네이버 부스트캠프 AI Tech 5 기 Project 2 랩업리포트

RecSys Team 3 : 렉돌이
강찬미, 박동연, 서민석, 이준영, 주혜인

1. Team Wrap-up Report

1-1. 프로젝트 개요

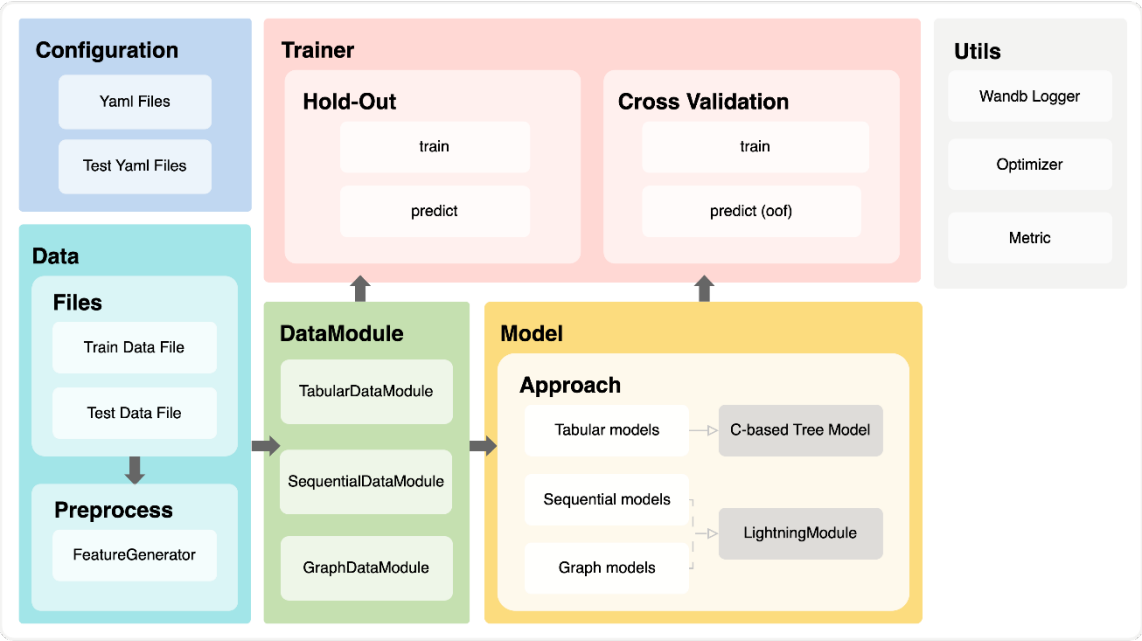
• 프로젝트 주제

시험은 학생이 얼마만큼 아는지 평가하는 좋은 방법이다. 시험 성적이 높은 과목은 이미 잘 아는 것을 나타내고 시험 성적이 낮은 과목은 반대로 공부가 더욱 필요함을 나타낸다.
그러나 시험은 개인의 맞춤화된 피드백을 제공하기 어렵다. 이를 보완하기 위해 Deep Knowledge Tracing(DKT)를 사용할 수 있다. DKT 는 딥러닝 방법론으로, 학생의 지식상태를 추적하는데 사용된다.
다만 이번 프로젝트에서는 학생 개개인의 지식상태를 예측하기 보다는, 아직 풀지 않은(Unseen) 문제에 대한 정오답을 예측하는 것을 목표로 한다. Iscream 데이터셋을 이용하여 각 학생의 푼 문제 목록과 정답 여부를 통해 최종 문제에 대한 정답 확률을 예측한다.

• 활용 장비 및 재료

- 서버 스펙 : AI Stage GPU (Tesla V100-SXM2)
- 협업 툴 : Github / GatherTown / Zoom / Notion / Google Drive
- 기술 스택 : Python / Pytorch / Pytorch-lightning / VScode / Wandb / Hydra / Scikit-learn

• 프로젝트 구조



• 사용 데이터의 구조

Column	설명
userID	사용자의 고유번호, 총 7,442 명의 고유 사용자가 있으며, train/test 셋은 9:1 비율로 구성되어 있음
assessmentItemID	문항의 고유번호, 총 9,454 개의 고유 문항이 있음
testId	시험지의 고유번호, 총 1,537 개의 고유 시험지가 있음
answerCode	사용자가 해당 문항을 맞췄는지 여부에 대한 이진 데이터, 0 은 사용자가 문항을 틀린 것, 1 은 맞춘 것이다.
Timestamp	사용자가 해당문항을 풀기 시작한 시점의 데이터
KnowledgeTag	문항 당 배정되는 태그

1-2. 프로젝트 팀 구성 및 역할

이름	역할
강찬미	EDA, LQTR/catboost 구현 및 HPO, T-fix up 구현, github action workflow 작성, Feature Engineering
박동연	Sequential baseline 구축, GPT2/GRUATTN 구현, Ensemble 구현, GPT2 & LSTMATTN HPO
서민석	Tabular baseline 구축, Feature Engineering, LightGBM HPO
이준영	EDA, XGBoost, Saint+ 구현 및 HYPO, 코드 테스트 구현, Feature Engineering
주혜인	Graph baseline 구축, Data Augmentation, LightGCN & LSTM HPO

