응용데이터애널리틱스

프로젝트 발표

✳

TEAM1

김수빈

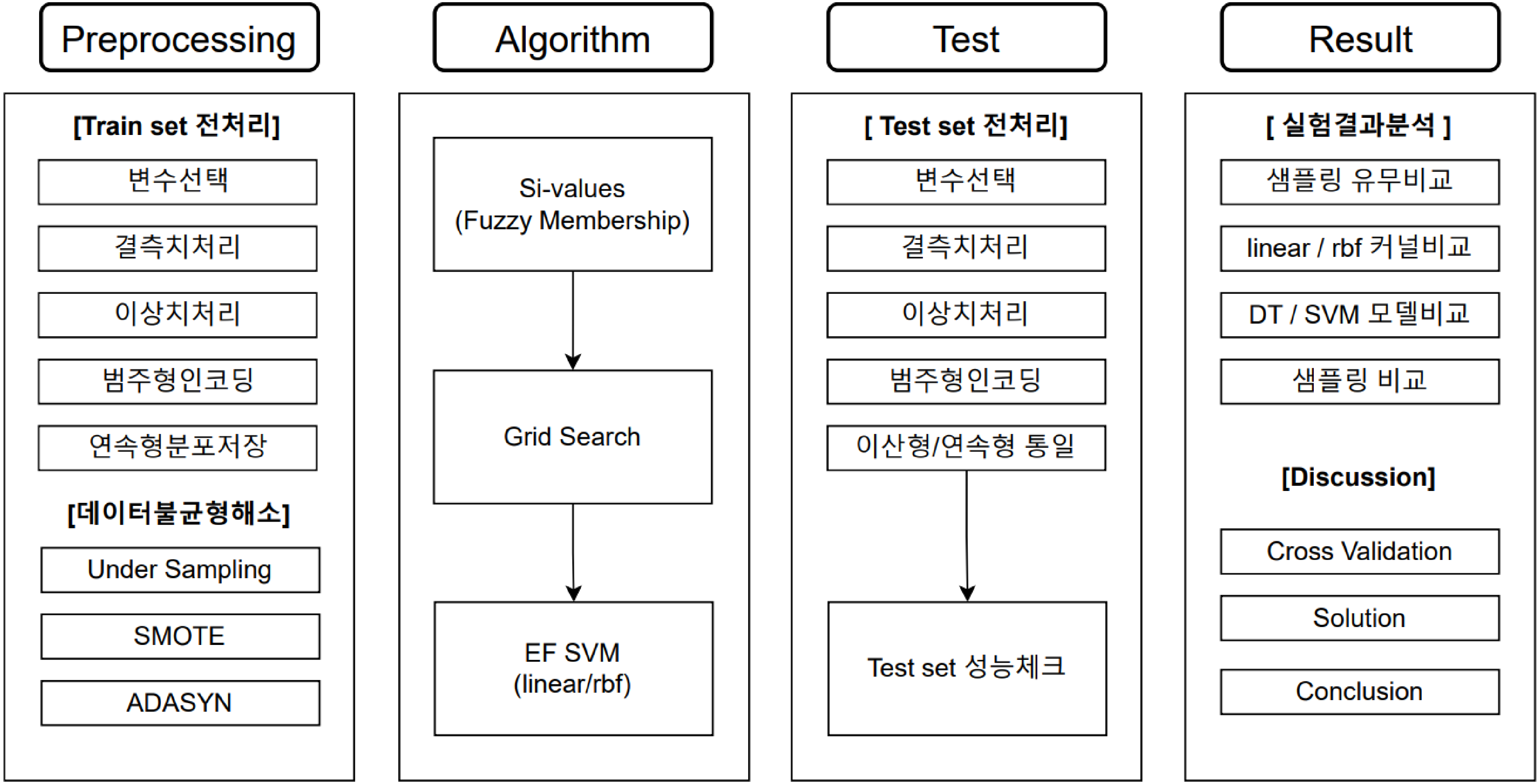
김주환

박주연

이채희

Contents

01 Executive Summary



02 Feature Engineering

1. Choosing Suitable Variables

1. **판단 시점 이전에 얻을 수 있는 데이터인가?**

   - i.e. P2P loan을 평가하기 이전에 주어진 데이터인가?

