[RecSys] DKT Wrap UP 리포트

작성자 : 추천시스템 8조

이나현_T4146　전해리_T4191

정의준_T4200　조원준_T4211

채민수_T4217

▼ Table of Contents

1. 프로젝트 개요

A. 개요

DKT는 Deep Knowledge Tracing의 약자로 이번 DKT 프로젝트를 통해 저희는 "지식 상태"를 추적하는 딥러닝 방법론을 사용하여 학생의 이해도를 측정하고 미래 학습 예측을 목표로 하였습니다.

B. 활용 장비(도구)

- 개발환경 Vscode, Jupyter
- 협업툴 GitHub, Wandb, MLflow
- 의사소통툴 Slack, Notion, Zoom, Trello, Gather Town

C. 기대 효과

- 학생의 문제 이해도를 파악할 수 있습니다.
- 학생에게 맞춤화된 교육을 제공할 수 있습니다.

2. 프로젝트 팀 구성 및 역할

전체	문제 정의, 계획 수립, 목표 설정, EDA, Feature Engineering, 모델 실험
이나현	LastQuery Transformer 구현, Feature Selection
전해리	lightGCN+lqTransfomer 구현, lightGCN+Seq model 구현, Feature Selection
정의준	TabNet, wandb, Feature Selection, Stacking Ensemble
조원준	CV Strategy, LGBM, XGBoost, MLflow, Feature Selection, Feature Importance, Ensemble
채민수	SAKT 구현, CatBoost 구현, Feature Selection

3. 프로젝트 수행 절차 및 방법

A. 목표 설정

① wandb, MLFlow, AutoML 등 다양한 툴 사용해보기

② Baseline에 있는 모델 외 다른 모델 사용해보기

③ 다양한 의사소통툴을 이용하여 원할한 소통하기

B. 프로젝트 사전 기획

① 프로젝트 일정

1주차	문제 정의 및 목표 설정, 역할 분담, 데이터EDA
2주차	모델 구현, 모델 실험
3주차	Feature Engineering, 모델 실험
4주차	Feature Engineering, 앙상블

② 역할 분담

대회시작할 때 이번 대회에서 어떤 것을 시도하면 좋을지 브레인스토밍을 먼저 진행하고, 그 결과를 토대로 역할 분담을 하였습니다.

대회 시작 브레인 스토밍

참고 링크 : https://www.kaggle.com/c/riiid-test-answer-prediction/discussion/210113

1. CatBoost 못찾았었던게 아쉬움 → Kaggle Riiid Winner's Solution → 찾아봤는데, SAKT라는 모델 사용 → SAKT 기반 모델 Saint.
   (채민수)
2. Last Query Transformer 구현
   (이나현)
3. wandb project 규칙 → 제목에 모델이랑 → 프로젝트 만들고, 공유하면서 진행 → 규칙 만들기 → 조원들 다 써보기.
   → 목표: 모델별 하이퍼파라미터와 성능 서로 한눈에 볼 수 있게.
   (코드에서 config 파일 만들어서 진행)
   (정의준)
4. mlflow 띄워서 연결해보기
5. GNN에서 GAT 구현해서 임베딩 추출, Transformer에 input으로 넣어보기
- +) ultra gcn
   (전해리)
6. FE → 고도화하면서, Boosting기반 (XGBM, LGBM, Catboost) 성능 올리기
   (조원준)
7. CV 전략 테스트 (User별 split, KFold 등)
   (조원준)