1-3. 프로젝트 수행 절차 및 방법

• 팀 목표

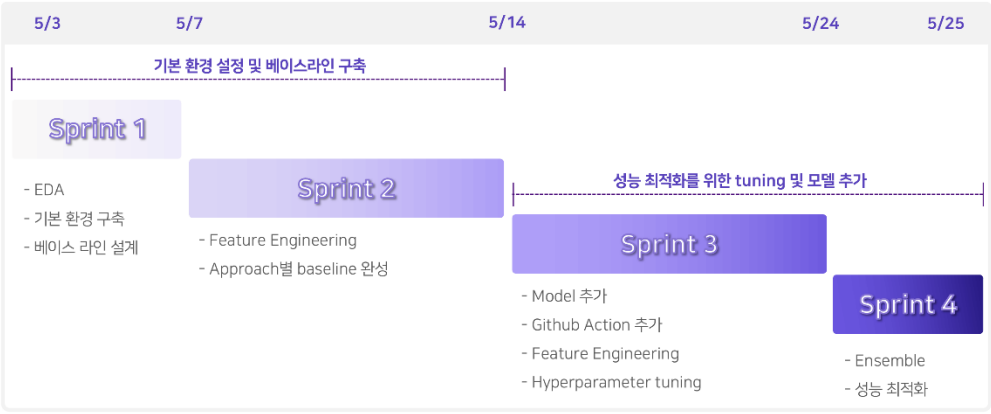
- **새로운 베이스라인 작성** : 제공된 베이스라인을 참고하여 우리 팀만의 자체적인 베이스라인 구축
- **스프린트 방식 도입** : 전반적인 계획 수립을 통한 체계적인 진행을 위해 스프린트 방식 도입
- **적극적인 Github 도입** : 이전보다 적극적인 issue 사용 및 PR 을 통한 코드 리뷰 활성화
- **다양한 Tool 사용** : Pytorch Lightning, hydra, github action 등과 같은 다양한 tool 경험

• 프로젝트 협업 문화

노션을 사용한 효율적인 협업
- **고민 해결 일지:** 고민 해결 일지에 프로젝트 관련하여 필요하다고 생각하는 것을 공유하고 해결하는 과정을 문서화하였음
- **칸반 보드**: 노션에 Jira 를 토대로 애자일 방법론을 적용하여 칸반보드를 제작해 협업을 수행하였음
- **스프린트 단위 자체 팀 회고 진행**: 스프린트 단위로 자체 팀 회고를 진행하여 매주 목표와 각 작업들의 완료 여부 및 계획 공유

적극적인 Git 활용
- **Git Convention**: commit 메시지, github flow 전략 도입
- **Pre-commit 활용**: black 포매터를 이용해 코드 스타일 통일
- **Github issue 활용**: 작업할 목록을 issue 에 정리
- **Github Action 을 이용한 자동 테스트**: pytest 를 이용한 모델 러닝 테스트

• 프로젝트 타임라인

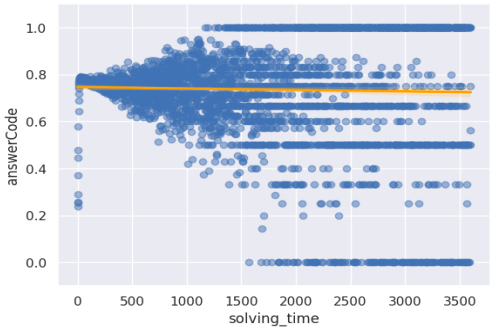


- Sprint 1 (5/3~5/7) : 기본 환경 설정, 베이스라인 구축, 탐색적 데이터 분석(EDA)
- Sprint 2 (5/7~5/14) : 기본 환경 설정, 베이스라인 구축, Feature Engineering, 각 Approach 별 베이스라인 완성
- Sprint 3 (5/14~5/24) : 모델 추가, Github Action 추가, Feature Engineering, 성능 최적화를 위한 하이퍼파라미터 튜닝
- Sprint 4 (5/24~5/25) : 성능 최적화를 위한 하이퍼파라미터 튜닝, Ensemble 구현 및 실험

1-4. 프로젝트 수행 결과

1. 탐색적 데이터 분석 (EDA)

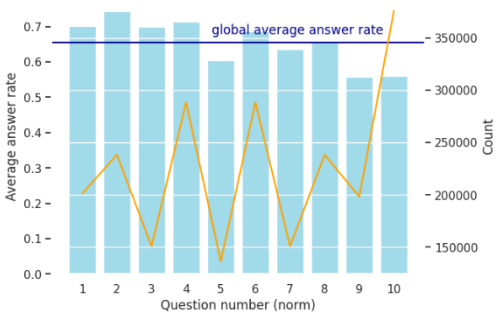
1-1. 풀이시간



풀이시간(n+1 행의 time stamp - n 행의 time stamp)에 따른 정답률 그래프를 그려본 결과 풀이시간이 증가함에 따라 정답률이 감소하는 경향을 확인할 수 있었음.

하지만 현재 문제에 대한 interaction이 끝난 후에 현재 문제의 풀이시간을 알 수 있다는 점에서 미래 정보를 활용하게 된다는 문제가 있어 직전 문제의 풀이시간을 활용하도록 함.

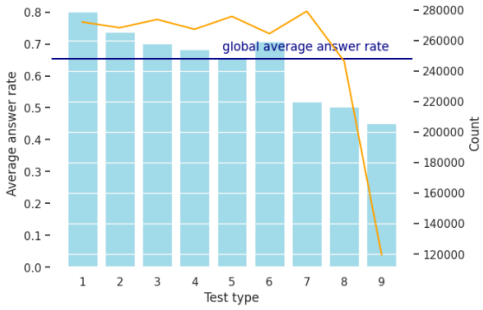
1-2. Question Number



Question Number(A000000XXX)에 따른 정답률 그래프를 그려본 후, 후반 번호로 갈수록 정답률이 떨어지는 경향을 확인할 수 있었음.