2. **중복되지 않는 데이터인가?**

   - 보다 Granularity가 높은 데이터 이용

   ex) ‘grade’ 대신 ‘subgrade’ 사용

1. Choosing Suitable Variables

Feature	Description
loan_amnt	The listed amount of the loan applied for by the borrower. If at some point in time, the credit department reduces the loan amount, then it will be reflected in this value.
funded_amnt	The total amount committed to that loan at that point in time.
funded_amnt_inv	The total amount committed by investors for that loan at that point in time.
int_rate	Interest Rate on the loan
installment	The monthly payment owed by the borrower if the loan originates.
grade	LC assigned loan grade
sub_grade	LC assigned loan subgrade
emp_length	Employment length in years. Possible values are between 0 and 10 where 0 means less than one year and 10 means ten or more years.
home_ownership	The home ownership status provided by the borrower during registration or obtained from the credit report. Our values are: RENT, OWN, MORTGAGE, OTHER
annual_inc	The self-reported annual income provided by the borrower during registration.
verification_status	Indicates if income was verified by LC, not verified, or if the income source was verified
loan_status	Current status of the loan (Charged Off, Fully Paid)
purpose	A category provided by the borrower for the loan request.
dti	A ratio calculated using the borrower's total monthly debt payments on the total debt obligations, excluding mortgage and the requested LC loan, divided by the borrower's self-reported monthly income.
delinq_2yrs	The number of 30+ days past-due incidences of delinquency in the borrower's credit file for the past 2 years
inq_last_6mths	The number of inquiries in past 6 months (excluding auto and mortgage inquiries)
open_acc	The number of open credit lines in the borrower's credit file.
pub_rec	Number of derogatory public records
revol_bal	Total credit revolving balance
revol_util	Revolving line utilization rate, or the amount of credit the borrower is using relative to all available revolving credit.
total_acc	The total number of credit lines currently in the borrower's credit file
initial_list_status	The initial listing status of the loan. Possible values are – W, F
total_pymnt	Payments received to date for total amount funded
total_pymnt_inv	Payments received to date for portion of total amount funded by investors
total_rec_prncp	Principal received to date
total_rec_int	Interest received to date
total_rec_late_fee	Late fees received to date
recoveries	post charge off gross recovery
collection_recovery_fee	post charge off collection fee
last_pymnt_amnt	Last total payment amount received
collections_12_mths_ex_med	Number of collections in 12 months excluding medical collections
acc_now_delinq	The number of accounts on which the borrower is now delinquent.
chargeoff_within_12_mths	Number of charge-offs within 12 months
delinq_amnt	The past-due amount owed for the accounts on which the borrower is now delinquent.
pub_rec_bankruptcies	Number of public record bankruptcies
tax_liens	Number of tax liens

Feature
loan_amnt
funded_amnt
funded_amnt_inv
int_rate
installment
grade
sub_grade
emp_length
home_ownership
annual_inc
verification_status
loan_status
purpose
dti
delinq_2yrs
inq_last_6mths
open_acc
pub_rec
revol_bal
revol_util
total_acc
initial_list_status
total_pymnt
total_pymnt_inv
total_rec_prncp
total_rec_int
total_rec_late_fee
recoveries
collection_recovery_fee
last_pymnt_amnt
collections_12_mths_ex_med
acc_now_delinq
chargeoff_within_12_mths
delinq_amnt
pub_rec_bankruptcies
tax_liens

02 Feature Engineering

2. Dealing with Missing Values

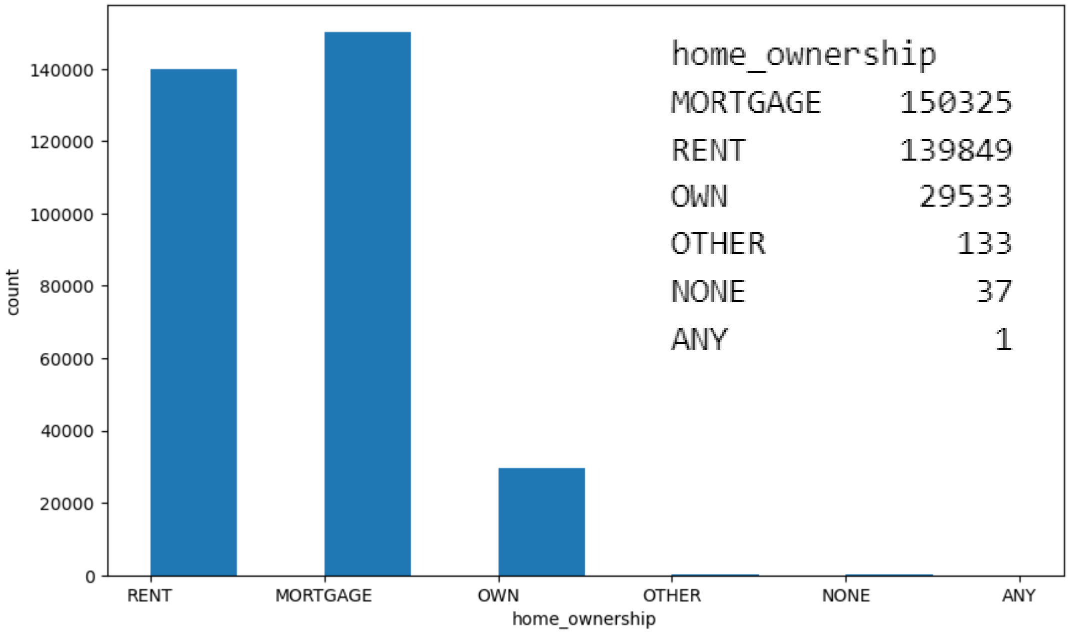
```
<Missing Value Count>

emp_length                  16269
revol_util                    213
collections_12_mths_ex_med     54
chargeoff_within_12_mths       54
pub_rec_bankruptcies          663
tax_liens                      38
```

- emp_length 는 가장 개수가 많았던 ‘10+ years’로 대체 (‘10+ years’ 가 전체의 30% 차지)
- 다른 결측치들을 중앙값으로 대체

```python
# 결측치를 중앙값으로 대체
df_1['collections_12_mths_ex_med']=df_1['collections_12_mths_ex_med'].replace(np.nan, df_1['collections_12_mths_ex_med'].median())
df_1['chargeoff_within_12_mths']=df_1['chargeoff_within_12_mths'].replace( np.nan, df_1['chargeoff_within_12_mths'].median())
df_1['pub_rec_bankruptcies']=df_1['pub_rec_bankruptcies'].replace(np.nan, df_1['pub_rec_bankruptcies'].median())
df_1['tax_liens']=df_1['tax_liens'].replace(np.nan, df_1['tax_liens'].median())
df_1['revol_util']=df_1['revol_util'].replace(np.nan, df_1['revol_util'].median())
# NaN 값을 '10+ years'로 대체
df_1['emp_length'].fillna('10+ years', inplace=True)
```

