NAVER Boostcamp AI Tech 7th Level.2 Project

수도권 아파트 전세가 예측 모델

Wrap-up Report
Recsys-08 TEAM RECONOVA

김영찬 | 박광진 | 박재현 | 박세연 | 배현우 | 조유솔

Oct.27 2024

목차

001 프로젝트 개요

프로젝트 개요

프로젝트명

수도권 아파트 전세가격 예측 모델

한 줄 소개

누구에게나 중요한 주거 환경,
전세가 예측 알고리즘 AI 모델을 개발하여
쉽게 정보를 제공해보자

기 간

2024.10.02 ~ 2024.10.24 (3주)

예측 Target

전세 실거래가 예측

팀 구성 및 역할

팀원 수

6명

팀원 별 역할

PM – 조유솔
EDA 및 도메인 리서치 – 전원
데이터 – 박광진, 박세연
모델링 – 김영찬, 박재현
엔지니어 – 배현우

001 프로젝트 개요

협업 및 개발 Tool

코드 공유

Github

실험 관리

Notion Teamspace (실험 개요)
Weight & Bias (실험 결과)

개발 환경

IDE: VSCode 및 Google Colab
작업 환경: 로컬, Google Colab 및 원격 GPU 서버(Linux V100)

Workflow

Notion

- 회의록 – 매일 진행한 회의내용을 기록하여, 나중에도 쉽게 찾아볼 수 있도록 하였음.
- 실험 일지 – 각자 진행한 실험의 의도와 결과를 기록하여 다른 팀원이 쉽게 파악할 수 있도록 하였음.

Github

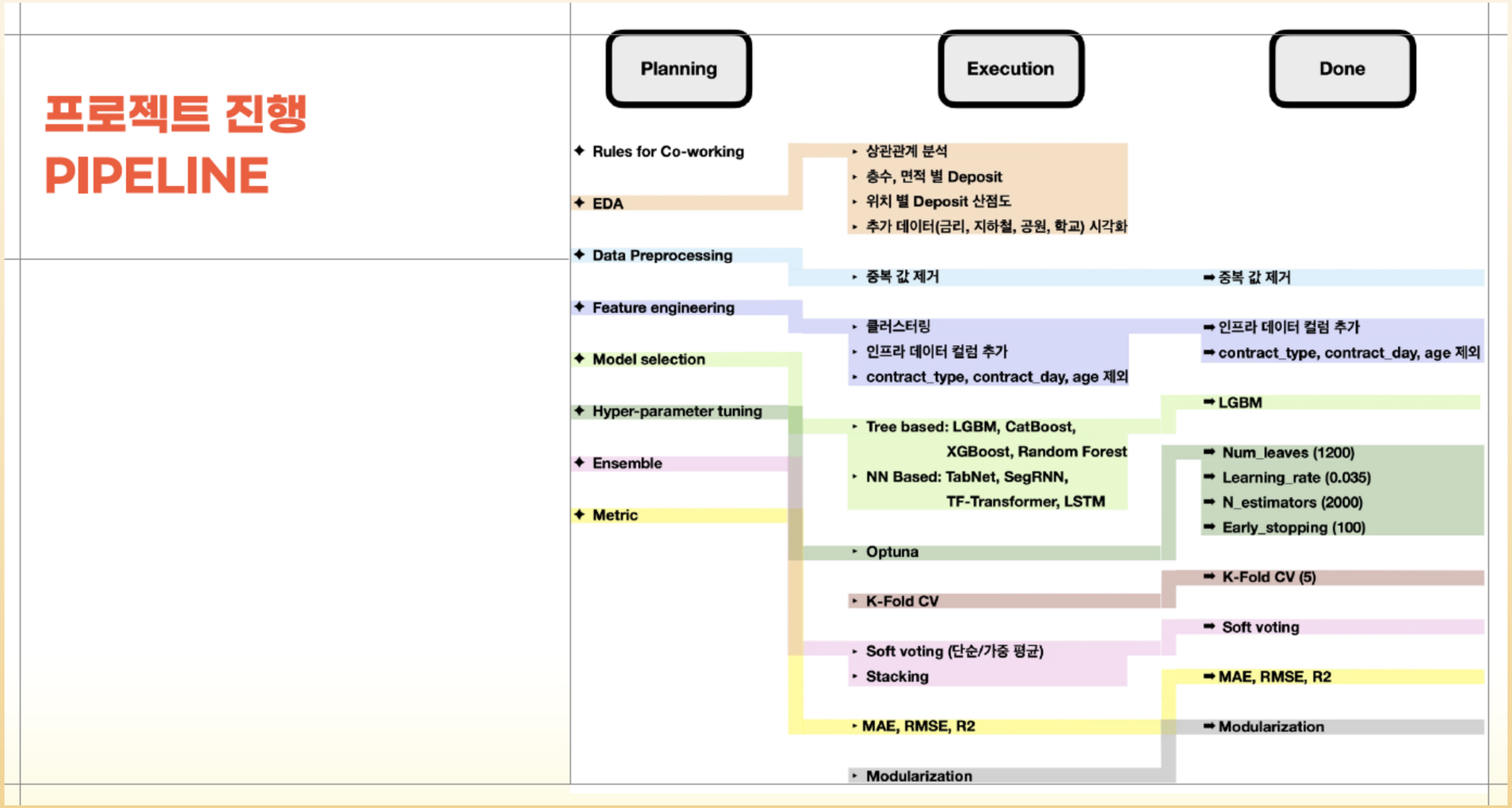
- 코드 버전 관리 – 프로젝트 전 과정에서, 개발 내용을 기록 및 공유함.
- Code Convention – 다음과 같은 Code Convention을 정하여서 같이 협업하는 팀원간 소통이 원활하도록 하였음(변수 및 함수 명명 규칙 / 함수 생성 / 코드 라인 길이 / 에러 핸들링 / 임포트 순서)
- 전략적인 Branch 활용
  main: 최종 결과물 제출용 branch
  develop: 실제로 개발 버전을 관리하는 작업용 기본 branch
  feat-XX: 기능별, 실험별 브랜치 개발하고, 완료 후 develop에 병합하는 branch
- Commit Convention – 유다시티 Convention으로 통일하여 다른 팀원의 커밋 내용을 쉽게 알아볼 수 있도록 하였음.

