Job Application Prediction with Consideration of Data Imbalance Problem in HPC

2024-2 AJOU SOFTCON

이 름 윤태영 지도교수 오상윤

연구배경

최근 인공지능 및 복잡한 계산 문제를 해결하기 위해 고성능 컴퓨팅(HPC) 자원을 요구하는 작업들의 수가 증가하고 있다. 이와 같은 상황에서 제한된 HPC 자원을 효율적으로 관리하기 위해서는 작업의 특성과 자원 요구 사항을 정확히 예측하는 것이 중요하다. 특히 실행되는 어플리케이션이 computation bound인지 I/O bound 인지를 사전에 파악하면, 시스템 자원의 적절한 할당이 가능해진다. 또한, 어플리케이션 예측을 통해 서로 다른 유형의 작업을 혼합 배치(즉, 작업 우선순위 조정)하여 자원 간의 경쟁을 최소화할 수 있으며, 결과적으로 전체 시스템의 성능을 최적화할 수 있다.

이를 위해 이전 연구(자기주도연구1)에서 실제 HPC 환경의 로그 데이터를 사용하여 작업의 어플리케이션을 예측하기 위한 머신 러닝 기반의 기법을 고안하였으나, 어플리케이션의 데이터 불균형이 존재하여 소수의 데이터를 가진 어플리케이션의 특징이 모델에 잘 학습되지 못해 예측이 어렵다는 문제가 존재하였다.

이에 본 연구에서는 이전 연구를 심화하여 데이터 불균형 문제의 분석 및 해결을 통한 더욱 개선된 어플리케이션 예측 기법을 제안하고자 한다.

결과 및 분석

표1은 각 모델별 전처리 전과 후 그리고 오버 샘플링 시 결과를 나타낸 것으로, Random Forest 모델을 사용하여 SMOTE기법으로 200% 오버 샘플링을 진행하였을 때, 예측 정확도는 95.28%, Macro Average는 0.74로 가장 높은 성능을 보였다.

표 1. 모델별 어플리케이션 예측 성능 평가

구 분	전처리 전		전처리 후		오버샘플링 후	
	정확도(%) (Accuracy)	F1_score 평균 (Macro avg)	정확도(%) (Accuracy)	F1_score 평균 (Macro avg)	정확도(%) (Accuracy)	F1_score 평균 (Macro avg)
Random forest	84.84	0.63	93.85	0.66	95.28	0.74
Decision tree	83.28	0.62	93.34	0.64	94.43	0.71
K-NN	82.02	0.45	93.27	0.66	95.08	0.69
Catboost	83.33	0.55	93.12	0.57	94.56	0.66

