부실 대출 예측 모델

이름: 지윤승 | 학번: 20259431

KAIST

개요

- 본 프로젝트는 Lending Club의 대출 데이터를 활용하여 대출의 부실 여부를 사전에 예측하는 분류 모델을 개발하는 것을 목표로 한다. 금융기관 입장에서는 대출 실행 이전에 부실 가능성이 높은 고객을 조기에 식별하는 것이 매우 중요하며, 이를 통해 신용 리스크 관리의 효율성을 높이고 부실률을 낮출 수 있다.

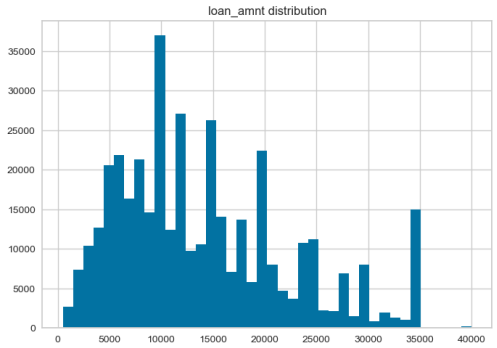
- 우선 탐색적 데이터 분석(EDA)을 통해 대출 금액, 금리, 신용 등급, 부채비율 등 다양한 변수들의 분포와 특성을 파악하고, 부실 대출과 관련된 주요 요인을 살펴보았다. 이후 결측치와 이상치를 처리하고, 범주형 변수에 대한 인코딩과 수치형 변수의 스케일링을 포함한 전처리를 수행하였다.

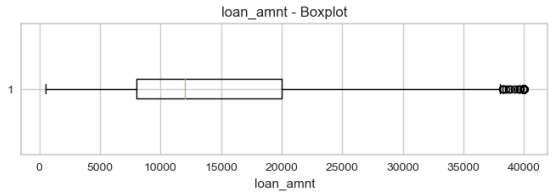
- 또한 interest_amount(loan_amnt × int_rate), installment_income_ratio(installment / annual_inc)와 같은 파생 변수를 생성하여 예측력을 강화하였다.

- 타깃 변수의 불균형 문제는 SMOTE 기법을 통해 해결하였으며, Logistic Regression, Random Forest, Gradient Boosting, XGBoost, LightGBM 등 다양한 머신러닝 알고리즘을 적용해 모델 성능을 비교하였다. 그 결과 LightGBM과 Gradient Boosting이 가장 우수한 AUC 성능을 보여, 최종 예측 모델로 선정하였다.

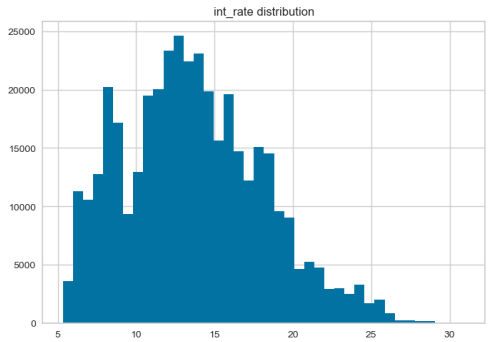
- 분석 결과, 높은 금리(int_rate), 높은 부채비율(dti), 높은 리볼빙 사용률(revol_util), 그리고 낮은 신용등급(grade)이 부실 대출과 밀접한 관련이 있는 것으로 나타났다. 본 모델을 통해 금융기관은 대출 심사 단계에서 위험 고객을 보다 효과적으로 선별할 수 있으며, 리스크 관리 체계 고도화 및 손실 최소화에 실질적으로 기여할 수 있을 것으로 기대된다.

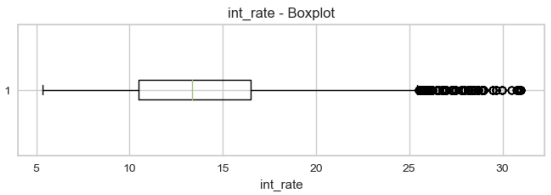
탐색적 데이터 분석(EDA)- 데이터 분포 시각화



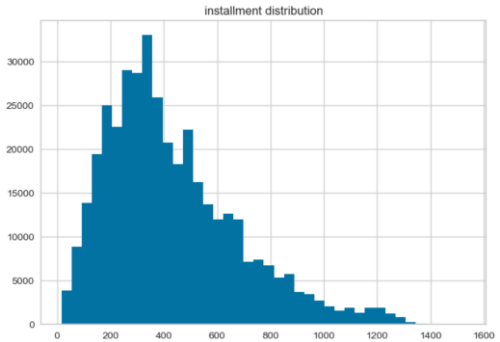


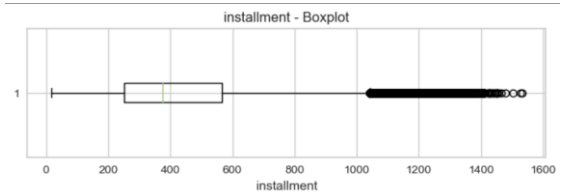
- loan_amnt는 약 0 ~ 40,000달러 구간에 분포하며, 5,000 ~ 15,000달러 구간에 데이터가 가장 많이 집중되어 있다.
- 특히 10,000달러 부근에서 뚜렷한 봉우리가 나타나며, 이는 금융기관이 대출 금액을 일정 단위(5천, 1만 달러 등)로 설정하는 관행과 관련이 있다.
- 분포 형태는 오른쪽으로 긴 꼬리(right-skewed)를 가진 다봉(multi-peak) 구조로, 30,000~40,000달러의 고액 대출이 소수 존재한다.
- 박스플랏을 통해 확인한 결과, 대부분의 대출은 8,000~18,000달러 구간에 몰려 있으며, 고액 대출이 이상치(outlier)로 나타난다.





