상 분포를 가지고 분류해보면 어떨까?

요약

단순히 픽셀의 색상 정보만으로는 분류가 어려움

04 방법 3

CNN 을 이용해 이미지를 분류해보기

방법 3

CNN 을 이용해 이미지를 분류해보기

CNN 을 이용해서 이미지를 분류하기

단순히 픽셀의 색상 정보를 넘어서 이미지 특성을 학습

방법 3

CNN 을 이용해 이미지를 분류해보기

CNN 을 이용해서 이미지를 분류하기

학습 속도 vs 분류 속도

방법 3

CNN 을 이용해 이미지를 분류해보기

CNN 을 이용해서 이미지를 분류하기

골빈해커의 3분 딥러닝 텐서플로맛

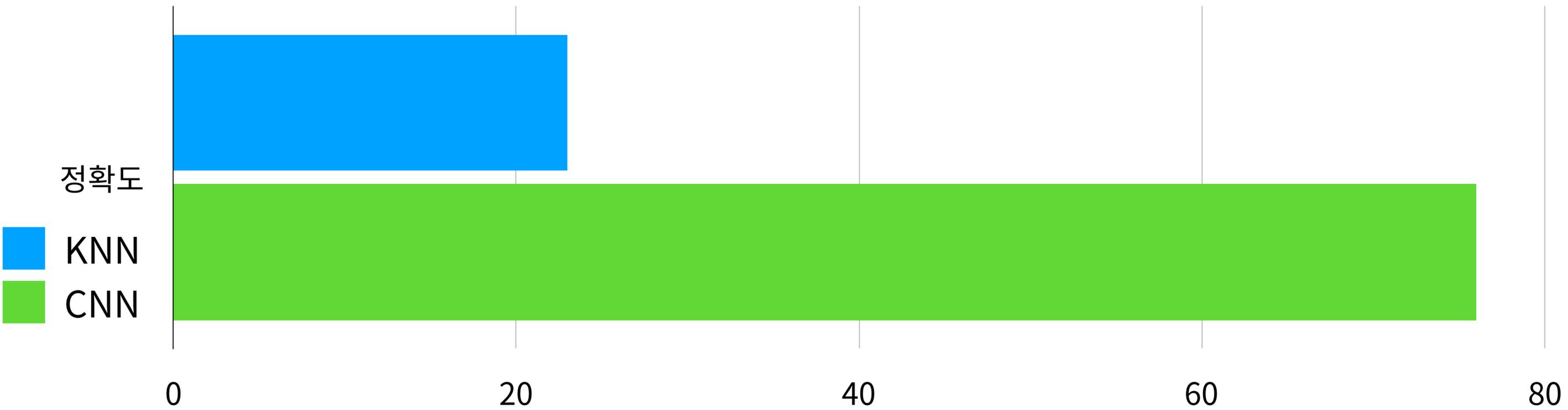
11장 - 구글의 핵심 이미지 인식 모델, Inception

방법 3

CNN 을 이용해 이미지를 분류해보기

KNN vs CNN (Inception v3)

23% $\xrightarrow{\text{정확도}}$ 76%



방법 2

CNN 을 이용해 이미지를 분류해보기

KNN vs CNN (Inception v3)

1분 10초

분류 시간 →

0.42s

방법 3

CNN 을 이용해 이미지를 분류해보기

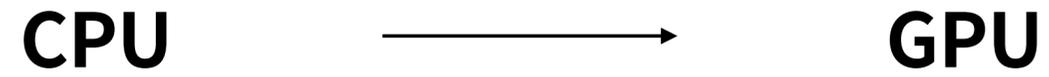
KNN vs CNN (Inception v3)



방법 3

CNN 을 이용해 이미지를 분류해보기

KNN vs CNN (Inception v3)



방법 3

CNN 을 이용해 이미지를 분류해보기

KNN vs CNN (Inception v3)

	KNN (Approximate)	CNN (Inception v3)
정확도	23%	76%
분류속도	1분 10초	0.4 초
학습속도	X	13 일
사용장비	CPU	GPU

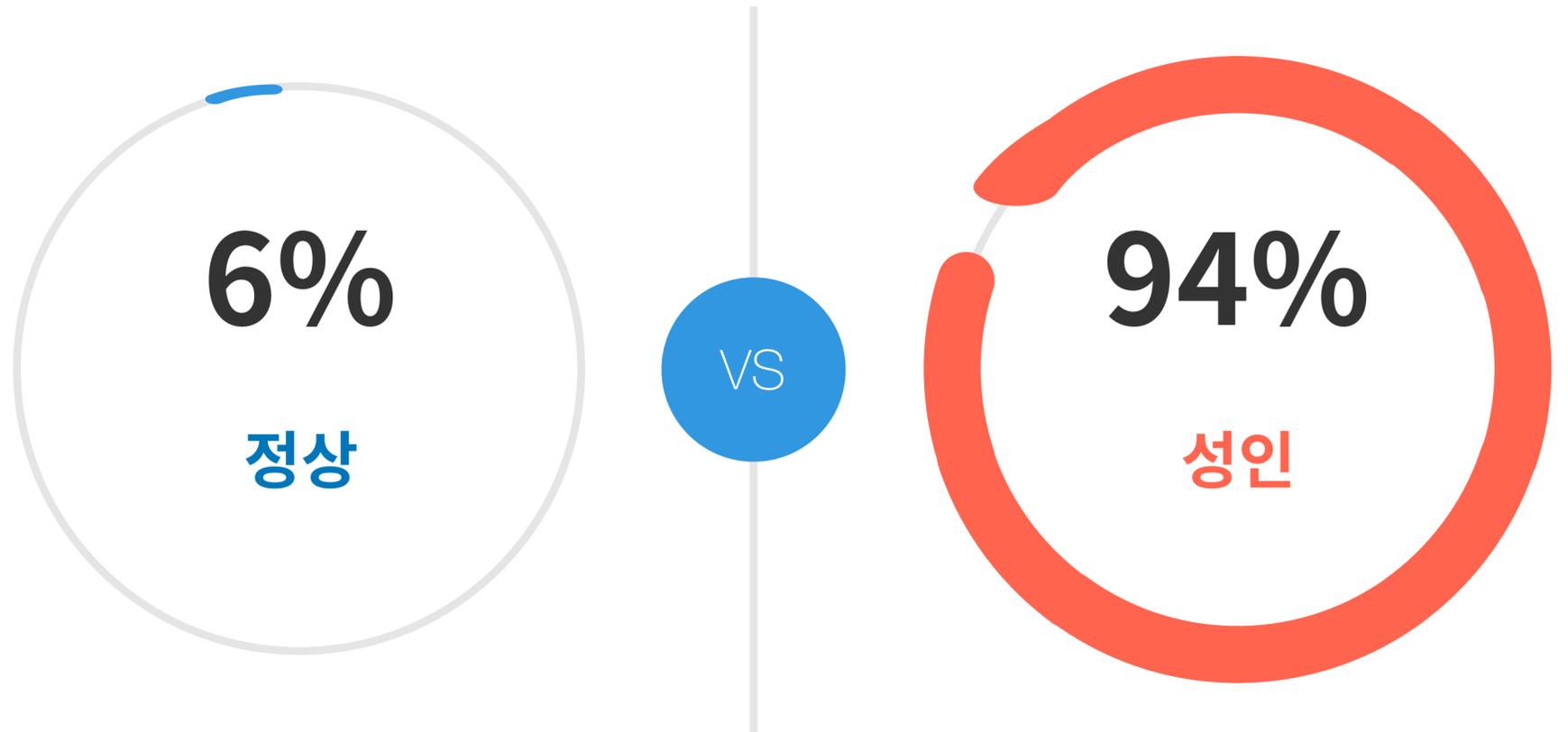
방법 3

CNN 을 이용해 이미지를 분류해보기

CNN 분류 결과 에서 발생한 문제점

만화책 이미지
(대사가 있고 펜 느낌)