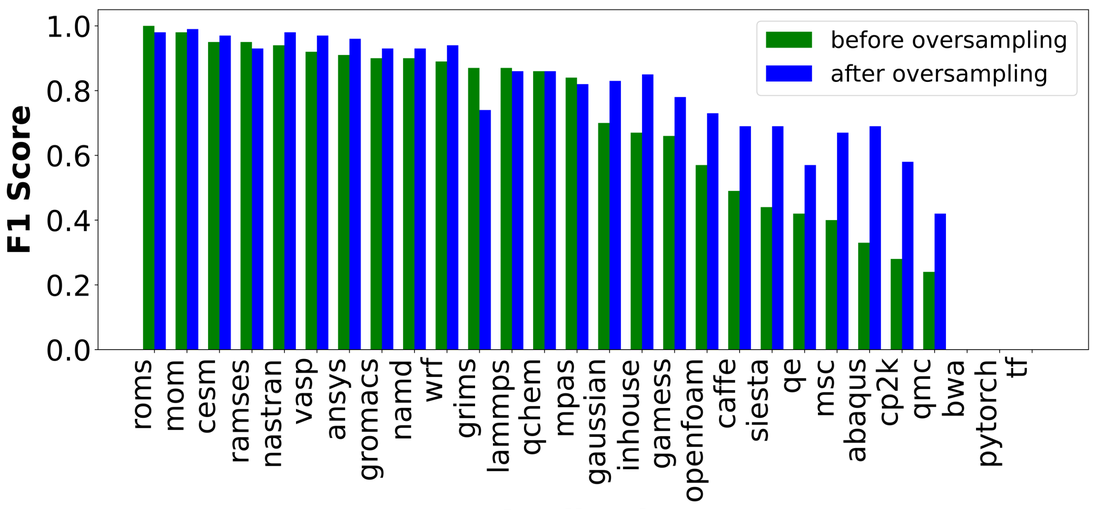

그림 6. 오버 샘플링 전과 후 어플리케이션별 F1-score 비교 그래프

그림6은 오버 샘플링 적용 전과 후의 각 어플리케이션별 F1-score를 나타낸 그래프로, 데이터의 양이 20개 미만으로 매우 적은 경우에는 오버 샘플링 적용 시에도 한계가 존재해 예측이 불가능하였다. 오버 샘플링 적용 전에 예측 성능이 높았던 어플리케이션들의 경우 오버 샘플링 적용 후에도 예측 성능이 유지되거나 미약하게 증가되었으며, “abaqus”의 경우는 F1-score가 약 0.42가 증가하는 등 소수의 데이터를 가진 어플리케이션들의 예측 성능이 크게 상승한 것을 확인할 수 있다. 본 연구의 결과는 HPC 환경에서 작업의 특성을 고려할 필요가 있는 스케줄링 및 자원 할당 최적화, 자원 소모량 예측 등의 여러 연구에 기여할 수 있을 것으로 예상된다.

연구 진행 과정

그림1은 어플리케이션별 데이터 분포를 내림차순으로 나타낸 그래프로, “vasp”의 경우 523,953개(57.3%)인 것에 반해 “mom” 이하의 어플리케이션의 경우는 데이터의 수가 1,000개(0.1%) 미만으로 어플리케이션별 데이터의 분포가 매우 불균형한 것을 확인할 수 있다.

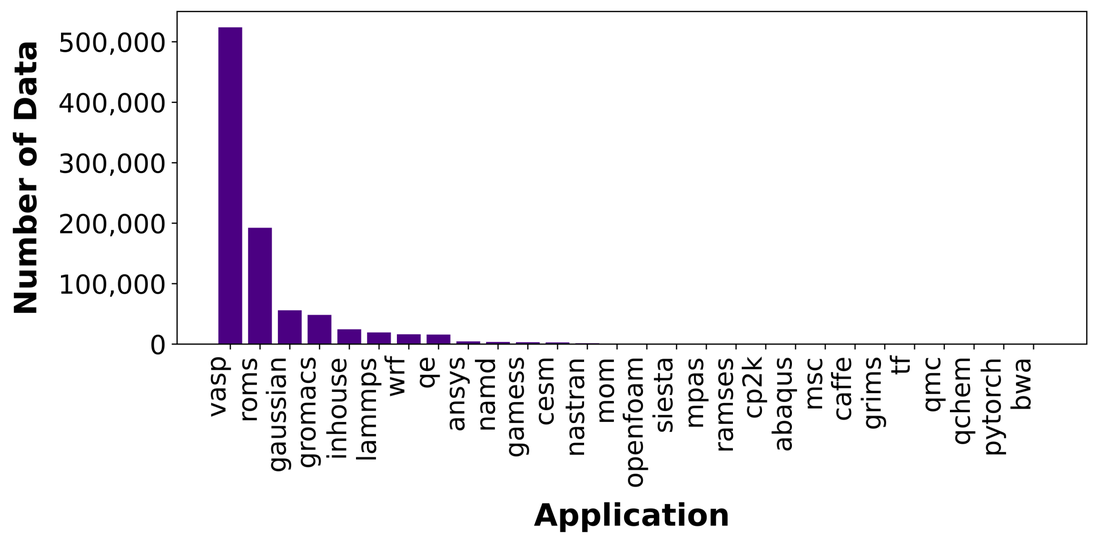

그림 1. Random forest 모델의 오버 샘플링 성능 그래프

본 연구에서는 이와 같은 데이터 불균형 문제를 해결하기 위해 소수의 데이터를 가진 어플리케이션마다 이웃한 같은 클래스의 데이터를 선택하여 데이터 값에 대한 Euclidean 거리를 구해 그 사이에 새로운 데이터를 생성하는 오버 샘플링 방식을 적용하였다. 실험에서는 4가지 오버 샘플링 기법(“Random sampling”, “SMOTE”, “Borderline SMOTE”, “ADASYN”)을 사용하였으며, 최적의 샘플링 비율을 찾기 위해 50%씩 비율을 증가시키며 성능을 비교하는 실험을 진행하였다.