Weight & Bias

- 실험 결과 로깅: 각자 진행한 실험의 내용을 Wandb에 기록하여, 다른 팀원의 실험결과를 손쉽게 트래킹할 수 있도록 하였음.

001 프로젝트 개요

프로젝트 진행 파이프라인



002 데이터셋 소개

데이터셋 개요 및 구성

소 개

전세 실거래가 예측을 위한 데이터셋

구 성

크게 3가지 종류의 데이터로 구성되어있다.

구 분	csv 데이터	내 용
모델의 학습 및 성능평가	test.csv train.csv	· 대회 훈련 및 추론용
Feature Engineering	subwayInfo.csv schoolInfo.csv parkInfo.csv	· 지하철, 학교, 공원의 위치정보
	interestRate.csv	· 월별 금리정보 (2018.12-2024.05)
예측 결과	sample _submission.csv	· 예측 결과 제출 Sample

데이터 Column 구조

test.csv train.csv

각 10, 11(+deposit)개 column으로 구성

- index: 인덱스 번호
- area_m2: 면적(제곱미터)
- contract_year_month: 계약연월
- contract_day: 계약일
- contract_type: 계약 유형
- floor: 층수
- built_year: 건축 연도
- latitude: 위도
- longitude: 경도
- age: 건물의 나이

+ deposit: 전세 실거래가(Target)

	index	area_m2	contract_year_month	contract_day	contract_type	floor	built_year	latitude	longitude	age	deposit
0	0	84.9981	201906	25	2	9	2019	37.054314	127.045216	0	17000.0
1	1	84.9981	202003	26	2	20	2019	37.054314	127.045216	1	23000.0
2	2	84.9981	202003	28	2	8	2019	37.054314	127.045216	1	23000.0
3	3	59.3400	201907	15	2	1	1986	36.964647	127.055847	33	5000.0
4	4	59.8100	201904	12	2	6	1995	36.972390	127.084514	24	1800.0
...	...	...	...	...	...	...	...	...	...	...	...
1801223	1801223	114.8126	202311	25	0	5	2010	37.528394	126.659398	13	39000.0
1801224	1801224	101.9088	202311	28	0	6	2010	37.528394	126.659398	13	38000.0
1801225	1801225	114.7900	202312	3	0	19	2010	37.528394	126.659398	13	37000.0
1801226	1801226	101.9088	202312	4	1	15	2010	37.528394	126.659398	13	34400.0
1801227	1801227	114.7900	202312	16	0	10	2010	37.528394	126.659398	13	45000.0

