인공지능 학습 결과서

팀명 : 3 팀

팀원 : 강민지 김지은 김효빈 안채연 홍혜원

목차

1. 데이터 개요 및 Baseline 모델

1.1 데이터셋 및 전처리 요약

- 데이터 크기 : 71,892 명 고객, 11 개 변수
- 타깃 변수 : churn (0 = 유지, 1 = 이탈)
- 주요 전처리 내용
    - id 컬럼 삭제
    - download_avg, upload_avg 결측 행 381 개 제거
    - subscription_age = -0.02 인 이상치 행 제거
    - 계약 상태·구독 조합·구독 연수에 대해 파생 변수 생성
        - contract_type (no_contract / expired / active)
        - subscription_age_group (0~1 년 / 1~3 년 / 3~5 년 / 5 년 이상)
        - subscription_label (none / tv / movie / both)
    - 트리 기반 모델용 데이터 : 라벨 인코딩 + 구간화 적용
    - 비트리 기반 모델용 데이터 : 원-핫 인코딩 + 연속형 유지 + 로그 변환(bill_avg, download_avg, upload_avg)

전처리의 목적은 데이터 품질 확보와 모델 특성에 맞는 인코딩/변환 전략 적용을 통하여 이탈 예측 모델의 성능과 해석력을 동시에 높이는 것이다.

1.2 머신러닝 Baseline 결과

초기 Baseline 에서는 단일 학습(Train 80% / Test 20%) 만 수행하여 성능을 확인하였다.

1.2.1 단일 학습 결과 (테스트 세트 평가)

모델	Accuracy	F1-score
Logistic Regression	0.9000	0.9087
Decision Tree	0.9727	0.9755
Random Forest	0.9702	0.9732
XGBoost	0.9842	0.9858
LightGBM	0.9716	0.9745
CatBoost	0.9480	0.9532

- 대부분 모델에서 Accuracy·F1-score 가 0.95 이상으로 매우 높게 측정됨
- 특히 XGBoost F1-score ≈ 0.986 으로 최상위 성능을 기록
- 그러나 데이터 불균형, 단일 분할의 편향 가능성을 고려할 때, 단일 테스트 결과만으로는 과대평가 위험이 존재. -> 교차검증 진행

1.2.2) 교차검증

모델	**Fold 1**	Fold 2	Fold 3	Fold 4	Fold 5	평균 F1-score
LightGBM	0.7161	0.6932	0.6618	0.5557	0.0025	0.5259
XGBoost	0.7161	0.6935	0.6541	0.5748	0.0208	0.5318
RandomForest	0.7158	0.6913	0.6699	0.5702	0.0002	0.5295

단일 학습에서는 높은 성능을 보였지만, 교차검증에서는 낮은 점수를 보여 데이터 불균형과. 분할 편향으로 인한 과대평가 가능성이 나타났다.

1.2.3)

교차검증 결과, XGBoost 와 LightGBM 이 단일 학습 대비 낮은 F1-score 를 보여 데이터 불균형의 영향을 받는 것으로 확인되었다. 이에 단일 학습 시 안정적으로 높은 성능을 보였고, 해석이 용이하며 과적합에 강한 Random Forest 를 베이스라인 모델로 선정하였다.

비교 대상으로는 Decision Tree 와 Logistic Regression 을 함께 사용하였다.
모든 모델은 기본 하이퍼파라미터 기준으로 성능을 비교하였다.

모델	**Accuracy**	Precision (0)	Recall (0)	F1-score (0)	Precision (1)	Recall (1)	F1-score (1)	Macro Avg (F1)	Weighted Avg (F1)
Decision Tree	0.91	0.90	0.89	0.89	0.92	0.92	0.92	0.91	0.91
Random Forest	0.94	0.92	0.94	0.93	0.96	0.93	0.94	0.94	0.94
Logistic Regression	0.92	0.89	0.94	0.91	0.95-	0.91	0.93	0.92	0.92

기본 설정 기준에서 Random Forest 는 Accuracy 와 F1-score 모두에서 가장 우수한 성능을 보여, 본 프로젝트의 머신러닝 베이스라인 모델로 선정하였다.

1.3 딥러닝 Baseline 결과 (sklearn MLPClassifier)

딥러닝 부분의 Baseline 은 sklearn MLPClassifier 로 구성하고,
GridSearchCV 로 최적 하이퍼파라미터를 탐색하였다.

- 탐색 범위
  - Hidden layer size : (128-64-32), (64-32), (64-32-16)
  - Activation : ReLU, Tanh
  - Learning rate : 0.001, 0.01, 0.1
- 최적 조합

- hidden_layer_sizes = (128, 64, 32)
- activation = 'relu'
- learning_rate_init = 0.001

Baseline MLP 의 성능은 아래와 같다.