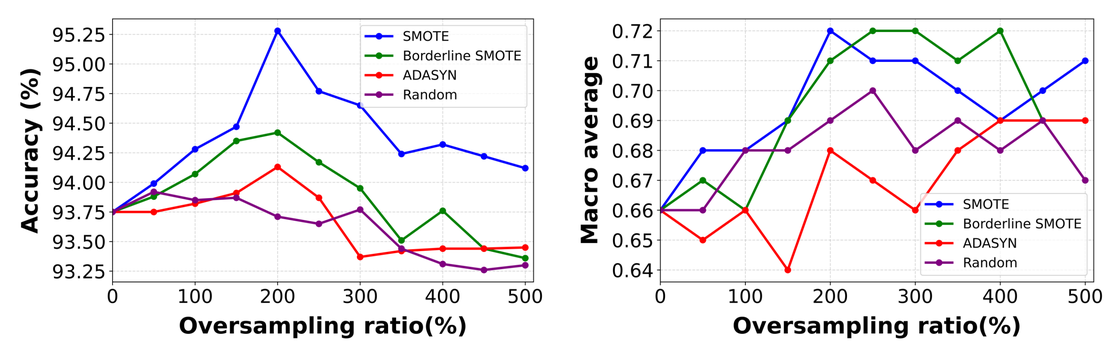

그림 2. Random forest 모델의 오버 샘플링 성능 그래프

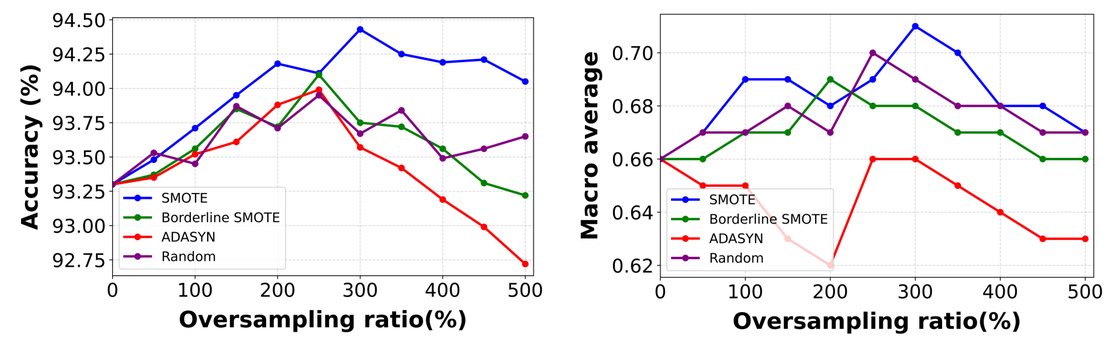

그림 3. Decision tree 모델의 오버 샘플링 성능 그래프

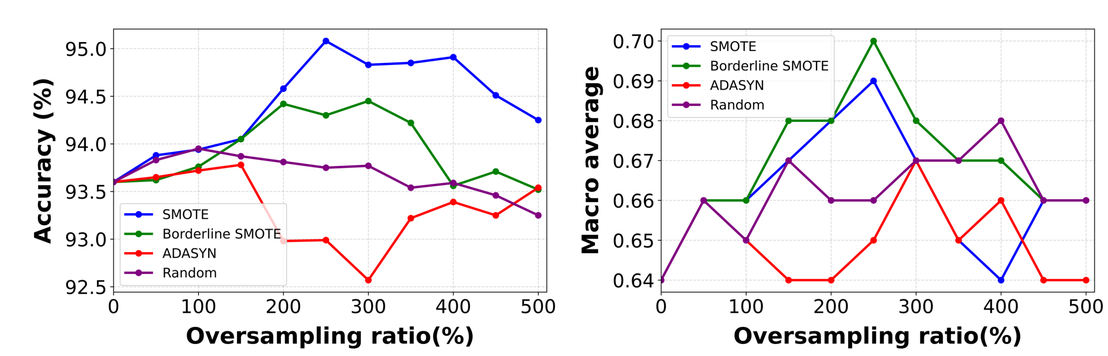

그림 4. K-NN 모델의 오버 샘플링 성능 그래프

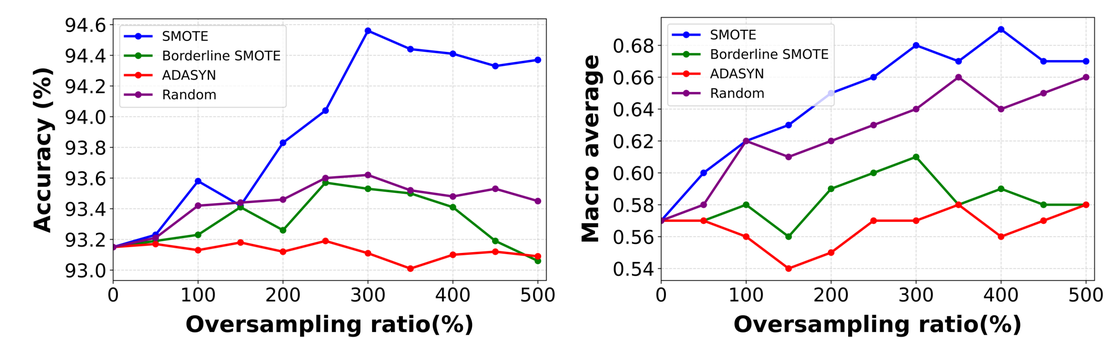

그림 5. Cat boost 모델의 오버 샘플링 성능 그래프