- int_rate분포를 보면, 대부분의 대출은 10%에서 15% 사이의 중간 금리 구간에 집중되어 있다.
- 전체적으로 오른쪽으로 긴 꼬리를 가진 형태라서, 일부 차주가 20% 이상 높은 금리를 적용받은 경우도 눈에 띈다.
- 박스플롯에서도 25% 이상에서 여러 개의 이상치가 확인되는데, 이는 신용도가 낮은 차주들에게 적용된 고이자 대출로 해석할 수 있다.

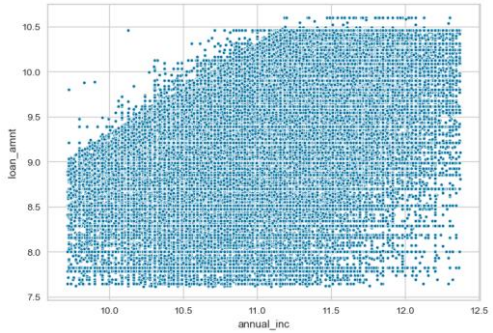




- installment는 대부분의 값이 200~600달러 구간에 집중되어 있고,1,000달러 이상의 고액 할부금은 소수의 이상치로 존재한다.
- 박스플롯에서도 중앙값 주변에 값이 몰려 있으며, 오른쪽 끝으로 긴 꼬리와 다수의 이상치 점들이 확인된다.
- 즉, 전형적인 우측 치우친 분포로, 고액 할부금 차주가 극히 일부임을 알 수 있다.

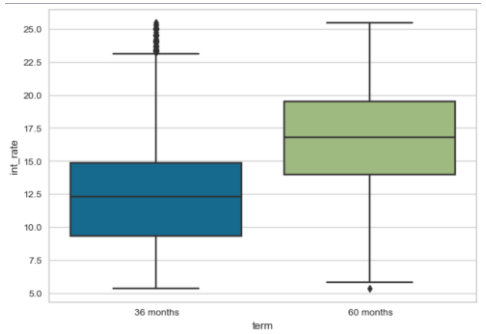
탐색적 데이터 분석(EDA)- 변수 간의 관계 파악

<loan_amnt ↔ annual_inc>



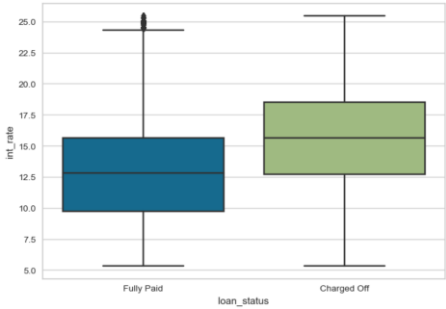
- -전반적으로 소득이 높을수록 대출금액도 커지는 경향이 보인다.

<int_rate ↔ term>



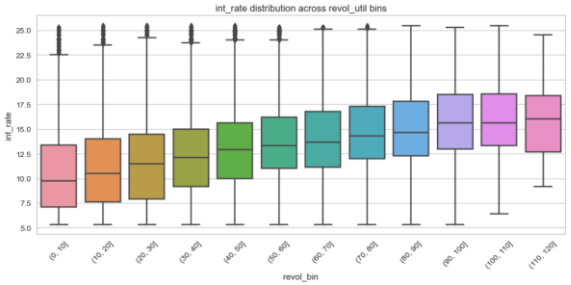
- 60개월 대출이 36개월 대출에 비해 전반적으로 이자율이 더 높게 분포한다.
- 대출 기간이 길수록 대출기관의 리스크가 커지기 때문에, 이를 반영해 더 높은 금리가 적용되는 것으로 해석 가능하다.

<int_rate ↔ loan_status>



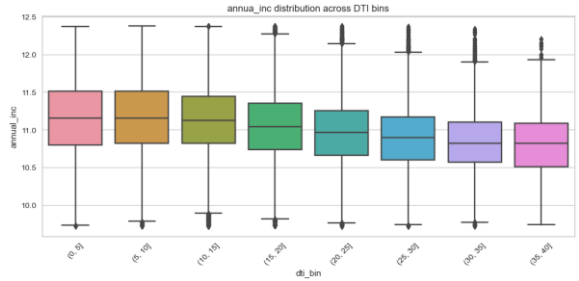
- 연체 및 부실(Charged Off)그룹의 이자율 중앙값이 정상 상환(Fully Paid)그룹보다 명확히 높다.
- 이는 이자율이 높은 대출일수록 부실 가능성이 커진다는 점을 시각적으로 보여준다.
- 또한 Charged Off 그룹의 분포는 전반적으로 상위 구간에 더 치우쳐 있는데, 이는 리스크 프리미엄과도 관련된다.

<int_rate ↔ revol_util>



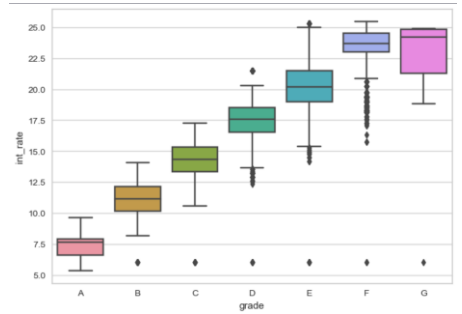
- revol_util(리볼빙 사용률)을 구간별(revol_bin)로 나누어 int_rate(이자율) 분포를 박스플롯으로 확인했다.
- 리볼빙 사용률이 높아질수록 이자율의 중앙값이 점진적으로 상승하는 패턴이 뚜렷하다.
- 특히 사용률이 80% 이상일 때 이자율의 중앙값이 크게 증가하며, 분포의 상단도 함께 올라갔다.
- 이는 신용카드 사용 한도가 많이 소진된 대출자일수록 더 높은 금리를 부과받는 경향을 시각적으로 보여준다.

