데이터 전처리를 통한 불균형 데이터의 분류 성능 비교 분석

김효원
27469 충북 충주시 대학로 50 한국교통대학교 소프트웨어전공

Comparative analysis of classification performance of imbalanced data through data preprocessing

Hyo Won Kim
Department of Software, Korea National University of Transportation, Chungju, 27469, Korea

ABSTRACT

데이터를 기반으로 한 기계학습은 학습 데이터의 양, 학습 모델 그리고 데이터 특징의 수 등 학습환경에 따라 모델의 성능이 크게 좌우된다. 특히 성능을 높이기 위해서는 데이터의 전처리 과정이 중요하다. 본 논문에서는 Kaggle에서 제공하는 신용카드 사기 검출 데이터셋을 가지고 불균형한 분포를 가진 컬럼이 있을 때 원본 데이터와 Data Scaling, 로그 변환, 이상치 제거, 오버 샘플링 한 후에 원본 데이터와 성능 비교를 한다.

Keywords: 데이터, Data Scaling, 로그 변환, 이상치 제거, 오버샘플링, 머신러닝

Ⅰ. 서론

데이터를 기반으로 한 기계학습은 학습 데이터의 양, 학습 모델 그리고 데이터의 특징의 수 등 학습 환경에 따라 모델의 성능이 크게 좌우된다. 또한 데이터의 전처리 방법에 따라서도 상이한 결과가 보인다. 그렇기 때문에 성능을 높이기 위해서는 데이터의 전처리 과정이 중요하다. 데이터 전처리는 데이터를 분석 및 처리에 적합한 형태로 만드는 과정이다.[1][2]
데이터 과학자의 시간의 60%는 데이터를 정리하고 구성하는데 보낸다. 그만큼 데이터 전처리는 매우 중요하다.[3]

그러므로 본 논문에서는 kaggle에서 제공한 신용카드 사기 검출 데이터를 사용하여 전처리 방법에 따른 모델의 성능을 비교하고자 한다.[3]

Ⅱ. 본론

1. Kaggle 신용카드 사기 검출

Time	V1	V2	V3	V4	V5	V6	V21	V22	V23	V24	V25	V26	V27	V28	Amount	Class
0	-1.36	-0.073	2.536	1.378	-0.338	0.462	-0.018	0.278	-0.11	0.067	0.129	-0.189	0.134	-0.021	149.6	0
0	1.192	0.266	0.166	0.448	0.06	-0.082	-0.226	-0.639	0.101	-0.34	0.167	0.126	-0.009	0.015	2.69	0
1	-1.358	-1.34	1.773	0.38	-0.503	1.8	0.248	0.772	0.909	-0.689	-0.328	-0.139	-0.055	-0.06	378.7	0
1	-0.966	-0.185	1.793	-0.863	-0.01	1.247	-0.108	0.005	-0.19	-1.176	0.647	-0.222	0.063	0.061	123.5	0
2	-1.158	0.878	1.549	0.403	-0.407	0.096	-0.009	0.798	-0.137	0.141	-0.206	0.502	0.219	0.215	69.99	0
2	-0.426	0.961	1.141	-0.168	0.421	-0.03	-0.208	-0.56	-0.026	-0.371	-0.233	0.106	0.254	0.081	3.67	0
4	1.23	0.141	0.045	1.203	0.192	0.273	0.168	-0.271	-0.154	-0.78	0.75	-0.257	0.035	0.005	4.99	0
7	-0.644	1.418	1.074	-0.492	0.949	0.428	1.943	-1.015	0.058	-0.65	-0.415	-0.052	-1.207	-1.085	40.8	0
7	-0.894	0.286	-0.113	-0.272	2.67	3.722	-0.073	-0.268	-0.204	1.012	0.373	-0.384	0.012	0.142	93.2	0
9	-0.338	1.12	1.044	-0.222	0.499	-0.247	-0.247	-0.634	-0.121	-0.385	-0.07	0.094	0.246	0.083	3.68	0

Fig. 1. 신용카드 사기 검출 원본 데이터셋

kaggle의 데이터셋인 신용카드 사기 검출은 2013년 9월 유럽 카드 소지자들의 이틀간 카드 거래 내역을 저장한 csv 파일 데이터셋이다. 사기거래의 경우 1, 정상 거래의 경우 0

으로 Class 컬럼에 저장되어있다.