분류결과



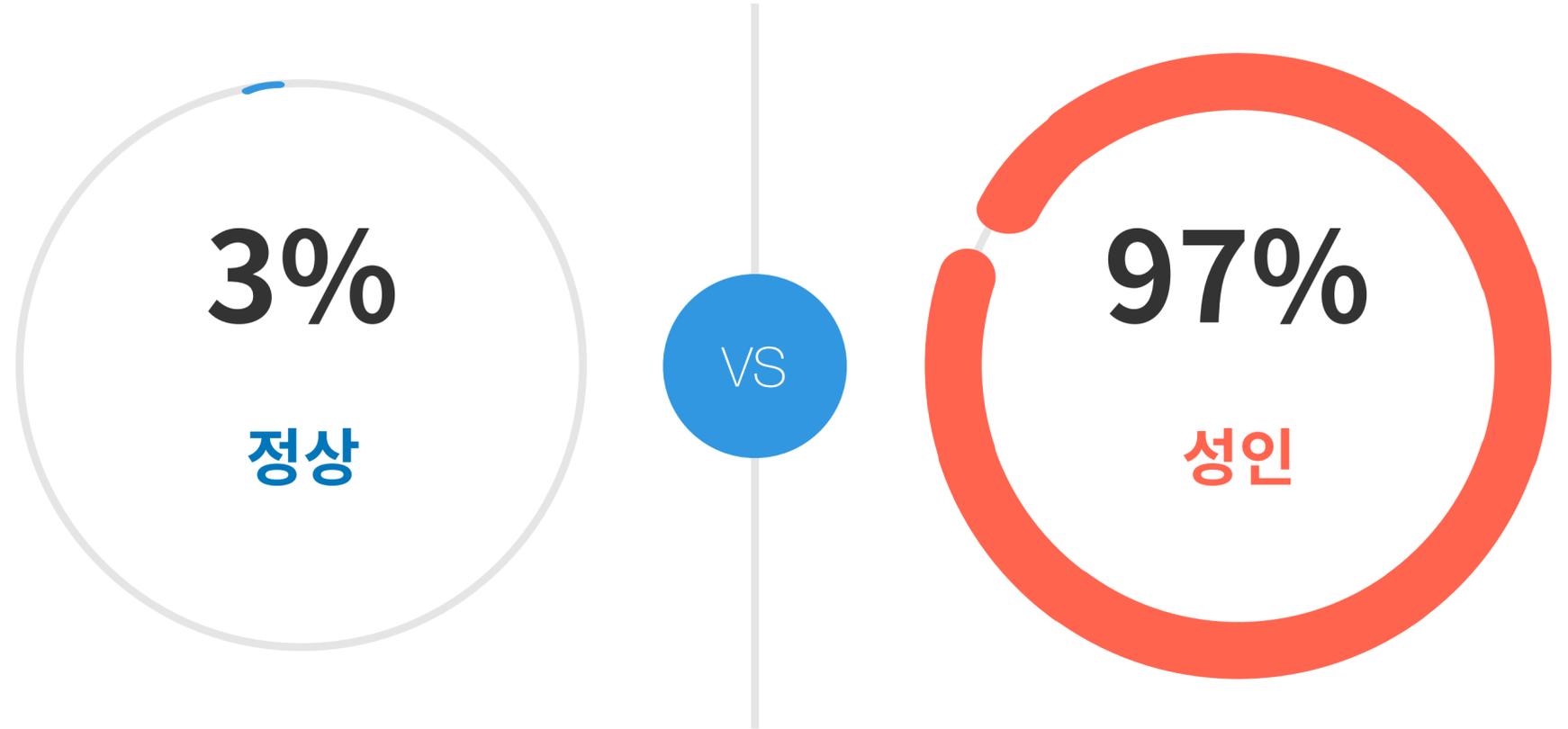
방법 3

CNN 을 이용해 이미지를 분류해보기

CNN 분류 결과 에서 발생한 문제점

강렬한 색감을 가진
카툰 이미지

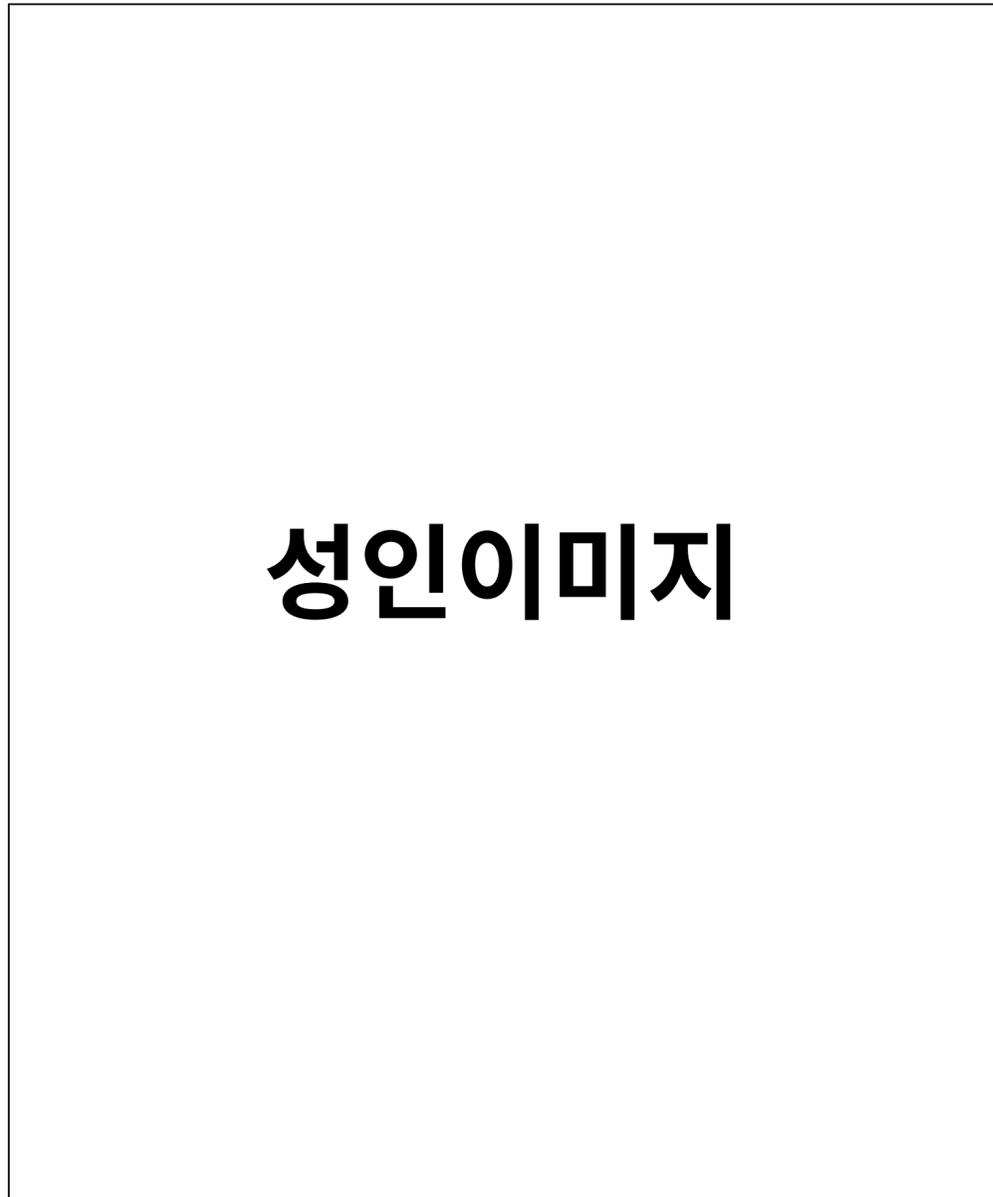
분류결과



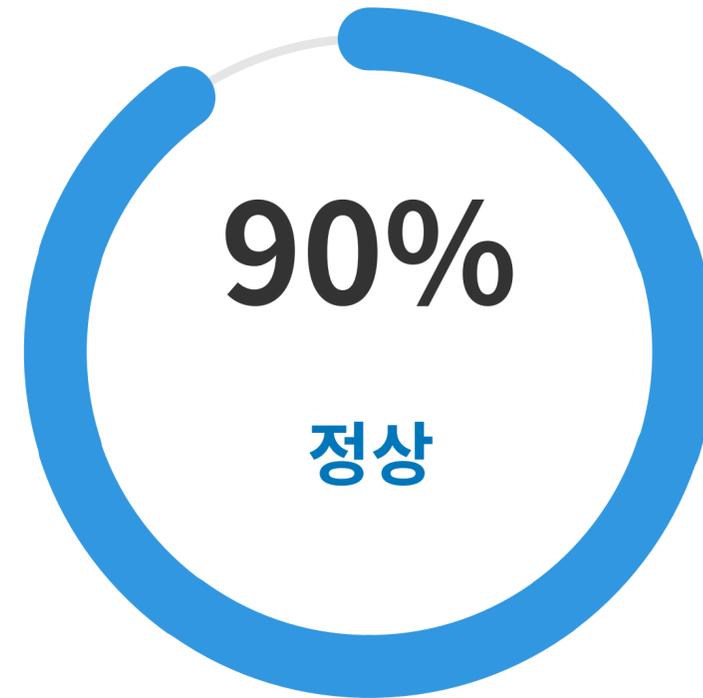
방법 3

CNN 을 이용해 이미지를 분류해보기

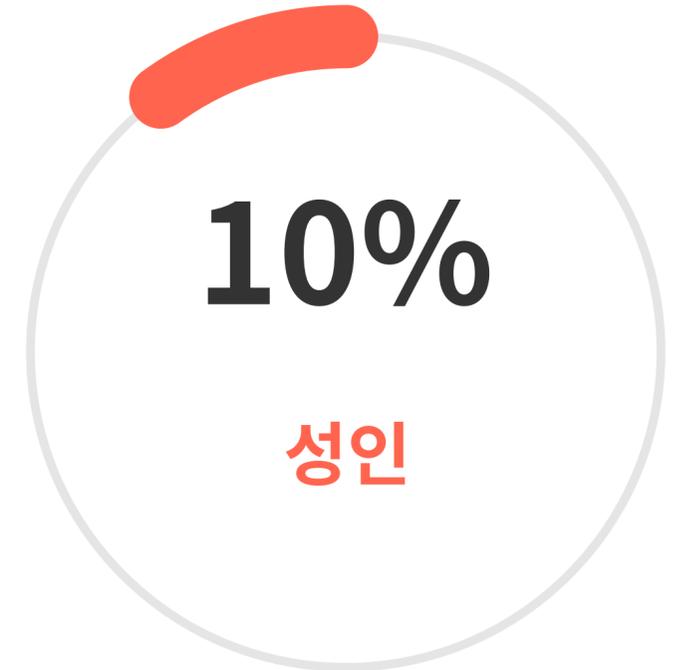
CNN 분류 결과 에서 발생한 문제점



분류결과



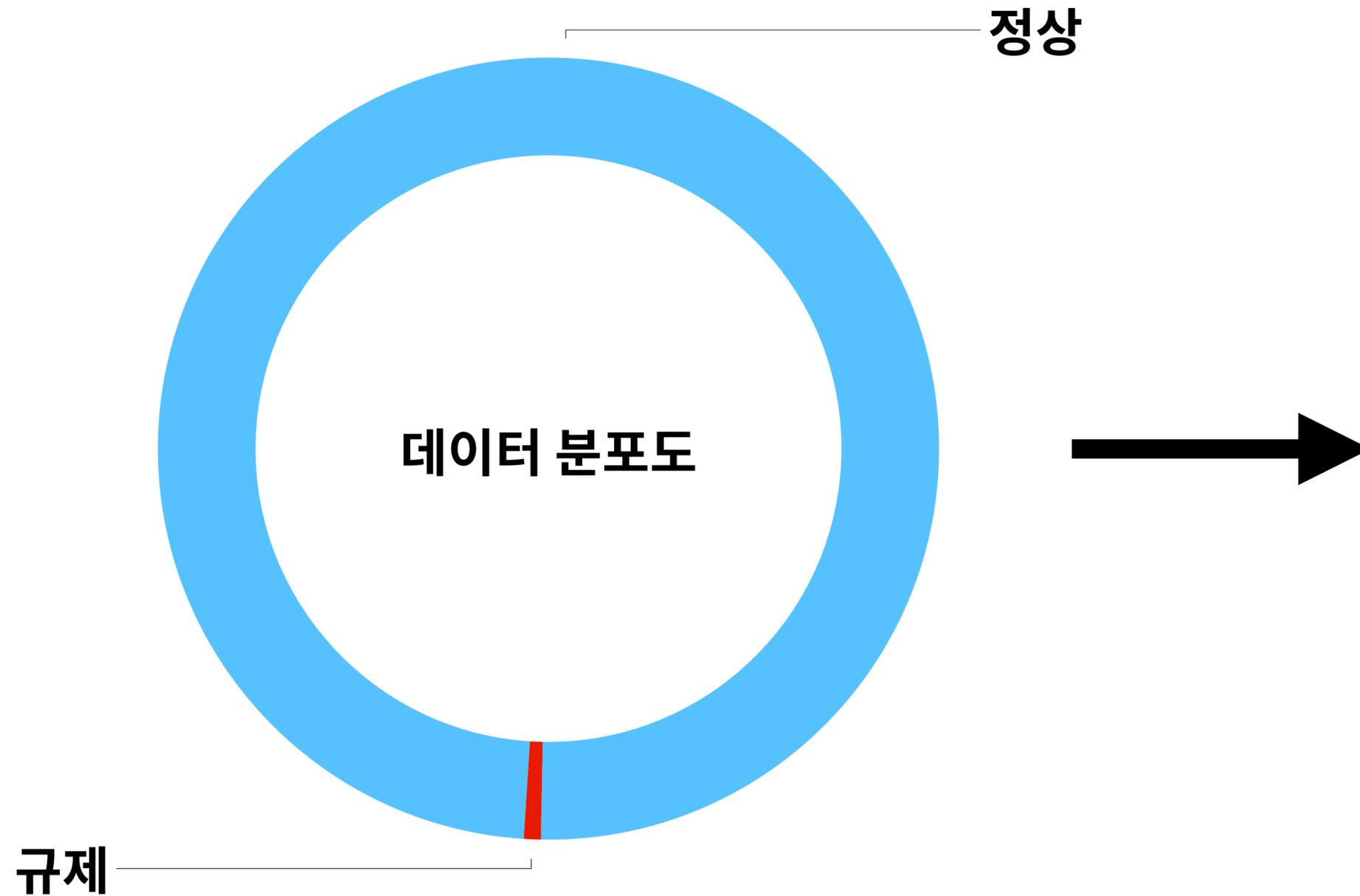
VS



방법 3

CNN 을 이용해 이미지를 분류해보기

1차 문제점 해결 - 데이터 불균형 문제



데이터 불균형 문제

정확도는 높아도 재현율은 떨어짐

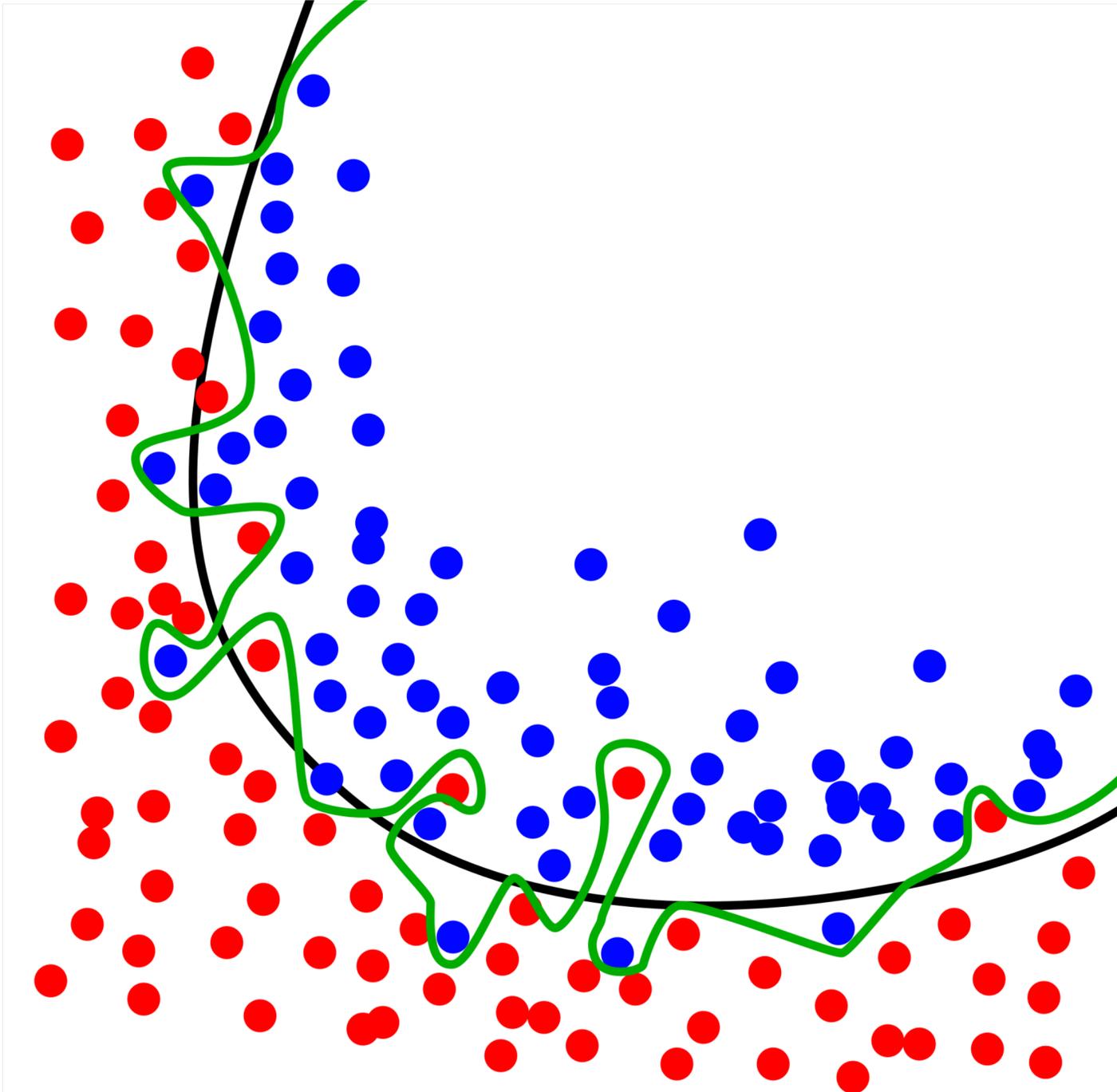
방법 3

CNN 을 이용해 이미지를 분류해보기

Overfitting ?

방법 3

CNN 을 이용해 이미지를 분류해보기



Overfitting ?