<dti ↔ annual_inc>



- dti(부채 상환 비율)를 5 단위 구간으로 나누고, 각 구간별annual_inc(연소득) 분포를 박스플롯으로 확인했다.
- DTI가 낮을수록 연소득의 중앙값이 높고, 구간이 커질수록 중앙값이 점차 감소하는 경향이 나타났다.
- DTI가 높은 구간에서는 소득 분포의 하위값이 낮아지고, 전반적으로 소득 수준이 낮은 차이가 뚜렷하게 보인다.

<int_rate ↔ grade>



- grade(등급)별 int_rate(이자율) 분포를 박스플롯으로 확인했다.
- 등급이 낮을수록(→ G 방향) 이자율이 높아지는 뚜렷한 패턴이 나타났다.
- A 등급의 경우 이자율이 낮고 분포 폭도 좁은 반면, 등급이 내려갈수록 중앙값과 상·하위 사분위 범위가 모두 증가했다.
- 이는 신용등급이 낮은 대출자의 경우 높은 이자율이 적용됨을 보여준다.

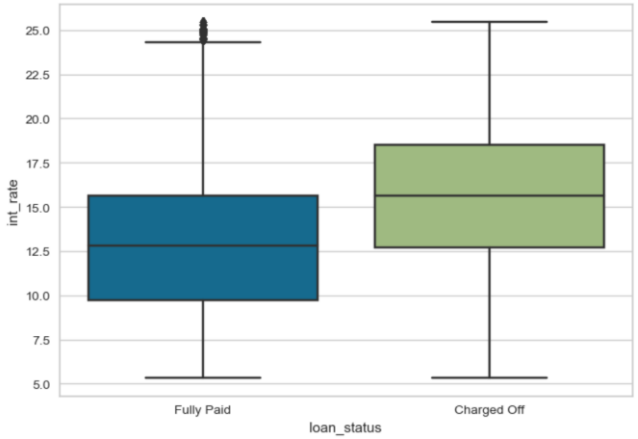
탐색적 데이터 분석(EDA)- 부실 여부(Target) 분석

<부실 처리된 대출 특징>

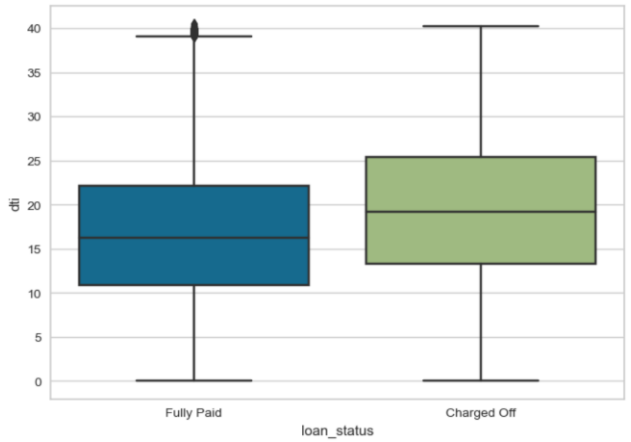
① 금리가 높다
부실 그룹의 금리는 상환 완료 그룹보다 전반적으로 높은 수준을 보인다.
이는 신용도가 낮은 차주에게 높은 금리가 적용되고, 이들이 상환에 실패할 가능성이 크다는 점을 반영한다.

② DTI(Debt-to-Income) 비율이 높다
부실 그룹의 DTI 중앙값이 상환 그룹보다 뚜렷하게 높다.
즉, 부채 부담이 큰 차주일수록 부실 위험이 증가하는 경향이 있다.
소득 대비 부채 수준이 높으면 상환 여력이 줄어드는 것이 자연스러운 결과이다.

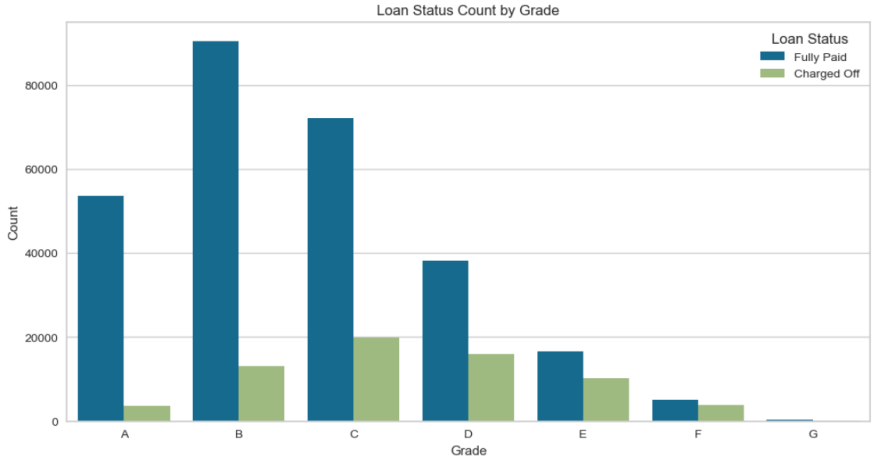
③ 신용등급(Grade)이 낮은 비중이 크다
A~C 등급은 부실 비중이 낮지만, C~E 등급에서 부실 건수가 크게 증가한다.
F~G 등급에서는 대출 규모는 작지만, 부실 비율이 높아 고위험군으로 볼 수 있다.