전체 284,807건의 거래내역 중 492건의 사기 거래가 보고 되어있으며, V1~V28은 기밀 유지로 인해 다른 정보는 제공되어 있지 않고, PCA 변환을 거친 상태의 실수 형태로 제공된다. Time은 거래시간, Amount는 거래금액을 나타낸다.[4]

이 데이터를 선정한 이유는 Amount 컬럼에 데이터 쏠림현상과 Class 컬럼에 데이터 불균형이 존재하기 때문이다.

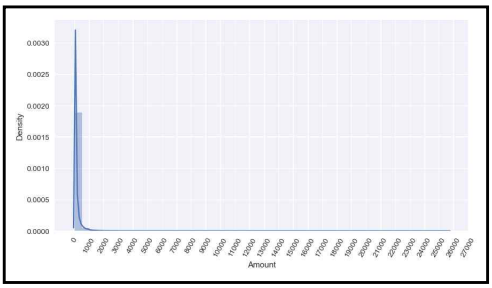

Fig. 2. Amount 컬럼 원본 데이터 분포도

Amount 컬럼을 살펴보면 1000달러이하에 데이터들이 쏠려있는 것을 확인할 수 있다.

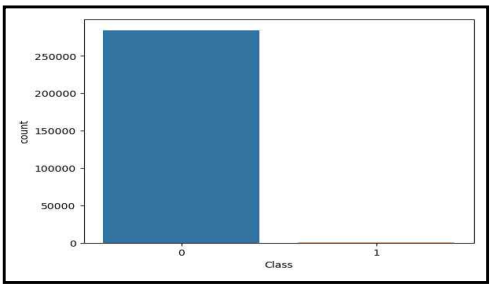

Fig. 3. Class 컬럼 원본

Class 컬럼을 살펴보면 대부분이 0으로 저장되어있는 것을 확인 할 수 있다. 컬럼 0의 비율은 99.8%이고 1의 비율은 0.2%이다.

2. 데이터 전처리에 따른 성능 결과

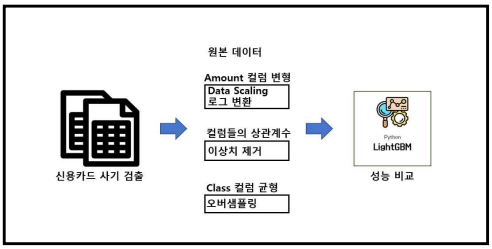

Fig. 4. 성능 평가 순서

데이터 전처리 후 성능 비교를 하기 때문에 원본 데이터를 먼저 성능 평가한 후에 다양한 데이터 전처리 방법을 사용해서 비교하였다.

신용카드 사기 검출 데이터셋은 Class 값이 명시되어 있기 때문에 성능 검증 방법으로 지도 학습 방식을 사용해 학습을 진행했다.

데이터셋은 학습/테스트 데이터의 비율을 train_test_split 모듈을 사용해서 70%:30%로 나누어서 사용하였다. 이때 데이터의 Class가 불균형하기 때문에 Stratify를 사용하여 분류하였다. Stratify는 Class의 비율을 학습/테스트 데이터에 한쪽에 몰아 분배되는 것을 방지해주는 기법이다. 만약 이것을 지정해주지 않고 분류 문제를 다루면 성능 차이가 발생하게 된다.[5]

Table 1. 데이터 레이블 값 비율

Class	학습 데이터	테스트 데이터
0	99.827451	99.826785
1	0.172549	0.173215

Table 1을 통해 학습 데이터와 테스트 데이터의 Class 0과 1의 비율이 잘 나뉜 것을 확인할 수 있다.

머신러닝 모델은 LightGBM을 사용하였다. LightGBM은 요즘 가장 많이 사용하는 모델 중 하나이다. 변수의 종류가 많고 데이터가 클수록 뛰어난 성능을 보여주기 때문에 선정하였다.[6]

성능 평가는 정확도와 혼동행렬 기반의 정밀도, 재현율, f1 score, AUC를 사용하였다.

(1) 원본 데이터 결과

원본 데이터만을 사용해서 LightGBM을 통해 성능을 검증하였다.