3. Dealing with Outliers

home_ownership	The home ownership status provided by the borrower during registration or obtained from the credit report. Our values are: RENT, OWN, MORTGAGE, OTHER

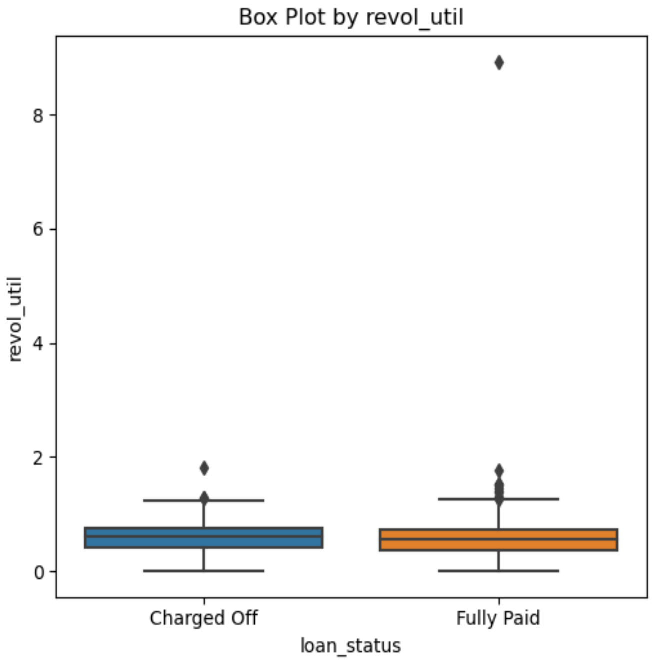


- home_ownership은 'RENT', 'OWN', 'MORTGAGE', 'OTHER' 4개 값들을 가지기 때문에 이외의 값들은 drop

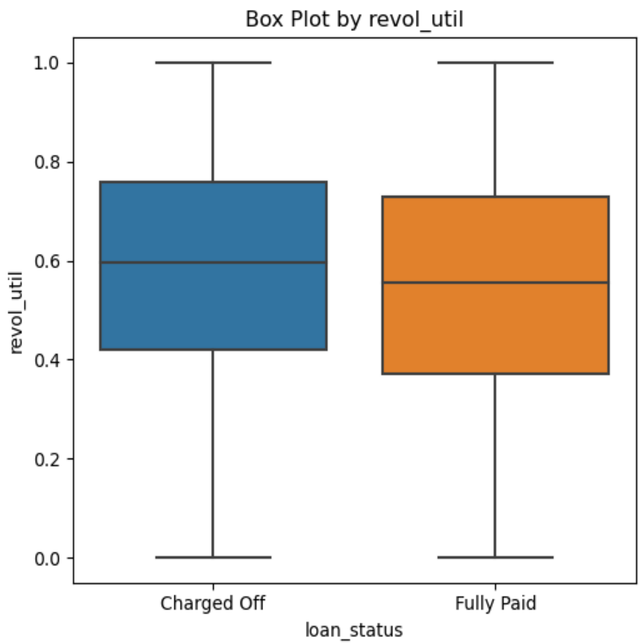
```
df_2 = df_2[df_2['home_ownership'].isin(['MORTGAGE', 'OTHER', 'OWN', 'RENT'])]
```

3. Dealing with Outliers

revol_util	Revolving line utilization rate, or the amount of credit the borrower is using relative to all available revolving credit.