1801228 rows × 11 columns

002 데이터셋 소개

데이터 Column 구조

subwayInfo.csv parkInfo.csv schoolInfo.csv

“ 각 위치정보 column + 기타 column ” 으로 구성

	latitude	longitude
0	37.759380	127.042292
1	37.541021	126.971300
2	37.529849	126.964561
3	37.514219	126.942454
4	37.513342	126.926382
...	...	...
695	37.378384	126.645168
696	37.386007	126.639484
697	37.393054	126.634729
698	37.399907	126.630347
699	37.467048	126.707938

700 rows × 2 columns

	latitude	longitude	area
0	37.509628	127.628406	856.0
1	37.508443	127.627414	847.0
2	37.493844	127.509326	1276.0
3	37.496021	127.408216	3300.0
4	37.496164	127.412326	394.0
...	...	...	...
17559	35.834179	128.619666	2642.0
17560	35.832236	128.618209	2654.0
17561	35.834941	128.626945	3315.0
17562	35.822580	128.625316	2237.0
17563	35.819858	128.635862	1644.0

17564 rows × 3 columns

	schoolLevel	latitude	longitude
0	elementary	37.703889	127.540156
1	elementary	37.676874	127.600664
2	elementary	36.987340	129.399471
3	elementary	34.808753	126.456974
4	elementary	35.854580	127.003365
...	...	...	...
11987	high	36.108950	128.355632
11988	high	36.507932	126.621914
11989	elementary	37.493705	126.900119
11990	middle	35.214234	129.011813
11991	middle	35.265227	129.220461

11992 rows × 3 columns

	공통 column	추가 column
subwayInfo.csv	latitude, longitude : 각 위치정보 column	-
schoolInfo.csv		schoolLevel: 초.중.고 여부
parkInfo.csv		area: 공원 면적
interestRate.csv	year_month: 연월 / interest_rate: 금리	

sample_submission.csv

index, deposit column으로 구성

	index	deposit
0	0	0
1	1	0
2	2	0
3	3	0
4	4	0
...	...	...
150167	150167	0
150168	150168	0
150169	150169	0
150170	150170	0
150171	150171	0

150172 rows × 2 columns

interestRate.csv

월별(year_month) + 금리 (interest_rate) column으로 구성

	year_month	interest_rate
0	202405	3.56
1	202404	3.54
2	202403	3.59
3	202402	3.62
4	202401	3.66
...	...	...
61	201904	1.85
62	201903	1.94
63	201902	1.92
64	201901	1.99
65	201812	2.04