너무 과하게 학습을 하는 상태

출처 : <https://ko.wikipedia.org/wiki/과적합>

방법 3

CNN 을 이용해 이미지를 분류해보기

1차 문제점 해결 - 데이터 불균형 문제

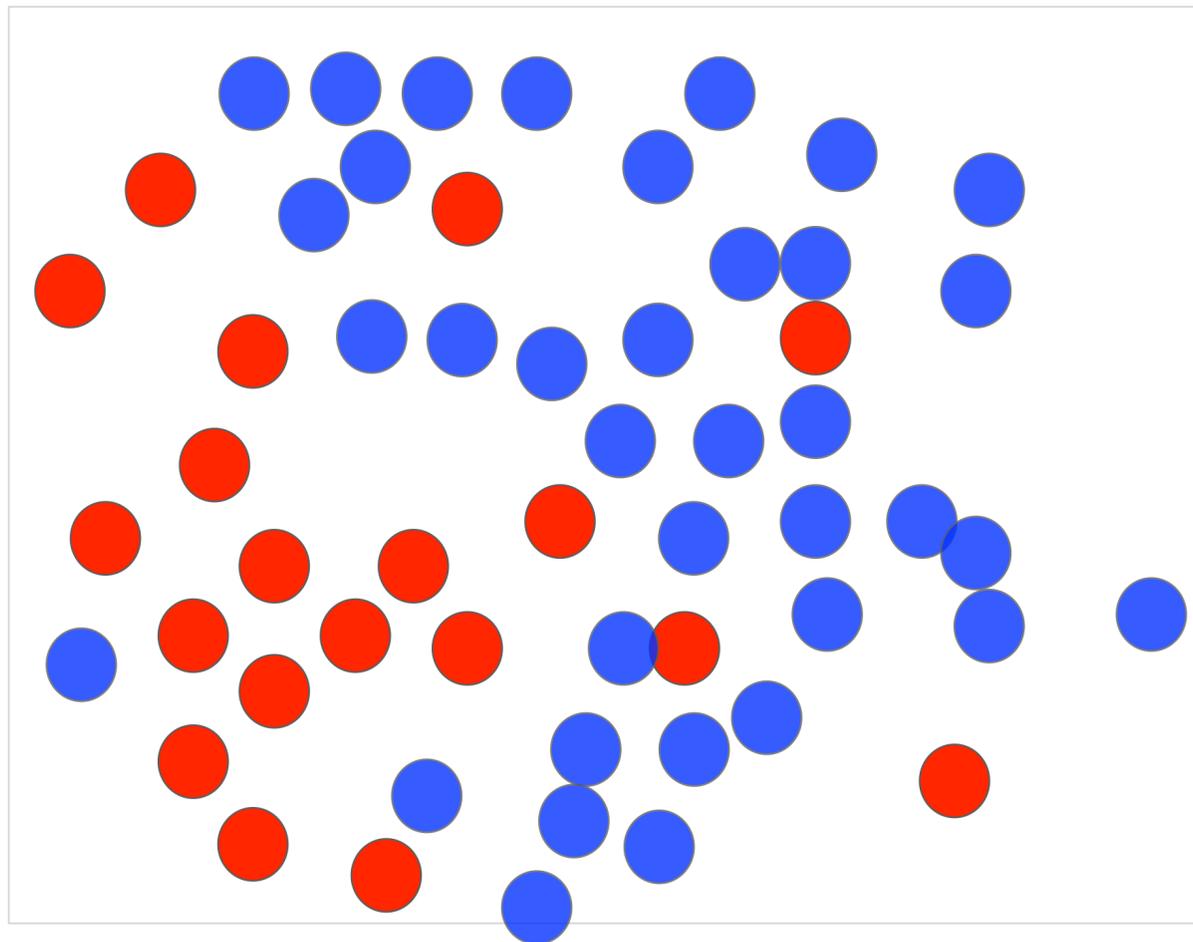
Random Under Sampling

무작위로 데이터를 제거함

방법 3

CNN 을 이용해 이미지를 분류해보기

1차 문제점 해결 - 데이터 불균형 문제



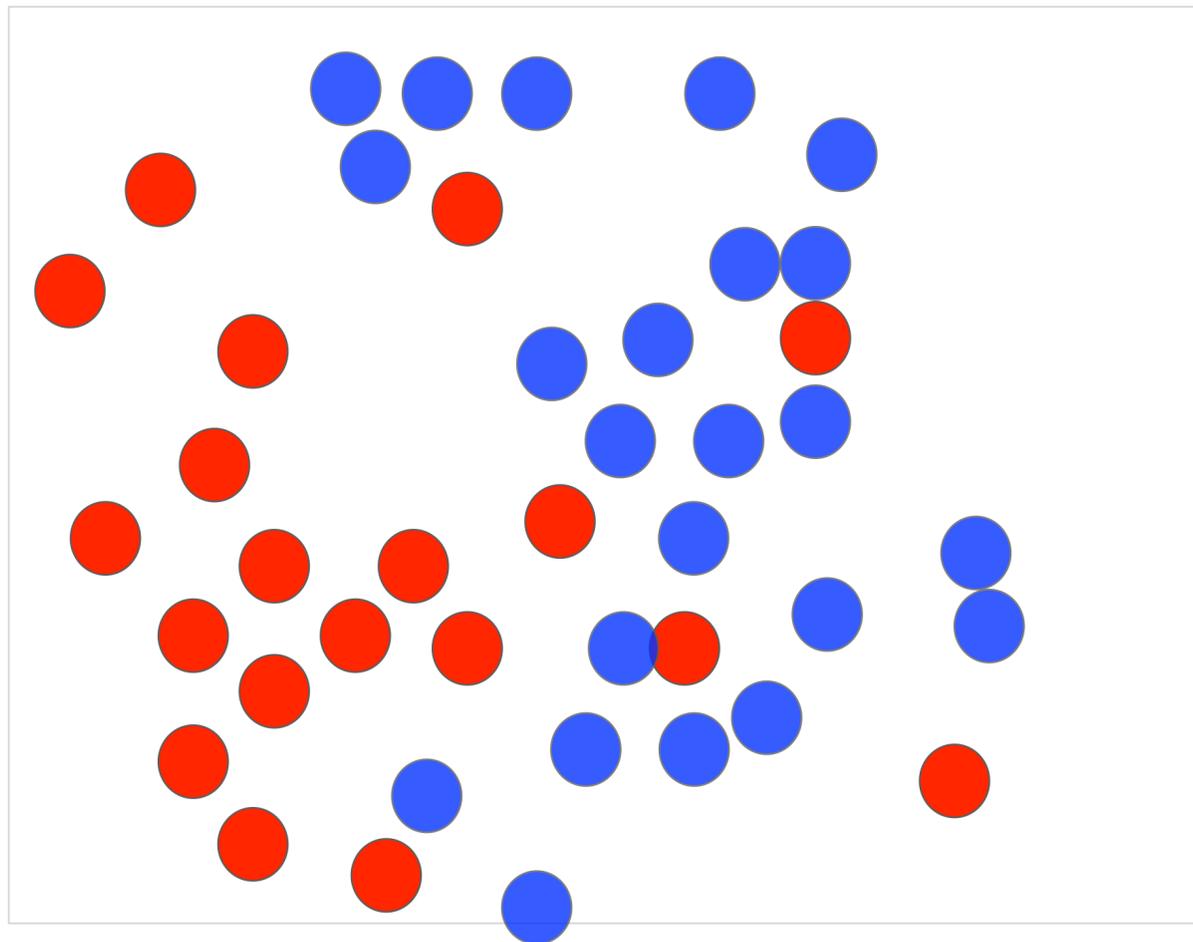
Random Under Sampling

무작위로 데이터를 제거함

방법 3

CNN 을 이용해 이미지를 분류해보기

1차 문제점 해결 - 데이터 불균형 문제



Random Under Sampling

무작위로 데이터를 제거함

방법 3

CNN 을 이용해 이미지를 분류해보기

1차 문제점 해결 - 데이터 불균형 문제

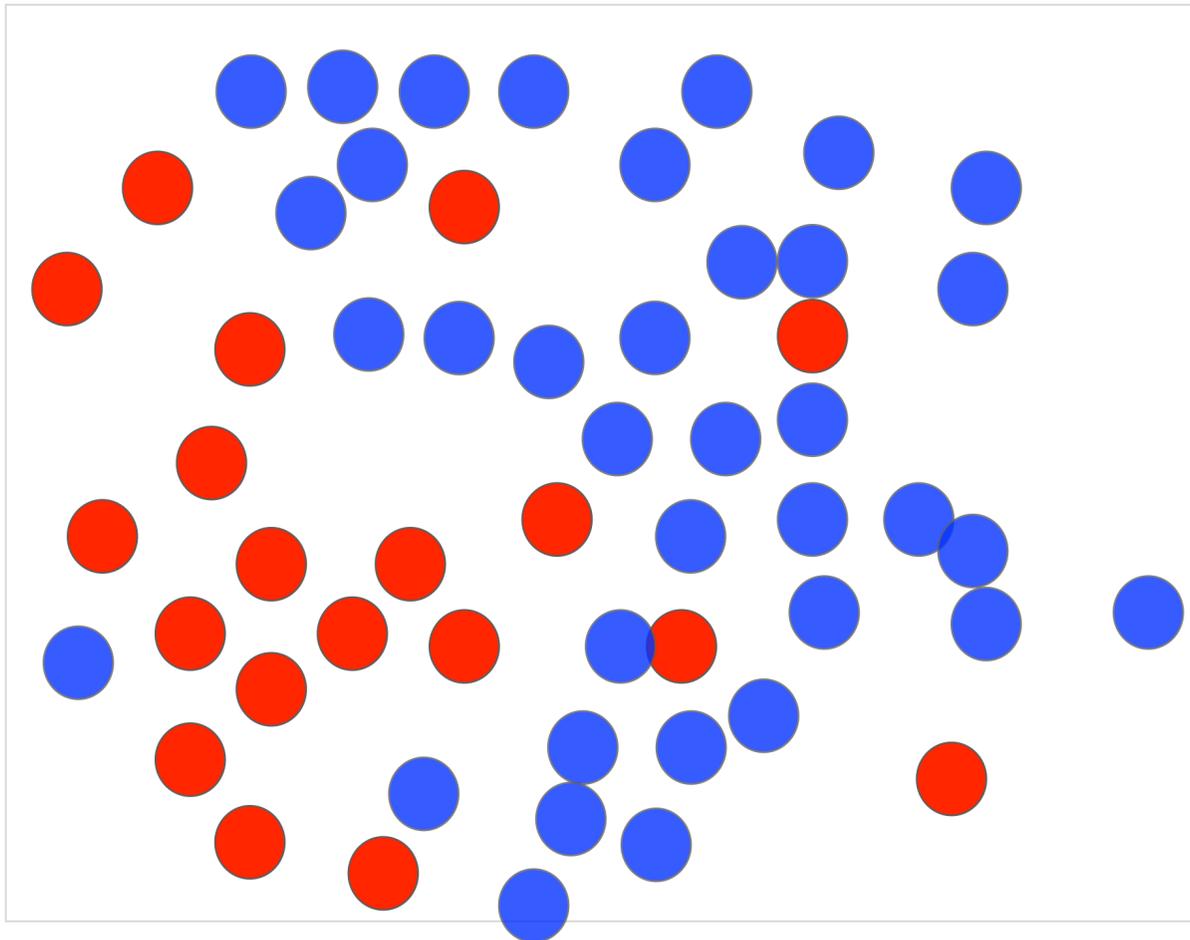
Tomek Link

서로 다른 클래스값이 거리값이 가까운 경우를 찾아서
데이터가 많은 클래스 값은 제거함

방법 3

CNN 을 이용해 이미지를 분류해보기

1차 문제점 해결 - 데이터 불균형 문제



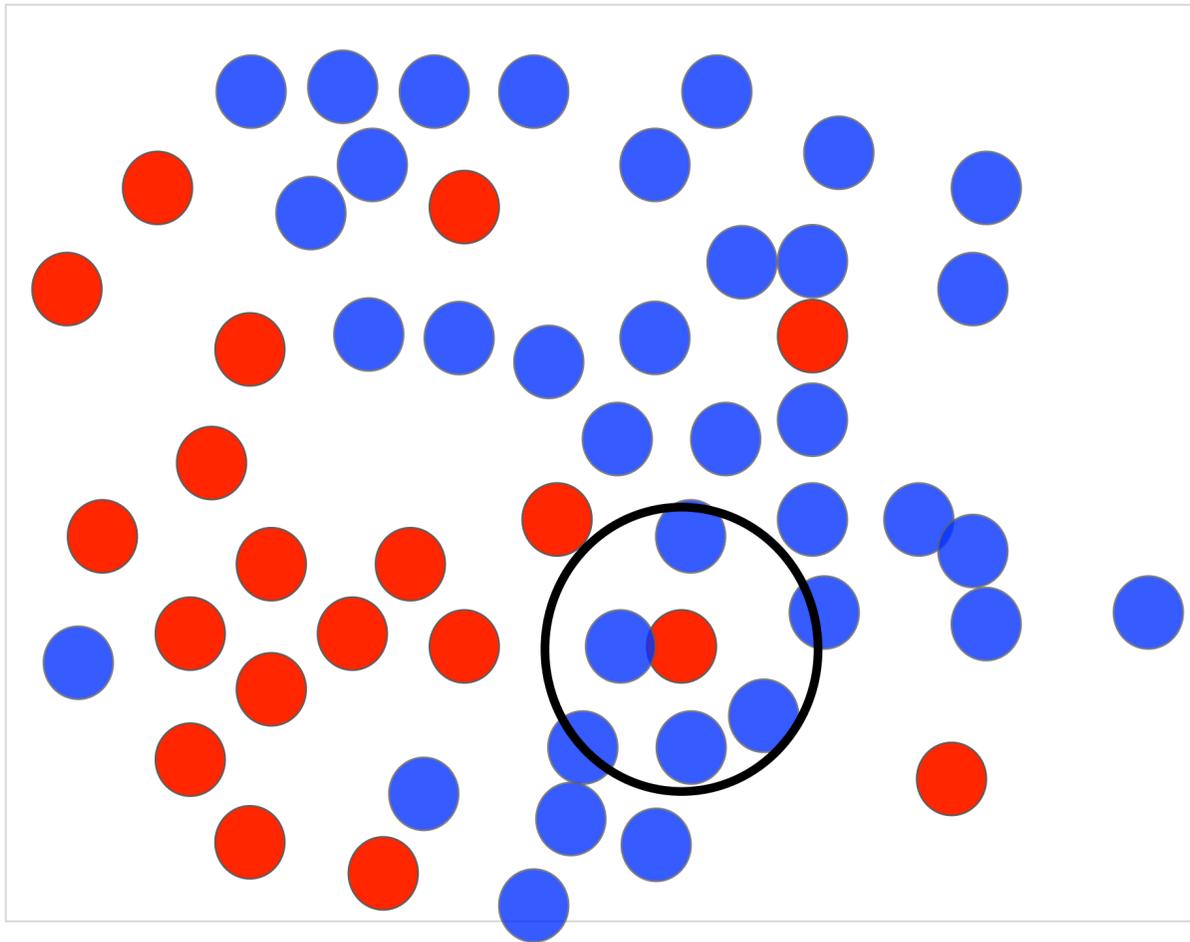
Tomek Link

서로 다른 클래스값이 거리값이 가까운 경우를 찾아서
데이터가 많은 클래스 값은 제거함

방법 3

CNN 을 이용해 이미지를 분류해보기

1차 문제점 해결 - 데이터 불균형 문제



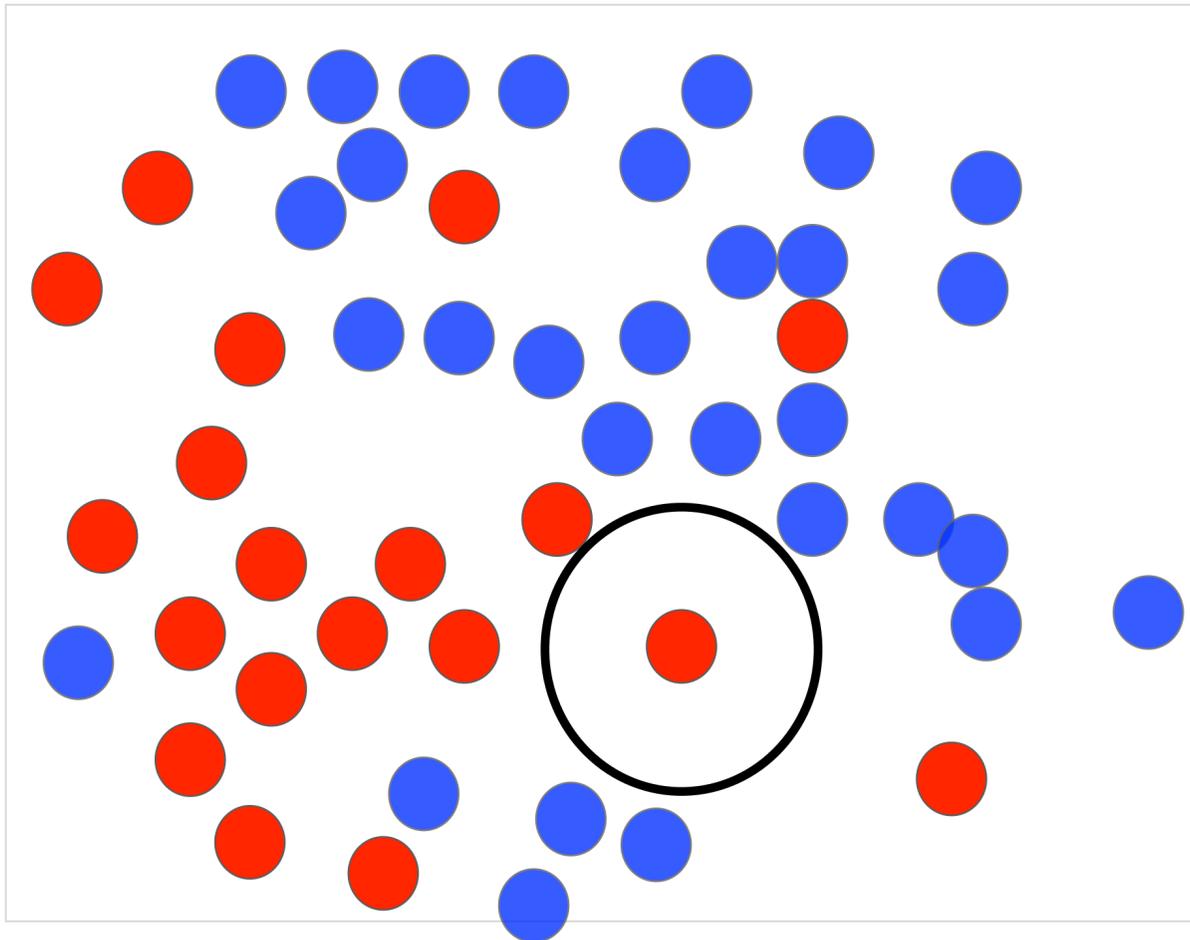
Tomek Link

서로 다른 클래스값이 거리값이 가까운 경우를 찾아서
데이터가 많은 클래스 값은 제거함

방법 3

CNN 을 이용해 이미지를 분류해보기

1차 문제점 해결 - 데이터 불균형 문제



Tomek Link

서로 다른 클래스값이 거리값이 가까운 경우를 찾아서
데이터가 많은 클래스 값은 제거함

방법 3

CNN 을 이용해 이미지를 분류해보기

1차 문제점 해결 - 데이터 불균형 문제

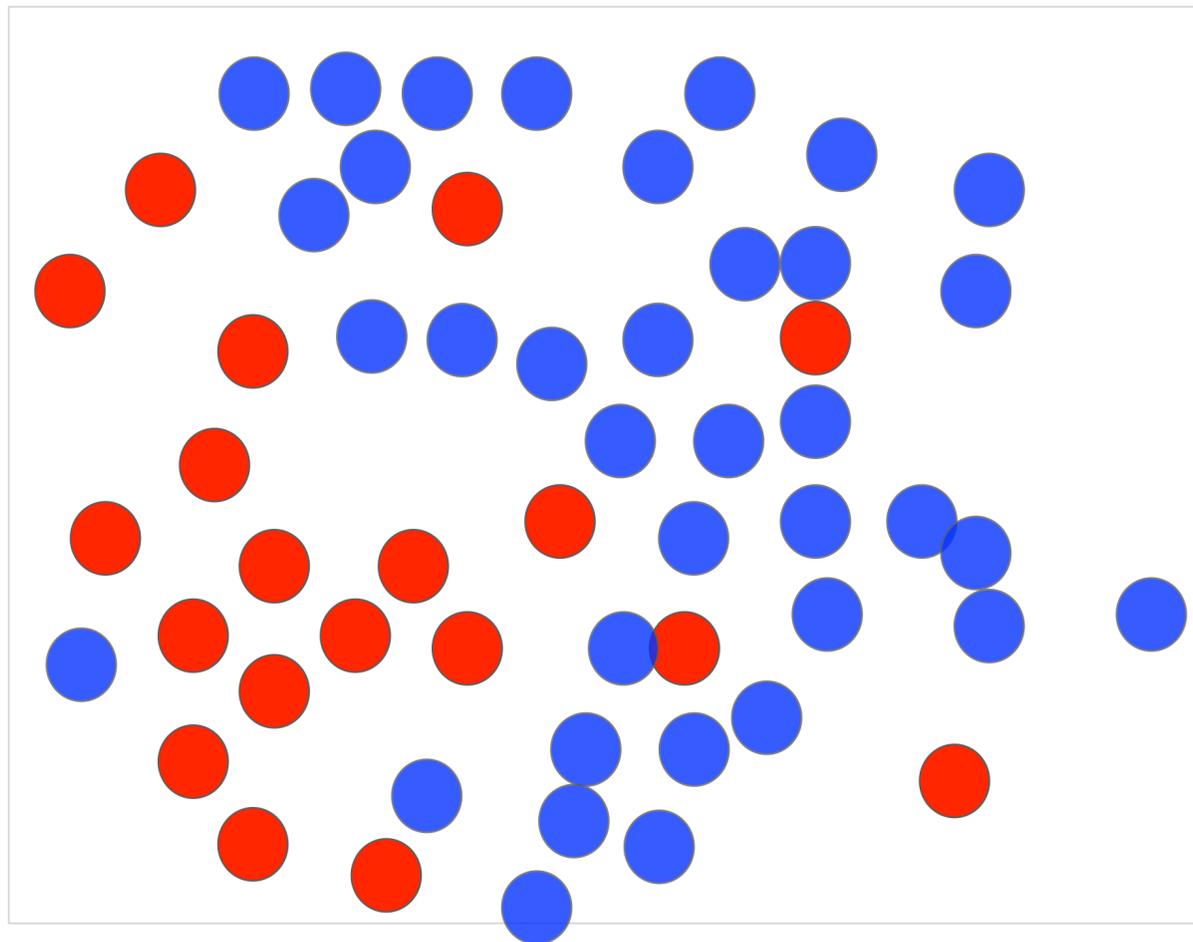
Condensed Nearest Neighbor

가까운 거리의 같은 클래스 인 경우 제거함

방법 3

CNN 을 이용해 이미지를 분류해보기

1차 문제점 해결 - 데이터 불균형 문제



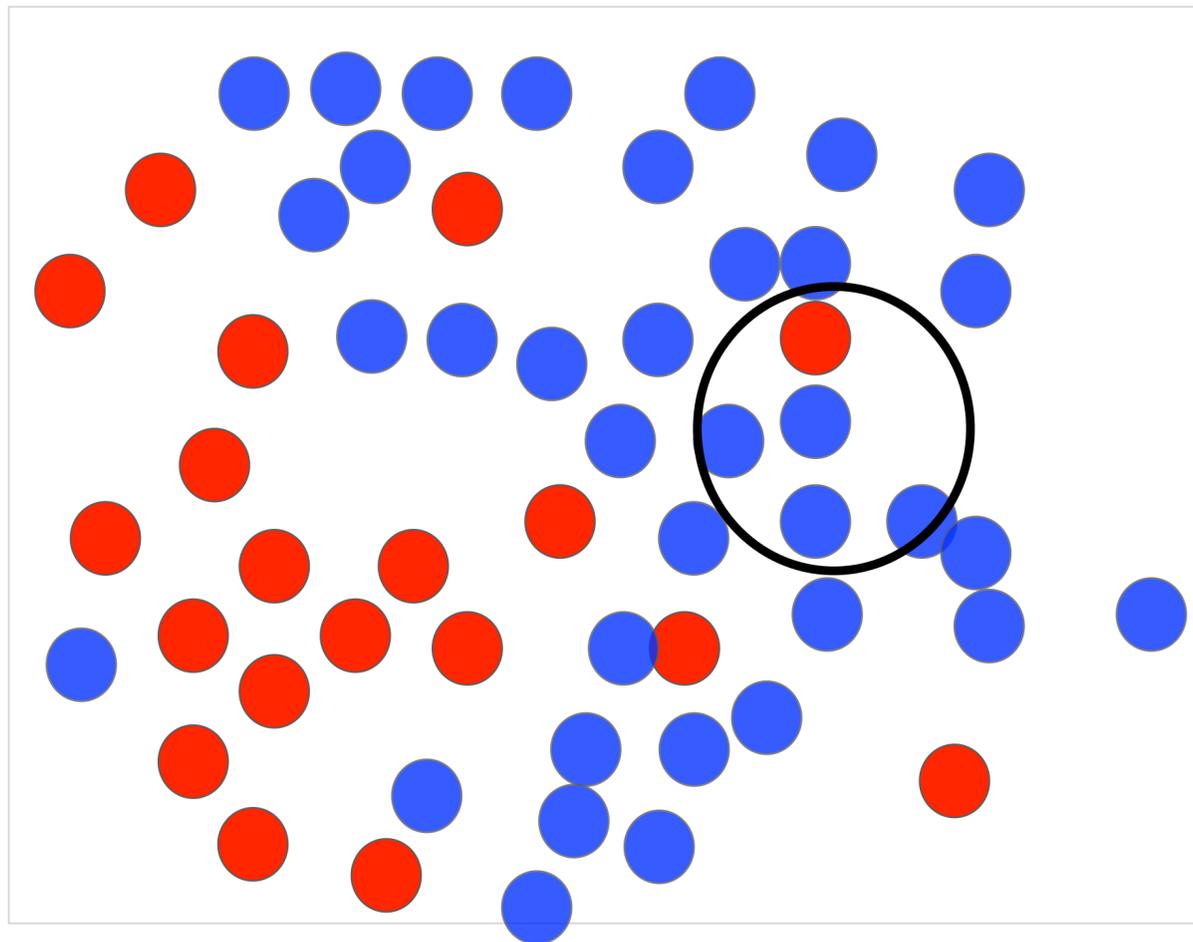
Condensed Nearest Neighbor

가까운 거리의 같은 클래스 인 경우 제거함

방법 3

CNN 을 이용해 이미지를 분류해보기

1차 문제점 해결 - 데이터 불균형 문제



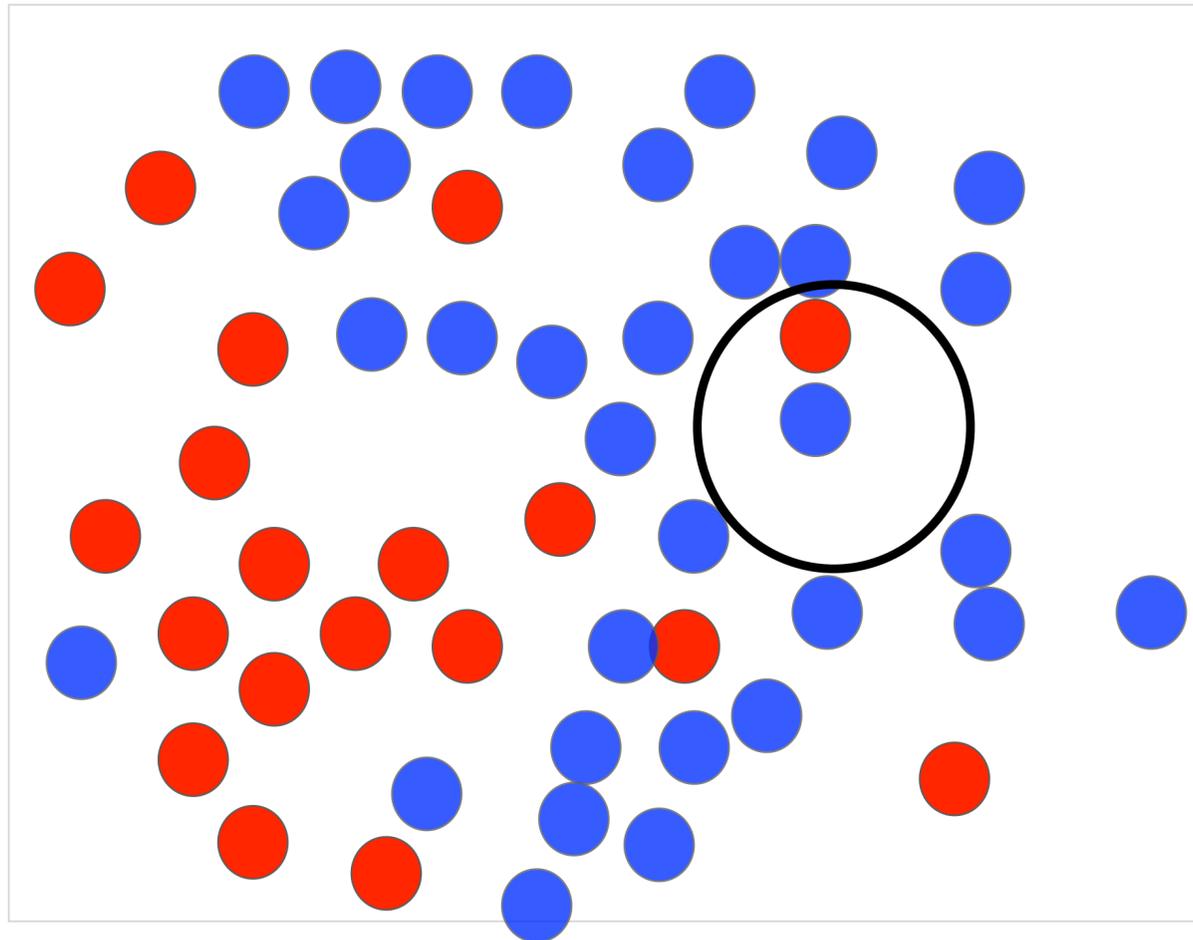
Condensed Nearest Neighbor

가까운 거리의 같은 클래스 인 경우 제거함

방법 3

CNN 을 이용해 이미지를 분류해보기

1차 문제점 해결 - 데이터 불균형 문제



Condensed Nearest Neighbor

가까운 거리의 같은 클래스 인 경우 제거함

방법 3

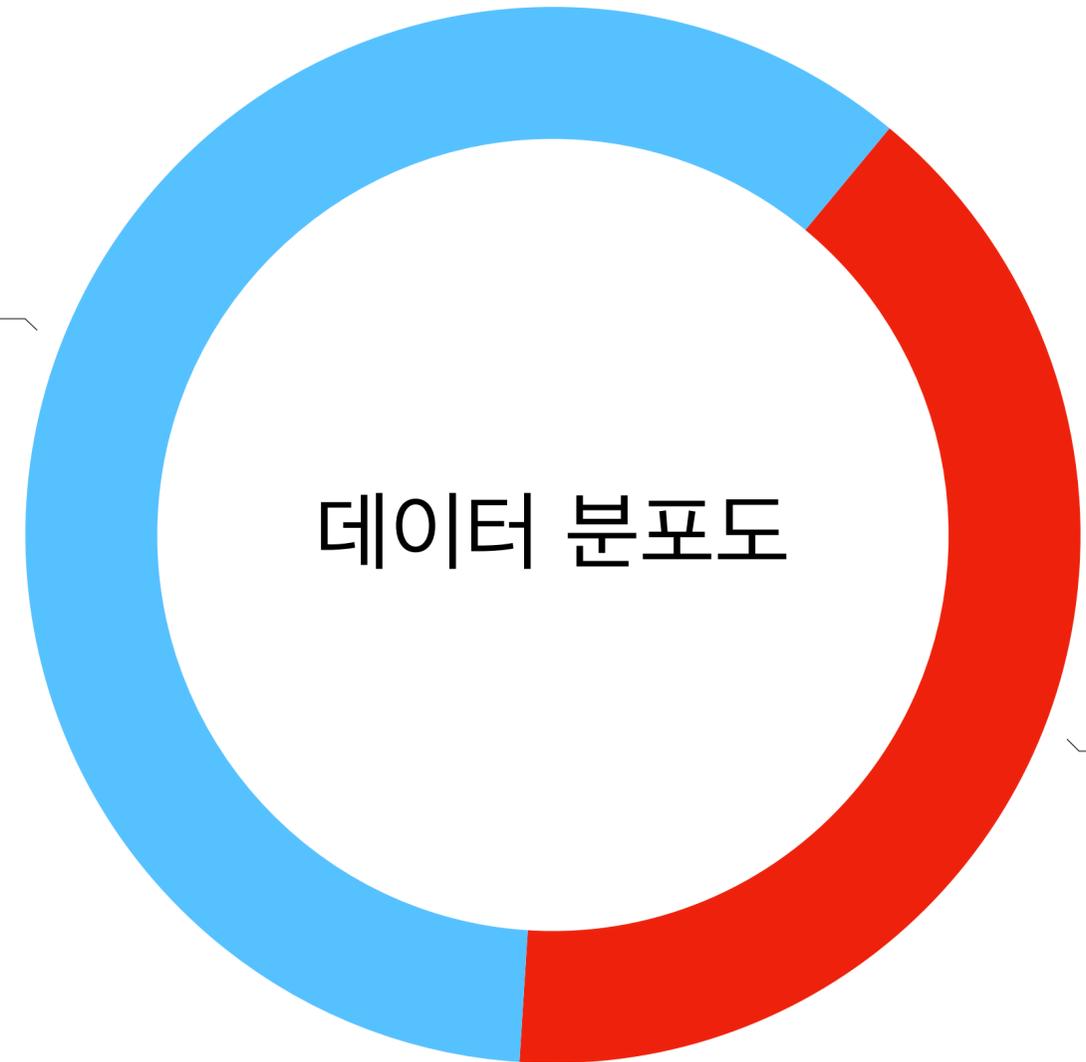
CNN 을 이용해 이미지를 분류해보기

1차 문제점 해결 - 데이터 불균형 문제

Random Under Sampling
+
Tomek Link
+
Condensed Nearest Neighbor



정상



데이터 분포도

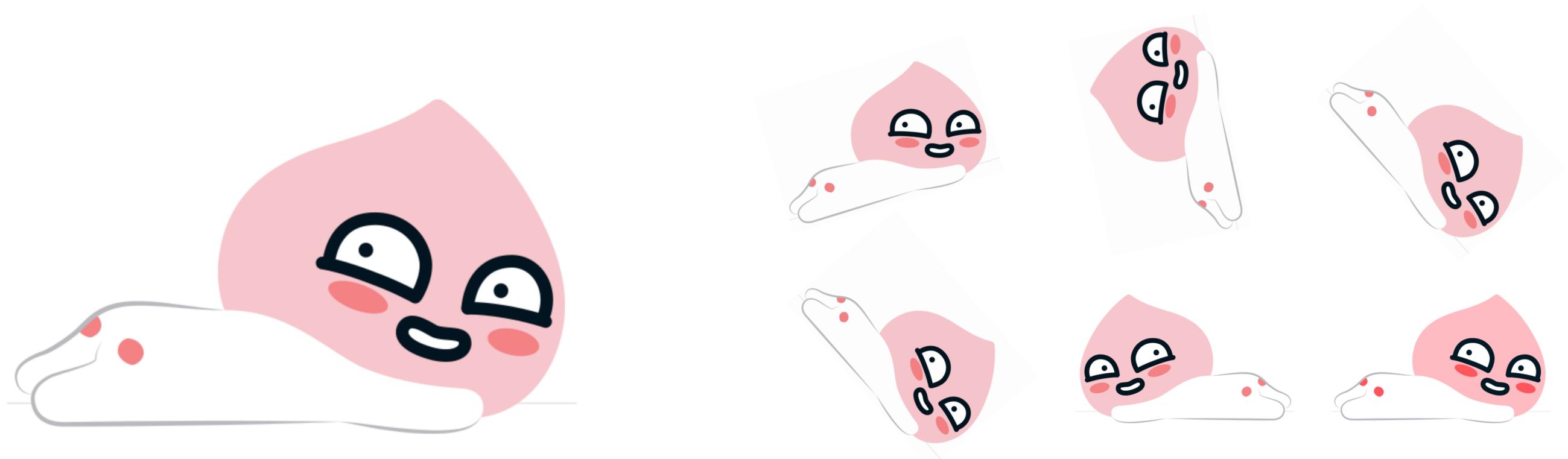
성인

방법 3

CNN 을 이용해 이미지를 분류해보기

1차 문제점 해결 - 데이터 불균형 문제

Data augmentation



방법 3

CNN 을 이용해 이미지를 분류해보기

1차 문제점 해결 - 반복되는 코드 작성

Tensorflow slim

```
./scripts/finetune_inception_resnet_v2_on_flowers.sh

# Download the pre-trained checkpoint.
if [ ! -d "$PRETRAINED_CHECKPOINT_DIR" ]; then
  mkdir ${PRETRAINED_CHECKPOINT_DIR}
fi
if [ ! -f ${PRETRAINED_CHECKPOINT_DIR}/${MODEL_NAME}.ckpt ]; then
  wget http://download.tensorflow.org/models/inception_resnet_v2_2016_08_30.tar.gz
  tar -xvf inception_resnet_v2_2016_08_30.tar.gz
  mv inception_resnet_v2.ckpt ${PRETRAINED_CHECKPOINT_DIR}/${MODEL_NAME}.ckpt
  rm inception_resnet_v2_2016_08_30.tar.gz
fi

# Download the dataset
python download_and_convert_data.py \
  --dataset_name=flowers \
  --dataset_dir=${DATASET_DIR}

# Fine-tune only the new layers for 1000 steps.
python train_image_classifier.py \
  --train_dir=${TRAIN_DIR} \
  --dataset_name=flowers \
  --dataset_split_name=train \
  --dataset_dir=${DATASET_DIR} \
  --model_name=${MODEL_NAME} \
  --checkpoint_path=${PRETRAINED_CHECKPOINT_DIR}/${MODEL_NAME}.ckpt \
  --checkpoint_exclude_scopes=InceptionResnetV2/Logits,InceptionResnetV2/AuxLogits \
  --trainable_scopes=InceptionResnetV2/Logits,InceptionResnetV2/AuxLogits \
  --max_number_of_steps=1000 \
  --batch_size=32 \
  --learning_rate=0.01 \
  --learning_rate_decay_type=fixed \
  --save_interval_secs=60 \
  --save_summaries_secs=60 \
  --log_every_n_steps=10 \
  --optimizer=rmsprop \
  --weight_decay=0.00004

# Run evaluation.
python eval_image_classifier.py \
  --checkpoint_path=${TRAIN_DIR} \
  --eval_dir=${TRAIN_DIR} \
  --dataset_name=flowers \
  --dataset_split_name=validation \
  --dataset_dir=${DATASET_DIR} \
  --model_name=${MODEL_NAME}

# .....
```

