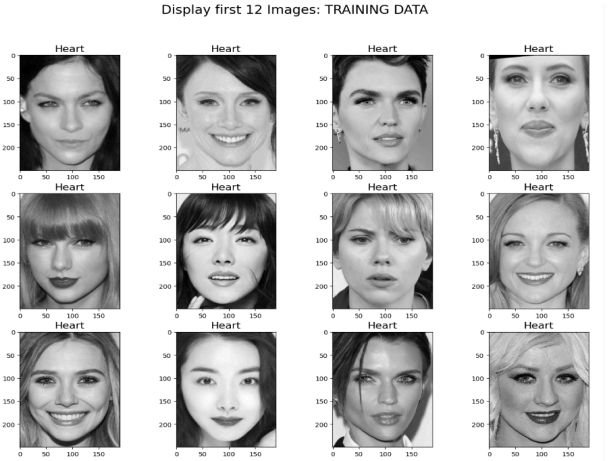
SK네트웍스 **Family AI**과정 **7**기

데이터 전처리 인공지능 데이터 전처리 결과서

☐ 개요

- 산출물 단계 : 데이터 전처리 / 인공지능 데이터 전처리 결과서
- 제출 일자 : 2025.04.04
- 깃허브 경로 : https://github.com/SKNETWORKS-FAMILY-AICAMP/SKN07-FINAL-2Team
- 작성 팀원 : 김태희

데이터 전처리 개요	• **문서유형** 얼굴형 분류 모델 개발을 위한 데이터 전처리 과정에 대한 상세한 결과를 기록하고 데이터 수집 및 전처리 과정 결과 기록 • **데이터 수집** Kaggle(출처) Face shape 데이터셋 • **데이터 수집일자** 2025년 03월 10일~2025년 03월 11일 • **데이터 양** - 전체 이미지 수: 4,979개 - 전처리 후 사용된 이미지 수: 4,973개 (이상치 6개 제거 후 학습 진행) - 카테고리별 이미지 수: 약 1,000개 (하트, 직사각형, 타원형, 둥근형, 정사각형) - 훈련 세트 이미지 수: 카테고리별 약 800개 - 테스트 세트 이미지 수: 카테고리별 약 200개 - 데이터셋 크기: 727.69MB • **전처리 도구** - 얼굴 탐지 및 랜드마크 추출: MediaPipe, MTCNN - 데이터 처리 및 분석: Pandas, NumPy - 이미지 처리 및 변환: OpenCV, PIL  ▲원본 데이터

전처리 과정 및 결과

- **데이터 전처리 방식**
  - 각 이미지에 대한 얼굴형을 라벨 데이터로 변환하여 y_train, y_test 생성
  - **이미지 얼굴형 탐지 기법**으로 MediaPipe, MTCNN 활용

① **MediaPipe** 활용 랜드마크 추출

- 얼굴 랜드마크 468개 포인트 추출
- 특징점(눈, 코, 입 등)의 상대적 위치를 정규화하여 사용

```python
# MediaPipe 얼굴 랜드마크 추출 설정
mp_face_mesh = mp.solutions.face_mesh
mp_drawing = mp.solutions.drawing_utils

face_mesh = mp_face_mesh.FaceMesh(static_image_mode=True, max_num_faces=1,
                                  min_detection_confidence=0.5, min_tracking_confidence=0.5)

# 흑백 이미지를 RGB로 변환하는 함수
def gray_to_rgb(images):
    # float64 -> uint8로 변환 (0~255 범위로)
    images_uint8 = np.clip(images * 255, 0, 255).astype(np.uint8)

    # 이미지를 RGB로 변환
    images_rgb = np.array([cv2.cvtColor(img.squeeze(axis=-1), cv2.COLOR_GRAY2RGB) for img in images_uint8])
    return images_rgb

# X_train과 X_test가 float64일 경우 이를 uint8로 변환
X_train_rgb = gray_to_rgb(X_train)  # (N, 224, 224, 1) -> (N, 224, 224, 3)
X_test_rgb = gray_to_rgb(X_test)  # (N, 224, 224, 1) -> (N, 224, 224, 3)

# 훈련 데이터에 대해 랜드마크 추출
train_landmarks_list = []
for img in X_train_rgb:
    # 이미지를 BGR 형식으로 변환
    img_bgr = cv2.cvtColor(img, cv2.COLOR_RGB2BGR)

    # 랜드마크 추출
    results = face_mesh.process(cv2.cvtColor(img_bgr, cv2.COLOR_BGR2RGB))  # RGB로 변환 후 처리

    # 얼굴 랜드마크가 추출되었을 때만 저장
    if results.multi_face_landmarks:
        train_landmarks_list.append(results.multi_face_landmarks[0])
```