그러나, 각 시험지마다 문제수가 달라 각 시험지에서 후반 번호의 범위가 달라지는 문제가 있음. 이를 해결하기 위해 각 시험지에 대해 Question Number를 1-10 값으로 정규화한 Normalized Question Num을 활용하도록 함.

1-3. Test Type



Test Type(A0X0000000)에 따른 정답률 그래프를 그려본 결과 Test Type이 커질수록 정답률이 감소하는 경향을 확인할 수 있었음.

또한, 하나의 Knowledge Tag(#7863)을 제외하고 모든 Tag가 하나의 Test Type에 독립적으로 속하는 것을 확인하였고 이를 통해 Test Type을 대분류, Knowledge Tag를 소분류로 볼 수 있다는 인사이트를 얻음.

2. Feature Engineering

- Feature Engineering을 통해 30개의 파생변수를 생성함
- EDA를 통해 train 과 test 데이터에서 공통적으로 발견되는 데이터의 특성을 파악하고 파생변수로 생성함
- 여러 DKT논문에서 사용되는 feature들을 본 대회 데이터와 각 approach에 적합한 방식으로 재구성함

3. Approach 1: Tabular

- 주어진 데이터를 정형 데이터로 취급하여 LightGBM과 같은 일반적인 지도학습 모델을 통해 예측하는 방식
- 30개 파생변수가 추가된 데이터를 LightGBM, CatBoost, XGBoost에 학습시키고 성능을 평가함

4. Approach 2: Sequential

4-1. 모델 개요

Model	특징
LSTM	Baseline 에서 제공된 모델
LSTMATTN	Baseline 에서 제공된 모델 LSTM 에 self-attention 레이어를 더한 모델
GRUATTN	데이터의 수가 적은 것을 감안, LSTM 보다 더 적은 양의 데이터셋에서 잘 동작하는 GRU 모델에 self-attention 레이어를 더한 모델
BERT	Baseline 에서 제공된 모델 Transformer 의 인코더 스택만 사용하는 모델
GPT2	데이터를 증강한 후 커진 규모의 데이터셋을 감안해 구현 Transformer 의 decoder 스택만 사용하는 모델
LQTR	강의의 실습 코드를 토대로 구현한 모델 Transformer 의 인코더에 LSTM, DNN 을 더함 인코더의 마지막 쿼리만 사용해 낮은 계산 복잡도를 가짐
SAINT+	강의와 논문을 토대로 구현한 모델 Transformer 에 시간 정보를 활용한 모델 본 대회 데이터셋에서 sequential 모델 중 가장 성능이 우수했음

4-2. Approach: Sequential - SAINT+

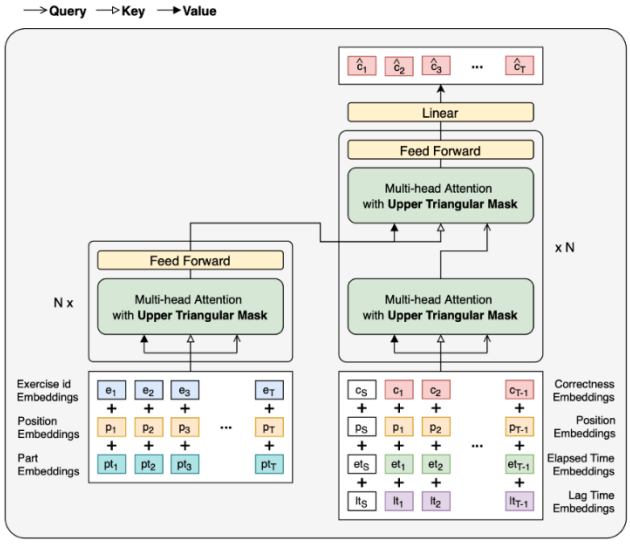


Figure 2: Model architecture of SAINT+.

- **입력 임베딩**: Elapsed Time 과 Lag Time 을 Solving Time(0~300s)로 구현, Test Id, Test Type 추가
- **T-fixup**: 모델의 깊이에 따라 parameter 를 scaling 하는 가중치 초기화 방법 도입
- **Data Augmentation**: sliding window 방식으로 하나의 sequence 에서 여러 개로 증강
- **Parameter Customization**: encoder layer, decoder layer 수를 2, head 개수를 2 로 수정

5. Approach 3: Graph

5-1. 데이터

- 사용 변수 : userID, AssessmentID
- 전처리 : 한 유저가 같은 문제를 여러번 푼 경우, 가장 최근의 기록만 취하여 edge 와 lable 을 만들었다. 이때, edge 는 2 차원 리스트로 각각 user 의 index 와 assessmentID 의 index 를 담고 있으며 같은 위치에 대응하는 label 이 1 인 경우 정답, 0 인 경우 오답을 의미한다.

5-2. 모델

- LightGCN : 이웃 노드의 임베딩의 가중합으로 GCN 을 적용한 모델

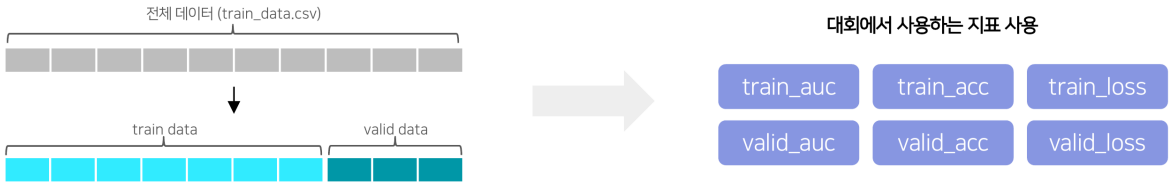
5-3. 결과

- layer 의 범위는 작을수록 AUC 가 높았으며 epoch 는 900 회가 넘었을 때 수렴하는 양상을 보였다.

