**신용카드 사용자 연체예측 데이터분석**

**2022.02.26 서재완**

1. **데이터 정보 파악**
   1. 총 데이터 수(row) : 26457, 피처 개수 :20
   2. 종속변수 – credit > 0,1,2로 구성, 테스트데이터의 모델 결과값이 0,1,2 로 나와야 해서 분류모델로 설계할 예정이다.
   3. 변수 별 정보파악 (더미변수화 하기 전에 질적변수의 구성원 확인)
      1. Car, reality는 y,n로 구성되어 있는데 이 값은 있다 없다를 의미하므로 y를 1, n을 0으로 하였다..
      2. Gender: F,M으로 구성 > 더미변수화
      3. Income\_type : 5개로 구성('Commercial associate', 'Working', 'State servant', 'Pensioner', 'Student')
      4. Edu\_type : 5개로 구성 ('Higher education', 'Secondary / secondary special', 'Incomplete higher', 'Lower secondary', 'Academic degree')

>> 여기서 범주 값이 순위가 정해져 있으므로 가장 높은 것을 5~1로 하려고 했지만 다시 생각해 보니 그 구간을 나누는 차이가 다를 수 있으므로 함부로 변환하지 않았다.

* + 1. Family\_type : 5개로 구성('Married' 'Civil marriage' 'Separated' 'Single / not married' 'Widow')
    2. House\_type: 6개로 구성('Municipal apartment' 'House / apartment' 'With parents' 'Co-op apartment' 'Rented apartment' 'Office apartment’ )
    3. Occyp\_type: : 18개의 직업과 nan값(8172개)

>> 전체 중 31퍼센트가 결측치

>> 직업별 credit차이를 보고 넣을지 결정, 즉 상관성을 보고 결정

>> 일단 이 피처를 빼고 모델링하기로 결정 > 차후 정확성을 보고 변수를 넣을지 재결정

1. **전처리**
   1. 전처리하기 전에 학습용 데이터와 테스트용 데이터를 병합하여 전처리하고 분리한다.
   2. Occyp\_type 변수 일단 데이터프레임에서 제외
   3. 질적변수들 더미변수화
   4. Days\_employed 예외처리 + not\_employed 변수 새로 생성
      1. 양수값은 실직자로 간주하므로 0으로 돌리고 days\_employed가 0인 값을 1로 가지는 not \_employed 변수 생성
   5. 양적 변수 스케일링 진행
2. **모델링**
   1. 로지스틱 모델
      1. ‘libliear’ : accuracy 0.64
      2. ‘newton-cg’ : accuracy 0.64

>> 목적이 다항 분류이기 때문에 로지스틱 모델로는 성능의 한계를 보인다.

* 1. 결정 트리모델
     1. Accuracy 0.61; 과적합의 위험성 존재한다.
  2. 랜덤 포레스트 모델 : 결정 트리 모델의 과 적합 해결
     1. Accuracy 0.70
     2. 피처 별 중요도를 보여주는 메서드로 중요도가 0.1 이상인 변수들만으로 모델 재구성

1. **성능을 올릴 수 있는 작업 진행**
   1. 하이퍼 파라미터 튜닝: 랜덤 포레스트의 파라미터를 성능을 최적으로 올리는 파라미터로 바꾸는 것. 랜덤 포레스트 모델 특성상 큰 변화는 있지 않았다.
   2. 이상치 처리 : 4개의 시나리오 중 가장 성능이 좋은 시나리오 선택

# 이상치 탐지 방법: q1과 q3사이를 벗어난 데이터를 이상치로 정의한다.

* + 1. 이상치 제거 : accuracy 0.725
    2. 이상치 변환 : accuracy 0.717

# q1과 q3를 벗어난 데이터들은 q1(하한값)과 q3(상한값)로 변환하여 모델링에 적용

* + 1. 로그변환 후 이상치 제거 : accuracy 0.723

# 양적변수들의 skew를 측정해보았을 때 생각보다 skewed한 분포를 띄고 있는 변수가 많았다. >> 그래서 로그변환을 하고 이상치 처리를 하는 방법을 생각해 보았다.

* + 1. 로그변환 후 이상치 변환 : accuracy 0.716
  1. 모델링 방법 변경
  2. 새로운 변수 추가 및 조합
  3. 데이터 추가

1. **프로젝트를 하면서 놓친 점**
   1. 중복데이터를 처리해 주어야 하는데 그 점을 미처 알지 못했다. 이는 모델의 성능에 큰 영향을 미침

**>> 해결점: 범주형 변수를 조합하여 개인정보를 나타낼 수 있는 id변수를 생성 했어야 했다.**

* 1. 범주형 변수는 ordinal\_encoder를 했다면?

# ordinal\_encoder는 범주형 데이터를 희소행렬로 그 결과 반환

encoder = OrdinalEncoder(categorical\_feats)

train[categorical\_feats] = encoder.fit\_transform(train[categorical\_feats],train['credit'])

test[categorical\_feats] = encoder.transform(test[categorical\_feats])

* 1. 로그변환과 정규화는 동시에 이루어지면 안된다. 한번에 하나만
  2. **새로운 모델링 방법을 선택했더라면? Catboost**
  3. **일반적으로 하는 스케일링과 전처리에 앞서 데이터 변수의 정확한 판단과 어떻게 사용할지를 잡는 게 데이터분석의 중요한 포인트이다.**
  4. 이상치를 제거할 때 모든 이상치를 제거하는 것이 아니라 성능에 영향을 미치는 변수들의 이상치만 제거하였더라면??
  5. **다중분류모델의 평가기법은 logloss**