▲**MediaPipe** 추출 함수

- 랜드마크가 부분적으로 감지되는 불필요한 이상치 데이터 제거

▲이상치 데이터 **6**개 제거

▲**MediaPipe** 적용 결과

전처리 과정 및 결과

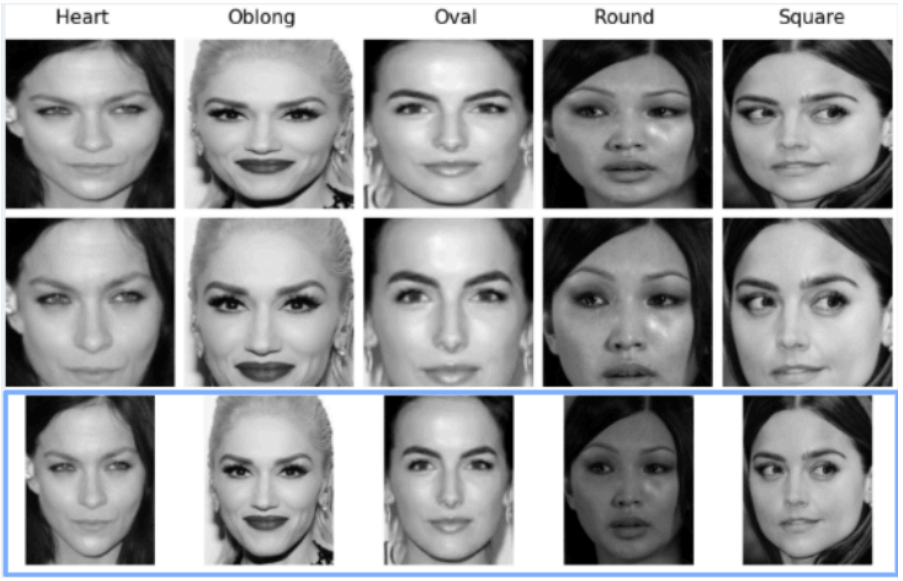
② MTCNN 활용 랜드마크 추출
- 얼굴 랜드마크 5개 포인트 추출
- 다각도 표정 얼굴 처리가 가능하며, 빠른 속도와 높은 정확성
- **이미지에서 얼굴 감지 후, Bounding box 정의**

```
#얼굴 부분만을 정확히 탐지해서 boxing 처리하기
from mtcnn.mtcnn import MTCNN
img2 = cv2.cvtColor(resized_lscp, cv2.COLOR_BGR2RGB)
# creates detector
detector = MTCNN()
# detect faces in an image
results = detector.detect_faces(img2)
results
```

```
[{'box': [24, 17, 174, 198],
  'confidence': np.float64(0.9992732405662537),
  'keypoints': {'nose': [np.int64(122), np.int64(150)],
   'mouth_right': [np.int64(153), np.int64(170)],
   'right_eye': [np.int64(159), np.int64(102)],
   'left_eye': [np.int64(74), np.int64(103)],
   'mouth_left': [np.int64(80), np.int64(169)]}}]
```

▲**MTCNN** 추출 함수

- 이미지 비율을 유지한 상태에서 **Crop & Resize**



▲**Boxing** 기법 적용 결과

- 최종적으로, 각각 다른 얼굴 탐지 기법으로 전처리한 데이터 기반으로 모델링 학습 진행
- 모델링 학습 결과 보고서 : **모델링 및 평가 / 수집된 데이터 및 전처리 문서 확인** 참고할 것