코드 작성 없이 학습 가능

방법 3

CNN 을 이용해 이미지를 분류해보기

1차 문제점 해결 - 정확도를 좀 더 향상 시키기

Pre-trained Model

+

Fine tune

이미 학습되어있는 모델에 학습 데이터를 추가하여 파라미터를 조정

방법 3

CNN 을 이용해 이미지를 분류해보기

1차 문제점 해결 - 정확도를 좀 더 향상 시키기

	Top-1	Top-5
Inception V3	78.8%	94.4%
Inception Resnet V2	80.1%	95.1%

방법 3

CNN 을 이용해 이미지를 분류해보기

1차 문제점 해결 - 정확도를 좀 더 향상 시키기

	Top-1	Top-5
Inception V3	78.8%	94.4%
Inception Resnet V2	80.1%	95.1%

방법 3

CNN 을 이용해 이미지를 분류해보기

1차 학습 문제점 해결 요약

방법 3

CNN 을 이용해 이미지를 분류해보기

1차 학습 문제점 해결 요약

학습 데이터의 균등화

방법 3

CNN 을 이용해 이미지를 분류해보기

1차 학습 문제점 해결 요약

학습 데이터의 균등화

학습 코드 작성을 최소화

방법 3

CNN 을 이용해 이미지를 분류해보기

1차 학습 문제점 해결 요약

학습 데이터의 균등화

학습 코드 작성을 최소화

Fine tune

방법 3

CNN 을 이용해 이미지를 분류해보기

1차 학습 문제점 해결 요약

학습 데이터의 균등화

학습 코드 작성을 최소화

Fine tune

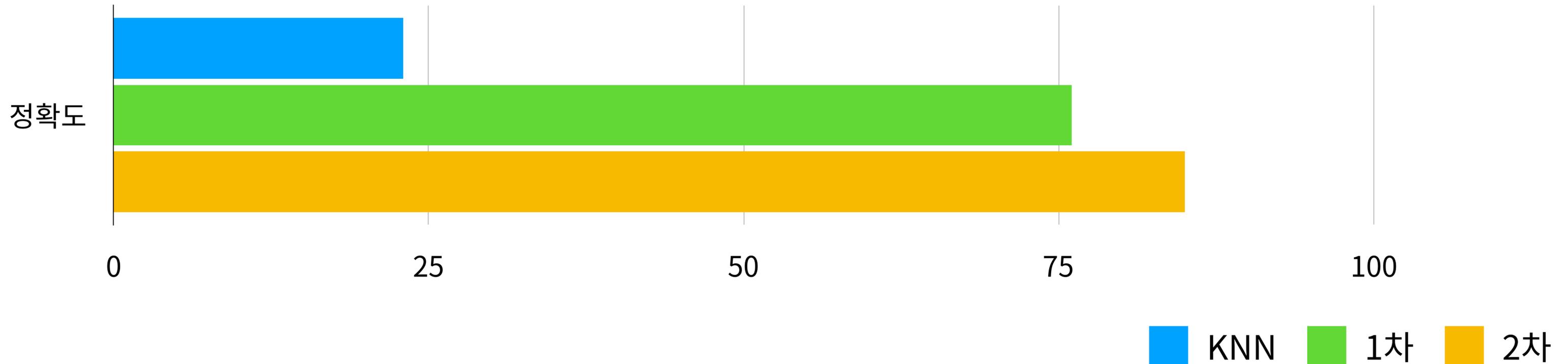
좋은 성능을 가진 Network로 변경

방법 3

CNN 을 이용해 이미지를 분류해보기

2차 학습 결과 (Inception Resnet V2)

76% $\xrightarrow{\text{정확도}}$ 85%



방법 3

CNN 을 이용해 이미지를 분류해보기

2차 학습 결과 (Inception Resnet V2)

13Days

학습 시간 →

25Days

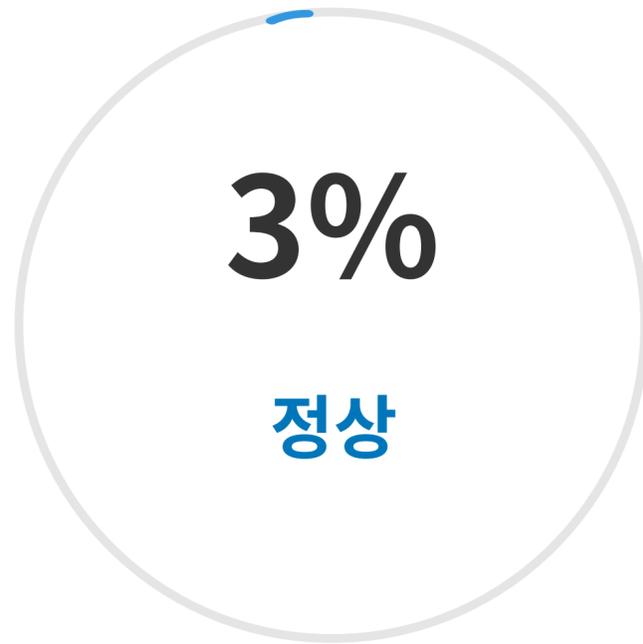
방법 3

CNN 을 이용해 이미지를 분류해보기

2차 학습 문제점 해결 - 특정 부위 (눈썹) 사진이 성인 이미지로 분류됨

눈만 클로즈업된
이미지

분류결과



VS



방법 3

CNN 을 이용해 이미지를 분류해보기

2차 학습 문제점 해결 - 특정 부위 (눈썹) 사진이 성인 이미지로 분류됨

기본 이미지 전처리

방법 3

CNN 을 이용해 이미지를 분류해보기

2차 학습 문제점 해결 - 특정 부위 (눈썹) 사진이 성인 이미지로 분류됨

이미지 전처리



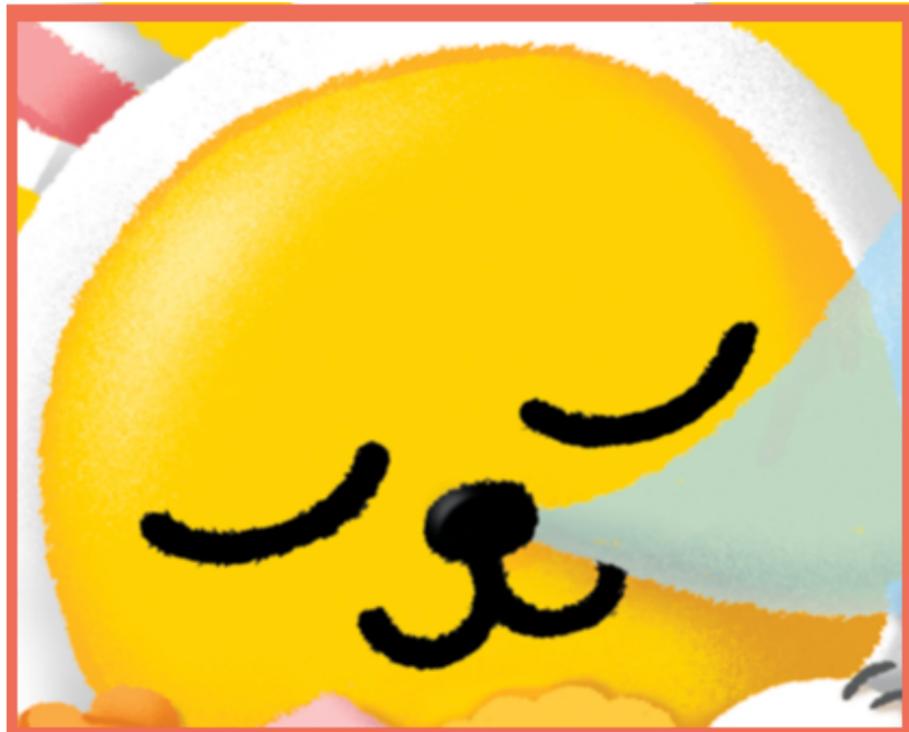
특정 위치에 BOX

방법 3

CNN 을 이용해 이미지를 분류해보기

2차 학습 문제점 해결 - 특정 부위 (눈썹) 사진이 성인 이미지로 분류됨

이미지 전처리



BOX CROP

방법 3

CNN 을 이용해 이미지를 분류해보기

2차 학습 문제점 해결 - 특정 부위 (눈썹) 사진이 성인 이미지로 분류됨

이미지 전처리



밝기 변경

방법 3

CNN 을 이용해 이미지를 분류해보기

2차 학습 문제점 해결 - 특정 부위 (눈썹) 사진이 성인 이미지로 분류됨

이미지 전처리



채도 변경

방법 3

CNN 을 이용해 이미지를 분류해보기

2차 학습 문제점 해결 - 특정 부위 (눈썹) 사진이 성인 이미지로 분류됨

원본



전처리



방법 3

CNN 을 이용해 이미지를 분류해보기

2차 학습 문제점 해결 - 특정 부위 (눈썹) 사진이 성인 이미지로 분류됨

목적에 맞는 전처리 요소 선택

방법 3

CNN 을 이용해 이미지를 분류해보기

2차 학습 문제점 해결 - 특정 부위 (눈썹) 사진이 성인 이미지로 분류됨

이미지 전처리



밝기 변경

방법 3

CNN 을 이용해 이미지를 분류해보기

2차 학습 문제점 해결 - 특정 부위 (눈썹) 사진이 성인 이미지로 분류됨

이미지 전처리



채도 변경

방법 3

CNN 을 이용해 이미지를 분류해보기

2차 학습 문제점 해결 - 특정 부위 (눈썹) 사진이 성인 이미지로 분류됨

원본



전처리



방법 3

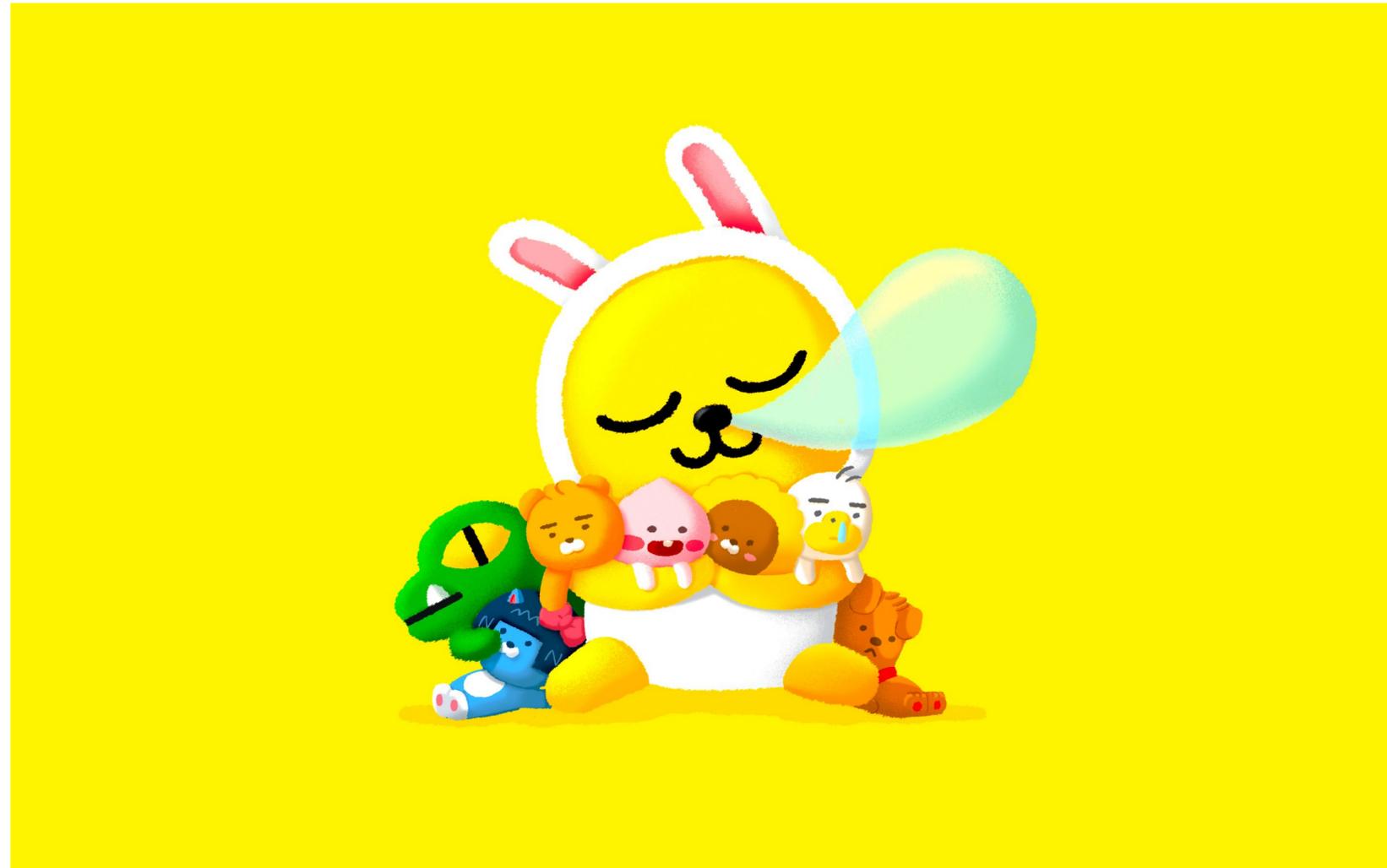
CNN 을 이용해 이미지를 분류해보기

2차 학습 문제점 해결 - 특정 부위 (눈썹) 사진이 성인 이미지로 분류됨

2차 학습 당시 전처리된 이미지



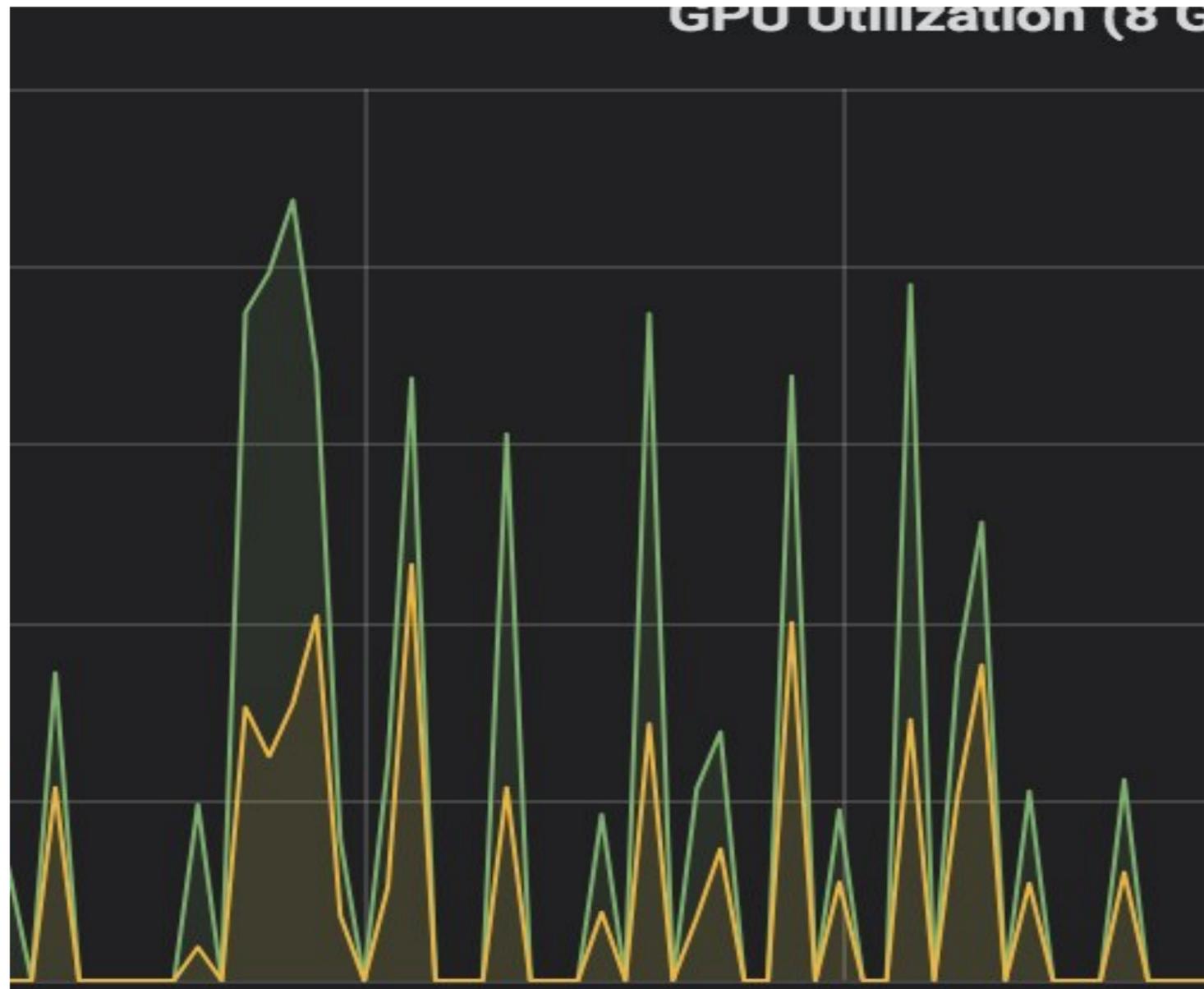
3차 학습 전처리된 이미지



방법 3

CNN 을 이용해 이미지를 분류해보기

2차 학습 문제점 해결 - 학습 속도



GPU 자원을 다 쓰지 못함

방법 3

CNN 을 이용해 이미지를 분류해보기

2차 학습 문제점 해결 - 학습 속도

적절한 Batch Size

방법 3

CNN 을 이용해 이미지를 분류해보기

2차 학습 문제점 해결 - 학습 속도

Tensorflow slim parameter 이해하기

```
tf.app.flags.DEFINE_integer('num_clones', 1,
                             'Number of model clones to deploy. Note For '
                             'historical reasons loss from all clones averaged '
                             'out and learning rate decay happen per clone '
                             'epochs')

tf.app.flags.DEFINE_integer(
    'num_preprocessing_threads', 4,
    'The number of threads used to create the batches.')
```

