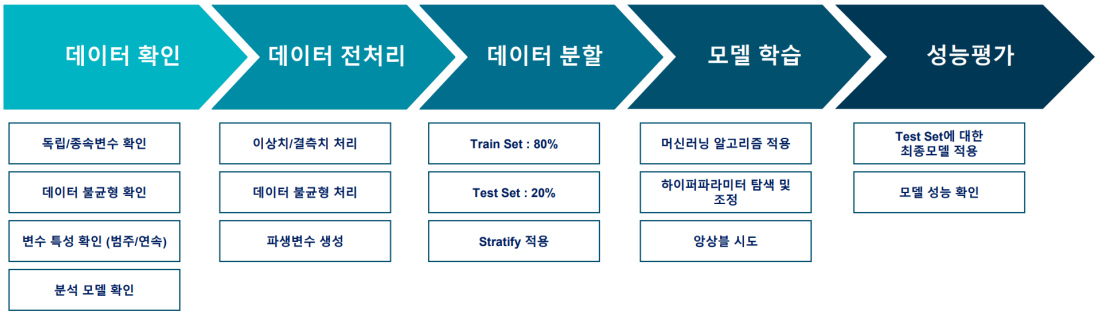
고객 채무 불이행 여부 예측

이어드림스쿨 4기 모의대회

프로젝트 요약

- P2P 대부 업체의 고객 데이터를 통한 채무 불이행 예측 모델 개발



1. 데이터 EDA

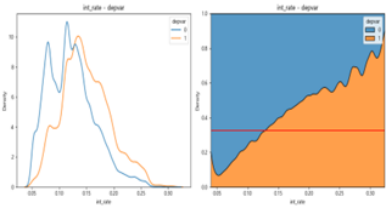
데이터 **전반적인 구조 및 의미 파악 및 예측에 있어 유의미한 변수**가 있는지 분석

[피처 ↔ 타겟]

- 범주형 피처
  - 시각화 : Barplot, Mosaic Plot
  - 통계 검정: 카이제곱 검정
- 수치형 피처
  - 시각화: Boxplot, Histogram, Kde Plot

[피처 ↔ 피처]

- 시각화: Pairplot, Heatmap



2. 파생변수 생성

데이터 탐색 및 도메인 조사를 기반으로 예측 성능을 향상시킬만한 주요 변수 생성

- **상환능력 관련 변수 생성**

```
# 상환기간
df['term']  = [36 if x == 1 else 60 for x in df['term1']]
# 총 이자
df['tot_rec'] = df['funded_amnt'] * (df['term'] / 12) * df['int_rate'] # 총 이자

# 총 이자 / 지금까지 납부된 이자
df['rec_rate'] = df['tot_rec'] / (df['total_rec_int']+1)

# 총 상환금액 / 지금까지 상환된 금액
df['fund_return'] = (df['installment'] * df['term']) / (df['funded_amnt'] - df['out_prncp'] + df['tot
al_rec_int'] )

# 지금까지 납부한 총 연체료가 0이 넘는가
df['total_rec_late_group'] = df.apply(lambda x : 1 if( x['total_rec_late_fee'] > 0)  else 0 , axis = 1)
```

이외에도 다양한 파생변수 생성 시도를 했으나, feature importance 를 기준으로 중요하게 작용한 피처는 다음과 같다.

- **총 상환 금액 / 누적 납부 금액**
- **총 이자 금액 / 누적 납부 이자 금액**
- 지금까지 납부한 총 연체료가 0이 넘는가에 대한 여부

3. 데이터 불균형 처리

타겟 데이터 불균형 확인 (0 : 67% , 1 : 32%)

시도1 . Over Sampling

- 다수 클래스 데이터 수에 맞춰 소수 클래스의 데이터를 생성하는 방식

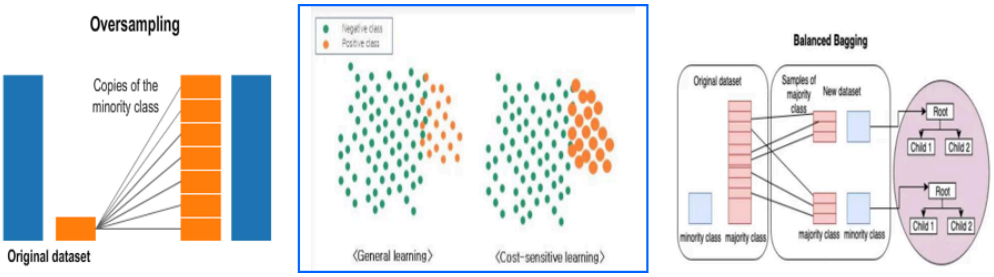
시도2. Cost-Sensitive Learning

- 데이터 자체를 생성하는 것은 아니며, **소수 클래스의 비용함수에 높은 가중치**를 두어 균형잡힌 학습을 하는 방식

시도3. Balanced Bagging Classifier

- 만일 300개의 인스턴스를 갖는 다수의 클래스와 100개의 인스턴스를 갖는 소수의 클래스가 있을 경우

- 다수의 클래스의 데이터를 부트스트래핑 방식으로 각각 100개씩 샘플링하여 총 3개의 서브셋 생성, 각 서브셋에 소수 클래스의 100개 데이터를 결합하여 학습 진행 → voting 방식으로 예측