- 부실 대출(Charged Off)은 높은 이자율을 가진 대출에서 상대적으로 많이 발생하는 경향이 뚜렷하다.
- 반대로 상환 완료(Fully Paid) 대출은 상대적으로 낮은 이자율 구간에 분포가 집중되어 있다.
- 이는 이자율이 대출 리스크를 반영하기 때문이며, 실제 모델링에서도 int_rate는 중요한 예측 변수로 작용할 가능성이 크다.



- 부실(Charged Off)그룹이 정상 상환(Fully Paid)그룹보다 DTI의 중앙값이 더 높다.
- 이는 부채 부담이 큰 차주일수록 부실 위험이 높아지는 경향을 잘 보여준다.



- A~C 등급은 상환 건수가 많고 부실은 상대적으로 적다.
- B 등급이 상환 규모에서 가장 크며, C~D 등급에서는 부실 건수가 눈에 띄게 증가한다.
- E~G 등급은 전체 규모는 작지만, 부실 비중이 높아 고위험군으로 볼 수 있다.

데이터 정제 및 전처리

(1) 결측치 처리

- **title**
→ 차주가 작성한 대출 제목으로, 예측에 큰 영향을 주지 않는 변수이며 결측치 비율도 낮다. 따라서 결측치가 있는 행은 삭제해도 전체 데이터에 영향이 크지 않다.
- **revol_util**
→ 신용위험 예측에서 중요한 수치형 변수이다. 임의로 값을 채우면 왜곡 위험이 크기 때문에, 결측치가 있는 행을 삭제하는 방식이 더 안전합니다. 결측치 자체도 많지 않다.
- **pub_rec_bankruptcies**
→ 파산 이력 변수로, 위험 예측에 매우 중요한 신호를 제공한다. 결측을 잘못 채우면 오신호가 생길 수 있어, 결측치가 있는 행은 삭제하는 것이 적절하다. 결측치가 적기 때문에 정보 손실도 작다.
- **emp_title**
→ 직함을 나타내는 범주형 변수이다. 값의 종류가 매우 다양하고 결측도 발생할 수 있는데, 결측 행을 삭제하면 데이터 손실이 커진다. 따라서 최빈값으로 대체해 정보 손실을 최소화했다.
- **emp_length**
→ 재직기간을 나타내는 범주형 변수이다. 중요 변수이지만 범주형이라 최빈값으로 결측을 채워도 분포 왜곡이 크지 않다. 따라서 최빈값으로 대체했다.
- **mort_acc**
→ 모기지 계좌 수를 나타내는 수치형 변수로, 값이 0~2개에 치우쳐 있다. 평균으로 채우면 이상치의 영향을 받을 수 있기 때문에, 중앙값으로 결측치를 대체해 분포를 안정적으로 유지했다.

(2) 이상치 처리

- IQR(Q3-Q1) 기준으로 상한(Q3+1.5×IQR)과 하한(Q1-1.5×IQR)을 설정한 뒤, 그 범위를 벗어나는 값을 이상치로 간주해서 해당 행을 제거하는 방식이다.
- **Loan_amnt, int_rate, installment, dti, revol_util, annual_inc, mort_acc, open_acc, total_acc** 등의 변수에 적용했다

(3) 변수 변환

- **emp_length**: 문자열을 직접 숫자로 변환 (예: “10+ years” → 10, “< 1 year” → 0)
- **grade, sub_grade**: 등급 간 순서가 존재하는 순서형 변수이므로, Label Encoding으로 순서를 유지하며 숫자로 변환
- **term, emp_title, loan_status, initial_list_status, application_type**: 범주값을 0, 1, 2… 형태로 변환하기 위해 Label Encoding 적용.
- **home_ownership, verification_status, purpose**: 순서가 없는 단순 범주이므로 One-Hot Encoding(더미 변수) 처리.

(4) 날짜 데이터 처리

- **issue_d**

→ 문자열 형태의 날짜를 datetime으로 변환한 뒤, 연도(issue_d_year)와 월(issue_d_month)로 나누어 새로운 컬럼을 만들었다. 이후 원래의 날짜 컬럼(issue_d)은 삭제했다.

- **Earliest_cr_line**

→ 마찬가지로 datetime으로 변환 후, 연도(earliest_cr_line_year)와 월(earliest_cr_line_month)로 분리했다.

피처 엔지니어링

(1) 데이터 스케일링

- StandardScaler를 사용해 표준화(Standardization) 방식으로 스케일링을 적용했다.
- 적용된 변수: **grade, sub_grade, emp_length, dti, revol_bal, address, interest_amount, installment_income_ratio, open_to_total_ratio, public_issue_score**
- 이 변수들에 대해 각 값에서 해당 변수의 평균을 빼고, 표준편차로 나누는 방식을 적용하여 각 변수를 평균 0, 표준편차 1 형태로 변환했다.
- 이렇게 하면 변수 간 단위 차이를 제거하여, 특정 변수의 크기에 모델이 치우치는 것을 방지할 수 있고, 특히 선형 모델이나 거리 기반 모델에서 학습이 더 안정적으로 이뤄질 수 있다.

