SKN5 4팀 모델 학습 결과서

1. 모델 선정

- 데이터의 전처리 후, LabelEncoder 를 통해 범주형 데이터의 라벨링 및 StandardScaler를 통한 정규화 후 다양한 머신러닝 모델을 활용해서 예측
- Boosting 계열의 모델이 잘 예측할 것 같다고 예상하여 Boosting 모델 4개 및 각자 다른 계열의 모델을 선정하여 예측 결과 비교

2. 모델 성능 비교 및 분석

1. Train_Test_split 진행

   ○ 추후 성능 예측 및 프로젝트 결론 내용 도출을 위해 Test, Validation, Train 데이터로 나눔

```
Test DataFrame:
Churn
0    848
1    165
Total 1013
```

```
Validation DataFrame:
Churn
0    673
1    138
Total 811
```

```
Train DataFrame:
Churn
0    2709
1     532
Total 3241
```

2. 초기 성능 확인

   ○ 우선 라벨링 및 정규화 후 초기 성능을 확인.

모델	원본데이터				
	recall		f1-score		
	0	1	0	1	accuracy
Logistic	0.98	0.31	0.93	0.45	0.87
DecisionTree	0.96	0.78	0.96	0.79	0.93
RandomForest	0.99	0.78	0.97	0.86	0.96
XGBoost	0.98	0.62	0.95	0.72	0.92
CatBoost	0.99	0.78	0.97	0.85	0.95
MLP	0.97	0.64	0.95	0.72	0.92
GradientBoosting	0.98	0.63	0.95	0.73	0.92
AdaBoost	0.98	0.59	0.95	0.69	0.91

   ○ 전체적인 F1 score 값은 높게 나타났으나, 클래스 1(이탈 고객) 에 대한 예측 값이 대체적인 모델에서 낮았음을 알 수 있음.

3. 1차 성능 개선

- 1차 성능 개선을 위해 Train 데이터의 오버 샘플링 여러 기법 시행

	오버샘플링 데이터				
	recall		f1-score		
모델	0	1	0	1	accuracy
Logistic	0.71	0.76	0.81	0.48	0.72
DecisionTree	0.97	0.82	0.97	0.82	0.94
RandomForest	0.99	0.83	0.98	0.87	0.96
XGBoost	0.88	0.89	0.93	0.73	0.89
CatBoost	0.97	0.91	0.98	0.89	0.96
MLP	0.87	0.87	0.92	0.7	0.87
GradientBoosting	0.89	0.89	0.93	0.74	0.89
AdaBoost	0.85	0.9	0.91	0.69	0.86

	SMOTE 데이터				
	recall		f1-score		
모델	0	1	0	1	accuracy
Logistic	0.72	0.7	0.81	0.46	0.72
DecisionTree	0.94	0.75	0.94	0.74	0.91
RandomForest	0.97	0.82	0.97	0.84	0.95
XGBoost	0.94	0.7	0.94	0.7	0.9
CatBoost	0.98	0.82	0.97	0.85	0.95
MLP	0.87	0.86	0.92	0.69	0.87
GradientBoosting	0.93	0.7	0.93	0.69	0.89
AdaBoost	0.92	0.78	0.93	0.71	0.89

	Borderline SMOTE 데이터				
	recall		f1-score		
모델	0	1	0	1	accuracy
Logistic	0.67	0.72	0.78	0.43	0.68
DecisionTree	0.95	0.79	0.95	0.77	0.92
RandomForest	0.97	0.81	0.97	0.83	0.94
XGBoost	0.94	0.73	0.94	0.72	0.9
CatBoost	0.98	0.82	0.97	0.85	0.95
MLP	0.84	0.8	0.9	0.63	0.84
GradientBoosting	0.93	0.74	0.94	0.72	0.9
AdaBoost	0.92	0.78	0.93	0.71	0.89

- F1 score 의 값들은 많은 모델에서 낮아졌으나, 클래스 1에 대한 F1 score 에 성능이 높아진 점에 있어 오버샘플링 기법을 채택하기로 함.

- 여러 오버 샘플링 기법 중 random oversampling 의 기법을 채택함.