③ GitHub 버전 관리 규칙

a. Commit 규칙 : [타입][이름] 간결하고 요약적인 서술

타입 : Feat, Fix, Docs, Style, Refactor, Test, Chore

b. Branch : master, develop, model, data

C. EDA

① 데이터프레임 확인

② raw 데이터 파악

③ 데이터 개별 속성값 관찰

④ 데이터 속성 간 관계 관찰

⑤ 이상치, 결측치 확인 및 처리

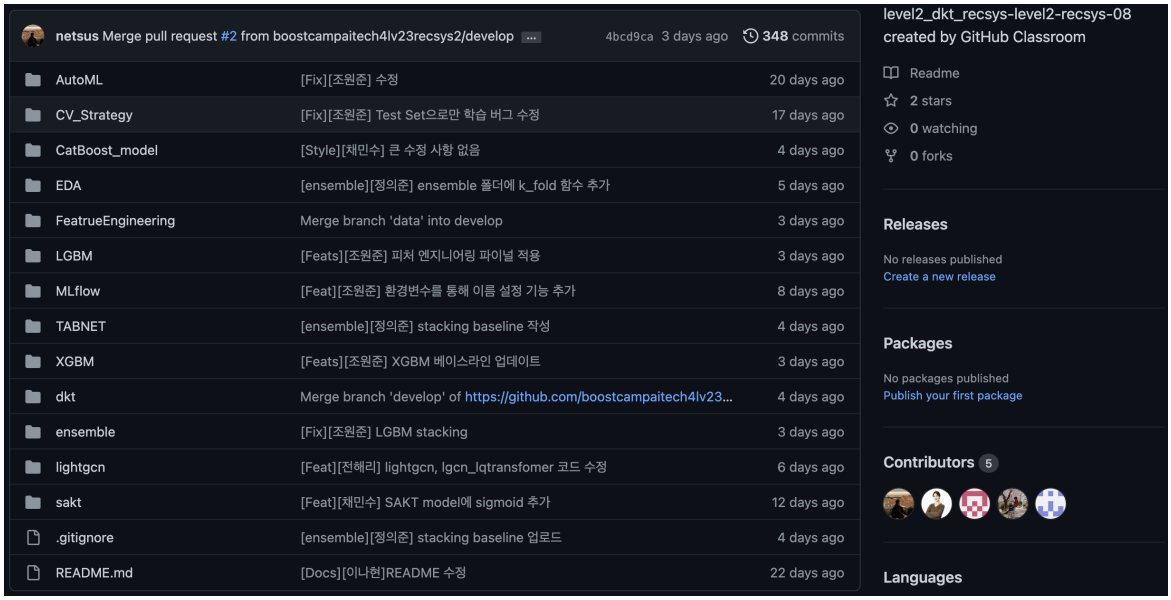
D. Model

Sequence / Transformer	SAKT, LSTM, LSTMATTN, BERT, Last Query Transformer
Boosting	CatBoost, LGBM, XGBoost
Tabular	TabNet
Graph	lightGCN