66 rows × 2 columns

003 데이터 전처리 및 Feature Engineering

Main Feature: Cluster

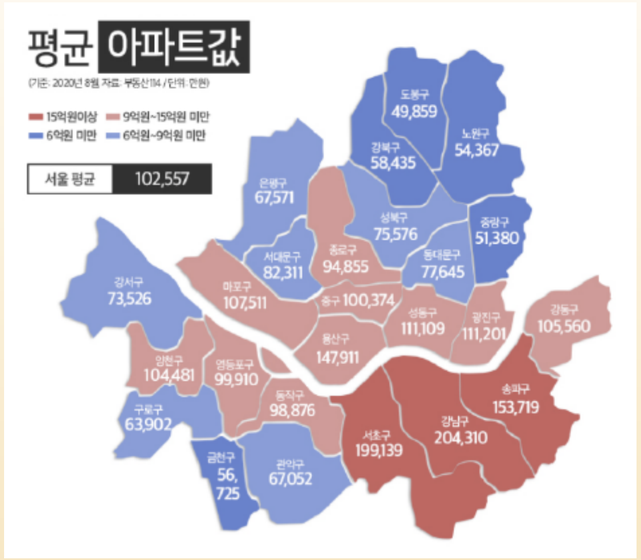
중점을 두게 된 계기

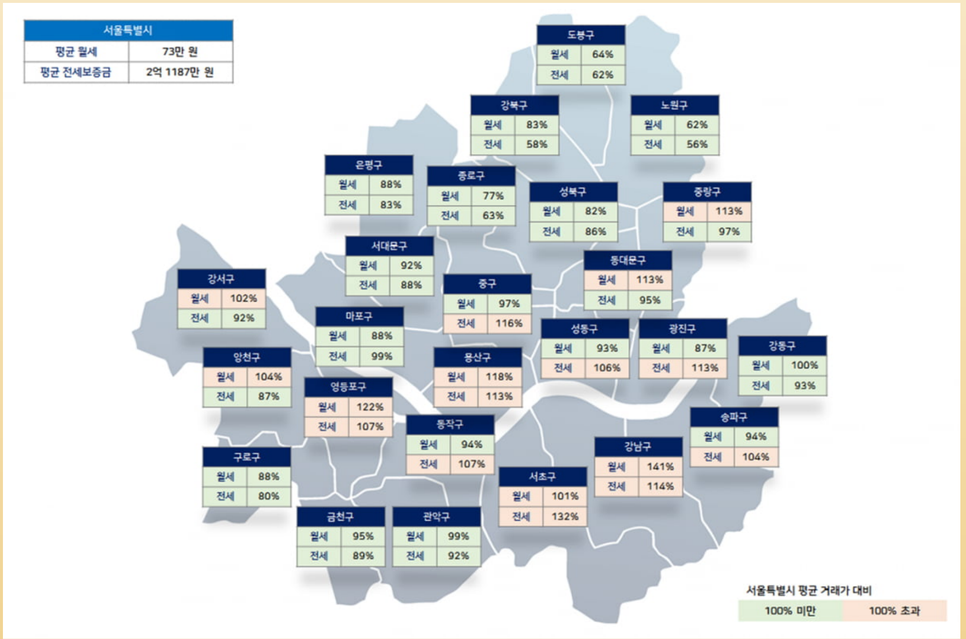
집값의 핵심 = “어느 동네에 위치했나?”

아파트 가격을 결정하는 요소는, 첫째도 둘째도 ‘입지’이다.

일자리, 교통, 학군, 녹지 등 모든 것이 갖추어진 곳이 곧 좋은 입지를 결정한다.
그리고 이를 모두 만족하는 지역은 많은 이들이 알고 있듯, “강남”이다.
대한민국 아파트 가격을 이끄는 강남3구(강남구, 서초구, 송파구)를 비롯하여 서울 동남권 지역을 중심으로 높은 아파트 가격을 형성하고 있다. “강남 접근성”이란 말은 주변 사람들 및 뉴스에서 쉽게 접할 수 있는 용어라 이해할 수 있을 것이다.
또한, 매매가격이 높을 수록 전세가격도 높아지는 것은 당연지사일 것이다.

이러한 사실에 기반하여,
“서울 동남권 지역에 가까울 수록 전세 가격이 높게 형성되어 있을 것이다.”
는 가정을 두고 cluster 변수를 생성 및 실험해 보았다.



