08 분류실습_ santander 고객 만족

⇒ XGBoost (느림) / LightGBM (비교적 빠름) 을 이용하여 실습을 진행

ROC-AUC 로 모델평가.

대부분이 만족 고객이고 일부가 불만족일 것이므로 정확도 보다는 ROC-AUC 가 적합.

[1] 데이터 전처리

- DataFrame 으로 로딩
- shape, info() 를 이용
- Target 속성 값의 분포로 확인 → 불만족 고객의 비율 4%에 불과.
- 학습 데이터, test 데이터 분리.

[2] XGBoost 학습 / 하이퍼 파라미터 튜닝

1) n_estimator = 500, early_stopping_rounds =100 으로 설정

⇒ ROC AUC: 0.8419

```
params = {'max_depth':[5, 7] , 'min_child_weight':[1,3] ,'colsample_
```

2) max_depth, min_child_weight, colsample_bytree 튜닝.

↳ 시간이 많이 걸리는 ML 모델은 2~3개의 파라미터의 최적값을 먼저 구하기 + 개별으로 1~2개 결합.

↳ 각 파라미터에 2가지 경우를 입력 → 총 8개의 경우의 수 ($2^3=8$)

3) 위에서 구한 최적의 파라미터에서 구한 최적값 + (n_estimators=1000, learning_rate=0.2, reg_alpha=0.3)

⇒ 처음 적합한 것보다 ROC AUC의 값이 증가함.

4) 피처 중요도 확인 → var38, var15 가 중요.

[3] LightGBM 학습 / 파라미터 튜닝

1) n_estimator =500, early_stopping_rounds=100으로 설정

⇒ XGBoost 보다 훨씬 시간 단축됨

2) num_leaves, max_depth, min_child_samples, subsample 튜닝

3) 위에서 구한 최적 값으로 다시 LightGBM을 이용.

⇒ ROC AUC 딱히 향상XX

09장. 분류실습 - 신용카드

⇒ 0이 정상, 1이 사기, 단 0.172% 만 차지 (극도로 불균형한 분포)

이럴 경우, 이상 레이블의 건수가 매우 적어, 오버 샘플링 또는 언더 샘플링 방식을 이용, 주로 오버 샘플링을 이용한다.

- 언더 샘플링: 많은 데이터를 적은 데이터 수준으로 감소. : 데이터 삭제 너무 많아 잘 이용하지 X
- 오버 샘플링: 적은 데이터를 증식. SMOTE 방식을 이용
  - ↳ KNN을 통해 데이터와 K개의 이웃들의 차이를 일정 값으로 만들어 새로운 데이터 생성

[1] 데이터 가공 및 모델 학습/예측/평가

1) preprocessed_df() 함수 생성 : 불필요한 Time 만 삭제.
2) get_train_test_dataset() 생성: get_preprocessed_df() 호출 → 레이블과 피처 값의 분리 →
   train_test_split()으로 학습, 테스트 데이터 분할 (stratify = y_target)
3) 학습데이터와 테스트 데이터가 비슷한 비율로 분리되었는지 확인
4) LogisticRegression 기반의 모델로 예측 진행
   ↳ 평가는 get_clf_eval() 이용. (오차행렬, 정확도, 정밀도, 재현율, F1, AUC 포함)
5) LGBM 기반의 모델로 예측 진행.
   ↳ get_model_train_eval() 생성: Estimator 객체와 학습/테스트 데이터 set을 입력..
   불균형한 데이터 분포를 가지므로, boost_from_average=False 를 포함.

[2] 데이터 분포도 변환 후 모델 학습/예측/평가

1) Amount 피처는 정상/사기를 결정하는 매우 중요한 속성으로 추측.
   ↳ 대부분 1000불 이하이나 2만불까지 포함하 매우 긴 분포를 가진다.
2) Amount를 표준 정규분포로 변환 (import StandardScaler)
   변환 후에 원래의 DataFrame에 결합 → 기존 Time, Amount 값 삭제.
3) 수정된 데이터로 LogisticRegression, LGBM을 이용해 성능 평가.
4) 위의 예측 성능 크게 개선 안됨. 로그변환 이용 (np.log1p())
5) 다시 LogisticRegression, LGBM으로 성능평가
   ⇒ 성능이 향상됨

[3] 이상치 제거 후 모델 학습/예측/평가

1) 이상치: 일반적으로 $(Q_1 - 1.5 \times IQR,\ Q_3 + 1.5 \times IQR)$ 밖의 범위에 있는 값을 이상치로 설정
   $IQR: (Q_3 - Q_1)$
   모든 데이터의 이상치를 다 제거하기에는 시간 소요 많다 → 가장 corr() 값이 높은 것을 선택.
2) heatmap을 이용하여 상관관계가 높은 피처 선택.

3) get_outlier( )을 설정: 위에서 결정한 column 데이터 추출 → 4분위값, IQR 구하여 → outlier 추출

4) 위에서 구한 outlier를 제거하는 get_processed_df( )를 생성.

5) LogisticRegression, LGBM 으로 예측

⇒ 예측 성능이 크게 좋아짐.

4 SMOTE 오버 샘플링 적용 후 모델 학습 / 예측 / 평가

1) SMOTE (import SMOTE)를 이용하여 오버 샘플링.

↳ 반드시 학습데이터에 대해서만 진행.

2) SMOTE 적용 후, LogisticRegression 성능 확인

↳ 정확도 증가하나 정밀도는 5% 대로 확 감소. 현업 업무에 적용X

3) Precision_recall_curve_plot 을 이용해 문제점 확인.

↳ 임계값의 민감도가 매우 심함을 확인.

4) LGBM을 이용해 예측성능 평가 ⇒ SMOTE를 적용하면 재현율↑, 정밀도↓ 같이 일반적.

좋은 SMOTE 일수록 재현율 증가분은 높이고 정밀도 감소분은 낮춘다.

결과) 원본 를 LR / LGBM을 이용하여 예측 진행, 평가를 통해 다양한 모델 이용해봄

- 로그변환
- 이상치 제거
- SMOTE