모델 / 방법	구조	LR	F1	ROC-AUC	PR-AUC
sklearn MLP (Baseline)	128-64-32	0.001	0.94	0.96	0.98

초기 기준 성능이 이미 F1 ≈ 0.94, ROC-AUC ≈ 0.96 수준으로 높게 형성되어 있음을 확인하였다.

2. 하이퍼파라미터 설정 및 탐색 전략

2.1 트리 기반 머신러닝 공통 GridSearch 설정

트리 기반 모델 4 개(CatBoost, LightGBM, Random Forest, XGBoost)에 대해
동일한 하이퍼파라미터 후보군을 적용하여 공정하게 성능을 비교하였다.

2.1.1 데이터 및 학습 설정

- 데이터셋 : tree_model_preprocessed.csv
- 입력(X) : 타깃 churn 을 제외한 모든 변수
- 타깃(y) : churn
- Train / Test 비율 : 80 / 20
- 랜덤 시드 : 42
- Stratified Sampling 으로 클래스 비율 유지

2.1.2 공통 하이퍼파라미터 후보

하이퍼파라미터	후보 값
max_depth	4, 6, 8
learning_rate	0.01, 0.05, 0.1
n_estimators	300, 500, 800

- 목적 : 동일한 탐색 공간에서 각 모델의 최적 조합을 찾고,
  ROC-AUC 기준 성능을 직접 비교하기 위함

2.2 딥러닝 하이퍼파라미터 및 RandomSearch 설정

딥러닝 MLP 는 프레임워크별로 다음과 같은 설정과 RandomSearch 를 수행하였다.

- 탐색 대상
    - Hidden Layer Size :
      (16-8), (64-32-64), (16-8-16), (16-32-16), (16-8-4), (128-64-32), (64-32), (64-32-16)
    - Activation : ReLU, Tanh
    - Optimizer : Adam, SGD
    - Learning Rate : 0.001, 0.01, 0.1
    - Weight Decay : 0.0, 0.0001, 0.001
    - Batch Size : 32, 64, 128
    - Epochs : 10, 20, 30
    - Scaling : StandardScaler, MinMaxScaler, RobustScaler
- RandomSearch 설정
    - CV : 3
    - n_iter : 300
    - 최적 조합 예시
        - Optimizer : Adam
        - Layers : [128, 64, 32]
        - Activation : Tanh
        - Epoch : 30, LR : 0.001, Batch size : 64
        -

3. 머신러닝 모델링 결과

3.1 단일 학습 vs 교차검증 결과 (XGBoost 예시)

- 단일 학습(Test) F1-score ≈ 0.986
- 5-Fold 교차검증 F1-score ≈ 0.532

→ 단일 테스트 점수와 교차검증 간 큰 차이가 존재하며,
데이터 불균형·특정 분할 편향으로 인해 과적합 가능성이 드러났다.

따라서 이후 분석에서는 GridSearchCV + 교차검증 기반 ROC-AUC 를 기준으로
모델을 재선정하였다.

3.2 트리 기반 모델별 최적 성능

다음 표는 GridSearchCV 를 통해 ROC-AUC 기준 최적 하이퍼파라미터를 찾은 후, 테스트 세트에서 측정한 성능이다.

3.2.1 최적의 하이퍼파라미터 요약

모델	learning_rate	max_depth	n_estimators	기타
LightGBM	0.05	6	300	
CatBoost	0.05	6	800	-
XGBoost	0.01	8	800	tree_method='hist'
Random Forest	-	8	800	max_features='log2'

3.2.2 테스트 세트 최적의 성능

모델	Accuracy	F1-score	ROC-AUC
LightGBM	0.9380	0.9433	0.9714
CatBoost	0.9380	0.9433	0.9721
XGBoost	0.9378	0.9432	0.9725
Random Forest	0.9372	0.9429	0.9673

- 네 모델 모두 Accuracy ≈ 0.937, F1 ≈ 0.943, ROC-AUC ≥ 0.967 로 매우 우수한 성능
- ROC-AUC 기준으로는 XGBoost(0.9725)가 가장 높고, CatBoost, LightGBM 이 근소한 차이로 뒤를 잇는다.

3.3 Logistic Regression 결과

3.3.1 최적 하이퍼파라미터

하이퍼파라미터	최적 값
C	20
penalty	l2
solver	lbfgs

3.3.2 테스트 세트 성능

지표	값
Accuracy	0.9325
F1-score	0.9385
ROC-AUC	0.9579

- 트리 기반 모델에 비해 소폭 낮지만, Accuracy·F1·ROC-AUC 모두 0.93 이상으로 안정적이고 해석이 쉬운 모델이라는 장점이 있다.
- 원-핫 인코딩, 스케일링 등 연속형·범주형 전처리 품질에 크게 의존한다.