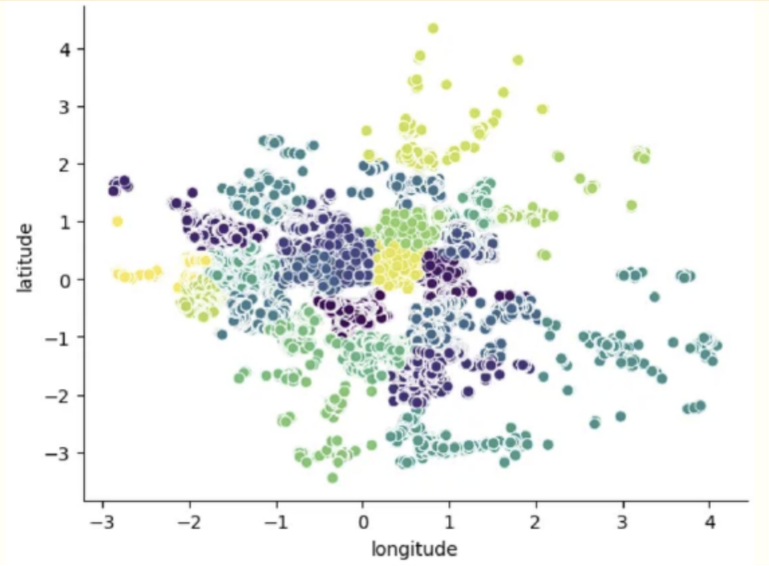


003 데이터 전처리 및 Feature Engineering

Main Feature: Cluster

적용한 Clustering 기법 - 1

K-Means Clustering

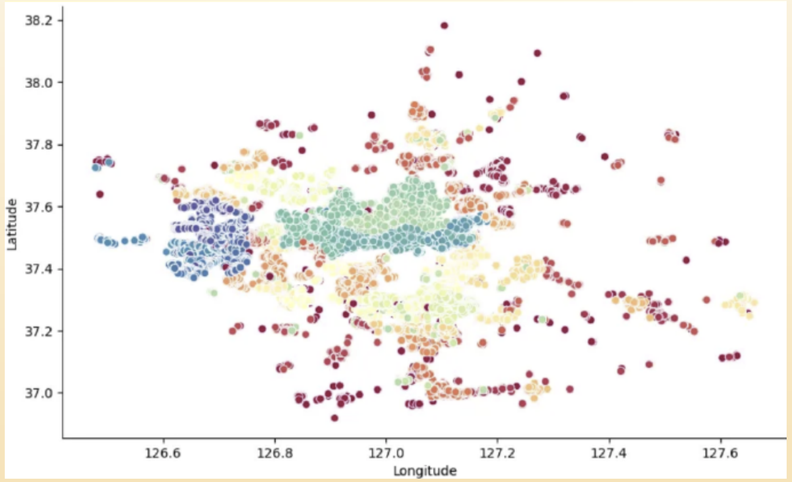


- 위도, 경도 변수를 기본으로 사용하되,
  k값/인코딩 방식/클러스터링 활용 변수를 달리해가며 거리 기반 군집화

- 무엇보다 거리기반 Clustering 기법이었기에 기존 가정과 가장 부합하였고, 활용하기에 간단하고 대용량 데이터에도 빠르게 적용 가능하여 선택하였다.

적용한 Clustering 기법 - 2

DBSCAN Clustering



- 하나의 샘플 주변에 충분히 많은 이웃 샘플이 존재해야 클러스터로 포함되고 주변 샘플이 적으면 해당 새샘플은 이상치로 분류

- 데이터의 밀도 차이를 고려하여 더 밀집된 의미 있는 그룹들로 클러스터링을 할 수 있을 것 같아 선택

- 하이퍼파라미터 값을 조정하며 클러스터의 갯수, 이상치로 분류되는 샘플의 양을 조절하려고 시도

003 데이터 전처리 및 Feature Engineering

Main Feature: Cluster

DBSCAN 적용 과정

1. X_train에 DBSCAN을 fit
2. X_train에 fit된 Core point로 X_test 군집화
3. 생성된 cluster 변수를 모델에 적용

train 데이터와 test 데이터 전체를 이용하여 clustering을 적용할 경우,
DATA LEAKAGE 발생

=> 훈련 데이터를 통해 클러스터링을 수행,
이후 테스트 데이터들은 가장 가까운 훈련 데이터의 라벨을 부여

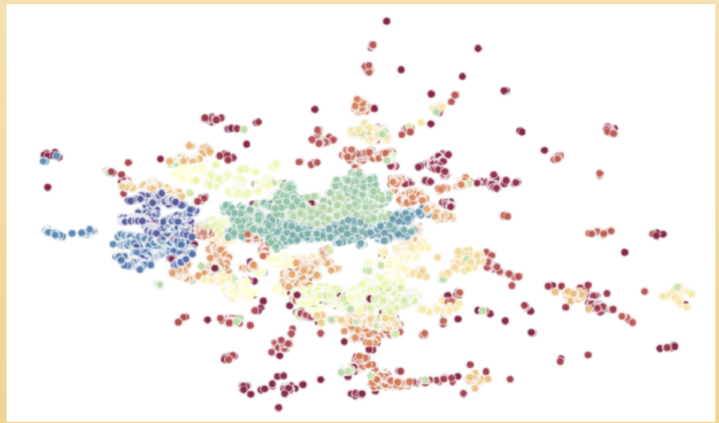
DBSCAN 적용 결과

vs. K-Means, Vanila

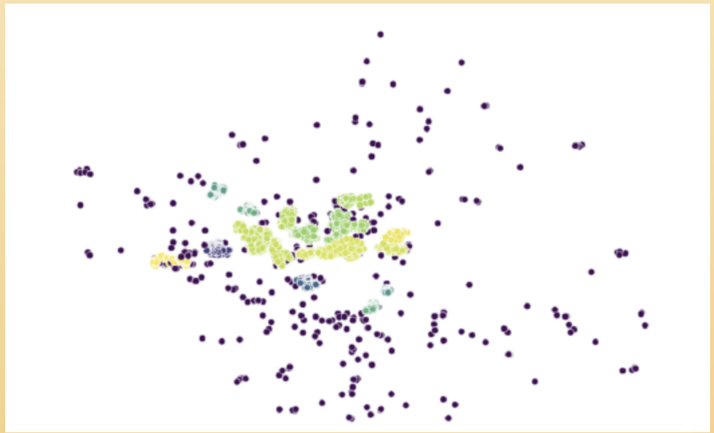
	MAE	RMSE
Vanila	6563	10481
K-Means	6490	10444
DBSCAN	6488	10407