E. 협업 과정

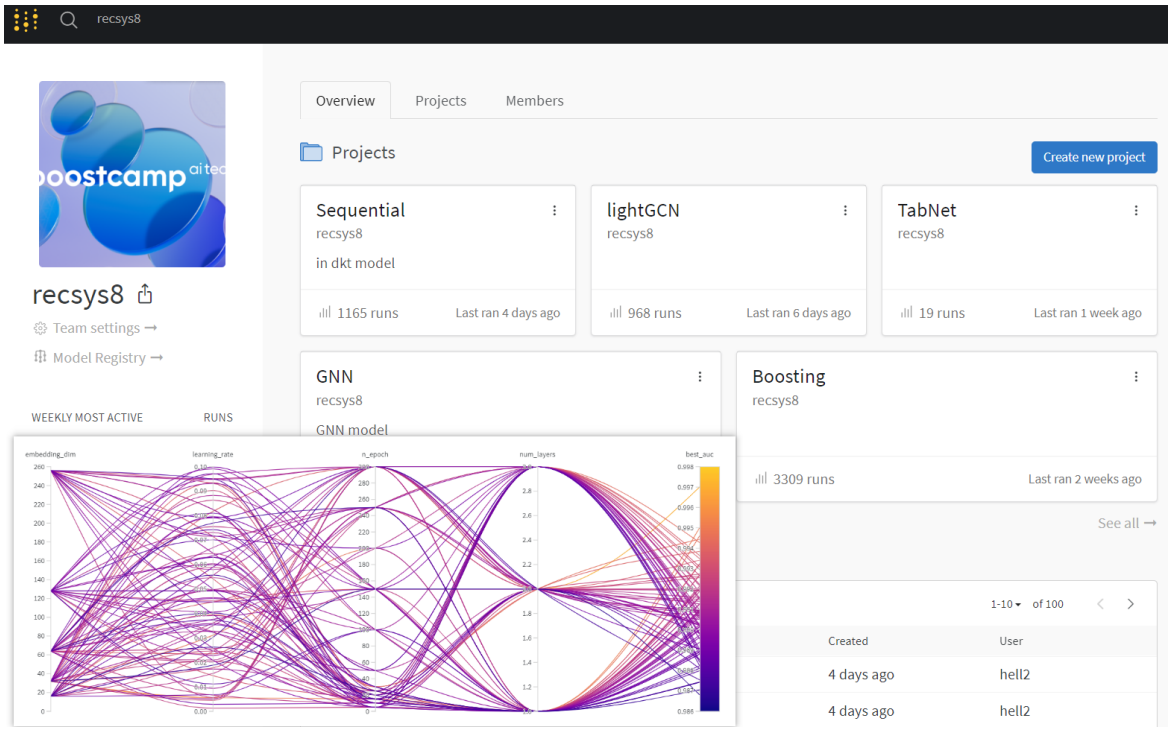
① Git을 통한 협업

→ 각 목적에 맞는 브랜치를 통해 버전관리를 체계적으로 할 수 있었습니다.



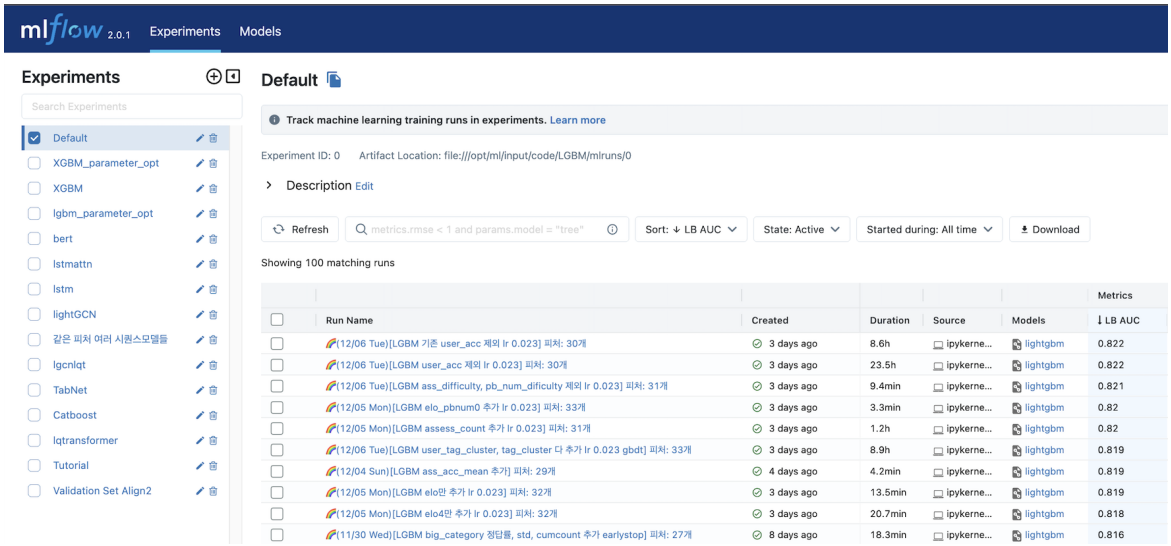
② WanDB를 이용한 H/P Tuning

→팀 페이지를 만들어 모델별로 프로젝트를 나누었습니다. Sweep을 통해 최적의 하이퍼 파라미터를 찾을 수 있었습니다.