6. 학습 방법 및 평가 지표

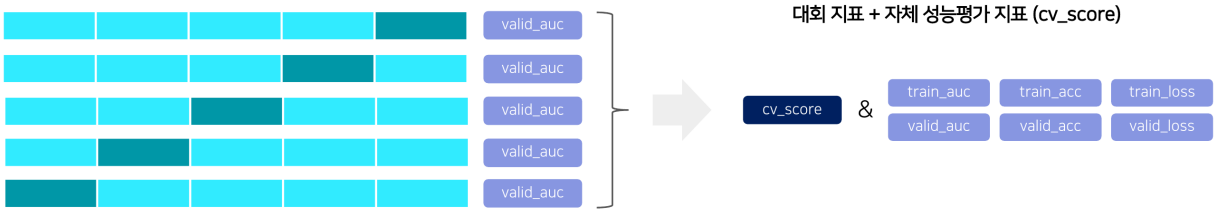
6-1. Holdout

- train_data.csv 전체를 7:3 의 비율로 train 데이터와 valid 데이터로 나누어서 학습 및 검증에 사용하였음
- 성능 평가 지표로는 train_auc, train_acc, train_loss, valid_auc, valid_acc, valid_loss 를 사용하였고 주요 성능을 확인하는 지표로는 대회에서 사용하는 것과 동일하기 'valid_auc'를 사용하였음.



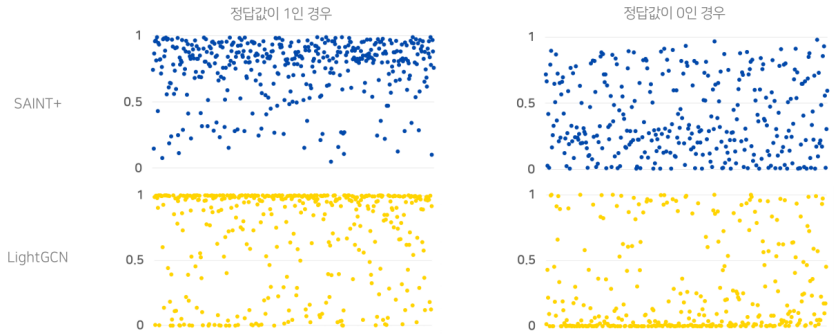
2-2. K-fold CV

- train_data.csv 전체를 동일한 크기의 5 개의 폴드로 나누어서 cross validation 방법을 적용
- 즉, 4 개의 폴드가 학습 데이터로 활용되고 1 개의 폴드가 검증 데이터로 활용되는 과정을 5 번을 반복
- 이를 거치면 5 개의 valid_auc 값을 얻어낼 수 있는데, 이 값의 평균을 cv_score 로 하여 주요 성능 지표로 사용



7. 앙상블 전략 수립

- (step 1) 각 모델에 따라서 정답을 예측하는 방법 및 추이, 잘 예측하는 경우가 다를 것이라고 가정
- (step 2) 각 모델이 예측한 값과 정답 값의 분포 추이를 비교하였음
  이 때, test_data 에서 (유저가 푼 문제수 -1) 번째 문제를 validation set 으로 구성하였음
- (step 3) 서로 다른 예측 분포를 가진 모델들을 다양한 기법으로 앙상블하여, 상호 보완하는 효과를 내는 것을 기대하였음



- 위의 사진을 보았을 때, SAINT+ 모델은 일반적으로 잘 예측하지만 오답의 경우는 중간값으로 예측하는 경향이 있음
- LightGCN 모델은 정답과 오답을 중간값보다는 극단값으로 예측하는 경향이 있음
- 앞서 설명한 추론 단계에 따르면 이 두 모델을 앙상블 하였을 때 좋은 예측 성능을 보일 것으로 기대할 수 있음

8. 최종 솔루션 모델

- 앞서 설명한 앙상블 전략에 따라서 최종 솔루션 모델을 선정함
- 첫 번째 솔루션으로는 public score 가 가장 좋았던 SAINT+ 모델과 LightGCN 모델을 8:2 의 가중치를 주어 앙상블함
- 두 번째 솔루션으로는 cv_score 가 가장 좋았던 SAINT+ 모델 두 개와 LigthGCN, LightGBM, LSTMATTN, LSTM 모델에 대해서 Stacking Ensemble 수행

Simple Weighted Ensemble	Stacking Ensemble
- SAINT+ (0.8) - LightGCN (0.2)	- 1st SAINT+ - 2nd SAINT+ - LightGCN - LightGBM - LSTMATTN - LSTM

1-5. 자체 평가 의견

[목표 달성]

- 주어진 baseline을 참고하여 자체적인 **baseline 구축**
- 체계적인 프로젝트를 위해 **스프린트 방식**으로 진행
- **Github** issue와 pr, code review의 **적극적인 활용**
- Pytorch lighting, hydra, github action 등 **다양한 Tool 경험**
- **체계적인 문서화**를 위한 노션 활용
- 최종 리더보드상 3 위라는 좋은 성과를 거둠

[배운 점]

- **코드 리뷰의 중요성** : 팀원 간 적극적인 코드 리뷰를 통해 더 나은 코드에 대해 고민해 볼 수 있었음.
- **문제상황 공유 및 해결방안 논의의 이점** : 문제 상황이 발생하면 빠르게 팀원들과 공유하여 함께 해결책을 모색함으로써 더 효율적으로 문제를 해결할 수 있었음.
- **Baseline에 대한 높은** 이해 : 직접 baseline을 구축하며 코드를 세세하게 살펴보고 직접 작성해보며 baseline에 대한 이해도를 높일 수 있었음.
- **효율적 실험 환경 구성의 이점** : 새로운 Tool을 사용하여 실험 환경을 구성함으로써 효율적인 실험 환경의 중요성을 느낄 수 있었음.
- **충분한 데이터의** 중요성 : Data augmentation을 적용함으로써 성능이 향상되는 것을 경험하며 충분한 양의 데이터의 중요성을 느낄 수 있었음.