- 적용 결과, DBSCAN이 가장 성능이 좋았음.
- 하지만, 성능 재현이 일정하게 이루어지지 못하고
  모델, Hyperparameter 등 조건에 따라 성능이 불안정한 모습을 보였음
  => 최종 반영 X

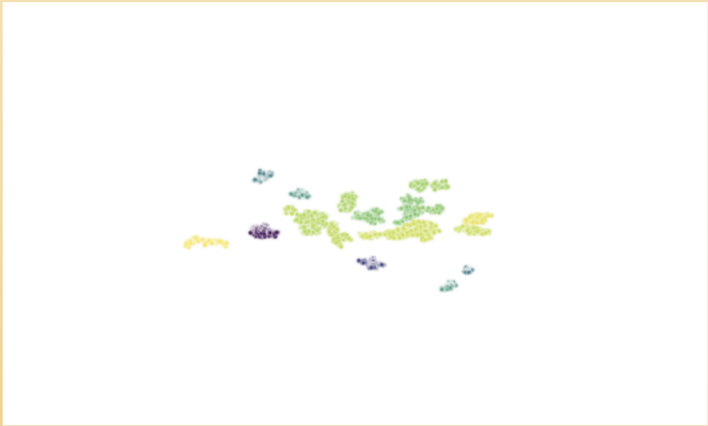
전체 클러스터



일부 상위 Cluster + 이상치



일부 상위 Cluster



003 데이터 전처리 및 Feature Engineering

Feature Engineering

기본 제공 Features

train.csv에 제공된 column

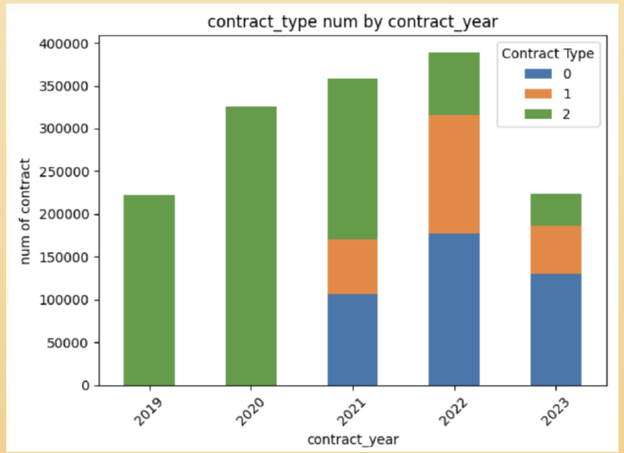
여러가지 요인을 파악하여, 사용할 column을 분류하여 적용함

포함 Features

- area_m2
- contract_year_month
- floor
- built_year
- latitude
- longitude

미포함 Features

- contract_day
  : Feature Importance 확인 결과, 중요도가 낮아서 제외하였음.
- age
  : built_year와의 강한 다중 공선성이 우려되어, 실험을 통해 둘 중 더 낮은 성능을 발휘하는 age 변수를 제거하였음.
- contract_type
  : 시간에 따른 노이즈가 포함된 데이터로 판단하여 제외하였음. 2019, 2020년도는 계약 유형을 알지 못하는 결측치라고 볼 수 있기 때문에 결측치가 많고 분포 확인 시 가격 예측에 큰 도움이 되지 못할 것 같아 제외하였음.



contract_type: 계약 유형
(0: 신규 1: 갱신 2: 모름)

003 데이터 전처리 및 Feature Engineering

Sub Features

가공한 Features

By. Feature Engineering – 모델링에 반영한 Feature

주어진 Feature를 가공하고 도메인 지식을 활용하여, 성능이 우수한 Feature를 실제로 반영함.

공원(park_density)

아파트와 공원 데이터에서 제공하는 위도와 경도를 이용해 하버사인 연산으로 거리를 구한 후, 각 아파트 데이터에 대해 특정 반경 내에 있는 공원의 총 면적을 계산한 후 그 특정 반경의 단면적을 나누어서 밀도를 계산

지하철(nearest_subway_distance)

EDA를 통해, 가장 가까운 지하철 역과의 거리가 가까울 수록 전세값의 median 값이 커지는 경향성을 확인함. 이러한 사실에 기반하여,
1. 가장 가까운 지하철 거리
2. 가장 가까운 지하철 거리를 범주화
3. 근처 특정 거리(0.1km~3km) 내 지하철 갯수

이렇게 3가지 경우에 대해 실험해본 결과, '가장 가까운 지하철 거리만 넣는 것'이 가장 좋은 성능을 발휘하여, 해당 Feature를 적용함.