방법 3

CNN 을 이용해 이미지를 분류해보기

2차 학습 문제점 해결 - 정확도 향상을 위한 파라미터 튜닝

Hyperparameter Tuning

Hyperparameter

learning_rate

optimizer

```
./scripts/finetune_inception_resnet_v2_on_flowers.sh

# Download the pre-trained checkpoint.
if [ ! -d "$PRETRAINED_CHECKPOINT_DIR" ]; then
  mkdir ${PRETRAINED_CHECKPOINT_DIR}
fi
if [ ! -f ${PRETRAINED_CHECKPOINT_DIR}/${MODEL_NAME}.ckpt ]; then
  wget http://download.tensorflow.org/models/inception_resnet_v2_2016_08_30.tar.gz
  tar -xvf inception_resnet_v2_2016_08_30.tar.gz
  mv inception_resnet_v2.ckpt ${PRETRAINED_CHECKPOINT_DIR}/${MODEL_NAME}.ckpt
  rm inception_resnet_v2_2016_08_30.tar.gz
fi

# Download the dataset
python download_and_convert_data.py \
  --dataset_name=flowers \
  --dataset_dir=${DATASET_DIR}

# Fine-tune only the new layers for 1000 steps.
python train_image_classifier.py \
  --train_dir=${TRAIN_DIR} \
  --dataset_name=flowers \
  --dataset_split_name=train \
  --dataset_dir=${DATASET_DIR} \
  --model_name=${MODEL_NAME} \
  --checkpoint_path=${PRETRAINED_CHECKPOINT_DIR}/${MODEL_NAME}.ckpt \
  --checkpoint_exclude_scopes=InceptionResnetV2/Logits,InceptionResnetV2/AuxLogits \
  --trainable_scopes=InceptionResnetV2/Logits,InceptionResnetV2/AuxLogits \
  --max_number_of_steps=1000 \
  --batch_size=32 \
  --learning_rate=0.01 \
  --learning_rate_decay_type=fixed \
  --save_interval_secs=60 \
  --save_summaries_secs=60 \
  --log_every_n_steps=10 \
  --optimizer=rmsprop \
  --weight_decay=0.00004

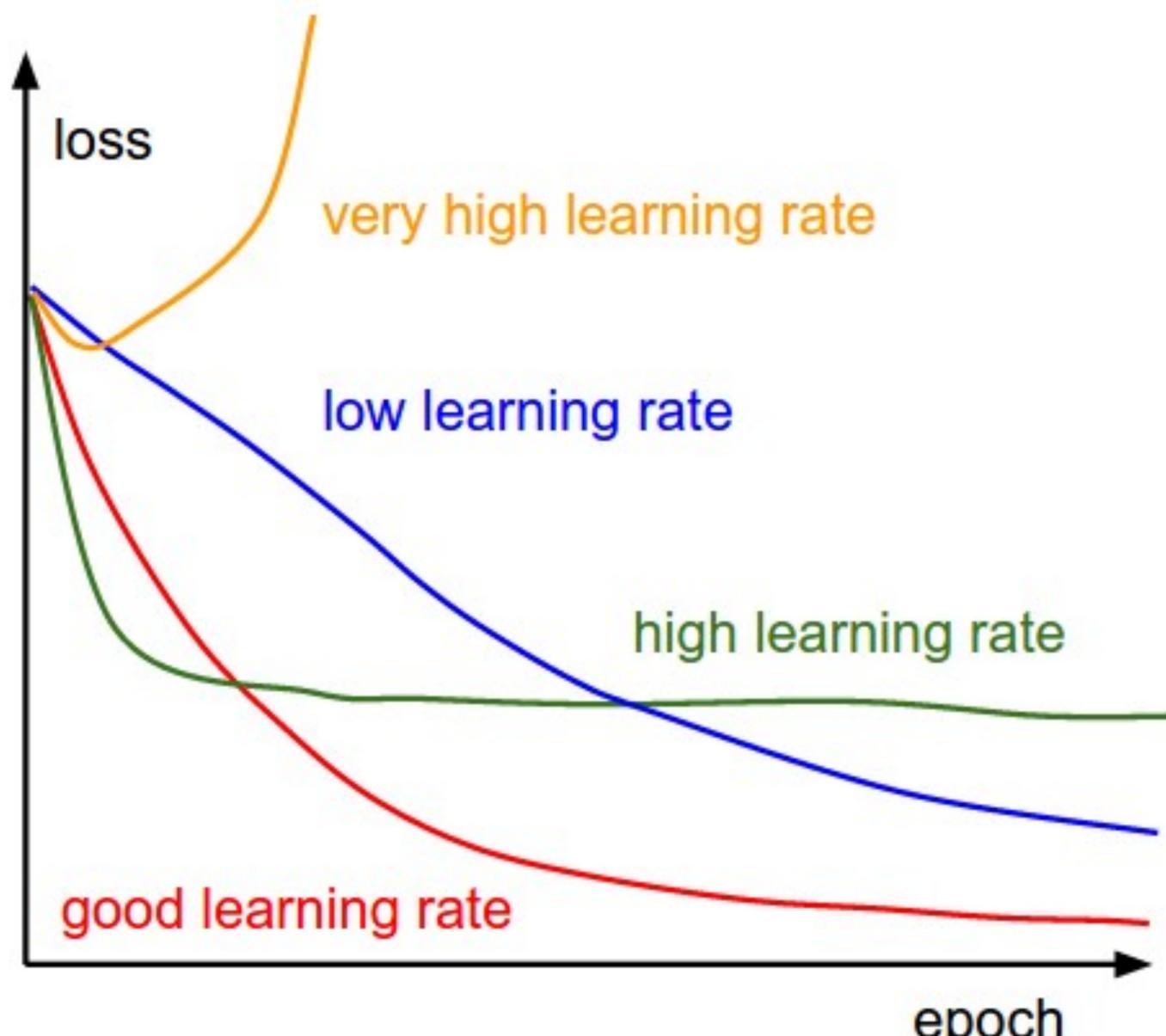
# Run evaluation.
python eval_image_classifier.py \
  --checkpoint_path=${TRAIN_DIR} \
  --eval_dir=${TRAIN_DIR} \
  --dataset_name=flowers \
  --dataset_split_name=validation \
  --dataset_dir=${DATASET_DIR} \
  --model_name=${MODEL_NAME}

# .....
```

방법 3

CNN 을 이용해 이미지를 분류해보기

2차 학습 문제점 해결 - 정확도 향상을 위한 파라미터 튜닝

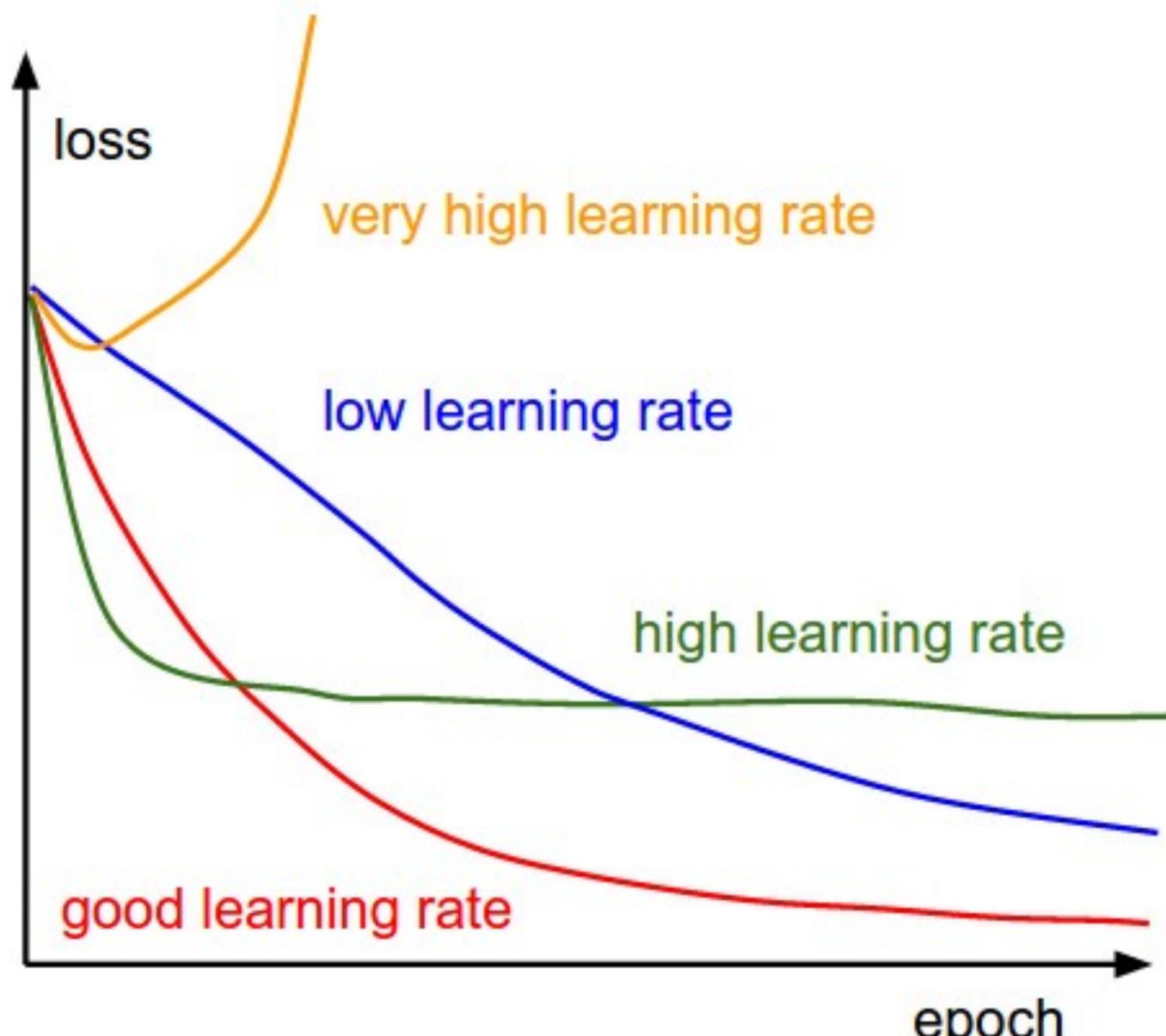


learning_rate

방법 3

CNN 을 이용해 이미지를 분류해보기

2차 학습 문제점 해결 - 정확도 향상을 위한 파라미터 튜닝

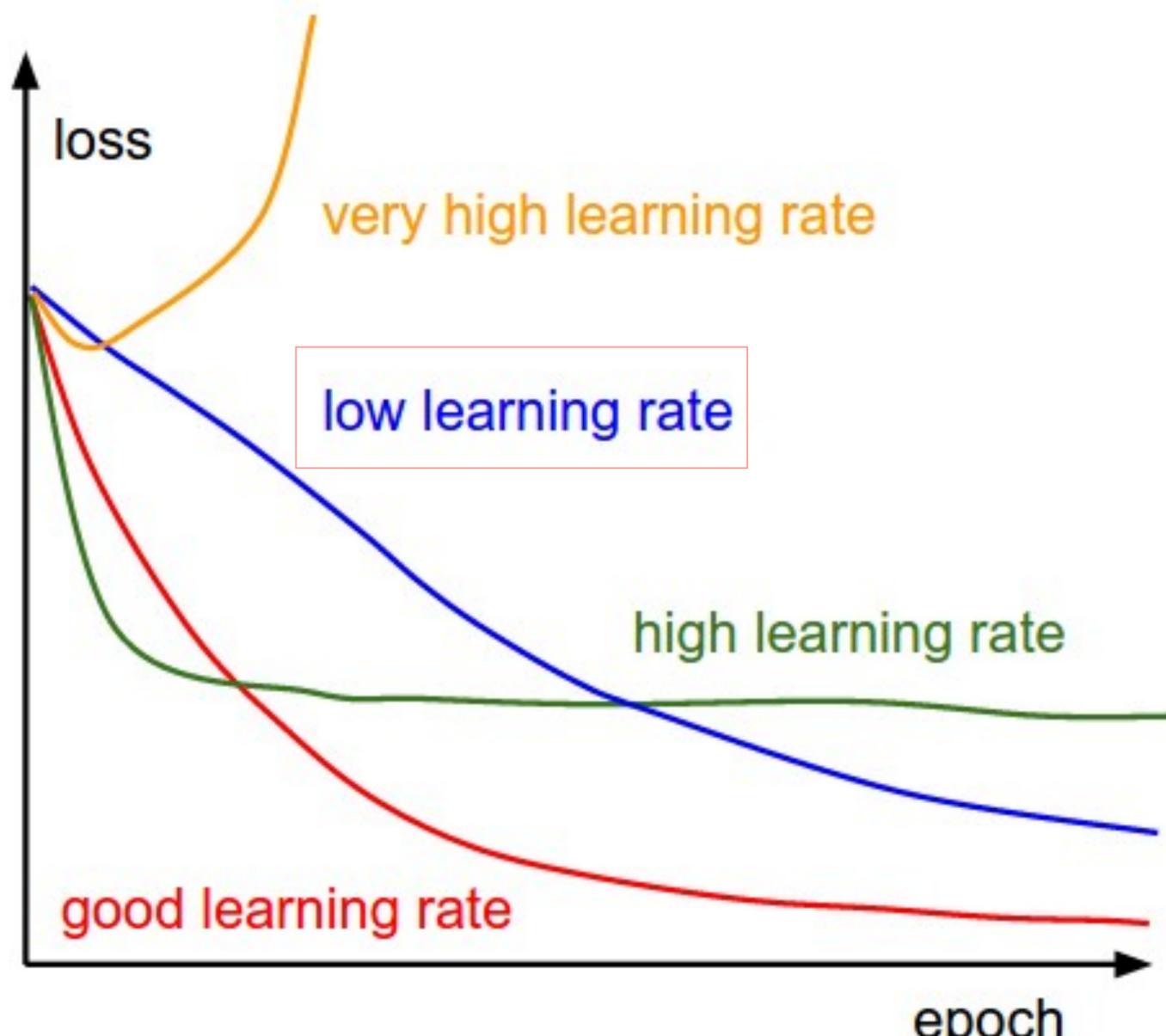


learning_rate (보폭)

방법 3

CNN 을 이용해 이미지를 분류해보기

2차 학습 문제점 해결 - 정확도 향상을 위한 파라미터 튜닝



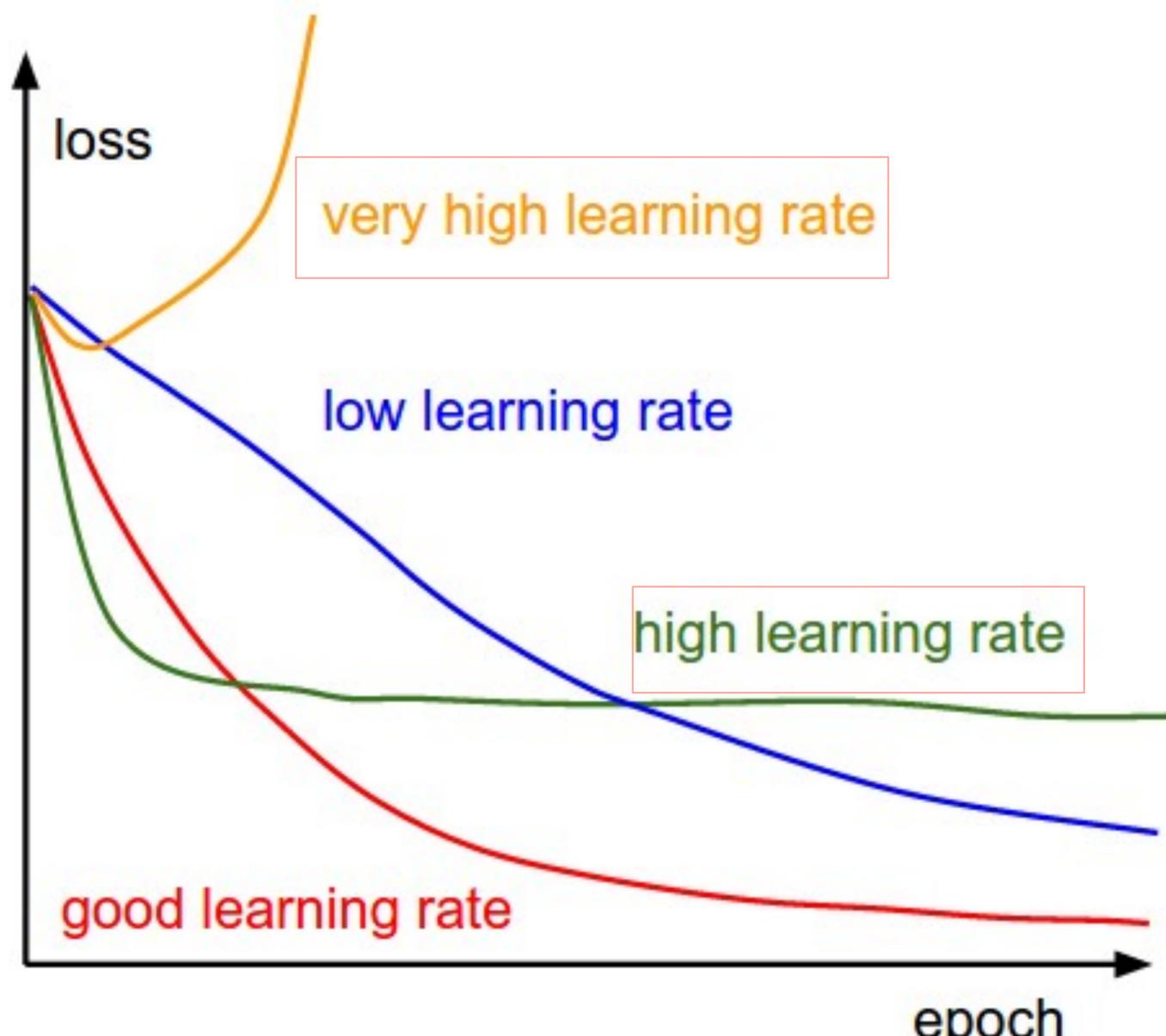
learning_rate (보폭)

너무 낮으면 목표에 도달하기 전에 학습 종료

방법 3

CNN 을 이용해 이미지를 분류해보기

2차 학습 문제점 해결 - 정확도 향상을 위한 파라미터 튜닝



learning_rate (보폭)