(2) 불균형 데이터 처리(Imbalanced Data Handling)

- 데이터의 가장 큰 문제 중 하나는 타깃 변수(loan_status)의 클래스 불균형 -> 즉, 정상 상환(Fully Paid) 데이터가 대부분을 차지하고, 부실(Charged Off) 데이터는 상대적으로 매우 적다.
- 이런 불균형이 있으면, 모델이 다수 클래스인 정상 상환 쪽으로 치우쳐 부실 대출을 제대로 예측하지 못하는 문제가 발생한다.
- 이 문제를 해결하기 위해, SMOTE(Synthetic Minority Over-sampling Technique)를 적용했다.
- SMOTE는 소수 클래스(부실 대출)의 데이터를 단순 복제하는 대신, 기존 데이터를 바탕으로 새로운 합성 샘플을 생성하여 학습 데이터를 양쪽 클래스가 균형을 이루도록 재구성한다.

(3) 다중공선성 제거

- 우리 데이터에서는 변수들 간에 서로 강하게 상관된 항목(다중공선성)이 있을 가능성이 있었기 때문에, 먼저 VIF(분산팽창지수)를 계산해 공선성을 진단했다.
- 그 결과, grade와 sub_grade처럼 비슷한 정보를 담고 있는 변수에서 VIF 값이 매우 높게 나타나 공선성이 확인됐다.
- 이에 따라 공선성이 심한 변수인 sub_grade를 제거해서 문제를 해소했다.

(4) 로그 변환/제곱근 변환

- **loan_amnt**와 **annual_inc**는 오른쪽으로 치우친 분포를 가지고 있었기 때문에, 로그 변환을 적용해 값의 스케일을 줄이고 분포를 보다 정규에 가깝게 만들었다. 이렇게 하면 극단값의 영향을 완화하고 모델이 안정적으로 학습할 수 있다.
- **revol_bal**은 소수의 차주가 매우 큰 값을 가지고 있어 제곱근 변환을 적용했다. 로그보다 완만하게 스케일을 줄이면서도 큰 값의 영향을 효과적으로 낮출 수 있기 때문이다.

(5) 파생변수 생성

- **interest_amount:** loan_amnt와 int_rate를 곱해서 생성한 변수로, 대출 기간 동안 지불해야 할 총 이자 금액을 나타낸다. 단순 금리보다 실제 부담 규모를 반영하는 지표다.
- **installment_income_ratio:** installment를 annual_inc로 나눈 값으로, 월 상환액이 연 소득에서 차지하는 비율이다. 차주의 소득 대비 상환 부담 수준을 보여주며, 부실 위험을 평가하는 데 유용하다.
- **open_to_total_ratio:** open_acc를 total_acc로 나눈 값으로, 보유 계좌 중 현재 열려 있는 계좌의 비율을 나타낸다. 신용 활동의 활발함과 현재 사용 중인 계좌 비중을 파악할 수 있는 변수다
- **public_issue_score:** pub_rec와 pub_rec_bankruptcies를 합한 값으로, 공공 기록과 파산 이력을 합산한 신용 위험 지표다. 공적 부정 기록이 많은 차주는 부실 위험이 높을 수 있다.

머신러닝 모델 구축

(1) 모델 비교 및 최적 모델 선정

- 로지스틱 회귀, 랜덤포레스트, 그래디언트 부스팅, XGBoost, LightGBM 등 다양한 분류 알고리즘을 미리 정의해두고, 각 모델에 대해 5-Fold 교차검증(cross validation)을 수행하면서 roc_auc 점수를 계산한다.
- 이렇게 얻은 AUC 점수의 평균값을 비교해서, 어떤 모델이 가장 좋은 성능을 내는지 순위를 매긴다 → LightGBM, Gradient Boosting 활용

(2) LightGBM

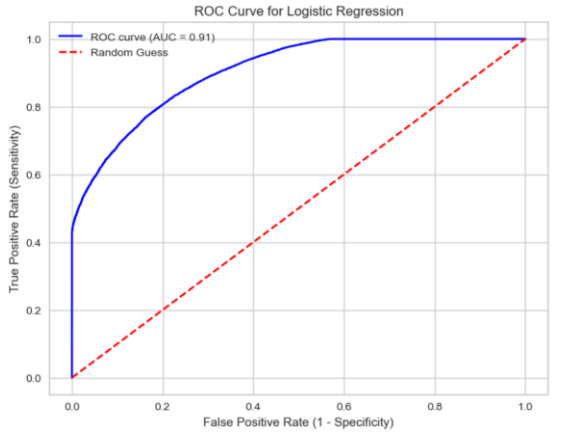
- Gradient Boosting 방식으로 여러 개의 약한 결정 트리를 순차적으로 학습시켜 강력한 예측 모델을 만드는 알고리즘으로 다른 부스팅 모델에 비해 학습 속도가 매우 빠르고 메모리 효율이 좋다.
- n_estimators=500 → 트리의 개수를 500개로 설정. 많은 트리를 학습시켜 성능을 높인다.
- learning_rate=0.05 → 각 트리의 학습 기여도를 조절하는 학습률. 작게 설정해 과적합을 방지하면서 점진적으로 학습한다
- max_depth=-1 → 트리의 깊이에 제한을 두지 않음. LightGBM은 자동으로 최적의 깊이를 탐색한다.