초/중/고 아파트 근처 학교 갯수(school: elementary, middle, high)

유사하게, 아파트 근처 학교의 갯수도 영향이 있을 것이라 판단, 초/중/고등학교별 아파트 근처에 있는 학교 갯수를 변수로 만들어 활용함. 초등학교는 어린 아이들이 다니는 학교이기에, 중/고등학교에 비해 비교적 가까운 거리에 있어야 한다는 도메인 지식을 적용하여 다음과 같이 변수를 만들어 적용함.
- elementary: 아파트 근처 1km 이내 초등학교의 갯수
- middle: 아파트 근처 5km 이내 중학교의 갯수
- high: 아파트 근처 5km 이내 고등학교의 갯수

아파트 근처 가장 가까운 초/중/고 거리(nearest_(school)_distance)

아파트와 학교가 떨어진 거리가 영향이 있을 것이라 판단, 초/중/고등학교 별 아파트와 떨어진 거리를 변수로 만들어서 활용함.

003 데이터 전처리 및 Feature Engineering

Sub Features

가공한 Features

By. Feature Engineering – 모델링에서 반영하지 않은 Feature

주어진 Feature를 가공하고 도메인 지식을 활용해 보았지만,
Feature Importance 및 성능을 체크해본 결과 역효과가 확인되어 제거하였음.

age_categorical

age 피처와 도메인 지식을 활용하여, 아파트 전세가격의 일반적 생애주기를 표현해본 Feature
age = 0–1: 신규 입주 아파트
(일반적으로 전세계약은 2년단위 + 입주장은 전세공급이 많음 → 2년간은 일반적인 주변 시세보다 저렴)
age = 2–5: 신축 아파트_전성기 – 전성기 (전세값 최고점)
age = 6–10: 신축 아파트_인기유지
age = 11–20: 준신축 아파트
age = 21–29: 구축 아파트
age > 30: 재건축 아파트 – 재건축 가능 연한(=30년) 도래 (전세값 최저점)

seasonal_month

일반적으로 이사수요가 많다고 알려진 시기를 고려하여,
수요가 많은달 vs 적은달을 이진 분류한 Feature

subway_within_1km **subway_density**

Subway 데이터를 활용하여 가공한 두 피처
– 가장 가까운 지하철 거리를 범주화
– 근처 특정 거리(0.1km~3km) 내 지하철 갯수

is_transfer_station

아파트 근처에 환승역이 있을 경우 비환승역이 있을 때에 비해 전세값이 높을 것이라는 가정을 고려한 Feature
지하철 데이터를 활용하여, (위도, 경도)쌍이 중복되는 횟수를 파악하여 환승역 여부를 확인하였음.

interest_rate_diff

금리는 전세값과 반비례 할 것이라 예상, 이전 금리와의 차이가 중요하다고 판단하여 차분하여 만들어본 Feature

003 데이터 전처리 및 Feature Engineering

Sub Features

가공한 Features

Feature Engineering 수행 결과,

포함 Features

- elementary
- middle
- high
- nearest_elementary_distance
- nearest_middle_distance
- nearest_high_distance
- nearest_subway_distance
- park_density

미포함 Features

- seasonal_month
- age_categorical
- is_transfer_station
- cluster
- subway_within_1km
- subway_density
- interest_rate_diff

004 Modeling

Tree 모델

시도한 Model

Baseline Model – LightGBM

Plus – CatBoost, XGBoost, Random Forest

기술지표

MAE – 평균 절대 오차, 대회 평가 지표

- Mean Absolute Error
- 예측값과 실제값의 차이가 평균적으로 얼마나 벌어졌나를 보여줌

RMSE – 제곱평균근 오차

- Root Mean Squared Error
- $\sqrt{MAE}$, 큰 오차에 민감하여, 큰 오차를 줄이는 데에 유리함

R^2 – 결정계수

- Coefficient of Determination
- 모델이 설명할 수 있는 데이터의 민감 비율을 나타낸다.
- 1에 가까울 수록, 모델이 데이터를 잘 설명한다는 반증임

Baseline - LightGBM

개요

Baseline 단일모델로, 데이터 전처리와 Feature Engineering으로만 꽤나 큰 성능 향상을 이루었음.

