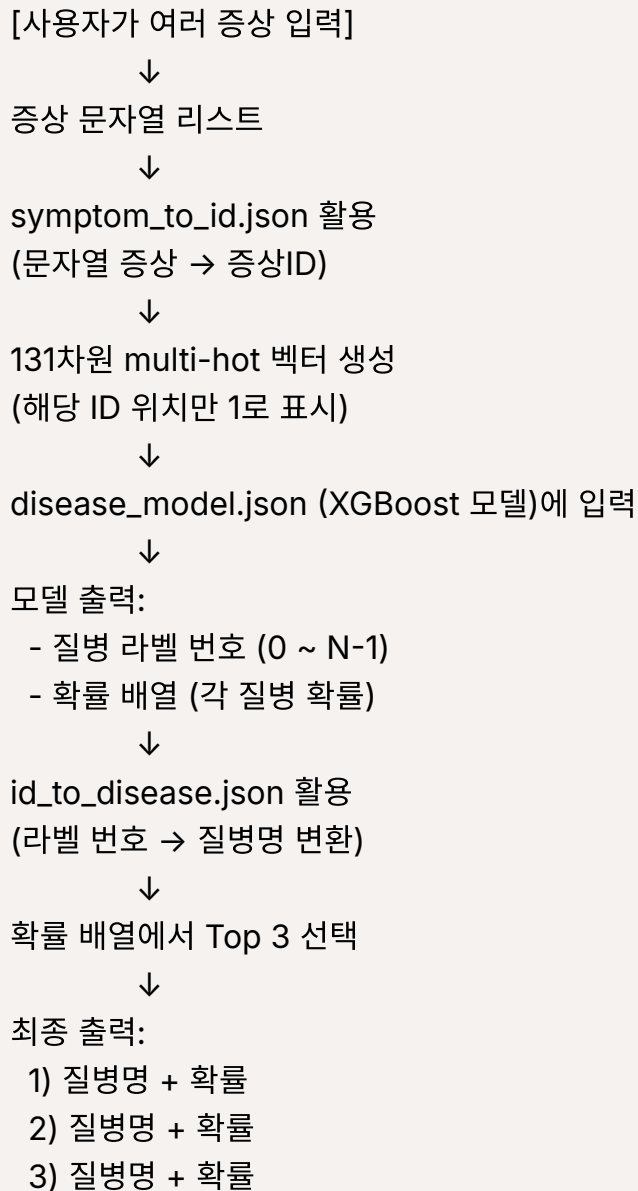


# 질병 분류 모델

## 1. 전체 파이프 라인



## 2. 파일 구성 및 역할

파일명	설명	역할
disease_model.json	학습된 XGBoost 모델	multi-hot 입력을 받아 질병 라벨 번호 예측
symptom_to_id.json	"fatigue": 26	입력 증상을 multi-hot 벡터의 위치로 매핑

파일명	설명	역할
id_to_symptom.json	"26": "fatigue"	숫자 ID를 문자열 증상명으로 변환 (선택적)
id_to_disease.json	"0": "Common Cold"	모델 출력(숫자 라벨)을 실제 질병명으로 변환
README.pdf	전체 구조 요약	각 파일의 사용 규칙 및 변환 절차 설명

## 3. multi-hot 입력 벡터 규칙

### 3-1. 증상 ID 정의

- 전체 증상은 131개로 구성됩니다.
- 각 증상은 1에서 131 사이의 정수 ID를 가집니다.
- 이 ID는 multi-hot 벡터에서 "1이 들어갈 위치"를 나타냅니다.

```
"itching" → 1
"skin_rash" → 2
"fatigue" → 29
```

### 3-2. multi-hot 벡터 생성 규칙

131차원 0으로 초기화된 벡터를 생성한 뒤, 입력된 각 증상에 대해 다음 연산을 수행합니다.

```
id = symptom_to_id[증상명]
multi_hot[id - 1] = 1 // 0-based index
```

예시 입력:

```
["itching", "fatigue"]
```

변환 후:

```
[1, 0, 0, 0, ..., 1, ..., 0] // 총 131차원
```

## 4. 모델 출력 처리 규칙

XGBoost 모델(disease\_model.json)은 multi-hot 벡터를 입력받아 다음 두 가지 값을 제 공합니다.

1. 질병 라벨 번호 (예: 12)
2. 각 질병별 확률 (예: [0.1, 0.03, 0.25...])

라벨 번호는 id\_to\_diseases.json 을 통해 실제 질병명으로 변환합니다.

예:

```
{
  "12": "Common Cold"
}
```

## 5. Top3 질병 반환 규칙

모델이 출력한 확률 배열을 기반으로 상위 3개의 index를 구합니다.

```
prob = booster.predict(...)
top3 = prob.argsort(descending=True)[0:3]
```

각 index는 id\_to\_disease.json 으로 문자열 질병명으로 변환합니다.

결과 예시:

```
[
  ("Common Cold", 72.91%),
  ("Flu", 13.20%),
  ("Allergy", 4.88%)
]
```

## 6. 모든 JSON 파일의 역할 요약

### 6.1 disease\_model.json

- XGBoost 모델 전체 구조를 포함
- 증상 벡터 → 질병 번호로 예측 수행

### 6.2 symptom\_to\_id.json

- "증상 문자열" → ID(1~131)
- multi-hot 벡터 생성 시 반드시 사용

### 6.3 id\_to\_symptom.json

- "ID" → "증상 문자"
- 디버깅 또는 앱에서 증상 텍스트 표시할 때 사용

### 6.4 id\_to\_disease.json

- "라벨 번호" → "질병명"
- 모델 예측 출력을 사람이 읽을 수 있게 변환