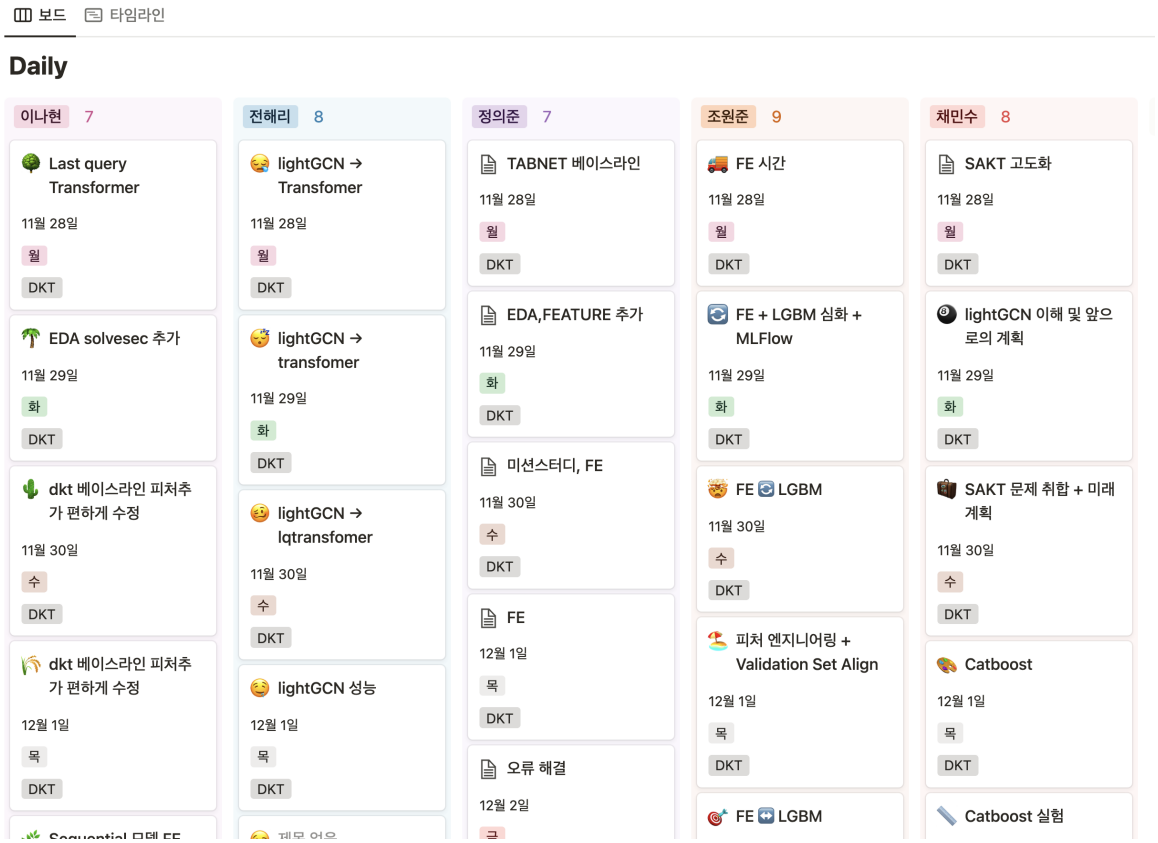


③ MLflow을 이용한 실험 관리

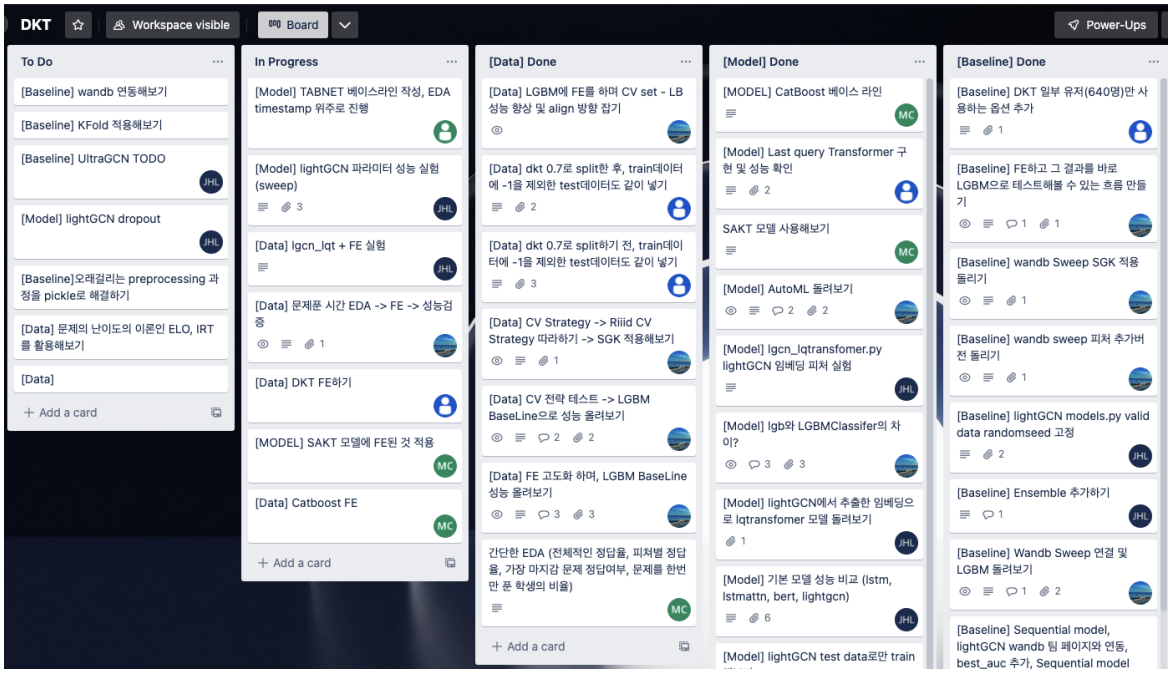
→ 각자 Experiments를 만들고 모델별로 자동으로 성능과 특성들이 기록되도록 하였습니다.



④ Notion을 이용한 데일리 스크럼
→ 데일리 기록을 바탕으로 매일 진행한 것, 어려운 점 등을 공유했습니다.



⑤ Trello를 활용한 업무 공유
→ 주요 업무별로 서로 어떤 업무를 진행하는지 알기위해 트렐로를 사용했습니다.



⑥ Gather Town을 활용한 실시간 소통

→ 서로 필요할 때 바로바로 소통하기 위해 게더 타운을 도입하였습니다.

4. 프로젝트 수행 결과

4-1. 탐색적 분석 및 전처리

A. 데이터 소개

train/test 합쳐서 총 7,442명의 사용자가 존재합니다. 한 행은 한 사용자가 한 문항을 풀었을 때의 정보와 그 문항을 맞췄는지에 대한 정보가 담겨져 있습니다. 데이터는 모두 Timestamp 기준으로 정렬되어 있습니다. 이 때 이 사용자가 푼 마지막 문항의 정답을 맞출 것인지 예측하는 것이 저희의 최종 목표입니다.

- `userID` 사용자의 고유번호입니다.
- `assessmentItemID` 문항의 고유번호입니다.
- `testId` 시험지의 고유번호입니다.
- `answerCode` 사용자가 해당 문항을 맞췄는지 여부에 대한 이진 데이터입니다.
- `Timestamp` 사용자가 해당문항을 풀기 시작한 시점의 데이터입니다.
- `KnowledgeTag` 문항 당 하나씩 배정되는 태그로, 일종의 중분류 역할을 합니다.

B. 데이터 분석 및 Feature Engineering

Base	userID, assessmentItemID, testId, Timestamp, KnowledgeTag
Static	user_acc, problem_num, test_mean, test_std, test_sum, tag_mean, tag_std, tag_sum, solvesec_cumsum, solvecumsum_category, big_category_acc, big_category_std, big_category_cumconut, big_category_user_acc, big_category_user_std
Category	big_category, mid_category, time_category
Time	month, day, hour, dayname, solvetime, solvesec, solvesec_3600 problem_elapsed_time, prob_elapsed_time, normalized_p_elapsed_time, user_prob_elapsed_time_aver, test_prob_elapsed_time_aver, BC_prob_elapsed_time_aver, MC_prob_elapsed_time_aver, KT_prob_elapsed_time_aver, normalized_solvesec,
answer	user_correct_answer, user_total_answer, big_category_answer, big_category_answer_log1p, user_kt_ans_rate prev_answer, prev_answer_rate correctRatio__by_user, correctRatio__by_test_paper, correctRatio__by_tag, correctRatio__by_prob, correctRatio__by_BC, correctRatio__by_TC
Difficulty	elo_assessmentItemID, elo_problem_num
Clutering	user_tag_cluster, tag_cluster