4. 2차 성능 개선

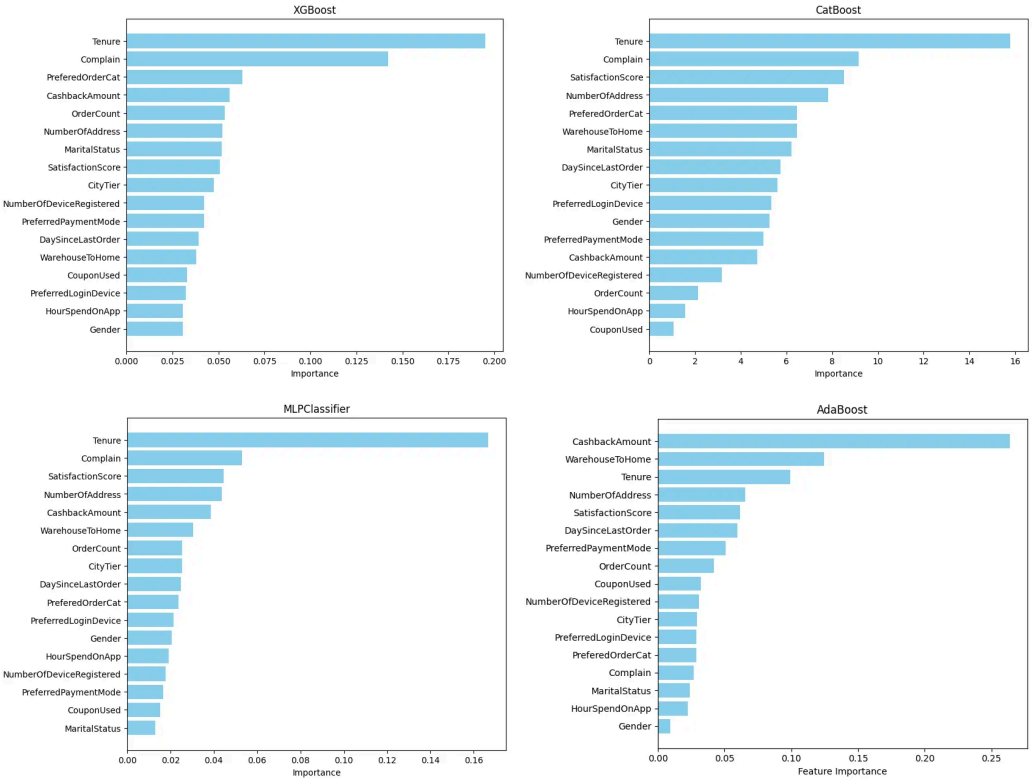
- 2차 성능 개선을 위해 GridSearchCV 사용

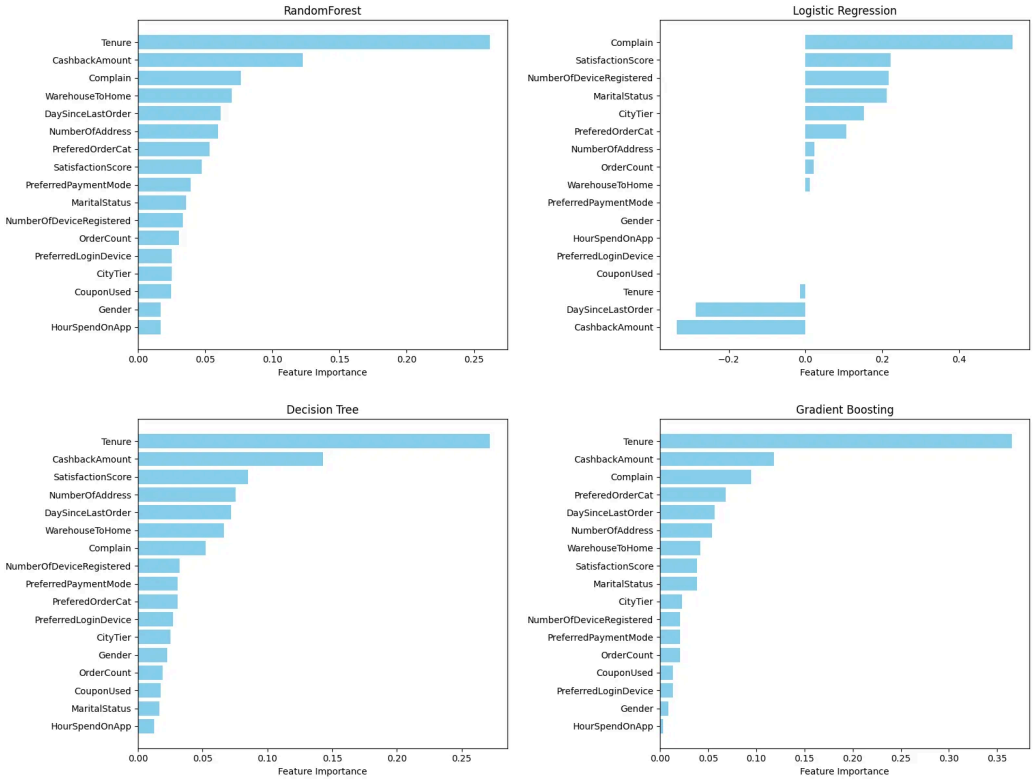
	GridSearchCV 데이터				
	recall		f1-score		
모델	0	1	0	1	accuracy
Logistic	0.71	0.77	0.81	0.49	0.72
DecisionTree	0.97	0.79	0.97	0.83	0.94
RandomForest	0.99	0.84	0.98	0.88	0.96
XGBoost	0.99	0.87	0.98	0.9	0.97
CatBoost	0.99	0.86	0.98	0.89	0.96
MLP	0.9	0.88	0.93	0.74	0.9
GradientBoosting	0.93	0.91	0.96	0.81	0.93
AdaBoost	0.98	0.78	0.97	0.83	0.95

- 모델 중에서 Boosting 계열의 모델 성능이 큰 폭으로 상승했다고 결론

5. 최종 모델 분석

- 가장 높은 성능을 보인 4개의 모델에 대한 feature importance





- 전체적인 Feature 를 확인했을 때 모든 모델에서 Tenure, CashbackAmount, Complain 칼럼의 Importance 가 대체적으로 높았음.

6. 최종 모델 선택

- GridSearchCV 진행 후 Boosting 계열의 모델 성능이 크게 상승해 해당 계열의 모델을 채택하고자 함.
- 해당 계열 중 가장 높은 성능을 보인 XGBoost 를 채택하기로 함.