너무 낮으면 목표에 도달하기 전에 학습 종료

너무 높으면 목표를 지나쳐서 학습 종료

방법 3

CNN 을 이용해 이미지를 분류해보기

2차 학습 문제점 해결 - 정확도 향상을 위한 파라미터 튜닝

Optimizer

파라미터를 업데이트 최적화 방법

방법 3

CNN 을 이용해 이미지를 분류해보기

2차 학습 문제점 해결 - 정확도 향상을 위한 파라미터 튜닝

Optimizer

파라미터를 업데이트 최적화 방법

방향

Momentum, NAG

방법 3

CNN 을 이용해 이미지를 분류해보기

2차 학습 문제점 해결 - 정확도 향상을 위한 파라미터 튜닝

Optimizer

파라미터를 업데이트 최적화 방법

방향

Momentum, NAG

보폭

Adagrad, AdaDelta, RMSProp

방법 3

CNN 을 이용해 이미지를 분류해보기

2차 학습 문제점 해결 - 정확도 향상을 위한 파라미터 튜닝

Optimizer

파라미터를 업데이트 최적화 방법

방향

보폭

방향+보폭 = Adam

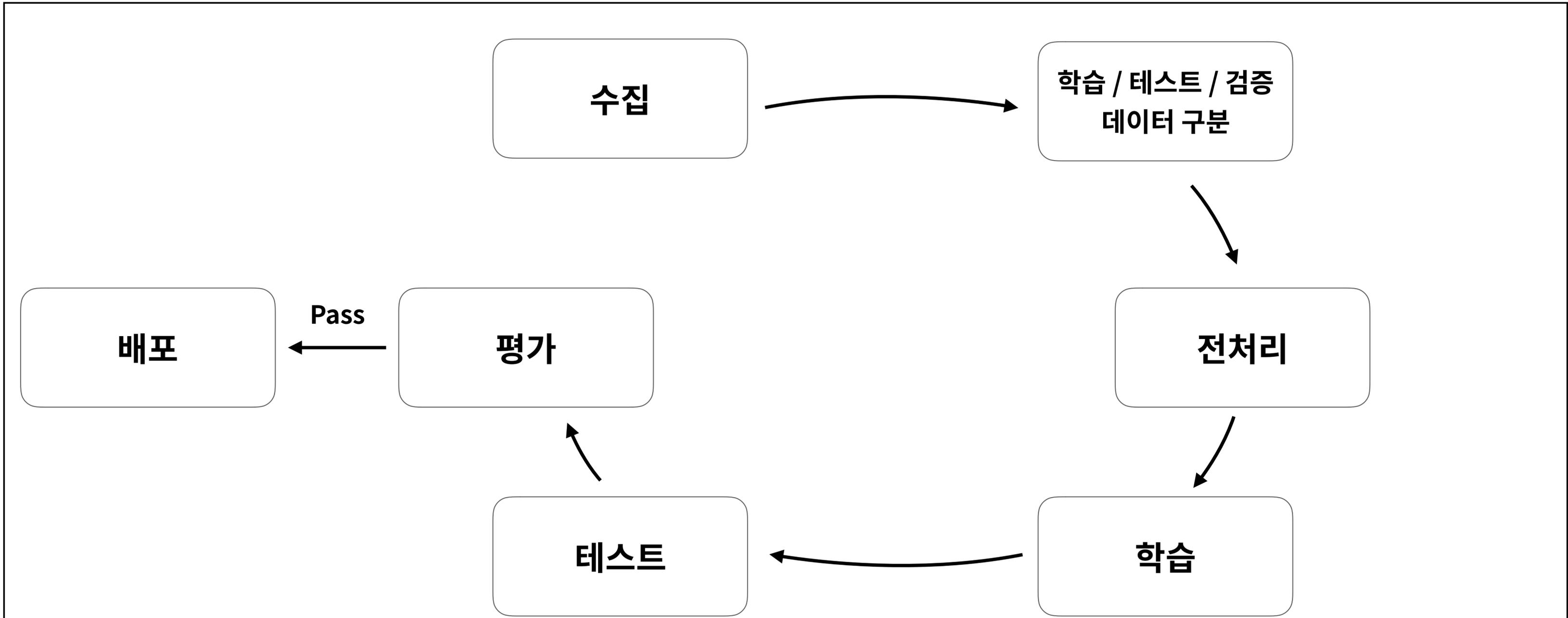
Momentum, NAG

Adagrad, AdaDelta, RMSProp

방법 3

CNN 을 이용해 이미지를 분류해보기

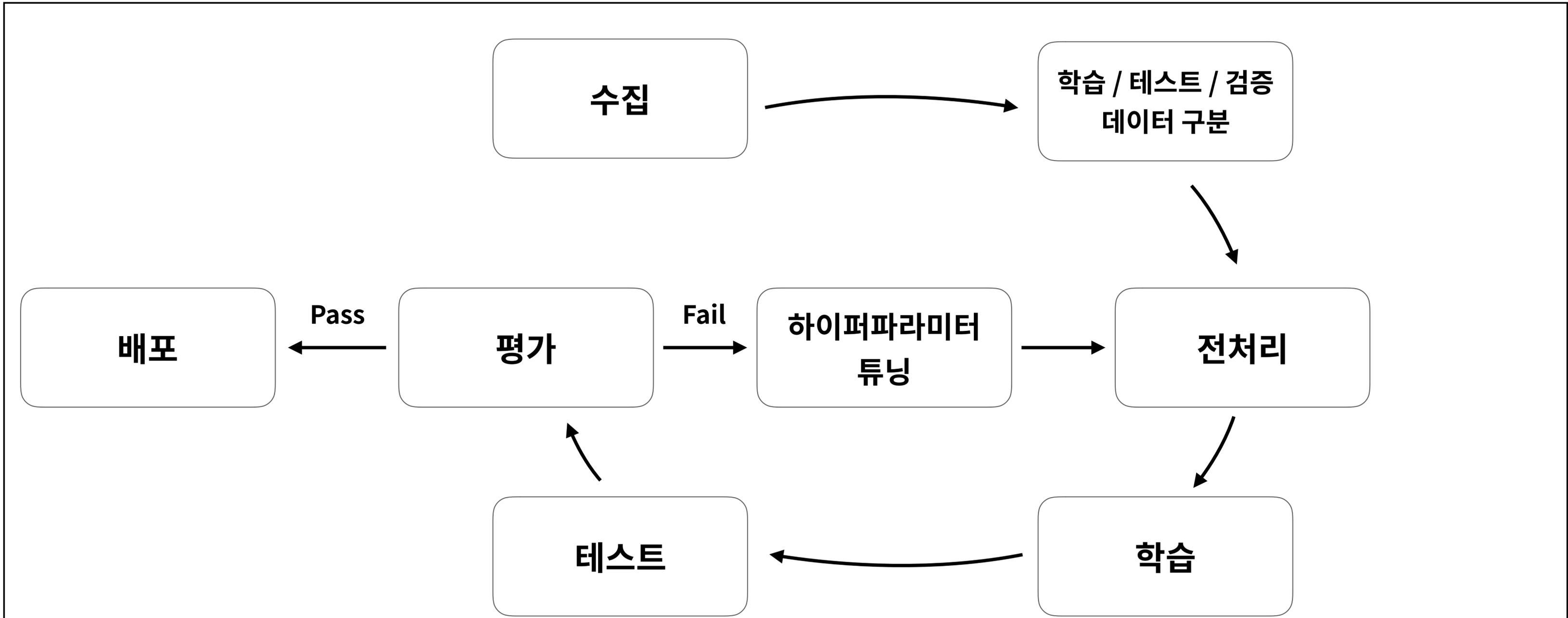
2차 학습 문제점 해결 - 정확도 향상을 위한 파라미터 튜닝



방법 3

CNN 을 이용해 이미지를 분류해보기

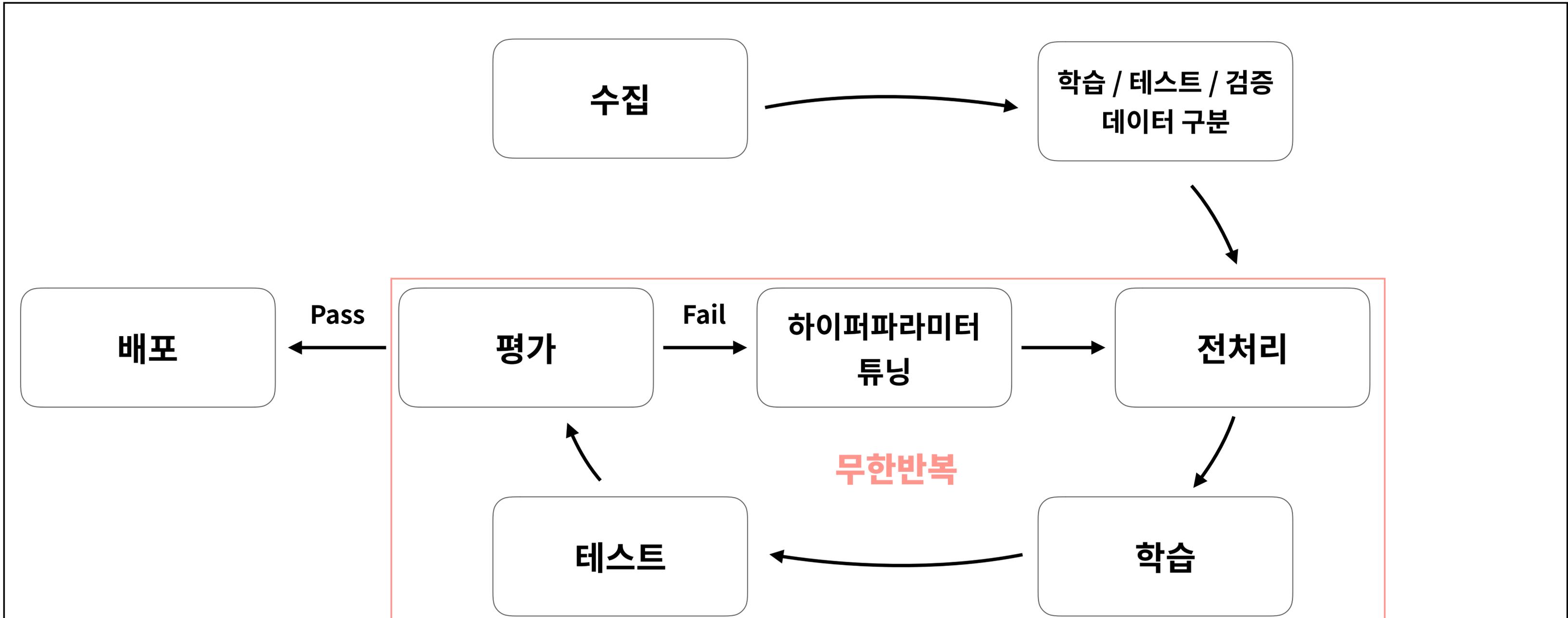
2차 학습 문제점 해결 - 정확도 향상을 위한 파라미터 튜닝



방법 3

CNN 을 이용해 이미지를 분류해보기

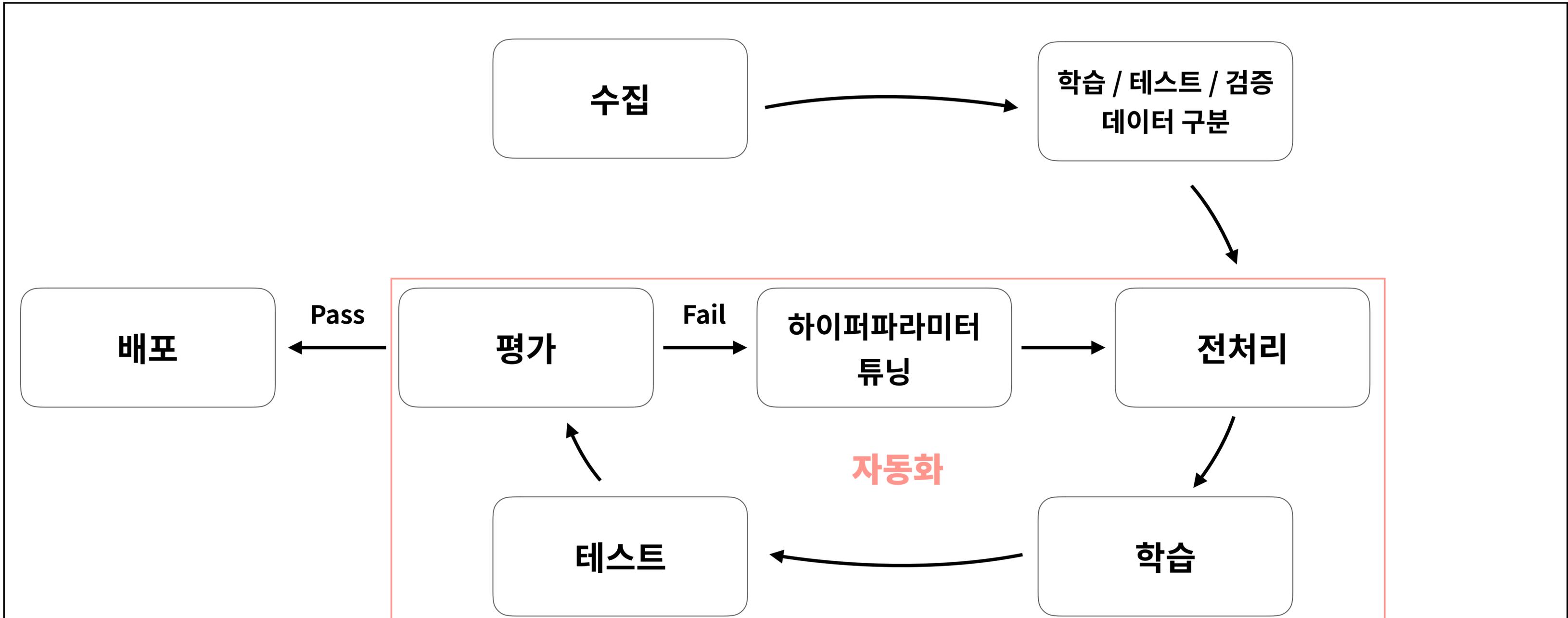
2차 학습 문제점 해결 - 정확도 향상을 위한 파라미터 튜닝



방법 3

CNN 을 이용해 이미지를 분류해보기

2차 학습 문제점 해결 - 정확도 향상을 위한 파라미터 튜닝



방법 3

CNN 을 이용해 이미지를 분류해보기

2차 학습 문제점 해결 요약

방법 3

CNN 을 이용해 이미지를 분류해보기

2차 학습 문제점 해결 요약

목적에 맞는 전처리

방법 3

CNN 을 이용해 이미지를 분류해보기

2차 학습 문제점 해결 요약

목적에 맞는 전처리

배치 사이즈 및 Parameter 변경으로 학습속도 상승

방법 3

CNN 을 이용해 이미지를 분류해보기

2차 학습 문제점 해결 요약

목적에 맞는 전처리

배치 사이즈 및 Parameter 변경으로 학습속도 상승

Hyperparameter Tuning

방법 3

CNN 을 이용해 이미지를 분류해보기

2차 학습 문제점 해결 요약

목적에 맞는 전처리

배치 사이즈 및 Parameter 변경으로 학습속도 상승

Hyperparameter Tuning

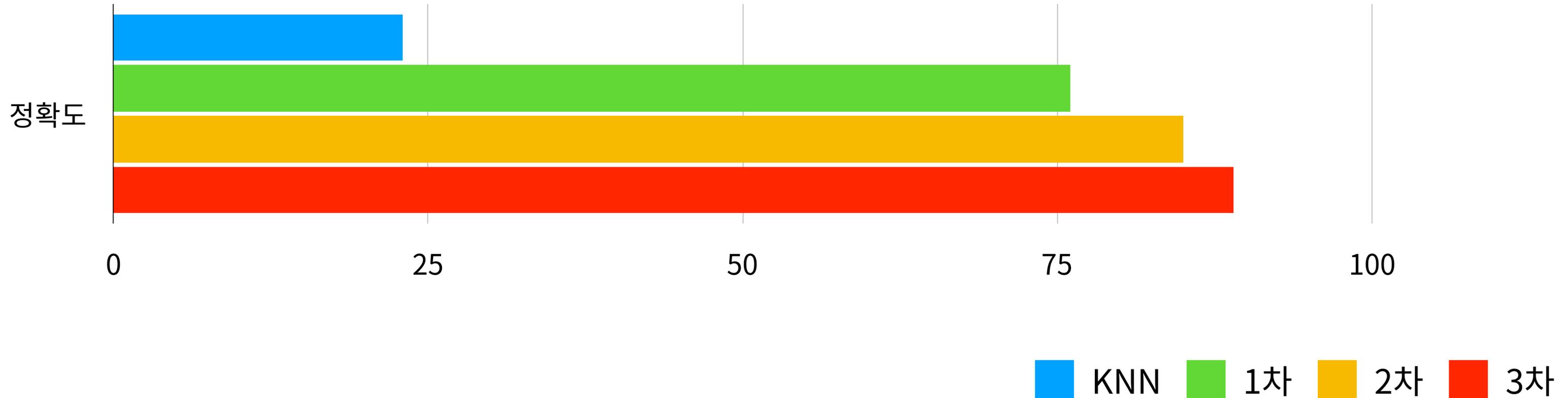
튜닝을 위한 학습 자동화

방법 3

CNN 을 이용해 이미지를 분류해보기

3차 학습 결과

85% $\xrightarrow{\text{정확도}}$ 89%



방법 3

CNN 을 이용해 이미지를 분류해보기

3차 학습 결과

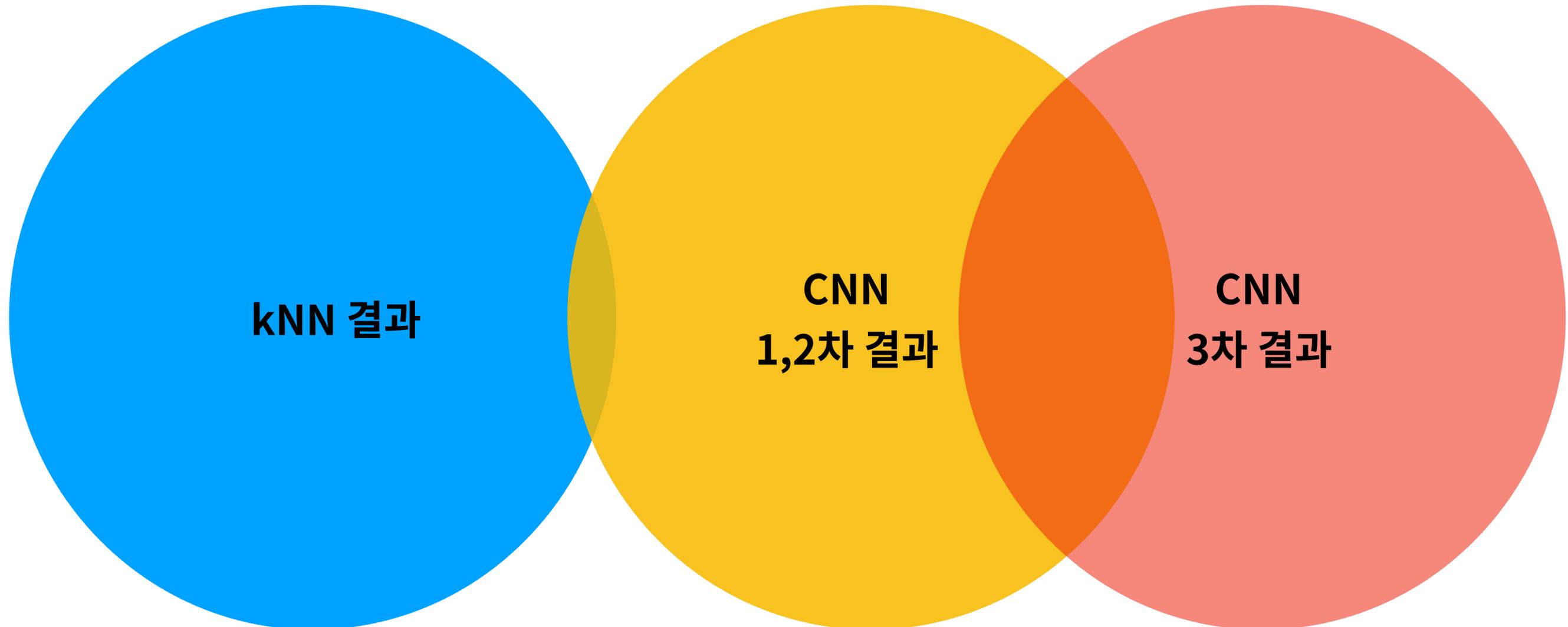


05 방법 4 Ensemble

방법 4

Ensemble

가설 - 서로 다른 모델의 결과는 정답의 종류, 범위가 다를 것이다.



방법 4

Ensemble

가설 - 서로 다른 모델의 결과는 정답의 종류, 범위가 다를 것이다.

방법 4

Ensemble

가설 - 서로 다른 모델의 결과는 정답의 종류, 범위가 다를 것이다.

kNN 정답 + 2차 CNN 정답 + 3차 CNN 정답

방법 4

Ensemble

가설 - 서로 다른 모델의 결과는 정답의 종류, 범위가 다를 것이다.

Ensemble method (learning)

방법 4

Ensemble

가설 - 서로 다른 모델의 결과는 정답의 종류, 범위가 다를 것이다.

Ensemble method (learning)

