2022147034 박정현 클래스 불균형 문제 해결 보고서

1. 데이터 로드 및 기본 탐색

1.1 데이터 개요

- 데이터셋: Kaggle Credit Card Fraud Detection
- 데이터 크기: 284,807건, 31개 변수
- 결측치: 없음
- 주요 변수: Time, V1~V28, Amount, Class

1.2 클래스 분포

- 정상 거래: 284,315건 (99.83%)
- 사기 거래: 492건 (0.17%)
- 불균형 비율: 577.88:1

데이터셋은 심각한 클래스 불균형을 보이며, 사기 거래가 전체의 0.17%에 불과합니다.

2. 샘플링

2.1 샘플링 전략

- 사기 거래: 전부 유지 (492건)
- 정상 거래: 10,000건 무작위 샘플링
- 샘플링 후 총 데이터: 10,492건

2.2 샘플링 후 클래스 분포

- 정상 거래: 10,000건 (95.31%)
- 사기 거래: 492건 (4.69%)
- 불균형 비율: 20.33:1

샘플링을 통해 불균형 비율을 577.88:1에서 20.33:1로 개선했습니다.

3. 데이터 전처리

3.1 변수 표준화

- Amount 변수: StandardScaler로 표준화
- 새 변수: Amount_Scaled 생성 후 원본 Amount 변수 제거
- 표준화 결과: 평균 0, 표준편차 1

3.2 데이터 분리

- X: Time, V1~V28, Amount_Scaled (총 30개 변수)
- y: Class

4. 학습 데이터와 테스트 데이터 분할

4.1 분할 설정

- 비율: 학습셋 80%, 테스트셋 20%
- 옵션: stratify=y, random_state=42

4.2 분할 결과

- 학습 데이터: 8,393건

  - 정상 거래: 7,999건 (95.31%)
  - 사기 거래: 394건 (4.69%)
- 테스트 데이터: 2,099건
  - 정상 거래: 2,001건 (95.33%)
  - 사기 거래: 98건 (4.67%)

5. SMOTE 적용

5.1 SMOTE를 적용하는 이유

클래스 불균형 문제로 인해 모델이 정상 거래에 편향되어 학습될 수 있습니다. SMOTE는 사기 거래의 합성 샘플을 생성하여 모델이 사기 거래 패턴을 더 잘 학습할 수 있도록 하여 Recall과 F1-score를 향상시킵니다.

5.2 SMOTE 적용 결과

- SMOTE 적용 전: 사기 거래 394건, 정상 거래 7,999건
- SMOTE 적용 후: 사기 거래 7,999건, 정상 거래 7,999건
- 증가한 사기 거래 건수: 7,605건

6. 모델 학습

6.1 모델 선정

4가지 모델을 비교 평가했습니다:

1. Logistic Regression
2. Random Forest
3. Gradient Boosting
4. SVM

6.2 모델 성능 비교

Model	Precision	Recall	F1-Score	PR-AUC
Random Forest	0.9457	0.8878	0.9158	0.9537
Logistic Regression	0.8125	0.9286	0.8667	0.9508
Gradient Boosting	0.8165	0.9082	0.8599	0.9449
SVM	0.0571	0.5408	0.1032	0.0826

선택된 모델: Random Forest

6.3 하이퍼파라미터 튜닝

GridSearchCV로 Random Forest의 하이퍼파라미터를 최적화했습니다.

최적 파라미터:

- n_estimators: 200
- max_depth: 25
- min_samples_split: 2
- min_samples_leaf: 1

최적 CV 점수: 0.9932

7. 최종 성능 평가

7.1 Threshold 조정

다양한 threshold 값에 대해 성능을 평가하여 최적 threshold를 찾았습니다.

최적 Threshold: 0.600

7.2 최종 모델 성능

목표 성능:

- Recall ≥ 0.80
- F1-Score ≥ 0.88
- PR-AUC ≥ 0.90

실제 성능:

- Precision: 0.9770
- Recall: 0.8673
- F1-Score: 0.9189
- PR-AUC: 0.9533

모든 목표를 달성했습니다.

7.3 Class별 성능

Class 0:

- Precision: 0.9935
- Recall: 0.9990

- F1-Score: 0.9963

Class 1:

- Precision: 0.9770

- Recall: 0.8673

- F1-Score: 0.9189

7.4 Confusion Matrix

```
                    예측
실제        정상    사기
정상        1999      2
사기          13     85
```

- True Negative: 1,999건

- False Positive: 2건

- False Negative: 13건

- True Positive: 85건

8. 결론 및 개선 방안

8.1 결론

신용카드 사기 탐지 데이터셋의 클래스 불균형 문제를 해결하기 위해 다음 접근을 사용했습니다:

1. 데이터 샘플링: 정상 거래를 10,000건으로 샘플링하여 불균형 비율을 577.88:1에서 20.33:1로 개선

2. SMOTE 적용: 소수 클래스를 오버샘플링하여 학습 데이터의 클래스 비율을 1:1로 균형화

3. 모델 선정: Random Forest를 선택하고 하이퍼파라미터 튜닝 수행

4. Threshold 조정: 최적 threshold 0.600을 찾아 성능 최적화

최종적으로 목표 성능을 모두 달성했습니다.

8.2 추가 개선 방법 제안

더 높은 성능을 위해 다음 방법들을 시도할 수 있습니다:

1. 앙상블 기법: Voting, Stacking 등 여러 모델을 결합

2. 다른 오버샘플링 기법: ADASYN, Borderline-SMOTE 등

3. 특성 엔지니어링: 새로운 특성 생성, 특성 선택

4. Cost-sensitive Learning: 클래스 가중치 조정