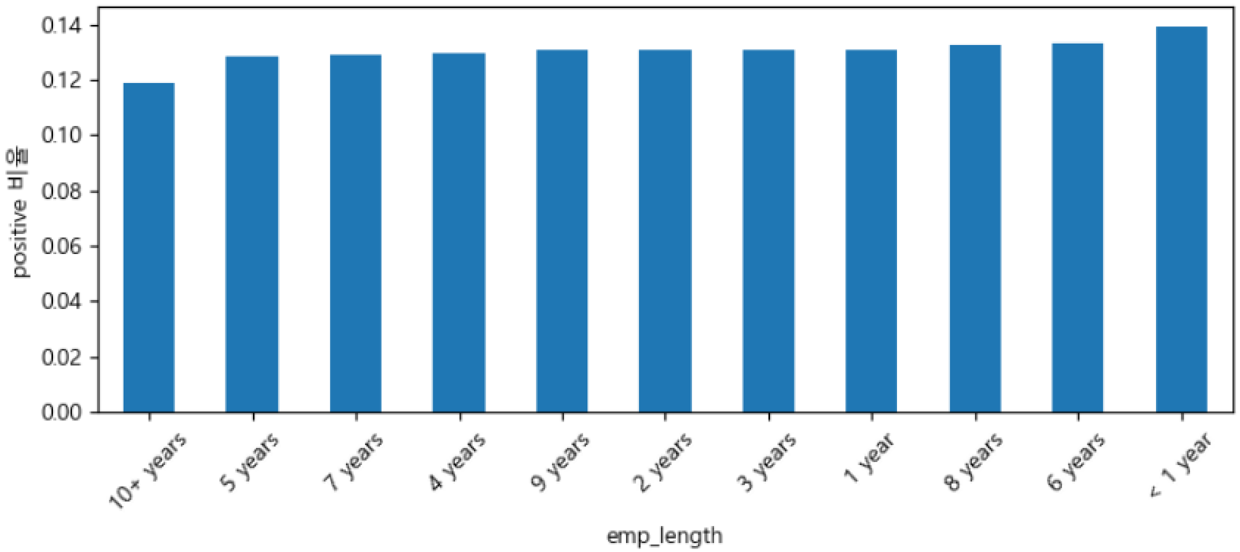


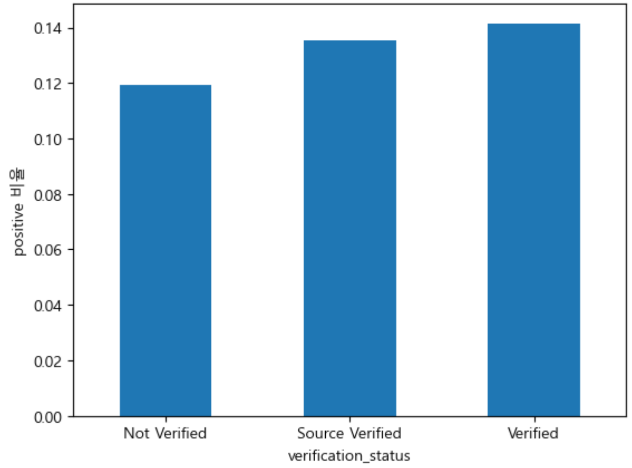
- Revol_util은 리볼빙 이용률이므로 1(100%)를 넘길 수 없다고 판단하여 1이 넘는 값들은 이상치로 제거




```
df_2 = df_2[df_2['revol_util'] <= 1.00]
```

4. Encoding Categorical Variable – Label Encoding

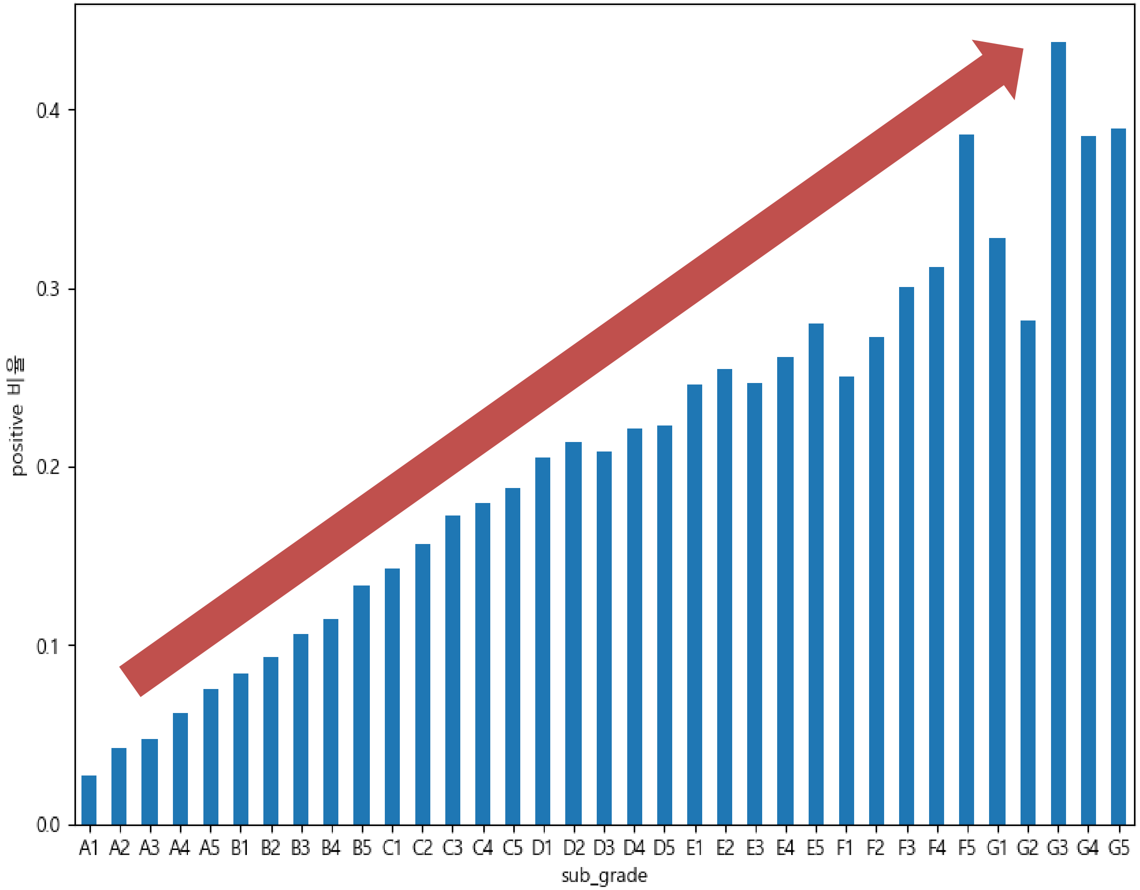




- emp_length와 verification_status는 5% 이내에서 변동
  → Label encoding

```
selected_list=['emp_length', 'verification_status']

label_encoders = {}
for column in selected_list:
    le = LabelEncoder()
    df_3[column] = le.fit_transform(df_3[column])
    label_encoders[column] = le
```