여러개의 모델을 학습시켜 결과를 이용해 하나의 모델보다 좋은 결과를 얻는 방법

방법 4

Ensemble

여러 모델의 결과를 이용해 결정하는 방법



Voting Classifier
다수결 의사결정 방식

방법 4

Ensemble

여러 모델의 결과를 이용해 결정하는 방법



Voting Classifier
다수결 의사결정 방식

=> 정확도에 큰 변함 없음

방법 4 Ensemble

여러 모델의 결과를 이용해 결정하는 방법



Voting Classifier
다수결 의사결정 방식

=> 정확도에 큰 변함 없음

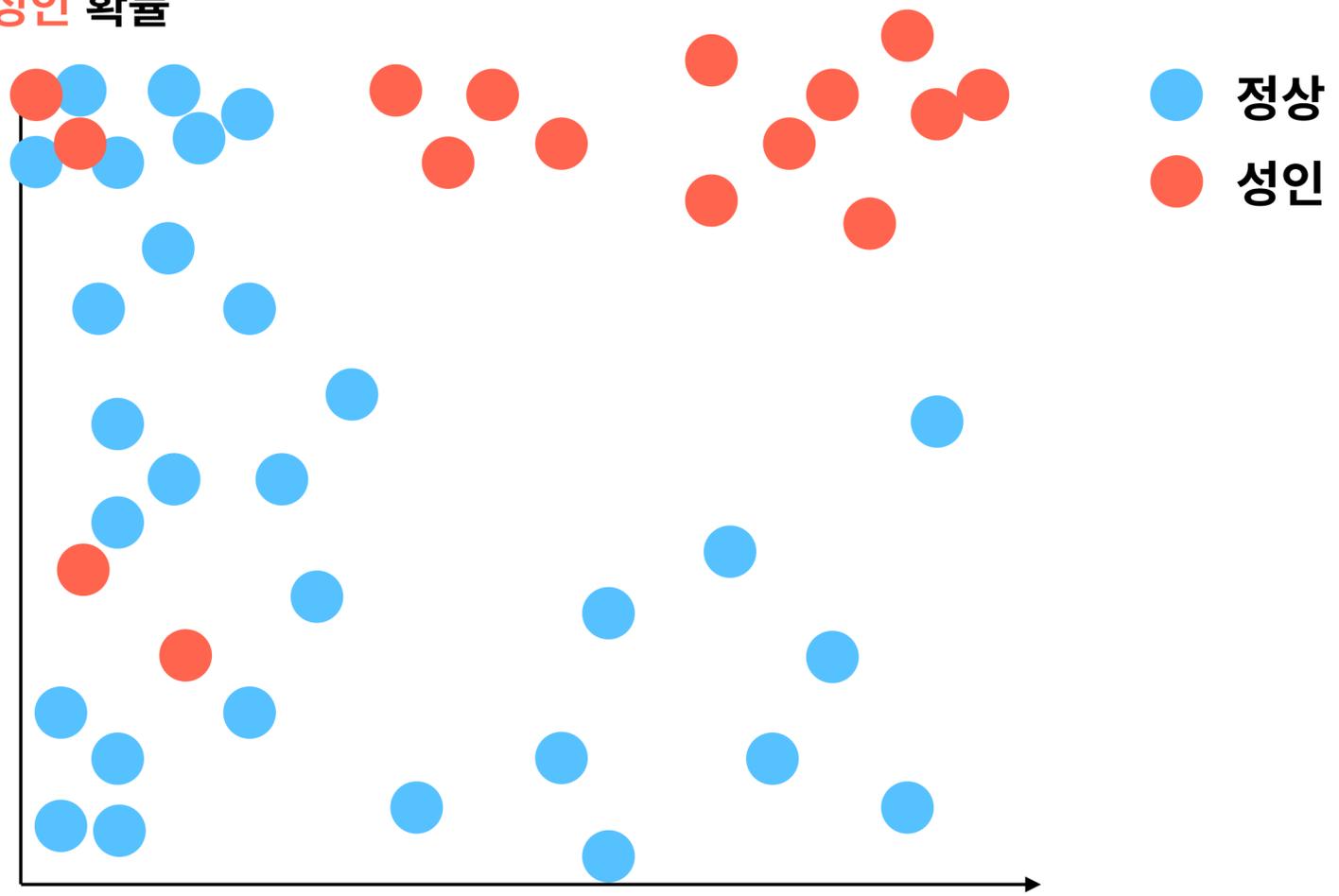
=> 오차에 서로 상관 관계를 가지기 때문

방법 4

Ensemble

여러 모델의 결과를 이용해 결정하는 방법

kNN 성인 확률

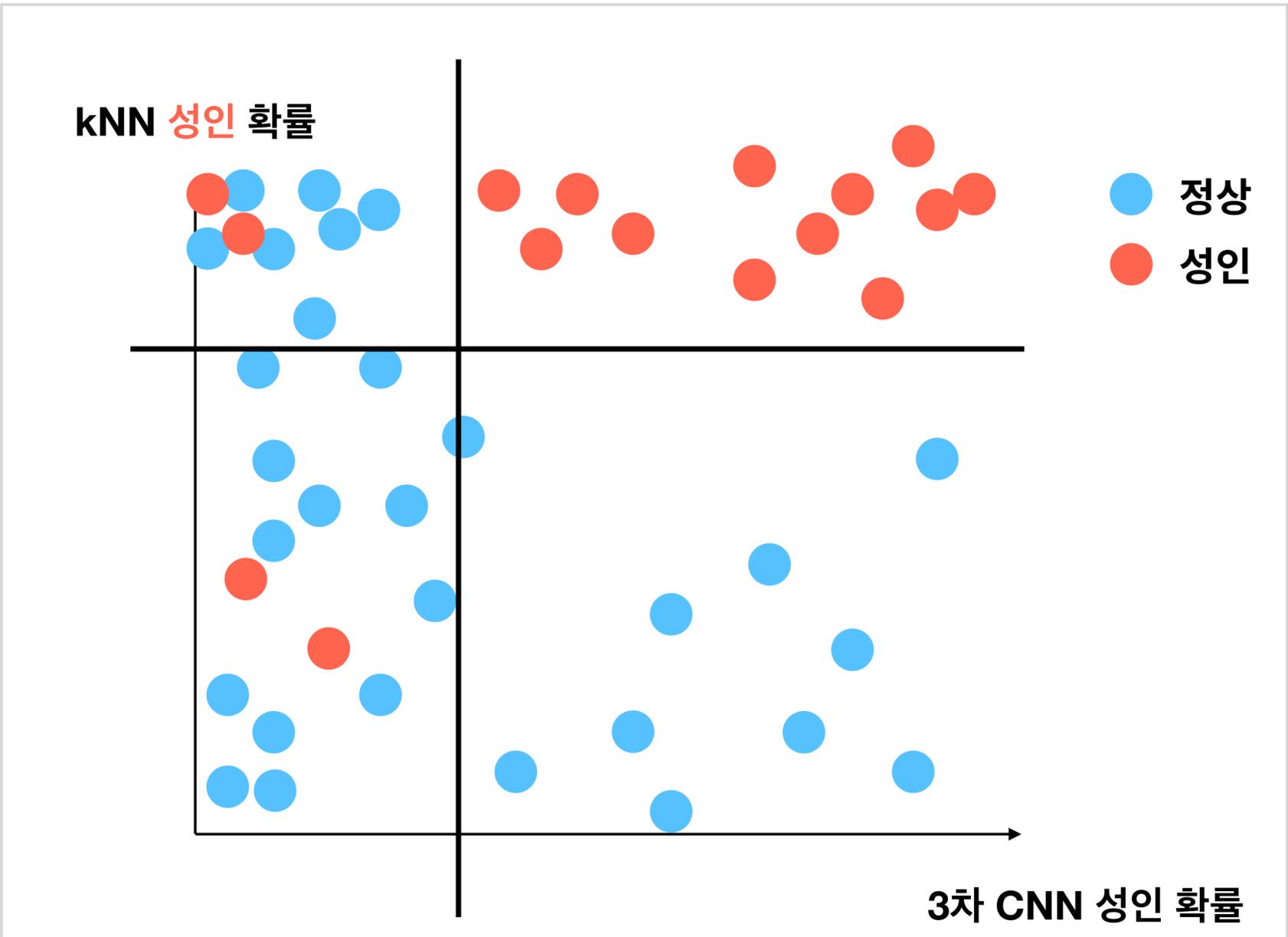


3차 CNN 성인 확률

방법 4

Ensemble

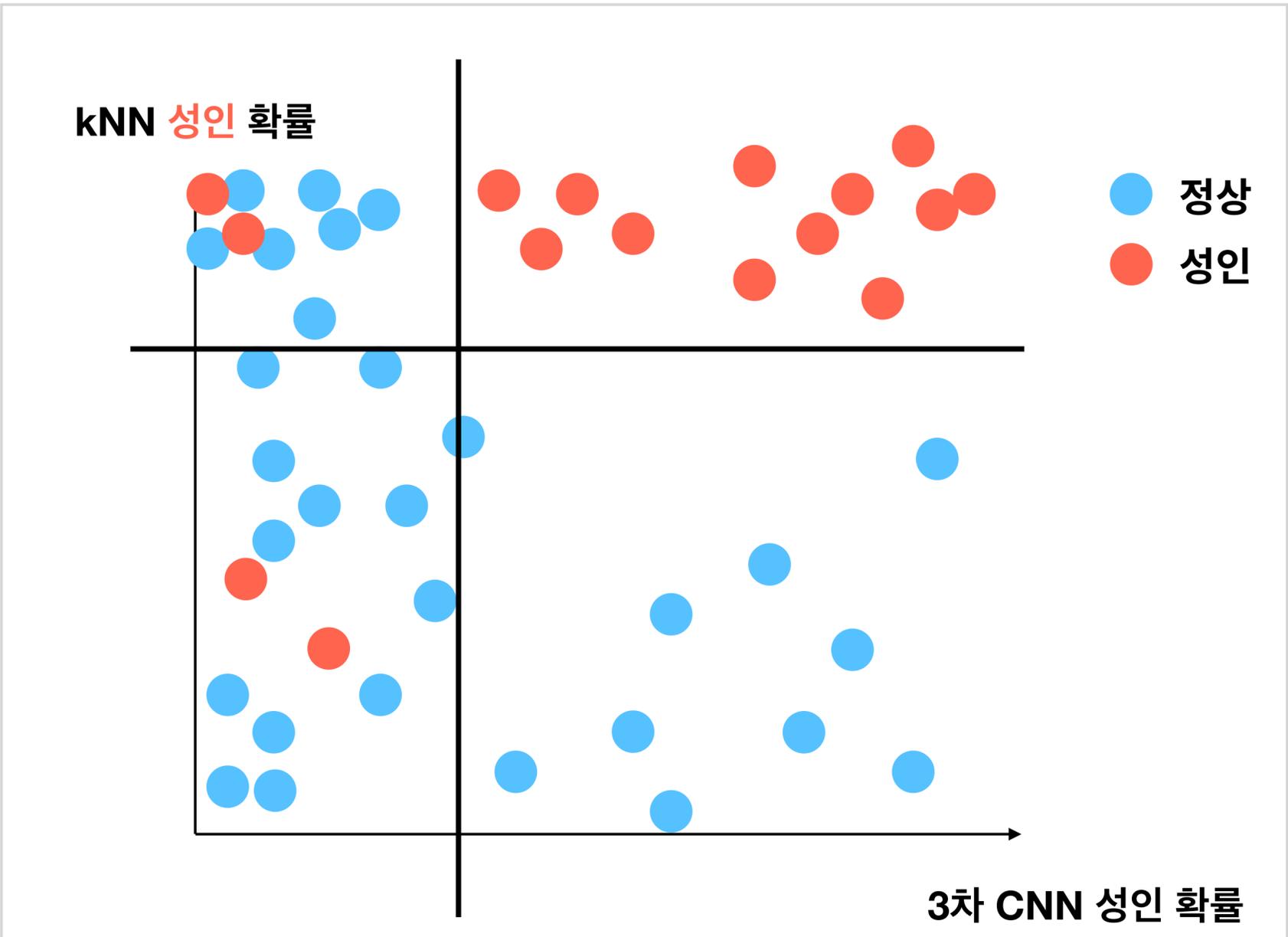
여러 모델의 결과를 이용해 결정하는 방법



방법 4

Ensemble

여러 모델의 결과를 이용해 결정하는 방법

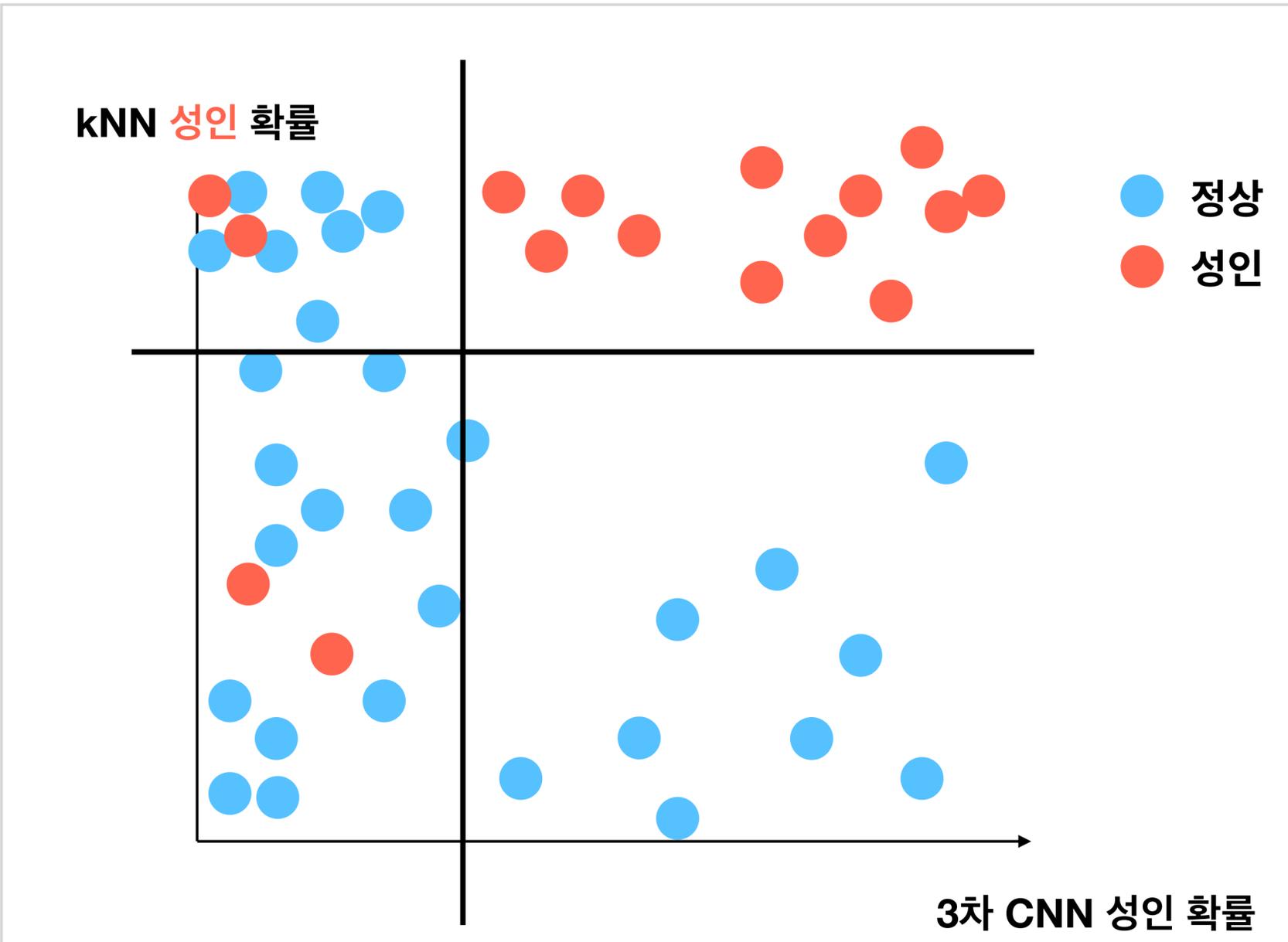


$kNN \text{ 성인} \geq N \ \&\& \ \text{CNN 성인} \geq N$

방법 4

Ensemble

여러 모델의 결과를 이용해 결정하는 방법



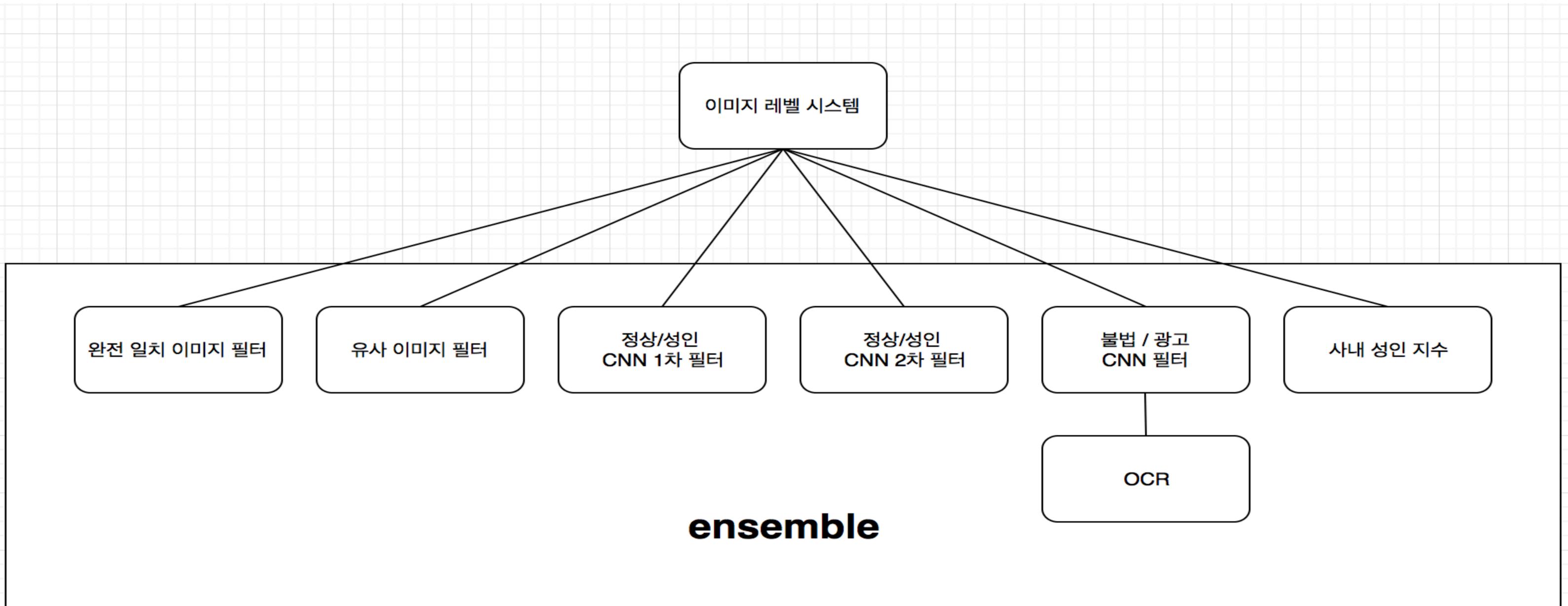
kNN 성인 $\geq N$ && CNN 성인 $\geq N$

=> 의사결정트리

방법 4

Ensemble

정확도 향상을 위한 앙상블 기법



06 고민하고 있는 주제

고민하고 있는 주제

if (kakao) dev 2019

Model Debugging

Black Box

Model Debugging

Grad CAM



어느 데이터를 보고 학습하였는지 유추가 가능함

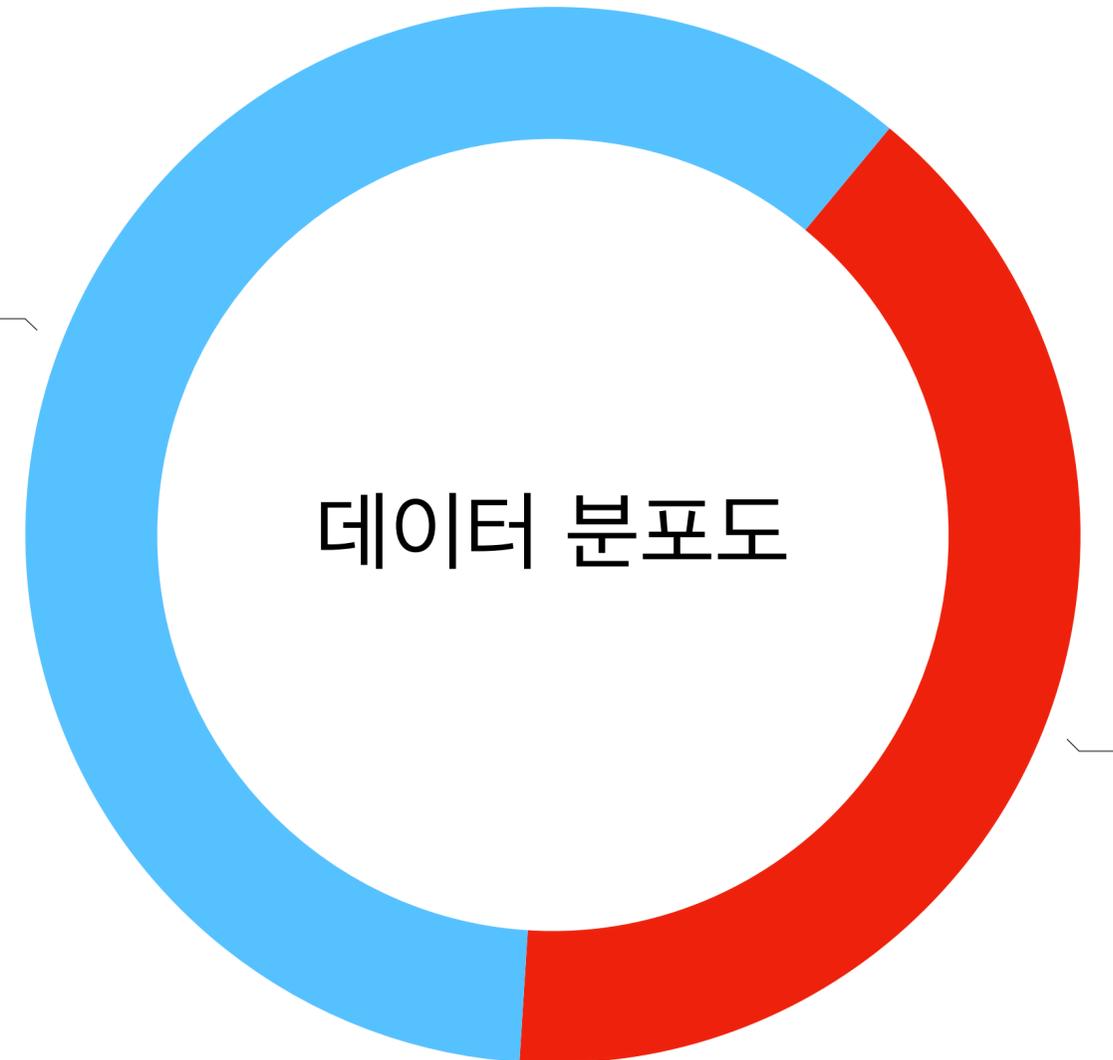
고민하고 있는 주제

데이터 불균형 문제

Random Under Sampling
+
Tomek Link
+
Condensed Nearest Neighbour



정상



데이터 분포도

성인

데이터 불균형 문제

Under Sampling

다수의 데이터에서 일부만 사용하는 방법

고민하고 있는 주제

if (kakao) dev 2019

데이터 불균형 문제

Under Sampling

중요 특징이 누락될 여지가 존재함

고민하고 있는 주제

if (kakao) dev 2019

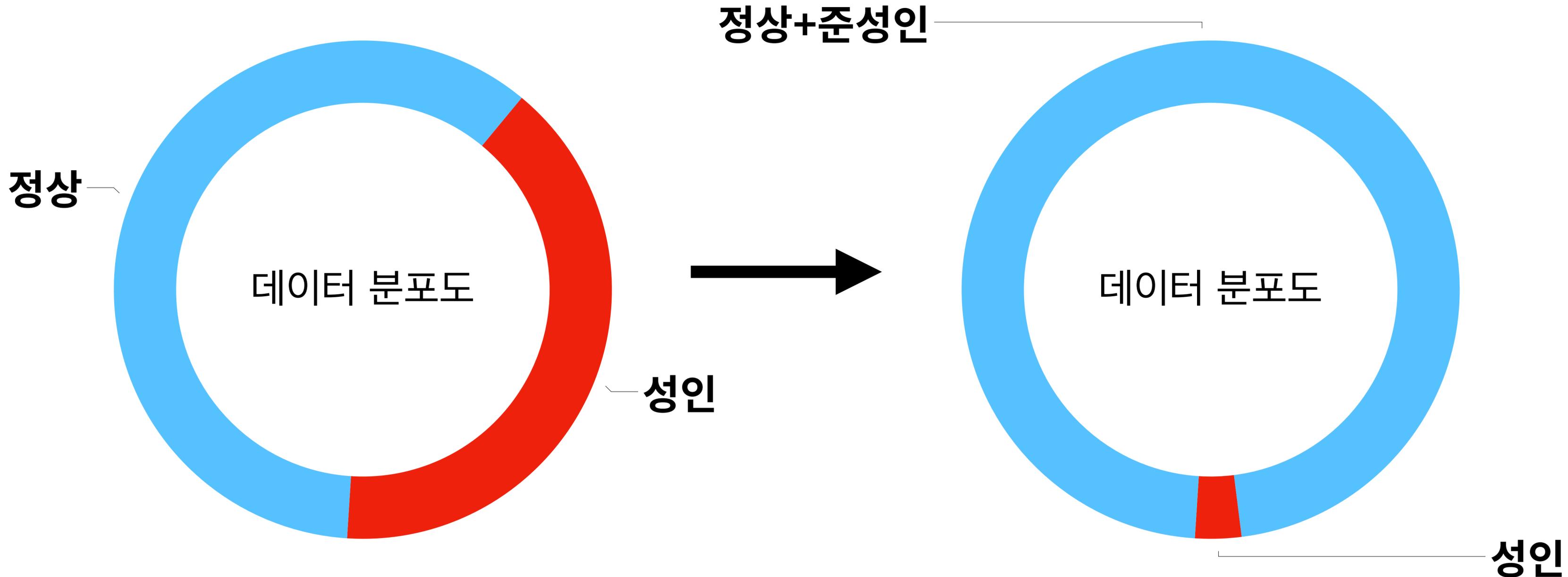
데이터 불균형 문제

그럼 Over Sampling 은?