[아쉬운 점]

- **다양한 접근 방식** : 시퀀스 구성 방식을 달리 해보는 등의 다양한 접근 방식을 시도해보지 못함.
- **ML project의 unit test** : ML project에서 적합한 unit test 방식을 찾지 못함.
- **Feature 및 data version 관리의 어려움** : Feature 및 data version을 효율적인 방식으로 처리하지 못함.
- **불필요한 log file 관리** : 다양한 Tool을 사용함으로써 생성되는 log file을 효율적으로 관리하지 못함.
- **Baseline 구축 소모 시간 :** Baseline 구축에 예정보다 많은 시간을 소요함.

2. Personal Wrap-up Report

2-1. 강찬미_T5009

1. 내 학습목표를 달성하기 위해 한 노력

- **다양한 경험하기 :** 이번 프로젝트를 시작하며 목표했던 것은 가능한 다양한 파트를 경험해 보는 것이었다. 우선 EDA를 담당하며 다양한 가설을 생각하고 이를 시각화를 통해 검증해본 후, 그 결과를 바탕으로 추가되면 좋을 feature에 대한 고민을 해보았다. 또한 Tabular에 catboost를, sequential에 Last query model을 추가하였으며, github action을 통한 자동화 test에 참여하였다. 그 외에도 scheduler, 가중치 스케일링 부분에 참여하며 다양한 파트를 경험해볼 수 있었다.
- **함께 고민하기:** 이번 프로젝트를 진행하며 잘 안 되거나, 확신이 들지 않거나, 궁금한 부분은 주저 않고 팀원과 공유하였다. 팀원들의 의견을 들으며 놓치고 있던 부분을 빨리 발견할 수 있었고 좋은 아이디어를 얻을 수 있었다. 이를 통해 조금 더 효율적으로 문제를 해결할 수 있었다.

2. 내가 모델을 개선하기 위해 한 노력

- **모델 추가 :** 강의에서 소개된 Last query model과 catboost model을 구현해 추가하였다.
- **T-fix up 구현:** saint+ 모델에 강의에서 소개된 가중치 초기화 및 스케일링 방법론 중 하나인 T-fix up을 추가하였다.
- **하이퍼 파라미터 튜닝:** Last query, catboost 모델을 집중적으로 튜닝하였다.
- **Linear warm up scheduler 추가:** Transformer model에서 사용해볼 수 있는 scheduler 추가하였다.

3. 내가 한 행동의 결과로 달성한 것 및 얻은 깨달음

- **생각보다 어려운 일이 아닐 수 있다**: 위의 작업들을 하기 전 '내가 과연 할 수 있을까?'하는 걱정이 앞서 섣불리 시도하지 못했었다. 하지만 실습 자료를 포함해 다양한 참고 자료를 보며 하니 예상보다 수월하게 해낼 수 있었다. 한 번 성공하고 나서 조금 자신감을 얻어 이것저것 시도해볼 수 있었다. '할 수 있을까?' 생각만 하는 것보다는 직접 해보고 확인하는 것이 중요하다는 것을 깨닫게 되었다.

4. 내가 새롭게 시도한 변화와 효과

- **디버깅**: 팀원분의 도움으로 디버깅을 제대로 사용하게 되었다. 오류 메시지 한 줄만으로 문제를 해결하던 이전과는 달리 변수들에 어떤 값이 들어가 있는지, 좀 더 깊이 들어가 어느 부분에서 오류가 나는지 확인하며 오류를 해결하니 더 빠르고 효율적으로 할 수 있었다.
- **Github action:** Github action을 통해 main에 merge되는 코드의 전체 학습 과정 중 오류가 있지 않은지 간편하고 빠르게 테스트할 수 있었다. 또한, workflow 작성에 참여하며 이에 대한 이해도를 높일 수 있었다.

5. 마주한 한계와 아쉬웠던 점

- **다른 작업에 대한 이해**: 맡은 부분이 아닌 다른 부분에 신경을 많이 쓰지 못해서 코드에 대한 이해도가 떨어지는 점이 아쉬웠다.
- **좁아지는 시야**: EDA, 실험을 하며 한가지에 초점을 맞추다 그것에 매몰돼 다른 아이디어나 가설을 떠올리지 못했다는 점이 아쉬웠다.
- **Task에 대한 이해**: 프로젝트 초반 Task에 대해 정확하게 이해하지 않고 진행하여 놓치고 넘어가 뒤늦게 깨닫게 되는 부분이 많아서 아쉬웠다.

6. 다음 프로젝트에서 시도해볼 것

- **Baseline 구축:** 이번 프로젝트에서 baseline에 대한 이해도를 높였으니 다음 프로젝트에서는 Baseline 구축에 참여해보고 싶다.
- **Streamlit을 활용한 시각화:** 강의에서 배운 Streamlit을 활용하여 시각화를 진행하면 조금 더 편리할 것 같아 도전하고 싶다.