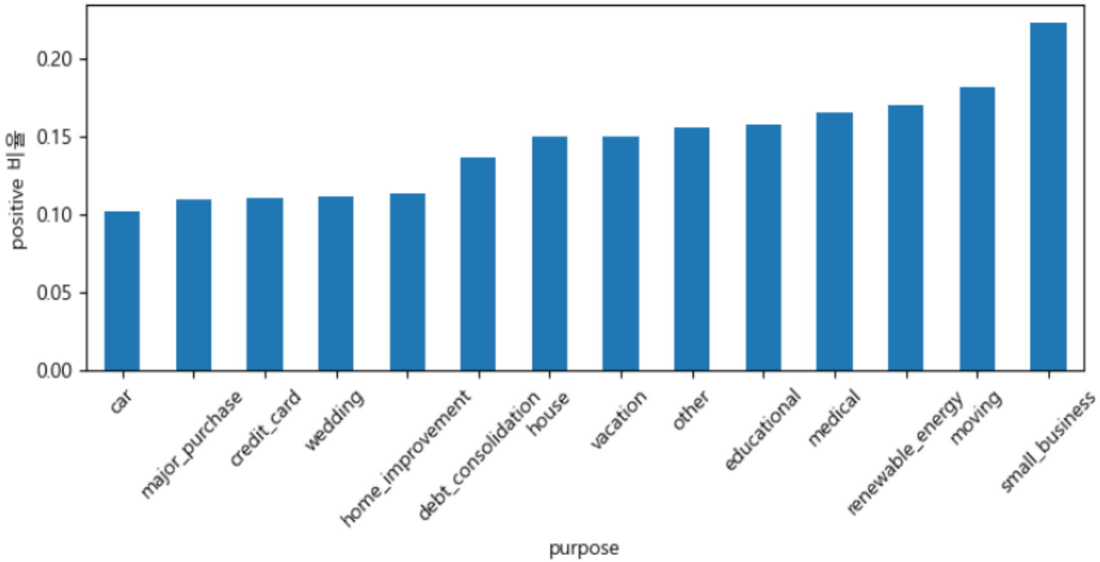
4. Encoding Categorical Variable – Ordinal Encoding

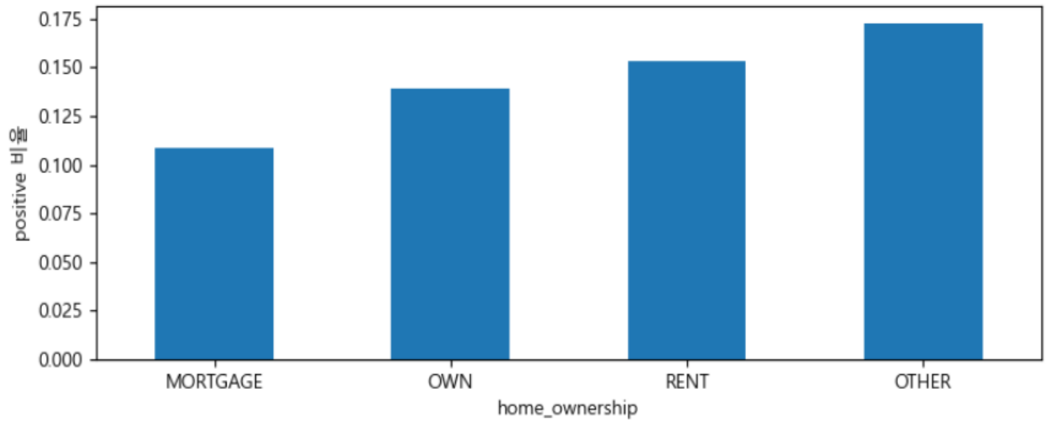


- sub-grade가 높을수록 positive 비율도 낮다
  → Ordinal Encoding

```
sg_order = sorted(df_3['sub_grade'].unique())
encoder_3 = OrdinalEncoder(categories = [sg_order])
df_3['sub_grade'] = encoder_3.fit_transform(df_3[['sub_grade']])
```

4. Encoding Categorical Variable – Ordinal Encoding





- Purpose와 home_ownership의 경우 유의미한 차이가 났다고 판단

  → Ordinal encoding

```
purpose_order = ['car', 'major_purchase', 'credit_card', 'wedding',
                 'home_improvement', 'debt_consolidation', 'house',
                 'vacation', 'other', 'educational', 'medical',
                 'renewable_energy', 'moving', 'small_business']
encoder_1 = OrdinalEncoder(categories=[purpose_order])
df_3['purpose'] = encoder_1.fit_transform(df_3[['purpose']])
```

```
home_order = ['OTHER', 'RENT', 'OWN', 'MORTGAGE']
home_order = home_order[::-1]
encoder_2 = OrdinalEncoder(categories=[home_order])
df_3['home_ownership'] = encoder_2.fit_transform(df_3[['home_ownership']])
```

1. Model

샘플링 방식에 따라 생성한 데이터 셋에 대해 다음 예측 모델 학습

1. Soft SVM – linear, rbf kernel
2. Entropy Fuzzy SVM– linear, rbf kernel
3. Scikit learn: Decision Tree, softSVM (Benchmark)