..Overfitting..

고민하고 있는 주제

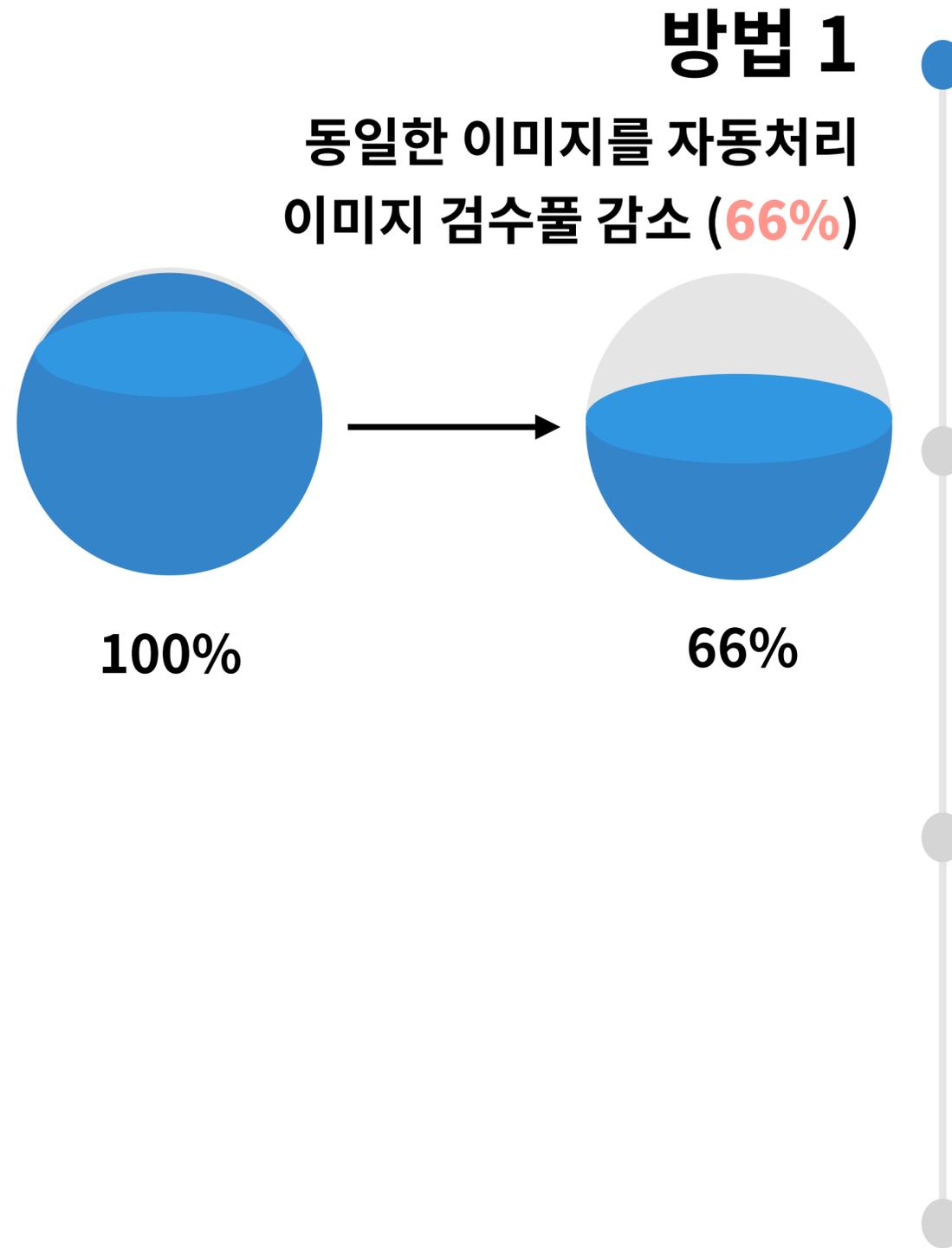
데이터 불균형 문제 - 불균형 조차 특성이 아닐까?

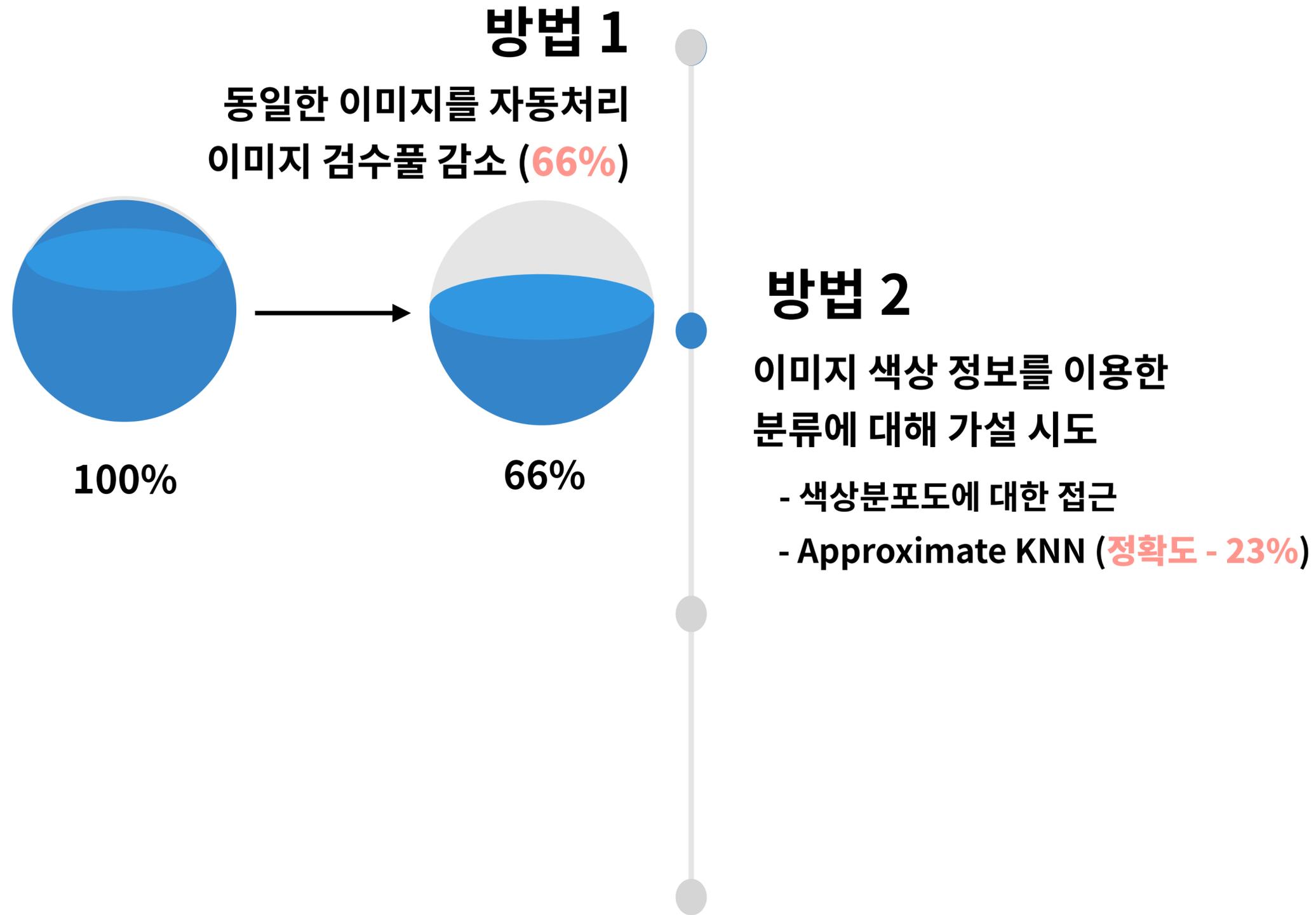


07 정리

정리

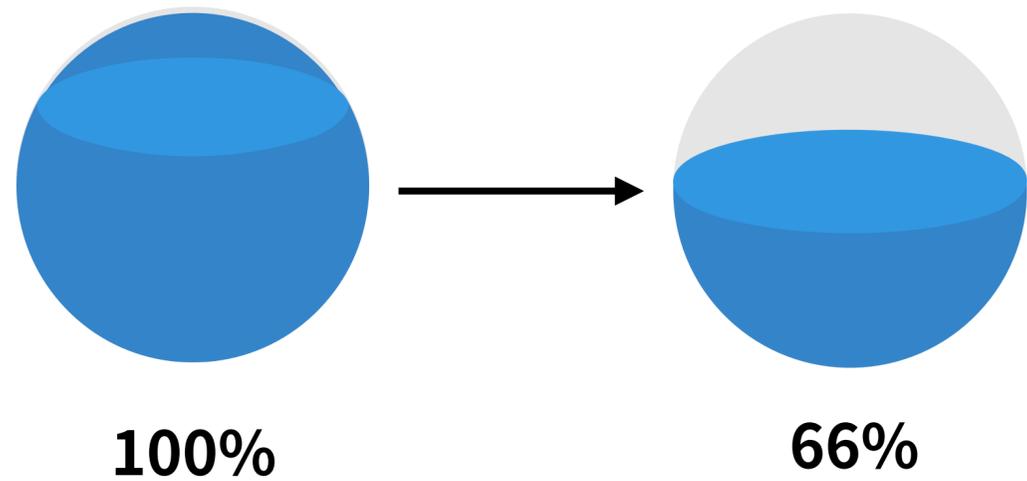






방법 1

동일한 이미지를 자동처리
이미지 검수풀 감소 (66%)



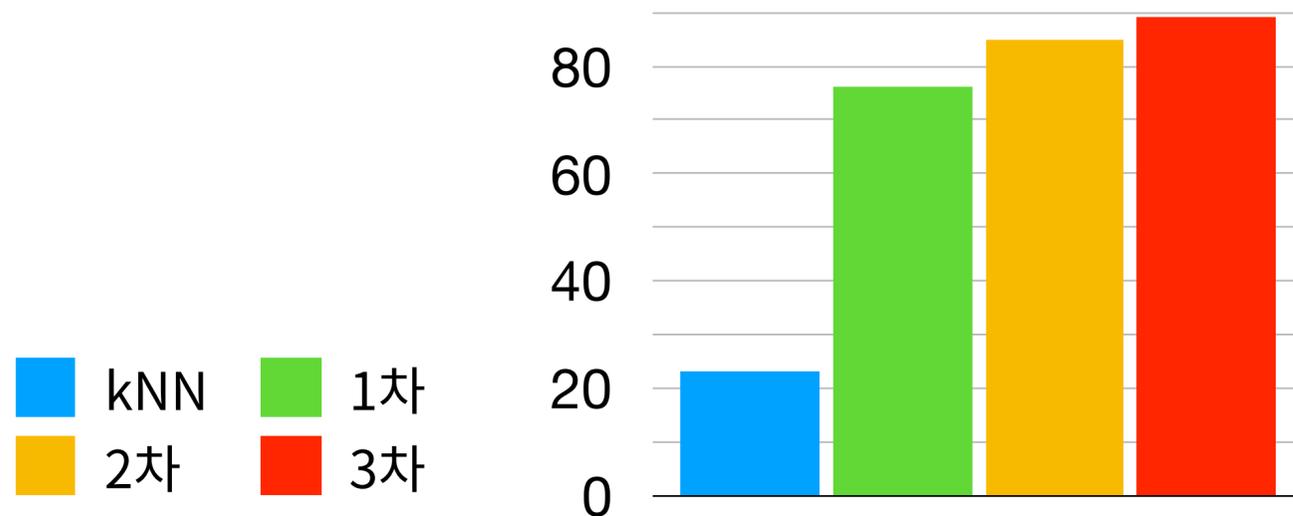
방법 2

이미지 색상 정보를 이용한
분류에 대해 가설 시도

- 색상분포도에 대한 접근
- Approximate KNN (정확도 - 23%)

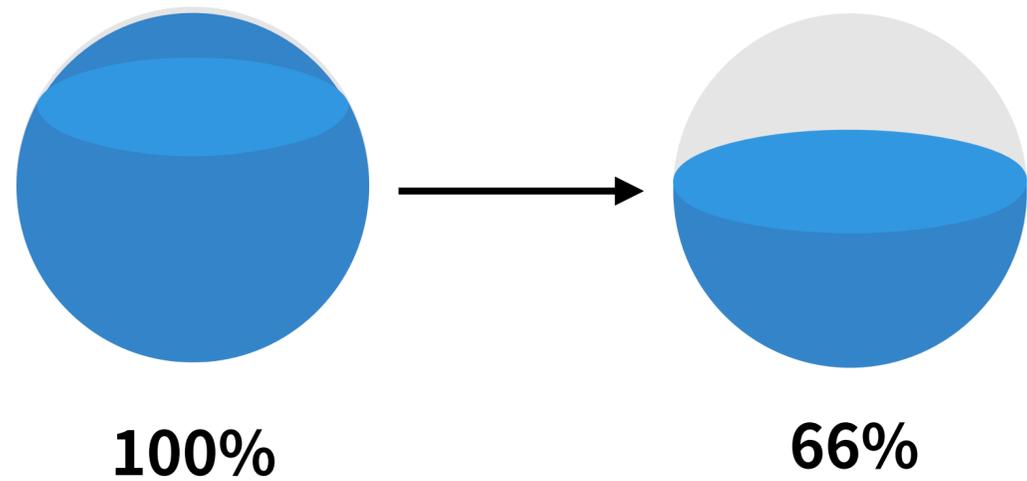
방법 3

CNN 을 이용한 이미지 학습 분류 (정확도 - 89%)



방법 1

동일한 이미지를 자동처리
이미지 검수풀 감소 (66%)



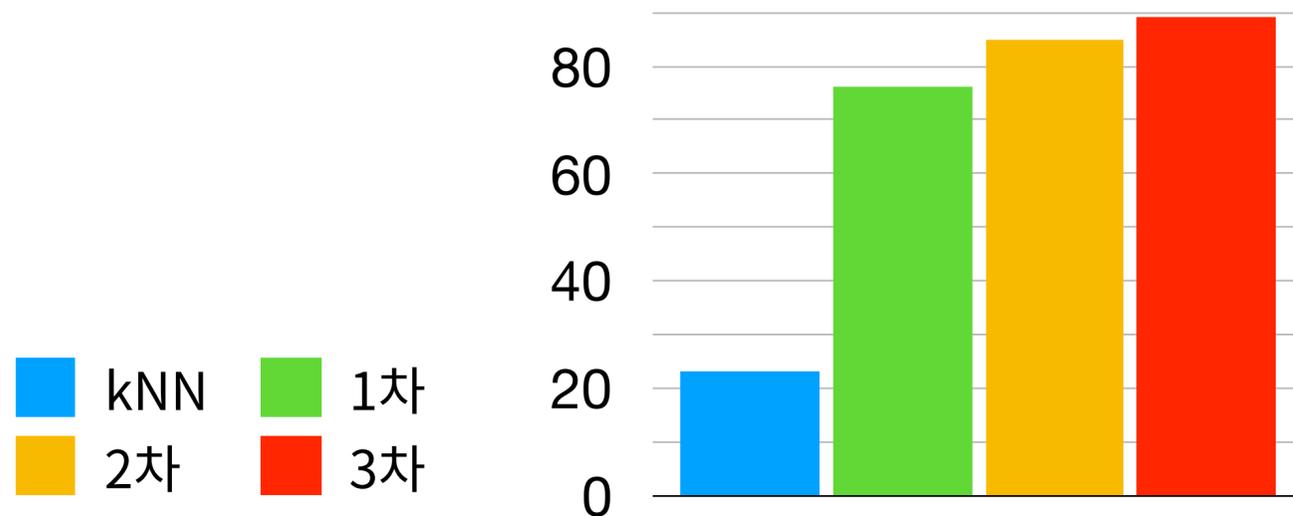
방법 2

이미지 색상 정보를 이용한
분류에 대해 가설 시도

- 색상분포도에 대한 접근
- Approximate KNN (정확도 - 23%)

방법 3

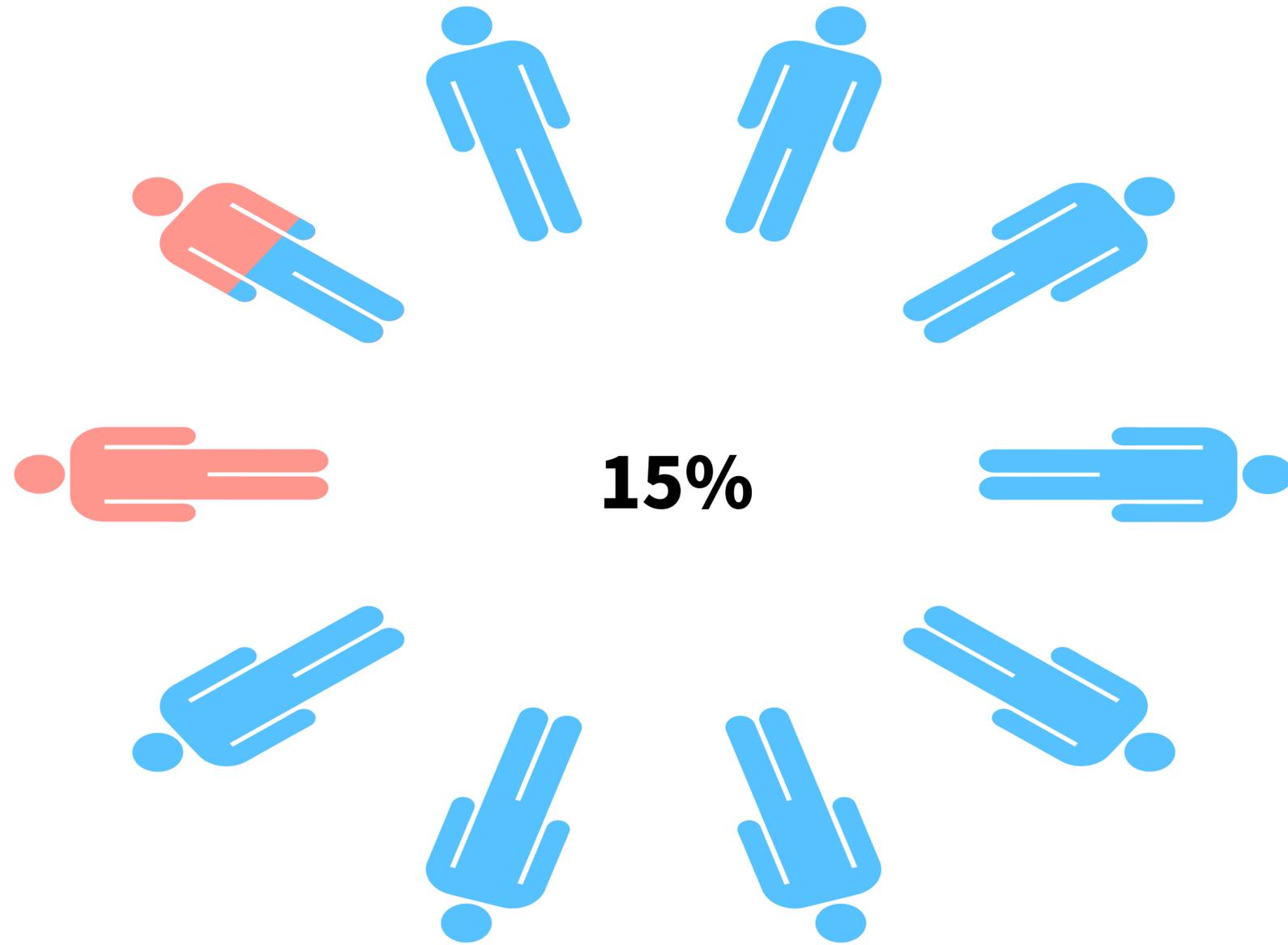
CNN 을 이용한 이미지 학습 분류 (정확도 - 89%)



방법 4

머신러닝 앙상블 기법을 이용한 추천시스템

- 여러 학습 결과를 이용해서 또다른 모델을 만듦



좀 더 많은 이야기가 하고 싶었으나..

우린 답을 찾을 것이다. 늘 그랬듯이

Q&A