2-2. 박동연_T5080

1. 내 학습목표를 달성하기 위해 한 노력

- 난관에 봉착하여 포기하고 싶거나 막막한 부분이 있다면 **팀원들에게 '함께 자라기' 요청을 하여 문제를 해결**하였다! 예전같은 습관이었다면 해결할 때까지 혼자 머리를 싸매고 끙끙 앓고 있었을 텐데, 이제는 좋은 동료에게 잘 물어보는 것도 좋은 능력이라는 것을 알기에 좀더 자주 물어보고 문제를 공유하게 되었다!
- 테크 블로그의 글 보다는 **공식 문서나 공식 깃헙의 코드, PR, issue, 다른 사람의 코드**를 위주로 참고하는 습관을 들였다.

2. 내가 모델을 개선하기 위해 한 노력

- **데이터 크기에 맞는 모델 추가**: Data Augmentation 을 하기 전과 후에 따라 데이터의 양이 매우 큰 차이를 보였다. 따라서 베이스라인에서 제공되는 LSTM 기반의 모델보다 적은 수의 데이터에서 좀 더 잘 동작할 것이라고 추정되는 GRU 기반의 모델을 추가하였다. 그리고 Data Augmentation 후에는 더 큰 모델인 GPT2 를 추가하였다.
- **하이퍼 파라미터 최적화:** Sequential 방법론의 모델들에 대해서 하이퍼 파라미터 최적화 과정을 수행하였다.

3. 내가 한 행동의 결과로 달성한 것 및 얻은 깨달음

- **태스크와 데이터를 꼼꼼히 살펴보고 아이디어를 내자**: 이번 프로젝트에서는 구현과 실험을 위주로 참여하느라 데이터를 꼼꼼히 살펴보고 이에 대해서 많은 아이디어를 내보지 못했었다. 그래서 딥러닝을 공부하면서 개괄적으로 알고 있던 정보들을 토대로 모델 추가 및 실험들을 해보았는데, 크게 성능을 향상시키지는 못했다. 하지만 팀원들이 데이터를 꼼꼼히 살펴보고 수행했던 피처 엔지니어링을 통해서는 성능이 꽤 올랐던 것으로 기억한다. 이를 통해 EDA를 통해 뽑아내는 아이디어의 구현이 얼마나 중요한 것인지 알게 되었다.

4. 내가 새롭게 시도한 변화와 효과

- **추상화와 안티패턴 발견:** 제법 정말 다양한 기능 구현에 적극적으로 참여하면서 다른 사람들의 코드를 많이 뜯어보게 되었다. 그러다보니 평소에 제대로 쓰지 못하고 개념만 이해한 채로 있었던 class 도 비교적 이전보다 잘 활용하게 되었고, 내 코드에 내재되어 있었던 안티패턴 같은 것들을 발견하고 어떤 방향으로 고치면 좋을 지 알게 되었다.
- **Level1 보다 더 적극적인 Git 활용:** 각 작업을 issue 로 업로드하고, 이를 해결하는 PR 을 올리며 꽤나 성취감도 있었고, 각 컨벤션도 잘 도입이 되어서 저장소를 더 풍부하면서도 잘 정돈시킬 수 있었다.

5. 마주한 한계와 아쉬웠던 점

- **너무 많은 오토로깅:** 다양한 프레임워크와 라이브러리를 사용하면서 각 프레임워크들이 자동으로 추적해주는 로그 파일들이 정말 많이 생성되어 불편했었다.
- **언제나 건강이 문제:** 의식하지 못했는데 바르지 않은 자세로 오랫동안 앉아있다보니 허리에 부담이 가서 또 병원 신세를 졌다. 한창 최고의 성능을 뽑아내야 할 프로젝트 막바지에 아프게 되어서 팀원들에게 더욱 미안했다. 정말로 건강관리를 제대로 해야겠다고 다짐했다.

6. 다음 프로젝트에서 시도해볼 것

- **EDA 및 피처 엔지니어링: 지**난 프로젝트에서는 모델 변경 및 실험, 이번 프로젝트에서는 구현 및 실험 위주로 참여하느라 데이터를 꼼꼼히 살펴보지 못했던 것이 아쉽다. 따라서 다음 프로젝트에서는 EDA 와 이를 통한 피처 엔지니어링을 위주로 참여하고, pandas 라이브러리와 더 친해져보고자 한다.
- **streamlit 으로 분포 시각화 자동화:** 이번 프로젝트에서는 각 모델의 예측 분포 결과를 늦게 살펴본 것이 아쉽다. 따라서 초반에 streamlit 을 활용하여 바로 모델별 예측 분포를 시각화해볼 수 있도록 하고싶다.

2-3. 서민석_T5102

1. 내 학습목표를 달성하기 위해 한 노력

- Baseline 구축: 지난 프로젝트에서 가장 아쉬웠던 점은 저장소를 public 으로 전환하는 과정에서 저작권 이슈로 인해 작업물을 온전히 공개할 수 없었던 점이었다. 이번 프로젝트에서는 baseline 을 직접 구축하여 저작권 이슈가 없는 온전한 작업물을 공개할 수 있어 기쁘다.
- Hydra 도입: 프로젝트 중에 sweep 을 이용해 실험을 하다 보면 config 파일 관리에 어려움이 생긴다. Hydra 를 사용하면 config 를 계층적으로 구성하여 관리할 수 있다는 점이 유용하게 느껴졌다.