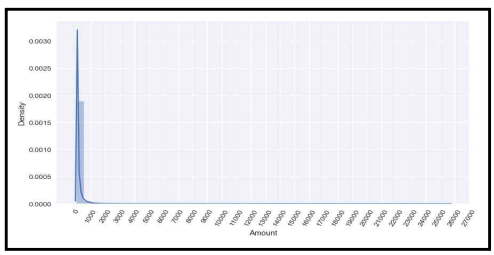

Fig. 5. Amount 컬럼 원본 데이터 분포도

Table 2. Amount 컬럼 원본

상위 하위 데이터 5개

	Amount
0	149.62
1	2.69
2	378.66
3	123.50
4	69.99
...	...
284802	0.77
284803	24.79
284804	67.88
284805	10.00
284806	217.00

Table 3. 원본 데이터만 사용 결과

	LightGBM
정확도	0.9995
정밀도	0.9573
재현율	0.7568
F1	0.8453
AUC	0.9790

(2) StandardScaler

Amount 컬럼의 값이 한쪽으로 쏠려있어서 상당히 불균형한 분포를 가지고 있다. 그래서 StandardScaler를 통해 Scaling을 하였다.

StandardScaler는 모든 피처들이 평균이 0, 분산이 1인 정규분포를 갖도록 만들어준다.[8]

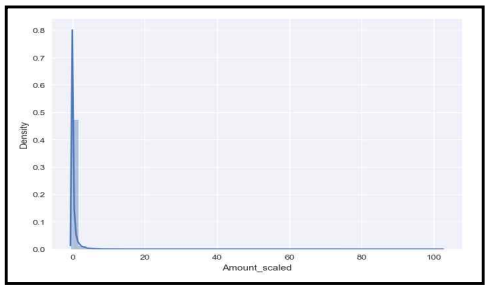

Fig. 6. StandScaler_Amount 컬럼

Table 4. StandScaler_Amount 컬럼

상위 하위 데이터 5개

	Amount
0	0.244964
1	-0.342475
2	1.160686
3	0.140534
4	-0.073403
...	...
284802	-0.350151
284803	-0.254117
284804	-0.081839
284805	-0.313249
284806	0.514355

Table 5. StandardScaler 사용 결과

	LightGBM
정확도	0.9995
정밀도	0.9569
재현율	0.7500
F1	0.8409
AUC	0.9779

결과는 원본 데이터 Table 3과 성능을 비교하면 정밀도, 재현율, F1, AUC는 낮고 정확도는 동일한 것을 확인하였다.

(3) RobustScaler

RobustScaler는 StandardScaler와 방법이 비슷하다. RobustScaler는 중간값을 0으로 하고 사분위값을 1인 정규분포 사용해서 이상치의 영향을 최소화하는 방법이다.[7]

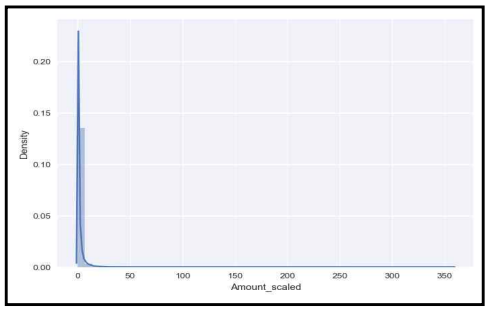

Fig. 7. RobustScaler_Amount 컬럼

Table 6. RobustScaler_Amount 컬럼

상위 하위 데이터 5개

	Amount
0	1.783274
1	-0.269825
2	4.983721
3	1.418291
4	0.670579
...	...
284802	-0.296653
284803	0.038986
284804	0.641096
284805	-0.167680
284806	2.724796

Table 7. RobustScaler 사용 결과

	LightGBM
정확도	0.9995
정밀도	0.9658
재현율	0.7635
F1	0.8528
AUC	0.9788

RobustScaler와 원본 데이터 Table3과 비교하면 정확도는 동일하고 정밀도, 재현율, F1의 성능은 높게 나온 것을 확인하였다.