4. 딥러닝 모델링 결과

4.1 PyTorch MLP – 학습률(LR) 실험

- 구조 : Input → 128 → 64 → 32 → Output
- Activation : ReLU
- BatchNorm + Dropout(0.2) + EarlyStopping 적용
- Loss : BCEWithLogitsLoss, Optimizer : Adam
- Batch size : 32, Epoch : 100

4.1.1 학습률별 성능 비교

Learning Rate	Accuracy	F1-score	Precision	Recall
0.001	93.76	0.9432	0.9570	0.9298
0.01	93.75	0.9431	0.9572	0.9293
0.1	93.63	0.9419	0.9577	0.9266

- 학습률 변경에 따른 성능 차이가 거의 없음
- 데이터 전처리 및 MLP 구조가 이미 안정적으로 학습되고 있음을 의미

4.2 TensorFlow MLP – 은닉층 구조 실험

- Activation : ReLU(hidden), Sigmoid(output)
- Loss : Binary Crossentropy, Optimizer : Adam
- Batch size : 16, Epoch : 50, EarlyStopping 적용

4.2.1 Layer 구조별 성능

(대표 값만 요약)

Layer 구조 예시	Accuracy	F1-score	Precision	Recall
(16-8-1)	0.94	0.94	0.95	0.94
(32-16-1)	0.93	0.94	0.95	0.93
(64-32-1)	0.93	0.94	0.95	0.93
(16-32-16-1)	0.94	0.94	0.95	0.93
(64-32-64-1)	0.94	0.94	0.95	0.94
(16-8-4-1)	0.94	0.94	0.95	0.93
(128-64-32-1)	0.93	0.94	0.95	0.93

- Layer 깊이·구조 변화에도 성능 차이가 거의 없음
- 이는 학습률 실험과 마찬가지로,
  데이터 전처리와 기본 구조만으로도 충분히 수렴한다는 것을 시사

4.3 RandomSearch 기반 최적화 결과

최종적으로 TensorFlow MLP + RandomSearch 조합에서
다음과 같이 딥러닝 모델들의 종합 성능을 비교하였다.

모델 / 방법	구조	LR	F1	ROC-AUC	PR-AUC
sklearn MLP (Baseline)	128-64-32	0.001	0.94	0.96	0.98
PyTorch MLP	128-64-32	0.001	0.94	0.96	0.97
TensorFlow MLP + RandomSearch	64-32-64	0.001	0.94	0.96	0.97

- 프레임워크와 최적화 기법(RandomSearch)을 달리해도
  F1 ≈ 0.94, ROC-AUC ≈ 0.96 수준에서 성능이 거의 고정되었다.
- 즉, 이 문제에서는 복잡한 딥러닝 구조보다 데이터 전처리와 피처 품질이 더 영향력 있음을 확인.

5. 종합 성능 비교 및 결론

5.1 머신러닝 vs 딥러닝 성능 비교

- 머신러닝 (트리 기반)
  - ROC-AUC : 0.967 ~ 0.9725
  - F1-score : 0.942 ~ 0.943
  - 특히 XGBoost, CatBoost, LightGBM 이 최고 수준 성능
- 머신러닝 (Logistic Regression)
  - ROC-AUC : 0.9579
  - F1-score : 0.9385
  - 트리 기반 대비 약간 낮지만, 단순성과 해석력이 강점
- 딥러닝 (MLP 계열)
  - F1-score : 약 0.94
  - ROC-AUC : 약 0.96
  - 프레임워크(sklearn, PyTorch, TensorFlow)와
    하이퍼파라미터 변화를 주어도 성능 변화 폭이 크지 않음

→ 종합적으로는 트리 기반 머신러닝이 딥러닝보다 소폭 우수한 성능을 보였다.

5.2 인사이트 및 모델 선택 기준

1. 트리 기반 모델 vs Logistic Regression
    - 트리 기반 모델 4 개는 높은 ROC-AUC 와 F1-score 로
      이탈/유지 고객을 잘 구분함.
    - Logistic Regression 은 성능은 약간 낮지만,
      선형 해석 가능·변수 영향력 설명이 쉬워 실무 적용에 용이하다.
2. 딥러닝 실험의 의미
    - 학습률, Layer 수, 은닉 노드 변경 등 여러 시도를 했음에도
      성능 변화가 크지 않았다는 점에서, 모델 복잡도보다는 데이터 전처리와 피처 설계가
      핵심임을 확인.
3. 하이퍼파라미터 튜닝 효과
    - GridSearch 및 RandomSearch 를 통해 모델들을 공정한 조건에서 비교할 수 있었고,
      튜닝 전·후 성능이 안정적으로 상향되었음.
4. 최종 결론 및 실무 적용 시 권장 모델
    - 예측력(Performance)을 최우선으로 할 경우
      → XGBoost / CatBoost / LightGBM 중 하나를 선택
    - 설명력·단순성(Interpretability)을 중시할 경우
      → Logistic Regression 선택 가능
    - 딥러닝 MLP 는 머신러닝과 비슷한 수준의 성능을 내지만,
      이 문제에서는 추가 복잡도 대비 이득이 크지 않으므로
      데이터 확장·피처 추가 전까지는 보조 모델로 활용하는 것이 적절함.