03 Algorithm

2. Hyperparameter

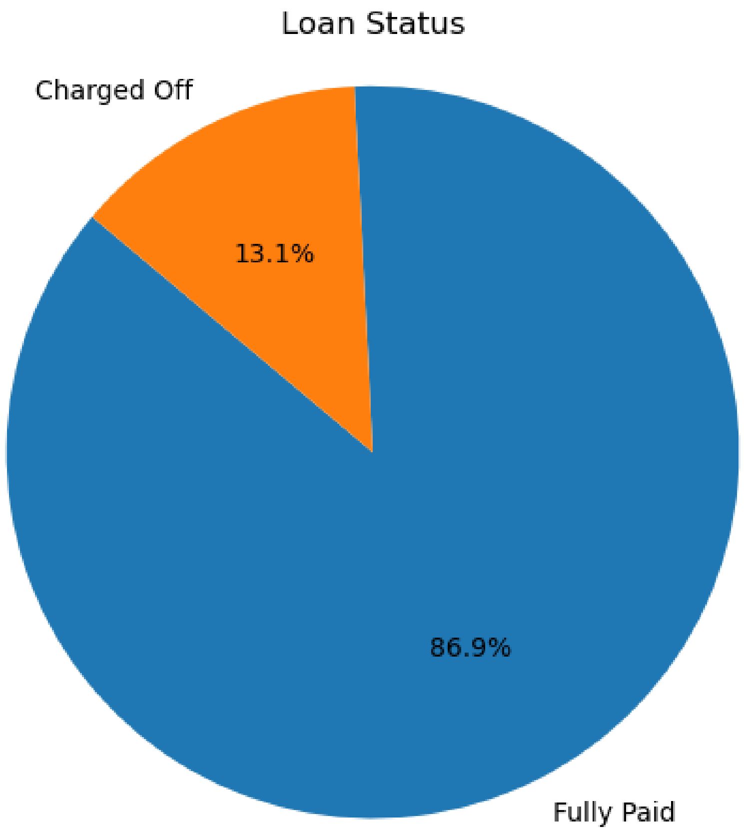
- 기본적인 hyperparameter tunning 시에는 grid search를 사용함
- β 는 다음과 같은 boundary 를 가짐

$$0 \leq \beta \leq \frac{1}{m-1}$$

- β = [$\frac{1}{4m}$, $\frac{1}{5m}$, $\frac{1}{10m}$]
- C = [0.1, 1 ,10]
- Gamma = [0.01, 2]
- m = [5, 10]
- Train size = 2500, validation size = 2000, test size = 2000

(테스트셋에서 개수는 이상치 제거로 정확히 2000은 아님.)

3. Sampling



- Positive class 가 약 13%인 불균형 데이터
- Computation time을 고려하여 일부 데이터만 이용해야 하기 때문에 Under sampling 사용
- 성능 비교를 위해 Over sampling도 사용
- 해당 프로젝트에서 사용한 sampling 방법

  1. Under Sampling
  2. SMOTE
  3. ADASYN

1. 샘플링 유무비교

- 불균형을 해소하지 않은 경우 테스트 셋에서 전부 음성으로 예측함.
- 하이퍼파라미터 추정을 더 넓게 하면 양성예측이 나올 수 있음.

NONE				
	EFSVM		SoftSVM	
	linear	rbf	linear	rbf
Accuracy	0.8705	0.8705	0.8705	0.8705
Precision	Nan	Nan	Nan	Nan
Recall	0.0	0.0	0.0	0.0

04 Result & Discussions

1. 샘플링 유무비교

```python
from sklearn.svm import SVC
model = SVC(kernel='linear', probability=False)
model.fit(X_train_norm, y_train)
y_pred = model.predict(X2.values)

accuracy = accuracy_score(Y2.values, y_pred)
recall = recall_score(Y2.values, y_pred)
precision = precision_score(Y2.values, y_pred)

print("accuracy :", accuracy)
print("precision", precision)
print("recall :", recall)
```

Executed at 2024.06.08 20:51:06 in 2s 259ms

```
accuracy : 0.870546914199699
precision 0.0
recall : 0.0
```

```python
from sklearn.svm import SVC
model = SVC(kernel='rbf', probability=False)
model.fit(X_train_norm, y_train)
y_pred = model.predict(X2.values)

accuracy = accuracy_score(Y2.values, y_pred)
recall = recall_score(Y2.values, y_pred)
precision = precision_score(Y2.values, y_pred)

print("accuracy :", accuracy)
print("precision", precision)
print("recall :", recall)
```

Executed at 2024.06.08 20:51:35 in 333ms

```
accuracy : 0.870546914199699
precision 0.0
recall : 0.0
```

2. Linear / rbf 커널비교

EF SVM		Accuracy	Precision	Recall	f1
linear	Under	0.1339	0.13	1.0	
	SMOTE	0.1786	0.1349	0.9883	
	ADASYN	0.1495	0.1317	0.9961	
rbf	Under	0.6201	0.2026	0.6589	0.3097
	SMOTE	0.6392	0.1804	0.7286	0.2909
	ADASYN	0.5007	0.1747	0.7674	0.2857

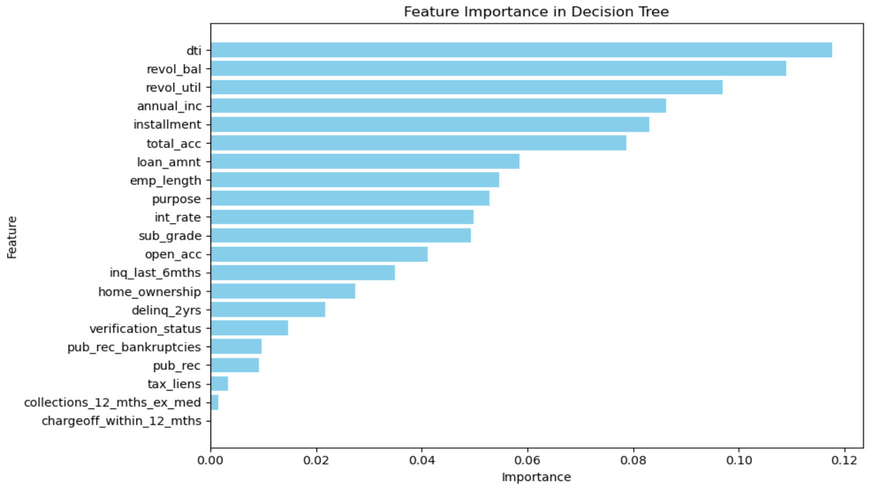
Soft SVM		Accuracy	Precision	Recall	f1
linear	Under	0.1339	0.13	1.0	
	SMOTE	0.1786	0.1349	0.9883	
	ADASYN	0.1495	0.1317	0.9961	
rbf	Under	0.6196	0.2023	0.6589	0.3095
	SMOTE	0.5369	0.1806	0.7287	0.2912
	ADASYN	0.5022	0.1752	0.7674	0.2863