(4) 로그 변환

로그 변환은 컬럼의 큰 분포를 작게 만들어서 복잡한 계산을 쉽게 하기 위해 사용한다. [8]

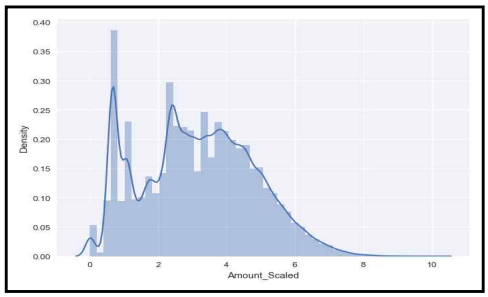

Fig. 8. log_Amount 컬럼

Table 8. log_Amount 컬럼

상위 하위 데이터 5개

	Amount
0	5.014760
1	1.305626
2	5.939276
3	4.824306
4	4.262539
...	...
284802	0.570980
284803	3.249987
284804	4.232366
284805	2.397895
284806	5.384495

Table 9. 로그 변환 사용 결과

	LightGBM
정확도	0.9995
정밀도	0.9576
재현율	0.7635
F1	0.8496
AUC	0.9796

로그 변환을 사용하였더니 원본 데이터 Table 3과 비교하면 정확도는 동일하지만 정밀도, 재현율, F1, AUC 모두 높게 나온 것을 확인할 수 있었다. Amount 컬럼을 변형주어 가장 성능이 높게 나오는 방법은 로그 변환이었다.

(5) 이상치 데이터 제거

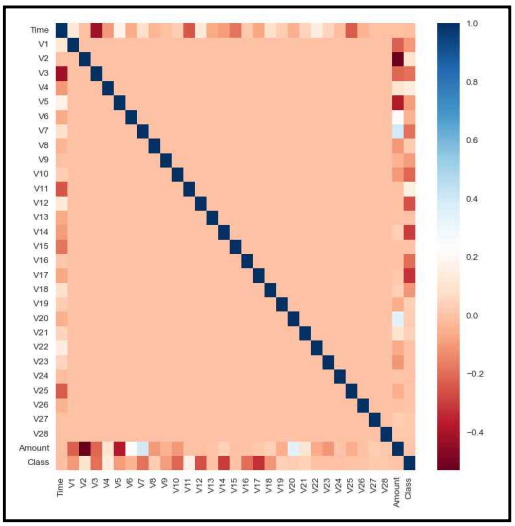

Fig. 9. 각 컬럼들의 상관계수

맨 아래 Class를 보면 음의 상관관계가 가장 높은 컬럼인 V14와 V17에 대한 이상치 검출을 하였다.

음의 상관관계란 한 변수가 증가함에 따라 다른 변수는 감소하는 경우를 뜻한다. [9]

Table 10. 이상치 데이터 인덱스

	이상치 인덱스
V14	8296, 8615, 9035, 9252
V17	

V14의 이상치가 4개 검출되었고, V17은 이상치가 검출이 안됐다. V14의 이상치를 제거한 후 성능을 측정을 하였다.

Table 11. 이상치 제거

	LightGBM
정확도	0.9996
정밀도	0.9603
재현율	0.8288
F1	0.8897
AUC	0.9780

이상치 제거와 원본 데이터 Table3를 비교하면 정확도, 정밀도, 재현율, F1은 성능이 높은 것을 확인 할 수 있고 AUC만 성능이 낮은 것을 확인하였다.

(6) 오버 샘플링

오버 샘플링이란 적은 데이터 세트를 증식하여 학습을 위한 충분한 데이터를 확보하는 방법이다. 단순하게 동일 데이터를 늘리면 오버 피팅이 되기 때문에 SMOTE 방법을 사용하여 오버 샘플링을 진행하였다.[11]

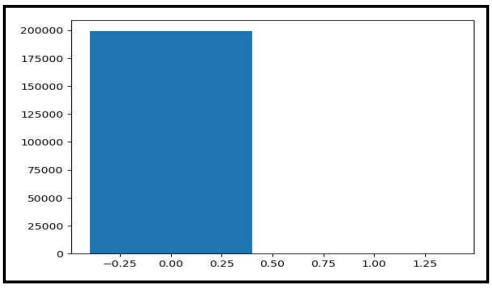

Fig. 10. 오버 샘플링 적용 전 레이블 값 분포

SMOTE 적용 전 학습용 피처/레이블 데이터 세트는 (199364, 30) (199364,)이고 적용 전 레이블 값의 분포는 0은 199013개 1은 351개가 나왔다.

