Genomic Relatedness Classifier의 불균형 데이터 처리 방법 (Rebalancing Methods)

이 문서는 `genomic_relatedness_classifier` 프로젝트에서 구현된 4가지 클래스 불균형 처리 전략에 대해 포괄적으로 설명합니다. 이 방법들은 훈련 데이터의 클래스 불균형 문제를 해결하여, 특정 클래스의 데이터가 부족하더라도 모델(MLP, CNN, Random Forest)이 친족 관계를 효과적으로 학습할 수 있도록 설계되었습니다.

아래에서 논의되는 구현 로직은 주로 `scripts/train_models.py`에 위치해 있습니다.

각 방법의 효과를 설명하기 위해 **cM_1 데이터셋**을 사례 연구로 사용합니다.

사례 연구 데이터: cM_1

`cM_1` 데이터셋의 원래 훈련 분포는 "UN"(비혈연) 클래스가 압도적으로 많은 고도의 불균형 상태입니다.

원본 클래스 분포 (훈련 세트 근사치):

- **UN (비혈연)**: ~2,404개 (다수 클래스)
- **4촌 (4th Degree)**: ~338개
- **5촌 (5th Degree)**: ~266개
- **3촌 (3rd Degree)**: ~184개
- **2촌 (2nd Degree)**: ~144개
- **1촌 (1st Degree)**: ~127개
- **6촌 (6th Degree)**: ~44개 (소수 클래스)

총 샘플 수: ~3,507개

---

1. Zero (베이스라인)

정의 (What)

"Zero" 방식(CLI 인자: `zero`)은 **명시적인 재조정(rebalancing)을 수행하지 않는** 베이스라인 훈련 접근 방식입니다. 모델은 train/test 분할 후 제공된 데이터셋 그대로 훈련됩니다.

방법 (How)

- **데이터**: 모델은 2,404개의 "UN" 샘플과 44개의 "6촌" 샘플이라는 원본 분포를 그대로 학습합니다.
- **손실 함수 (Loss Function)**: 가중치가 없는 표준 `nn.CrossEntropyLoss()`를 사용합니다.

이유 (Why)

- **베이스라인 비교**: 다른 재조정 전략들의 효과를 측정하기 위한 대조군 역할을 합니다.
- **성능**: 테스트 결과(cM_1 데이터셋의 MLP 모델), 이 방식은 **0.4737의 F1-Macro 점수**를 기록했습니다. 모델이 "UN" 클래스에 크게 편향되어 높은 정확도(Accuracy)를 보일 수는 있지만, 더 가까운 친족 관계를 정확하게 식별하는 데는 실패할 가능성이 큽니다.

---

2. Weighted (클래스 가중치)

정의 (What)

"Weighted" 방식(`weighted`)은 데이터셋을 수정하는 대신 손실 함수 내에서 알고리즘적으로 불균형을 해결합니다. 소수 클래스의 오분류에 대해 더 높은 페널티(손실)를 부여합니다.

방법 (How)

- **가중치 계산**: 가중치는 클래스 빈도에 반비례하게 계산됩니다.
  - "6촌"(44개)에 대한 가중치는 "UN"(2404개)에 대한 가중치보다 대략 **54배 더 높습니다**.
- **손실 함수**: `nn.CrossEntropyLoss(weight=class_weights)` 를 사용합니다.

이유 (Why)

- **비용 민감 학습 (Cost-Sensitive Learning)**: 모델이 소수 클래스에 더 주의를 기울이도록 강제합니다.
- **성능**: **0.4363의 F1-Macro 점수**를 기록했습니다.
  - *관찰*: 놀랍게도 이 특정 케이스에서는 베이스라인보다 성능이 낮았습니다. 이는 가중치가 너무 공격적이어서 모델이 노이즈가 섞인 소수 클래스 예제에 과도하게 집중하거나, 경사 하강법(gradient descent)을 불안정하게 만들었기 때문일 수 있습니다.

---

3. SMOTE (Synthetic Minority Over-sampling Technique)

정의 (What)

SMOTE(`smote`)는 기존 소수 클래스 샘플들 사이를 보간(interpolate)하여 **합성(synthetic)** 데이터를 생성함으로써 다수 클래스의 샘플 수에 맞추는 데이터 수준의 재조정 기법입니다.

방법 (How)

- **목표**: 모든 클래스를 다수 클래스(UN: 2,404개)의 수에 맞춰 오버샘플링합니다.
- **결과 분포**:
  - UN: 2,404 (원본)
  - 6촌: 44 원본 + ~2,360 합성 = 2,404
  - ...나머지 모든 클래스도 동일
- **총 훈련 크기**: 2,404 * 7개 클래스 = **16,828 샘플**.
- **설정**: 극소수 클래스를 처리하기 위해 `k_neighbors=1` 을 사용합니다.

이유 (Why)

- **완전한 대표성**: 모델이 모든 클래스에 대해 동일한 수의 예제를 학습하도록 보장합니다.
- **성능**: **0.5207의 F1-Macro 점수**를 기록했습니다. 이는 베이스라인 대비 유의미한 향상이며, 합성 데이터를 생성하는 것이 소수 클래스에 대한 일반화 성능을 높이는 데 도움이 됨을 증명합니다.

---

4. OverUnder (하이브리드 샘플링)

정의 (What)

"OverUnder" 방식(`overunder`)은 **오버샘플링**, **언더샘플링**, 그리고 **데이터 정제(cleaning)**를 결합하여 특정하고 효율적인 목표 크기를 달성하는 정교한 하이브리드 전략입니다.

방법 (How)

1. **목표 설정**: **클래스당 400개 샘플**이라는 고정된 목표를 설정합니다.
2. **오버샘플링 (SMOTE)**: 400개 미만인 소수 클래스(예: 44개인 6촌)를 SMOTE를 사용하여 400개까지 늘립니다.
3. **언더샘플링**: 다수 클래스(2,404개인 UN)를 무작위로 400개로 줄입니다.
4. **정제 (ENN/Tomek)**: SMOTEENN과 같은 알고리즘을 사용하여 결정 경계(decision boundary) 근처의 "노이즈"나 모호한 샘플을 제거하여 클래스 분리를 명확하게 합니다.
5. **결과 분포**: 클래스당 약 400개 샘플.

- **총 훈련 크기**: ~2,800 샘플.

이유 (Why)

- **효율성**: 데이터셋 크기를 ~16,828개(SMOTE)에서 ~2,800개로 줄여, 균형을 유지하면서도 훈련 속도를 **6배 더 빠르게** 만듭니다.
- **품질**: 정제 단계를 통해 모호한 샘플을 제거합니다.
- **성능**: **가장 높은 0.5372의 F1-Macro 점수**를 달성했습니다.
    - *결론*: 이 방법은 정확도와 계산 효율성 간의 최적의 균형을 제공하며, 베이스라인과 순수 SMOTE 방식 모두를 능가합니다.