3. DT / SVM 모델비교

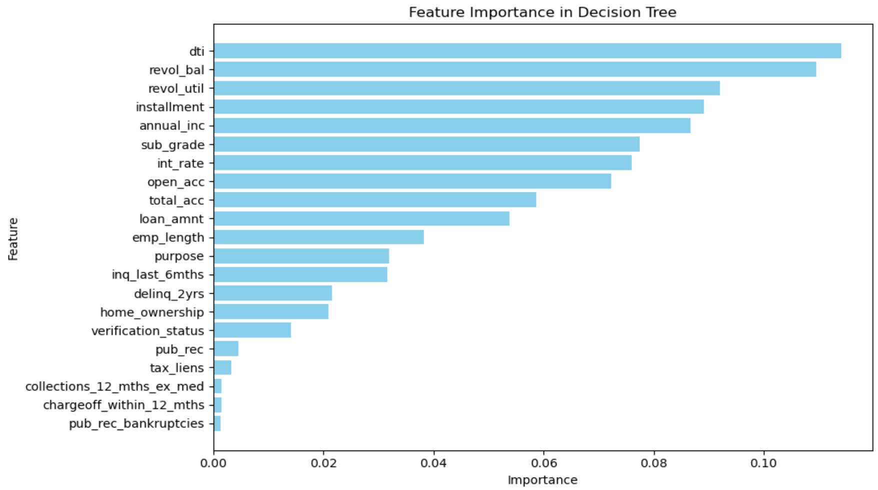
F1	Decision Tree	Soft SVM(scikit)	Soft SVM	EF SVM
None	0.1565	nan	nan	nan
Under	0.2063	0.3052	0.3095	0.3097
SMOTE	0.1798	0.2651	0.2912	0.2909
ADASYN	0.2012	0.2663	0.2863	0.2857

4. 샘플링 비교

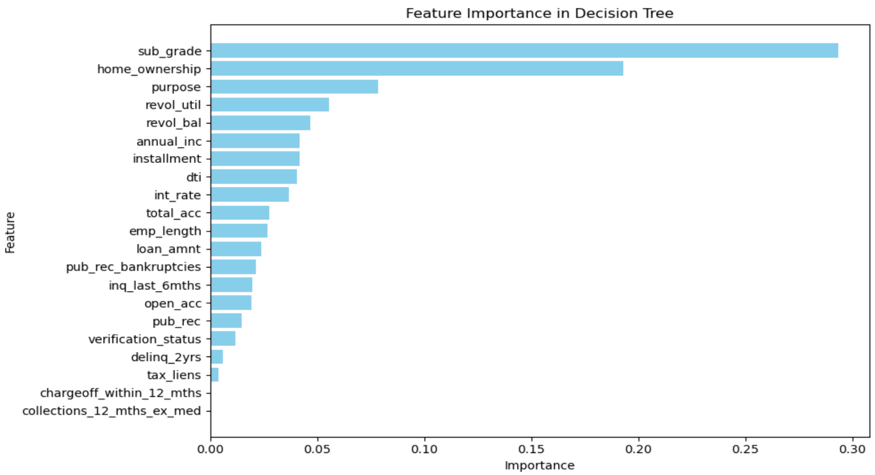
NONE



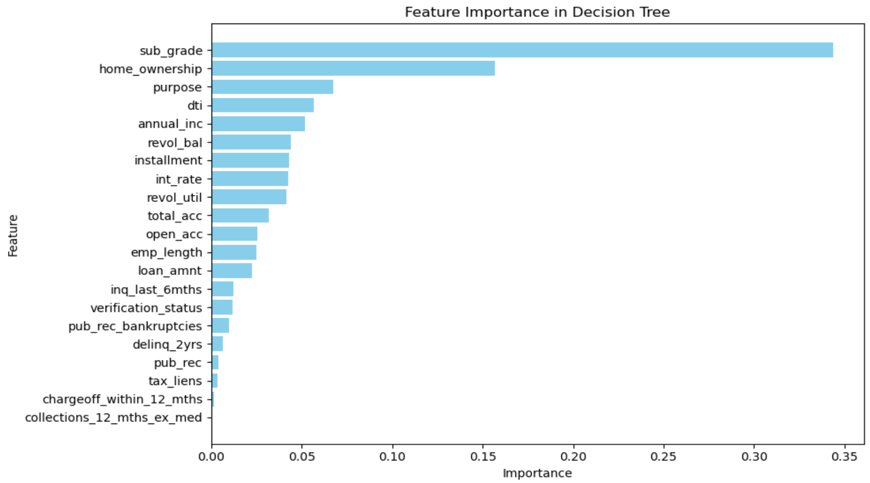
Under



SMOTE



ADASYN



5. Cross Validation

- Sampling을 단 한번 한 것으로 모형 Train 한 것이 과연 옳은가?
- Soft SVM (scikit) / rbf커널 / SMOTE

Cross Validation	1	2	3	4	5
Accuracy	0.6455	0.616	0.63	0.6265	0.64
Precision	0.6239	0.5883	0.6119	0.6084	0.5997
Recall	0.7710	0.7419	0.7622	0.7466	0.7798
F1-score	0.6393	0.6101	0.6227	0.6206	0.6337

6. Solution

- Sampling을 단 한번 한 것으로 모형 Train 한 것이 과연 옳은가?
- Soft SVM (scikit) / rbf커널 / SMOTE or Under

C=1, gamma=scale	model1	model2	model3	soft voting
F1(SMOTE)	0.2667	0.2588	0.2838	0.2796
F1(Under)	0.3174	0.3041	0.3133	0.3248

1. 전체 소결

(1) 프로젝트 목적

- P2P대출 데이터를 활용하여 대출 승인 여부 파악

(2) 결과 요약

- 데이터 불균형 해소는 매우 중요한 Task.
- 하나의 train set에서 얻은 초평면만으로는 설명력이 부족함. -> 확률적 SVM 앙상블 필요.