2. 내가 모델을 개선하기 위해 한 노력

- Feature 추가: 기존 DKT 연구에서 Transformer 기반의 모델에 여러가지 임베딩 feature 추가하여 예측 성능을 개선했던 사례가 있었다. 나는 연구에서 사용된 feature 들을 LightGBM 에 적합한 형태로 가공하여 적용해보는 시도를 했고, 결론적으로 유저가 과거에 푼 문제와 정답여부에 대한 interaction 을 파생변수로 만들어서 추가했을 때 유의미하게 성능이 개선되는 것을 확인할 수 있었다. 과거에 어떤 문제를 풀었고, 맞췄는지 틀렸는지에 대한 정보가 모델이 현재 풀고 있는 문제에 대한 정답 여부를 예측할 때 유용하게 작용되었던 것 같다.

3. 내가 새롭게 시도한 변화와 효과

- 실행시간 고려: 데이터 처리 과정에 상당한 실행시간이 소요된다는 사실을 발견하고 이를 단축시키기 위해 다양한 시도를 했다. 대표적으로 데이터 버저닝을 도입하고 기존 데이터 모듈에 '데이터 버전에 따라 데이터 처리를 생략하거나 최소한의 처리만 진행할 수 있도록 하는 기능'을 추가했다. 이를 통해 데이터 처리 반복을 최소화 함으로써 HPO 를 위한 실험에 소요되는 시간을 효과적으로 단축 시키는 데에 기여하였다.

4. 마주한 한계와 아쉬웠던 점

- Category 타입 Series 정렬: 프로젝트 초반에 userID 컬럼의 dtype 을 category 로 관리했고. 이로 인해 유저 별 예측값을 뽑고 정렬하는 과정에서 의도했던 것과 다르게 정렬되는 오류가 있었다. 프로그램은 userID 를 문자열로 취급하여 전혀 다르게 정렬된 submission 파일을 생성하고 있었던 것이었다. 제출 횟수를 낭비했다는 점, 그리고 오류를 찾는 과정에서 다소 많은 시간을 소비했던 점이 아쉬웠다.

5. 다음 프로젝트에서 시도해볼 것

- Transformer: 강의를 통해 다양한 Kaggle 솔루션을 공부하면서 참가자들이 각 대회 task 에 맞게 transformer 구조를 설계하는 과정이 흥미롭게 느껴졌다.. 다음 프로젝트에서는 대회 task 에 적합한 transformer 구조에 대한 고민을 해보고 싶다.
- Graph: 팀원 한 분이 유저와 문제를 node 로, 정답여부를 edge 로 하는 그래프를 만들고 LightGCN 모델을 사용하여 Link prediction task 를 풀도록 했을 때 예측 성능이 좋았다. 다음 프로젝트에서도 그래프 기반의 접근방식을 시도해볼 여지가 있다면 시도해보고 싶다.
- Ablation Study: 여러 딥러닝 논문의 실험 파트에서 각 component 를 제거해보면서 성능 추이를 관찰하는 실험을 많이 진행한다. 다음 프로젝트 때 내 아이디어가 성능 개선에 도움이 되는지를 검증해보는 데에 사용해보면 좋을 것 같다.

2-4. 주혜인_T5208

1. 내 학습목표를 달성하기 위해 한 노력

- Graph approach baseline 도전하기
- 스프린트 단위 회고하기 (사실은 데일리회고가 목표였으나 제대로 못했다)
- 계획을 잘 짜서 일을 마무리하기 - issue 를 쓰고 할 일을 시작하니 어느정도 해결이 되었던 것 같다.

2. 내가 모델을 개선하기 위해 한 노력

- **LightGCN hyperparameter tuning** - 실험 결과를 잘 정리해두고 분석한 뒤 다음 스윕을 돌려서 모델의 성능을 키워낼 수 있었던 것 같다.
- **data augmentation** - augmentation 을 한 뒤에 확실히 성능이 좋아져서 보람이 있었다. 그러나 shuffle 로 새로운 sequence 를 더 만들어내는 방법은 아직도 납득이 잘 안되는 일이고 우리 팀에서는 실험에서도 별로 효과가 없었다.
- validation set 시각화

3. 내가 새롭게 시도한 변화와 효과

- **소스코드 및 논문 구현 코드 찾아보기** - GNN 계열 모델에서는 데이터가 들어가는 타입이 다 다른 것 같아서 어쩔 수 없이(?) 소스코드와 공식 깃헙을 보게 되었다.
- **참견하기** - 다른 팀 일이더라도 관심 있는 부분이나 도전해보고 싶은 작은 일들을 한게 확실히 회의할 때 다른 팀의 상황과 사정을 이해하는데 많은 도움이 되었던 것 같다.

4. 마주한 한계와 아쉬웠던 점

- **Git 사용**: 이번에 pull 을 잘 모르고 강제로 하는 말도 안되는 실수를 저질러서 중간에 commit 들이 날아갔었다.
- **강의**: 프로젝트 기간에 강의를 열심히 듣는건 여전히 어려웠다!
- **다양한 모델** 도입: GAT, UltraGCN 등 GNN 계열의 다양한 모델들을 추가하려는 시도는 많았지만 결국 LightGCN 에서 그친게 아쉽다.
- 나는 아직도 **코드를 이해하는 속도**가 너무 느리다: 데이터 processing 을 어떻게 하는지 이해하는데 오래걸리고 정확하게 이해했다는 자신이 없어서 결국 클론받아서 중간 과정을 다 출력해보면서 이해했고 결론적으로는 시간을 너무 많이 소모했다.
- **꼼꼼한 검토를 안한다**: augmentation 에 shuffle 을 추가한 뒤에 shuffle 된 sequence 를 다른 sequence 라고 인지할 수 있게 새로운 userID 를 만들어줬어야 하는데 이 부분을 빠트려서 결국 hotfix 브랜치가 탄생하게 되었다.