4-2. 모델

A. 모델 개요

Model	개요
SAKT	Transformer의 Attention mechanism을 Knowledge Tracing에 적용한 모델
CatBoost	Boosting 모델의 일종으로 Boosting 모델의 과적합 문제를 개선하고 빠른 학습 속도를 가진 모델
LGBM	Light GBM(LGBM)은 트리 기반 학습 알고리즘인 Gradient Boosting 방식의 프레임워크 모델
XGBoost	병렬 처리및 CART 앙상블 모델을 이용하여 기존 GBM을 개선한 모델
TabNet	Tree기반 모델들의 장점을 계승하여 Tabular 데이터에 적용할 수 있는 딥러닝 모델
LSTM	Long-Short Term Memory 모델은 비선형 시계열 데이터의 의미있는 관계를 인식하는 인공 신경망 모델
LSTMATTN	LSTM 모델에 Self-Attention을 추가한 모델

Model	개요
BERT	Transformer의 Encoder부분과 동일한 구조를 가진 모델
Last query Transformer	Transformer의 Encoder를 이용하여 문제들 간의 관계를 파악하고, LSTM을 이용하여 연속적인 특성을 학습한 후 DNN으로 예측하는 구조의 모델
lightGCN	그래프 구조가 가진 장점을 이용한 GCN의 핵심적인 부분만을 사용하여 가볍게 만든 모델

B. 모델 선정 및 분석

- SAKT

  해당 대회의 목적은 학생들의 과거 학습 기록을 바탕으로 미래에 접하게될 문제의 정오답을 예측하는 것이다. 이때 이전에 풀었던 문제들 과의 연관성을 Transformer의 attention mechanism으로 학습에 반영한다면 좋은 성능을 모일것으로 판단했다. 따라서 동일한 아이디어에서 출발한 SAKT 모델을 적용해보기로 경정했다. 적용 결과 기본적인 구조에서 아쉬운 성능을 모여주어 고도화하여 사용하기로 하였으나 반복되는 에러 발생으로 최종적으로 체택하지 않았다.

- CatBoost

  Feature Engineering을 하는 과정에서 다양한 categorical feature를 추가로 생성하였고 범주형 변수가 많아짐에 따라 CatBoost 모델을 사용할 경우 높은 예측 성능을 가질 것으로 기대하여 채택하였다. 적용 결과 다른 Boosting 모델과 같이 auroc 0.8점대의 높은 성능을 모여주었다.

- TabNet

  Tabular형 데이터에 특화되어 있는 딥러닝 모델. Hyper Parameter보다는 Feature Engineering의 여부에 따라 성능이 크게 좌우되었다. 학습 시간이나 성능은 Boosting 계열보다 낮게나오는 경향이 있었고, 다른 딥러닝 모델과는 다르게 사용한 Feature를 시각화 할 수 있었고 그를 이용해 Feature Selection을 할 수 있었음.

- LGBM, XGBoost

  Time Series에 대해 Lag 기법을 적용한 피처들을 추가하고, 문제 유형별, 항목별 클러스터링 진행 및 elo Rating에 대한 피처를 추가하여 성능을 개선했습니다.

- Seq / Transfomer 계열 (LSTM, LSTM-ATTN, BERT, LastQuery Transfomer)

  : Baseline에 주어진 LSTM, LSTM-ATTN, BERT 모델들에 더하여 과제로 주어진 LastQuery Transfomer 모델까지 구현하여, 각 유저가 푼 문제들을 이용하여 Sequence 형태로 Seq 계열 모델에 적용하여 학습시켰습니다. 사용한 피처로 가장 성능이 잘 나온 모델은 LSTM이었습니다. 피처엔지니어링보다 모델링 위주로 개선을 이어나가 높은 성능을 내지 못했습니다.