소수 클래스에 더 높은 가중치를 두고 학습하는 방법 선택

(Class 샘플 수에 비례한 가중치 계산)

```
# class weight 설정
from sklearn.utils.class_weight import compute_class_weight

classes = classes = np.unique(y_train)
weights = compute_class_weight(class_weight = 'balanced' , classes = classes , y = y_train)
class_weights = dict(zip(classes , weights))
class_weights
```

4. 데이터 분리

target 데이터가 불균형이기에 train 또는 test에 클래스가 치우쳐 학습되지 않도록 설정

- **stratify** 를 설정하여 target이 동일한 비율로 클래스가 나눠지게 함

```
from sklearn.model_selection import train_test_split

target = 'depvar'

x = df.drop(target , axis = 1)
y = df[target]

x_train , x_test , y_train , y_test= train_test_split(x,y , test_size= 0.2 , random_state= 42 , stratify =y)
```

```
print(y_train.value_counts(normalize=True))
print(y_test.value_counts(normalize=True))
```

train 과 test 각각 target 비율을 확인해봤을 때 동일하게 나눠짐을 확인

5. 모델링

[베이스라인 모델 선정]

- 별도의 **전처리 없이** 여러 모델들을 학습하여 성능 비교
- **부스팅 계열 모델**이 전반적으로 우수한 성능을 나타냄
- 범주형 변수가 많은 Raw 데이터를 고려해 **CatBoost** 선정

Model	F1 Score
LogisticRegression	0.4036
RandomForest	0.6520
GradientBoost	0.6584
LightGBM	**0.7004**
XGBoost	**0.6984**
CatBoost	**0.7070**

[하이퍼파라미터 튜닝]

- **Optuna(하이퍼파라미터 최적화 라이브러리)** 를 활용해 파라미터 탐색
- F1 Score을 기준으로 가장 큰 값을 가진 파라미터로 모델 최적화

```
def objective_xgb(trial):
    params = {}
  model = XGBClassifier(**params)
  model.fit(x_train, y_train, verbose=0)
  preds = model.predict(x_test)
  return f1_score(y_test, preds, average="macro")

# Optuna study for XGBClassifier
study_xgb = optuna.create_study(direction="maximize")
study_xgb.optimize(objective_xgb, n_trials=50)
print("Best XGB params:", study_xgb.best_params)
```

[앙상블 시도]

베이스 모델 선정 시, 성능이 높게 나왔던 부스팅 계열 모델 **하이퍼파라미터 튜닝 후 앙상블** 시도

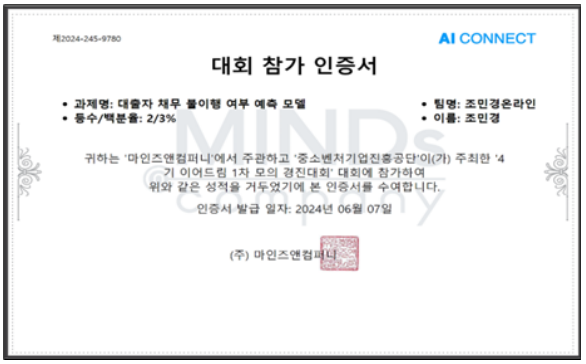
- Stacking , **Soft Voting (XGB + CatBoost + LGBM)** 시도

- **SoftVoting**을 활용했을 때 성능이 향상

```
# Voting
voting_clf = VotingClassifier(
    estimators=[
        ('cat', cat_clf),
        ('xgb', xgb_clf),
        ('lgbm', lgbm_clf)
    ],
    voting='soft'
)
```

6. 성과

- F1 Score(Macro) : 0.7070 → 0.7462
- 베이스라인 성능 대비 **5.55 % 성능 향상**
- **이어드림 4기 내부 모의 경진대회 2등**


제2024-245-9780

AI CONNECT

대회 참가 인증서

- 과제명: 대출자 채무 불이행 여부 예측 모델
- 등수/백분율: 2/3%
- 팀명: 조민경온라인
- 이름: 조민경

귀하는 '마인즈앤컴퍼니'에서 주관하고 '중소벤처기업진흥공단'이(가) 주최한 '4기 이어드림 1차 모의 경진대회' 대회에 참가하여 위와 같은 성적을 거두었기에 본 인증서를 수여합니다.

인증서 발급 일자: 2024년 06월 07일

(주) 마인즈앤컴퍼니