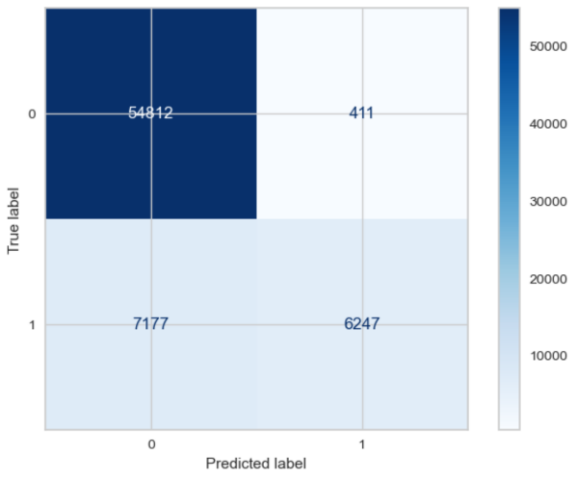
(3) Gradient Boosting

- 그 모델이 틀린 부분(잔차, 오류)에 대해 다음 트리가 집중해서 학습하는 과정을 순차적으로 반복한다. 이렇게 여러 트리를 합쳐 최종적으로 높은 예측력을 가진 모델을 완성한다. 과적합을 방지하고 성능을 부드럽게 조절하기 위해 학습률(learning rate)과 트리 개수 등을 함께 설정한다.
- n_estimators=300→ 트리 300개를 순차적으로 학습시킨다. 트리 수가 많을수록 더 정밀하게 학습할 수 있지만 과적합 위험도 커질 수 있다.
- learning_rate=0.05→ 각 트리의 학습 기여도를 조절하는 학습률이다. 작게 설정해 조금씩 천천히 학습하게 함으로써 과적합을 방지한다.
- max_depth=3→ 각 트리의 최대 깊이를 제한해 복잡도를 조절하고 과적합을 방지한다.

모델 성능 평가- 데이터 불균형 처리 전

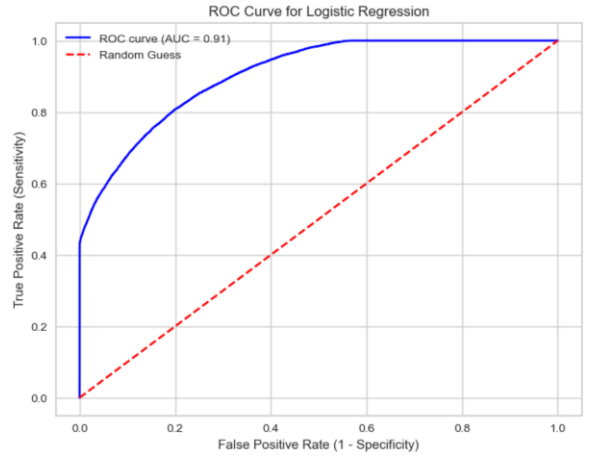
<LightGBM>

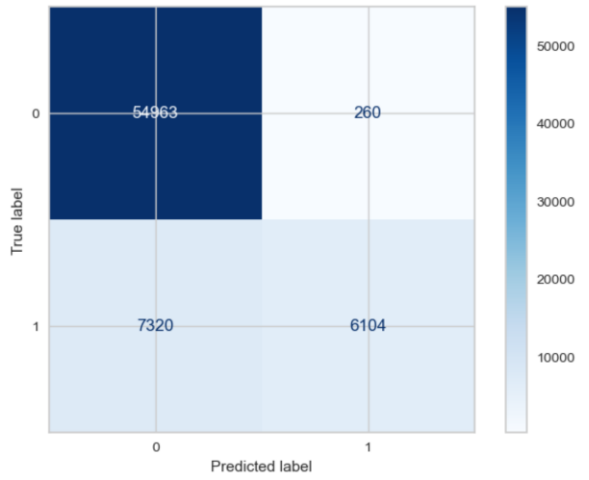




Accuracy : 0.8894634871152417
recall_score : 0.4653605482717521
precision_score : 0.9382697506758786
f1_score : 0.6221491883278558
AUC score : 0.905700793124041

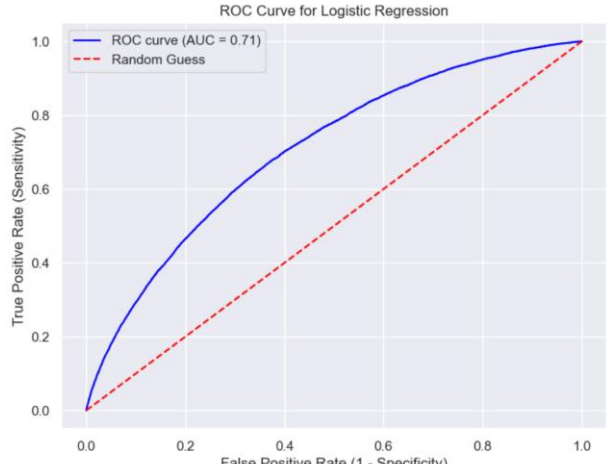
<Gradient Boosting>

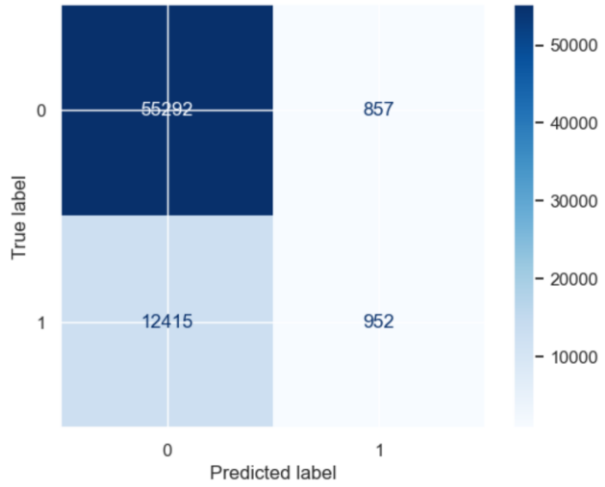




Accuracy : 0.8895800253470654
recall_score : 0.4547079856972586
precision_score : 0.9591451917033312
f1_score : 0.6169395593288862
AUC score : 0.9055647771295566