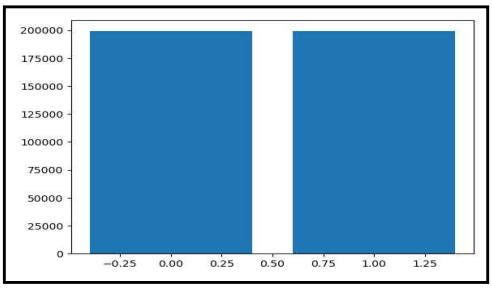

Fig. 11. 오버 샘플링 적용 후 레이블 값 분포

SMOTE 적용 후 학습용 피처/레이블 데이터 세트는 (398026, 30) (398026,)이고 적용 후 레이블 값의 분포는 0은 199013개 1도 199013개가 나왔다.

Table 12. 오버 샘플링 적용 결과

	LightGBM
정확도	0.9996
정밀도	0.9118
재현율	0.8493
F1	0.8794
AUC	0.9246

오버 샘플링과 원본 데이터 Table3을 비교하면 정확도, 재현율, F1은 성능이 높다. 하지만 정밀도와 AUC는 성능이 낮은 것을 확인할 수 있었다.

(7) 로그 변환 + 이상치 제거 + 오버샘플링

앞서 (2)~(4) 과정을 통해 Amount 컬럼에 변형을 주는 것에는 로그 변환이 가장 성능이 높았던 것을 알 수 있었다.

원본 데이터의 Amount 컬럼에 로그 변환을 주었고 (5) 이상치 데이터를 제거하고 마지막으로 (6) 오버 샘플링을 하여 성능을 측정하였다.

Table 13. 로그 변환+이상치 제거+오버 샘플링

	LightGBM
정확도	0.9995
정밀도	0.8732
재현율	0.8378
F1	0.8552
AUC	0.9188

원본 데이터 Table 3과 결과를 비교하면 앞서 로그 변환, 이상치 제거, 오버 샘플링을 각각 사용했을 때는 원본보다 성능이 높게 나왔다. 하지만 3가지 방법을 한 번에 사용한 전처리한 결과의 정확도는 동일하지만 정밀도, AUC는 성능은 낮은 것을 확인 할 수 있었다.

Ⅲ. 결론

Table 14. 성능 총 정리

Light GBM	원본	Standard	Robust	로그 변환	이상치 제거
정확도	0.9995	0.9995	0.9995	0.9995	0.9996
정밀도	0.9573	0.9569	0.9658	0.9576	0.9603
재현율	0.7568	0.7500	0.7635	0.7635	0.8288
F1	0.8453	0.8409	0.8528	0.8496	0.8897
AUC	0.9790	0.9779	0.9788	0.9796	0.9780

Light GBM	오버 샘플링	로그 변환+이상치 제거 +오버 샘플링
정확도	0.9996	0.9995
정밀도	0.9118	0.8732
재현율	0.8493	0.8378
F1	0.8794	0.8552
AUC	0.9246	0.9188

본 논문에서는 데이터 전처리가 머신러닝 성능 예측에 미치는 영향에 대해 알아보았으며, Kaggle 신용카드 사기 검출 데이터셋을 사용하여 원본 데이터와 Data Scaling, 로그 변환, 이상치 제거, 오버 샘플링을 사용하여 성능을 비교 분석하였다.

Amount 컬럼에 변형을 주는 Data Scaling, 로그 변환과 원본 데이터를 비교하면 정확도는 0.9995로 모두 같은 성능을 보였다. 하지만 정밀도, 재현율, F1, AUC에서 성능 차이가 보였다. StandardScaler는 원본 데이터에 비해 성능이 낮은 것을 확인 할 수 있었다. 그리고 RobustScaler는 정밀도, 재현율, F1이 높은 것을 확인 할 수 있었고, 로그 변환은 모든 부분이 더 높게 나왔다. 그 결과 Amount 컬럼에 변형을 줄 때에는 로그 변환이 효율적인 것을 알게 되었다.

그리고 데이터의 상관관계를 구하고 음의 상관관계가 가장 높은 컬럼에서 이상치를 찾아내 원본 데이터와 성능 비교를 하였다. 정확도, 정밀도, 재현율, F1 모두 원본 데이터보다 눈에 띄게 성능이 올라간 것을 확인할 수 있었다.