5. 다음 프로젝트에서 시도해볼 것

- **RecBole 도전**하기! 이번 프로젝트에서 언급만 많이 하고 결국 못한게 너무 아쉽다.
- 다양한 **loss function** 활용해서 실험해보기
- **github 의 project** 기능이 궁금하다. 이번 프로젝트에서 꽤 많은 팀들이 사용한 것 같아서 알아보고 싶다.

2-5. 이준영_T5158

1. 내 학습목표를 달성하기 위해 한 노력

- **자동화된 테스트**: Pytest와 hydra를 이용해 모든 모델에 대해 기초적인 테스트를 수행할 수 있었고, 이어 Github Action을 통해 자동화된 테스트를 구축했다. 추가로 더미데이터와 pip 모듈 캐싱으로 2~3분대로 테스트를 빠르게 수행할 수 있게 했다.
- **EDA에 집중하기**: 이번 대회에서는 초반부터 EDA에 집중하였고 유의미한 피쳐를 탐색해 이후 구현하는 모델에 활용해볼 수 있었다.

2. 내가 모델을 개선하기 위해 한 노력

- **모델링**: EdNet데이터 셋에서 가장 우수한 성능을 낸 SAINT+모델을 직접 구현했다. 또한 본 대회의 데이터셋에 맞게 입력 임베딩을 수정하였고, EDA에서 발견한 유의미한 피쳐 임베딩를 추가했다.
- **모델 및 데이터 튜닝**: wandb sweep을 이용해 SAINT+, BERT, XGBoost 모델을 튜닝 했다. 이를 통해 데이터 셋에 따라 필요한 하이퍼 파라미터가 다르다는 것을 알게 되었고, 데이터 수의 중요성 또한 알게 되었다. (데이터 수가 많을수록 학습이 잘되었기 때문)

3. 이번 프로젝트에서 도움이 되었던 것 및 깨달음

- **트랜스포머**: SAINT+논문을 리뷰하고 이번 프로젝트에 직접 구현함으로써 트랜스포머 모델에 대한 이해도가 깊어졌다. 또한, 임베딩 벡터들을 다루는 방법 또한 매우 도움이 되었고 강의와 논문에 나왔던 토큰에 대해 직접 구현함으로써 직관적으로 이해할 수 있었다.
- **Pytorch Lightning**: 이번 프로젝트에선 적극적으로 pytorch lightning 도입해보았다. 내 담당이 아니였지만, 코드를 보며 우리가 고민했던 설계에 대한 프레임을 잡아주어서 매우 도움이 되었다.
- **함께자라기**: 이해가 잘 안 되거나, 도저히 해결할 수 없는 오류를 머리를 맞대어 해결하고자 하였다. 이것을 우리는 "함께자라기"라고 했는데, 문제 해결뿐만 아니라 역으로 서로 담당한 분야에 대해 이해시키는 역할을 했다. (개인적으로 pr review보다 더 효과적일 수 있다는 생각이 든다.)

4. 다음 프로젝트에서 시도해볼 것

- **접근성 좋은 시각화**: 이번 프로젝트에서 EDA를 담당하였고 피쳐 엔지니어링을 시도해보았지만 좋은 시각화에 대해 심도 있는 고민을 하지 못했다. 다음 프로젝트에선 직, 간접적으로 streamlit등의 툴을 사용해 데이터에 대한 이해도를 팀원들과 공유하고 싶다.
- **병렬 학습**: 이번 프로젝트를 하면서 대부분의 모델이 GPU를 제대로 활용하지 못하는 것을 발견했다. Tabular 모델을 제외하면 많아도 30%를 넘지 못하면서 학습은 느렸다. 이런 특징은 작은 배치사이즈로 학습할 때 더욱 드러났다. 다음 프로젝트에서는 최대한 동일 학습 프로세스에서 최대한 GPU 효용성을 높이기 위한 노력을 기울일 것이다.

3. 부록 (Appendix)

3-1. 파생변수 목록

컬럼명	설명
TestType	시험지 유형
UserAcc	유저별 과거 평균 정답률
UserItemAcc	유저별 현재 풀고 있는 문제의 과거 평균 정답률
UserTag1Acc	유저별 현재 풀고 있는 문제의 KnowledgeTag 에 대한 과거 평균 정답률
UserTag2Acc	유저별 현재 풀고 있는 문제의 TestType 에 대한 과거 평균 정답률
UserLastTag1Correct	유저별 현재 풀고 있는 문제의 KnowledgeTag 에 대한 가장 최신 지식상태
UserLastTag2Correct	유저별 현재 풀고 있는 문제의 TestType 에 대한 가장 최신 지식상태
RollingTime	유저별 현재 풀고 있는 시험지에 대한 풀이시간 이동평균
UserTestRetakeCnt	유저별 현재 풀고 있는 시험의 재시험 횟수
UserRecencyN	유저별 N 시점 전 정답여부 (N=1,2,…,5)
UserInteractionN	유저별 N 시점 전 풀이한 문제와 정답여부 (N=1,2,…,5)
UserTag1InteractionN	유저별 N 시점 전 풀이한 문제의 KnowledgeTag 와 정답여부 (N=1,2,…,5)
UserTag2InteractionN	유저별 N 시점 전 풀이한 문제의 TestType 과 정답여부 (N=1,2,…,5)