(3) 추가 개선 방안

- EF SVM 의 파라미터를 더 다양하게 찾아 모델 선에서 불균형을 해소할 필요가 있음.
- Optuna: 베이지안 최적화 방법을 활용한 더 효율적인 하이퍼 파라미터 공간 탐색.
- SMOTE / ADASYN 등의 oversampling 방법은 customizing 될 필요가 있음.

Evaluation

- Accuracy : 전제 데이터 중 맞게 분류된 데이터 수

$$\text{Accuracy} = \frac{TP + TN}{TP + FP + FN + TN}$$

- Precision : Positive로 예측한 데이터 중 맞게 예측된 데이터 수

$$\text{Precision} = \frac{TP}{TP + FP}$$

- Recall : Positive인 데이터 중 Positive로 올바르게 예측된 데이터 수

$$\text{Recall} = \frac{TP}{TP + FN}$$

- F1-Score : Precision 과 Recall 의 조화평균(harmonic average, low value 에 더 가중치를 줌.)

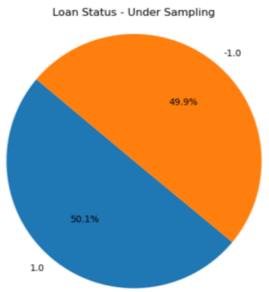
$$\text{F1 Score} = \frac{2 * (Precision * Recall)}{Precision + Recall}$$

→ Imbalanced classification 문제에서는 적은 class를 맞추는 일이 중요
따라서 재현율(Recall)과 정밀도(Precision) 모두를 고려하는 F1 score를 가지고
모델을 평가하는 것이 좋음.

Sampling

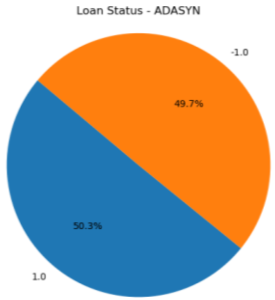
1. Under Sampling

- positive class의 크기만큼 무작위로 negative class 샘플링.



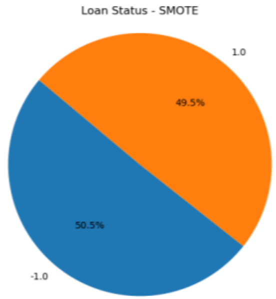
2. SMOTE (Synthetic Minority Over-sampling Technique)

- 랜덤으로 샘플링한 20000개의 데이터에 대해 knn을 사용하여 점들을 내분하는 positive class의 새로운 합성데이터를 생성.



3. ADASYN

- 랜덤으로 샘플링한 20000개의 데이터에 대해 학습난이도를 고려한 가중치를 두어 새로운 합성데이터를 생성.



→ Computation time 을 줄이기 위해 train set은 2500개, validation set은 2000개를 샘플링하여 사용.

Decision Tree

	Decision Tree			
	Under	SMOTE	ADASYN	None
accuracy	0.5660	0.6889	0.6969	0.7120
precision	0.1348	0.1365	0.154	0.1256
recall	0.4341	0.2636	0.2984	0.2054
F1-score	0.2063	0.1798	0.2012	0.1565

Under Sampling

Under Sampling				
	EFSVM		SoftSVM	
	linear	rbf	linear	rbf
Best parameter	C : 0.1 M : 5 beta : 0.05	C : 0.1 Gamma : 0.01 M : 5 Beta : 0.05	C : 10	C : 10 gamma : 0.01
Best score	0.3433	0.6053	0.3433	0.6053
Accuracy	0.1339	0.6201	0.1339	0.6196
Precision	0.1300	0.2026	0.1300	0.2023
Recall	1.0	0.6589	1.0	0.6589
F1 Score		0.3097		0.3095

SMOTE

SMOTE				
	EFSVM		SoftSVM	
	linear	rbf	linear	rbf
Best parameter	C : 0.1 M : 5 beta : 0.05	C : 0.1 Gamma : 0.01 M : 5 Beta : 0.05	C : 10	C : 10 gamma : 0.01
Best score	0.4080	0.6392	0.4080	0.6392
Accuracy	0.1786	0.5363	0.1786	0.5368
Precision	0.1349	0.1804	0.1349	0.1805
Recall	0.9883	0.7286	0.9883	0.7286
F1 Score		0.2909		0.2912

ADASYN

ADASYN				
	EFSVM		SoftSVM	
	linear	rbf	linear	rbf
Best parameter	C : 0.1 M : 5 beta : 0.05	C : 10 Gamma : 0.01 M : 5 Beta : 0.05	C : 10	C : 1 gamma : 0.01
Best score	0.3524	0.6110	0.3524	0.6110
Accuracy	0.1495	0.5007	0.1495	0.5022
Precision	0.1317	0.1747	0.1317	0.1752
Recall	0.9961	0.7674	0.9961	0.7674
F1 Score		0.2857		0.2863