그리고 오버 샘플링을 통해 Class 값의 불균형을 없애고 원본 데이터와 비교한 결과 정확도는 이상치 제거와 같은 정확도인 0.9996이 나왔지만 아무래도 데이터를 임의적으로 추가시키는 작업이라 정밀도의 성능은 떨어진 것을 확인하였다.

마지막으로 Amount 컬럼의 데이터를 변형했을 때 성능이 좋았던 로그 변환, 이상치 제거와 오버 샘플링을 함께 사용한 결과는 성능이 이상치 제거 하나만 한 결과에 비해 낮게 나온 것을 확인 할 수 있었다.

이번 연구를 통해 Amount 컬럼처럼 한 쪽에 몰려있는 데이터를 가질 때는 로그 변환을 통해 불균형을 줄여주는 것과 데이터의 상관 관계를 구해 음의 상관계수가 높은 컬럼에서의 이상치를 검출해 제거하는 방법, Class컬럼의 불균형을 오버샘플링 방법을 통해 균형을 맞추는 법 모두 다 원본 데이터보다 성능이 높게 나온 것을 확인 할 수 있었다. 하지만 이 3가지 방법을 한 번에 사용하면 성능이 오히려 떨어진 것을 확인 할 수 있었다. 이것을 통해 전처리를 많이 사용하면 무조건 성능이 올라가는 것이 아닌 성능이 떨어질 수 있다는 것도 확인하였다.

향후 연구에서는 본 연구에서 사용한 데이터

전처리 뿐만 아닌 다양한 데이터 전처리 방법을 사용하면서 성능을 올릴 수 있는 방법을 찾아보고자 한다.

감사의 글

이 논문은 2023년도 한국교통대학교 소프트웨어전공 졸업논문으로 제출한 것임

참고문헌

[1]김동현, 유승언, 이병준, 김경태, 윤희용. (2019.1). 효율적인 기계학습을 위한 데이터 전처리. 한국컴퓨터정보학회 학술발표논문집. p49~50
[2]우성우, “데이터 전처리란?-데이터 전처리 정의, 작업단계 순서”, https://modulabs.co.kr/blog/data-preproccesing/
[3]Gil Press, “Cleaning Big Data: Most Time-Consuming, Least Enjoyable Data Science Task, Survey Says”, https://www.forbes.com/sites/gilpress/2016/03/23/data-preparation-most-time-consuming-least-enjoyable-data-science-task-survey-says/?sh=74299b286f63
[4]Kaggle. Credit Card Fraud Detection, https://www.kaggle.com/datasets/mlg-ulb/creditcardfraud
[5]박균우, 윤성욱. (2022.12). Kaggle Credit Card Fraud Detection 데이터셋을 활용한 이상치 탐색 구현. 한국정보과학회 학술발표논문집, p1, 655~1,657
[6] “train_test_split 모듈을 활용하여 학습과 테스트 세트 분리”, https://teddylee777.github.io/scikit-learn/train-test-split/, 2020.01.17.
[7]hwi-e, “[ML] LightGBM 이해하고 사용하자”, https://hwi-doc.tistory.com/entry/%EC%9D%B4%ED%95%B4%ED%95%98%EA%B3%A0-%EC%82%AC%EC%9A%A9%ED%95%98%EC%9E%90-LightGBM, 2020.08.03
[8]sssssun, “sklearn으로 데이터 스케일링(Data Scaling)하는 5가지 방법”, https://dacon.io/en/codeshare/4526, 2022.02.13.
[9]cosmicdev, “데이터 분석 로그 변환”, https://velog.io/@cosmicdev/%EB%8D%B0%EC%9D%B4%ED%84%B0-%EB%B6%84%EC%84%9D-%EB%A1%9C%EA%B7%B8-%EB%B3%80%ED%9998, 2021.09.05.
[10]박용규,(2001),통계시리즈:상관분석과 회귀분석, 가정의학회지, p43~51
[11]정현승, 강창완, 김규곤. (2008). 불균형 데이터에 대한 오버샘플링 효과 연구. Journal of The Korean Data Analysis Society, 10(4), 2089-2098.