<Sample Code – Logistic Regression>

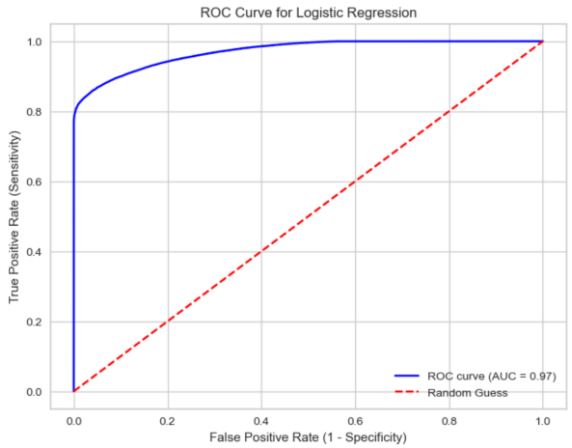


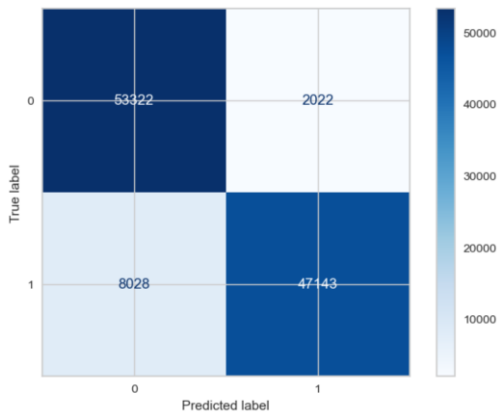


Accuracy : 0.8090799240462627 recall_score : 0.07122016907309045 precision_score : 0.5262576008844666 f1_score : 0.12546125461254612
AUC score : 0.7070128748255683

모델 성능 평가- 데이터 불균형 처리 후

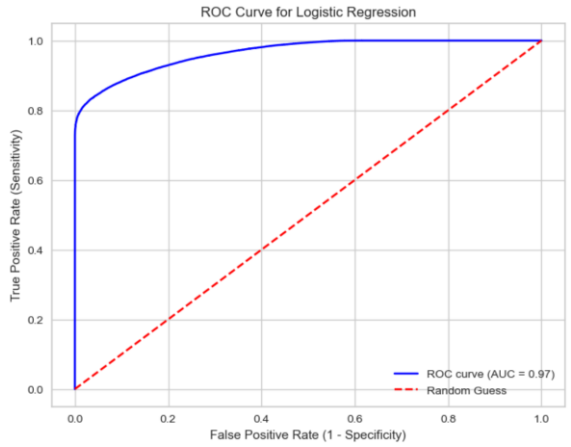
<LightGBM>

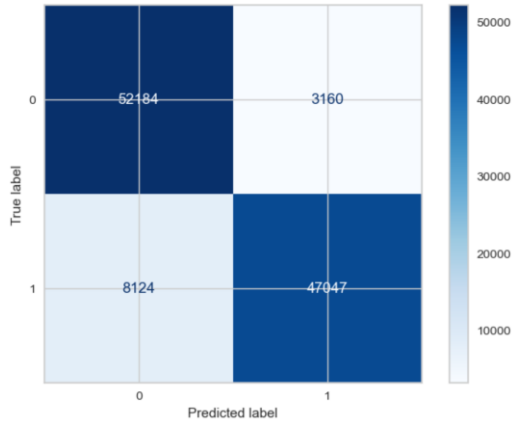




Accuracy : 0.909062118264489
recall_score : 0.854488771274764
precision_score : 0.9588731821417675
f1_score : 0.9036765833461126
AUC score : 0.9707442358547059

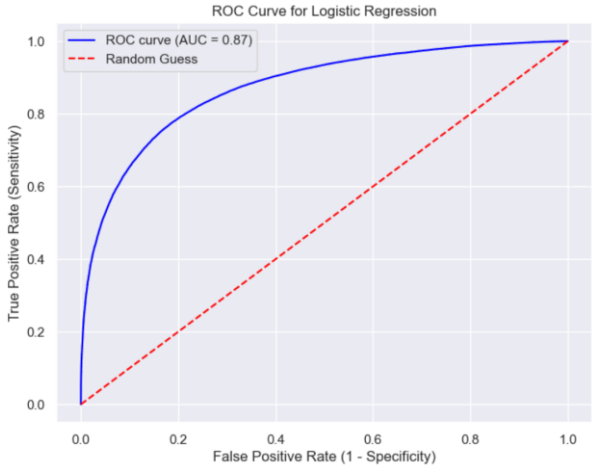
<Gradient Boosting>

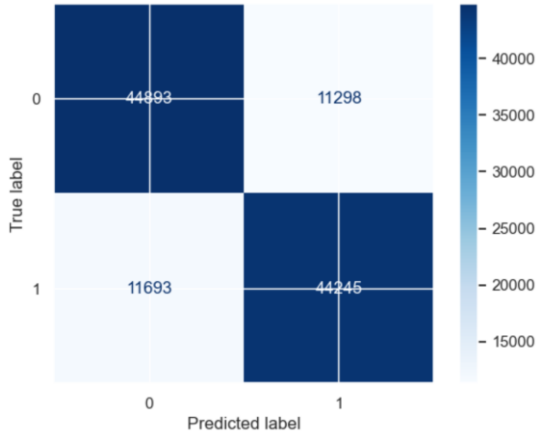




Accuracy : 0.8978962131837307
recall_score : 0.8527487266861213
precision_score : 0.9370605692433326
f1_score : 0.892918255613126
AUC score : 0.9652392772681433