- lightGCN

  : 그래프 구조를 이용하여 특징을 추출한다는 점이 흥미로웠고, 이를 이용하여 유저와 아이템의 관계와 상호작용을 잘 추출할 수 있을 것으로 판단되어 lightGCN 모델을 사용했습니다. 하지만 단독으로 모델을 사용할 시 데이터 양이 충분하지 않다면 분산이 낮아 오히려 유저와 아이템의 관계와 상호작용을 잘 표현하지 못하는 현상을 발견했습니다.

- Graph + Transfomer (lightGCN + lqtransfomer)

  : lightGCN을 이용하여 뽑은 임베딩 벡터를 사용하여 LastQuery Transfomer에 input으로 사용한다면 더 높은 성능을 보일 것으로 예상되어 두 모델을 함께 사용했습니다. 실제로 lightGCN 모델을 단순 사용한 것의 성능(0.7371)보다 lightGCN 임베딩 결과를 LastQuery Transfomer 모델에 넣어 함께 사용한 것의 성능(0.7632)이 더 높았습니다. (0.7371→ 0.7632)

4-3. 수행 결과 정리

A. 시연 결과

Model	Details	AUROC (제출)	Accuracy (제출)
SAKT	userID, assessmentItemID, answerCode 만 사용, Output에 sigmoid 적용	0.6425	0.6075
CatBoost	21개 피쳐 사용, learning rate = 0.01, iteration = 10000	0.8456	0.7769
TabNet	Sweep, Feature Selection	0.8160	0.6989
LSTM	16개 피처 사용, 연속형변수 batchnorm, data augmentation	0.7828	0.7124
LSTMATTN	16개 피처 사용, 연속형변수 batchnorm, data augmentation	0.7747	0.7177
LastQuery Transformer	16개 피처 사용, 연속형변수 batchnorm, data augmentation	0.7785	0.7151
LGCN+LSTM	LGCN 임베딩(assessmentItemID, testId) 사용, Sweep	0.7915	0.7339
LGCN+lqTransfomer	LGCN 임베딩(assessmentItemID, testId) 사용, Sweep, Feature Selection	0.8032	0.7285
LGBM	27개 피처 사용, 주요 피처 - Time Lag 관련 피처, learning rate = 0.023	0.8507	0.7715
XGBoost	32개 피처 사용, 주요 피처 - elo Rating, learning rate = 0.023	0.8481	0.7661

B. 앙상블

각 모델의 예측값에 가중치를 부여하는 Weighted Voting을 사용하였습니다.

LGBM+CatBoost, LGBM, XGBM, TabNet, LSTM, LGCN (가중치 : 40 - 30 - 15 - 5 - 5 - 5)

→ 최종 LB AUC : 0.8486 LB Accuracy : 0.7823

5. 자체 평가 의견

좋았던 점

1. 지난 대회의 회고록을 바탕으로 이번 대회를 시작하며 목표한 바를 이룬 점
2. 실시간 소통이 가능한 협업툴(게더타운, 줌)을 이용하여 좋은 협업을 이끌어 낸 점
3. DKT 과제에 적합한 다양한 모델(Sequential 모델, Boosting 모델, GNN 모델 등)을 시도한 점
4. Git 커밋 컨벤션을 통해 가독성 좋은 커밋메시지를 작성한 점
5. 앙상블을 통해 모두의 노력을 담은 제출결과가 좋은 성능을 보인 점
6. Wandb, MLflow, AutoML 등 다양한 툴을 사용해본 점
7. 데일리 스크럼을 통해 팀원들의 진행 상황을 기록하고 공유할 수 있었던 점

아쉬웠던 점

1. 훌륭한 모델로 최고의 성능까지 끌어내지 못했던 점

2. Public LB auroc 값에 중요도를 높게 매김에 따라 실제로 뛰어난 성능을 가진 모델들이 사용되지 못한 경우가 있다.
3. 각자 한 모델을 깊게 파는 구조로 진행되어, 서로 다른 모델에 대해 깊게 경험해보지 못한 점
4. 앙상블에 대한 실험을 충분히 하지 못한 점