<Sample Code – Logistic Regression>





Accuracy : 0.7937554067190468
recall_score : 0.7890342879616719
precision_score :0.7958061375356099
f1_score : 0.7924057450628367
AUC score : 0.8732325264822293

모델 성능 평가- 종합

(1) 데이터 불균형 처리 전 – 모델 간 성능 비교

모델	Accuracy	Recall	Precision	F1	AUC
LightGBM	0.889	0.465	0.938	0.622	0.906
Gradient Boosting	0.890	0.455	0.960	0.617	0.906
Logistic Regression	0.809	0.071	0.526	0.125	0.707

LightGBM / Gradient Boosting
→ 데이터 불균형 상황에서도 전체 분류 능력(AUC)은 매우 높다.
→ 하지만 부도(1) Recall이 0.45~0.47 수준으로, 실제 부도의 절반 이상을 놓치고 있다.
→ Precision은 0.94~0.96으로 부도라고 판단한 건 대부분 진짜 부도이지만, 탐지율 개선이 필요하다.

Logistic Regression
→ 데이터 불균형의 영향을 가장 심하게 받는다. Recall 0.071 Precision 0.526, F1 Score 0.125 수준으로 실제 부도를 거의 잡지 못한다.

(2) 데이터 불균형 처리 후 – 모델 간 성능 비교

모델	Accuracy	Recall	Precision	F1	AUC
LightGBM	0.909	0.854	0.959	0.904	0.971
Gradient Boosting	0.900	0.853	0.937	0.892	0.965
Logistic Regression	0.794	0.789	0.796	0.792	0.873

LightGBM
→ 전반적인 모든 지표에서 가장 우수한 성능을 보인다.
→ Recall이 0.854로 높아 부도 탐지에 탁월하고, Precision도 0.959로 오탐률도 낮다.

Gradient Boosting
→ Recall은 LightGBM과 비슷(0.853)이지만, FP가 조금 많아 Precision이 다소 낮다(0.937).
→ F1-score도 0.892 수준으로 전체적으로 안정적이지만 LightGBM에는 소폭 밀린다.

Logistic Regression
→ 비선형 관계를 충분히 반영하지 못해 AUC와 Precision 모두 트리 기반 모델보다 낮다.
→ 간단한 기준선 모델로는 적절하지만 실무 적용에는 한계가 있다.

(3) 데이터 불균형 처리 전/후 모델별 성능 비교

LightGBM
→ Recall이 0.465 → 0.854으로 약 2배 가까이 증가, 부도 대출을 놓치지 않는 탐지력이 크게 향상됐다.
→ Precision도 0.938 →0.959 로 향상되어 거짓양성(False Positive)도 줄어들었다.
→ F1-score가 0.622 → 0.904으로 크게 향상, 불균형 문제로 인해 놓치던 부실 대출을 균형 있게 잡기 시작했다.
→ AUC도 0.906 → 0.971로 개선되어 전반적인 분류 능력이 향상됐다.
→ LightGBM은 불균형 처리 후 전반적인 성능이 고르게 개선, 특히 Recall 향상이 두드러진다.

Gradient Boosting
→ Recall이 0.455 → 0.853으로 크게 향상, 부실 대출 탐지 능력이 강화된다.
→ Precision은 0.960 → 0.937로 약간 하락했지만 여전히 안정적이다.
→ F1-score는 0.617 → 0.892로 급상승했다.
→ AUC도 0.906 →0.965로 상승하여 분류 능력이 개선됐다.
→ Gradient Boosting은 Recall과 Precision을 동시에 개선, 특히 F1-score 향상이 두드러진다.

Logistic Regression
→ Recall이 0.071 → 0.794으로 극적으로 상승, 부실 대출을 거의 탐지하지 못하던 모델이 제대로 기능하기 시작했다.
→ Precision도 0.526 → 0.796으로 향상, 잘못된 부도 예측 비율도 크게 줄어들었다.
→ F1-score가 0.125 → 0.928로 가장 극적인 개선을 보였다.
→ AUC도 0.707 → 0.873로 향상되어 모델 분류 능력 자체가 개선됐다.
→ Logistic Regression은 단순 모델임에도 불균형 처리만으로 성능이 급격히 개선, 특히 Recall과 F1이 대폭 향상됐다.

(4) 결론

데이터 불균형 문제 해결의 중요성
→ 원본 데이터는 정상(0)과 부도(1) 비율이 불균형하여, 불균형 처리 전에는 모든 모델에서 부도 탐지 Recall이 낮고, Logistic Regression의 경우 Recall이 0.07 수준에 불과했다. SMOTE를 활용한 오버샘플링을 통해 부도 데이터의 비중을 조정하자, 모든 모델 에서 Recall과 F1-score가 크게 향상되었다.

모델 성능 비교 결과
→ 불균형 처리 후 LightGBM과 Gradient Boosting은 Precision·Recall 모두 높은 수준을 보여 부도 탐지 성능이 크게 개선되었고, F1-score가 각각 0.904, 0.892로 가장 우수했다. Logistic Regression 또한 불균형 처리만으로 Recall이 0.071 → 0.789으로 극적으로 향상되어 baseline 모델로서의 역할을 수행할 수 있게 되었다.

최종 모델 선정
→ 전반적인 지표(Recall, Precision, F1, AUC)를 종합했을 때, LightGBM 모델이 가장 균형 잡힌 성능을 보여주고 학습 속도와 예측 효율성 측면에서도 우수해, 실제 금융 현장 적용 시에도 확장성과 실용성이 높을 것으로 기대돼 부실 대출 예측에 가장 적합한 모델로 판단된다.