문제점

기존 Hyperparameter를 활용하다 보니, 모델 표현력이 부족하다는 문제점 발견

해결책

모델이 새로운 피처를 충분히 표현할 수 있도록, 파라미터를 재설정함.

실험 전략

– 모델 이해 및 특성을 반영하여 실험을 진행함

– 실험 시간 증가를 감수하더라도, 가설 검증을 확실히 하는 데에 집중

결론

모델의 이해 및 데이터에 따른 모델 설정의 중요성을 깨달을 수 있었음.

004 Modeling

모델 표현력 개선을 위한 방법: Optuna

Optuna 도입

앞서 발견된 문제점을 해결하기 위해, 단일 모델 및 Kfold CV 최적화를 시도하였다. 해당 과정에서, Hyperparameter의 원활한 Tunning을 위해 Optuna를 도입하였다.

Optuna 적용

- Baseline 뿐 아니라, CatBoost/XGBoost/Random Forest에도 적용하였다.
- 모델 별 공식 Docs를 읽으면서 각 세부 파라미터를 파악하였고, 튜닝할 부분을 선정하였다.
- 오래 걸리는 작업인 만큼, 밤새 서버 별로 Optuna를 실행시켜 파라미터 서칭을 진행하였다.

Optuna 파라미터 탐색 전략

- 큰 범위에서 Step 단위로 좁혀 나가며 초기 탐색(특히, iteration)
- 최적값 근처 세부 영역에서 추가 탐색

Optuna 적용 성과

결론적으로, 성능 개선에 크게 기여할 수 있었다.

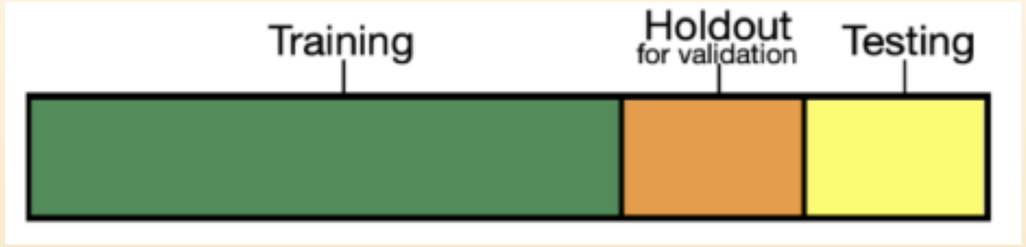
004 Modeling

일반화 성능을 위한 방법: K-Fold Cross Validation with Soft Voting

K-fold CV 도입

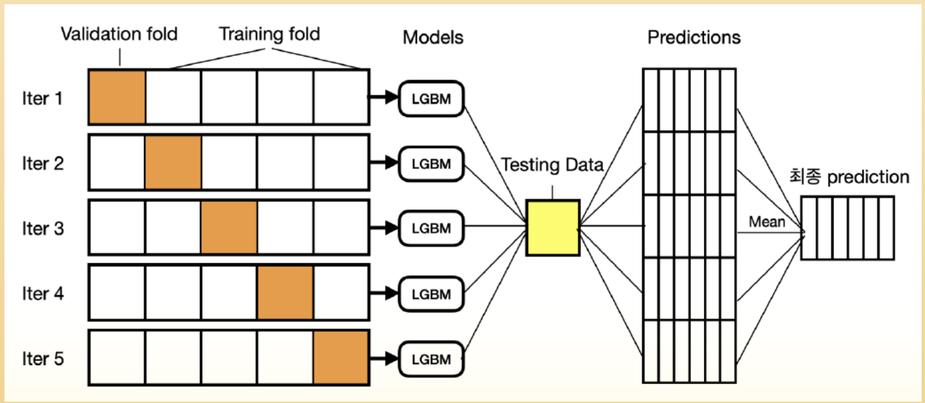
전체 데이터 학습 시, 문제점 발생할 수 있다는 사실을 인지하였음.

- 기존에는 Holdout Data로 검증한 파라미터와 Iteration으로 재학습 진행
- 이 때, 전체 데이터 재학습 부분에서는 마치 Black-Box처럼 미지의 영역이 될 수 있다는 사실이 문제로 작용할 수 있음.



K-fold CV 적용

- 동일한 크기를 지닌 5개의 Fold로 분할하여 교차 검증을 진행하였음.
- Validation Fold로 Underfitting 및 Overfitting 여부를 판별하고, Early Stopping을 할 수 있음.



Soft Voting

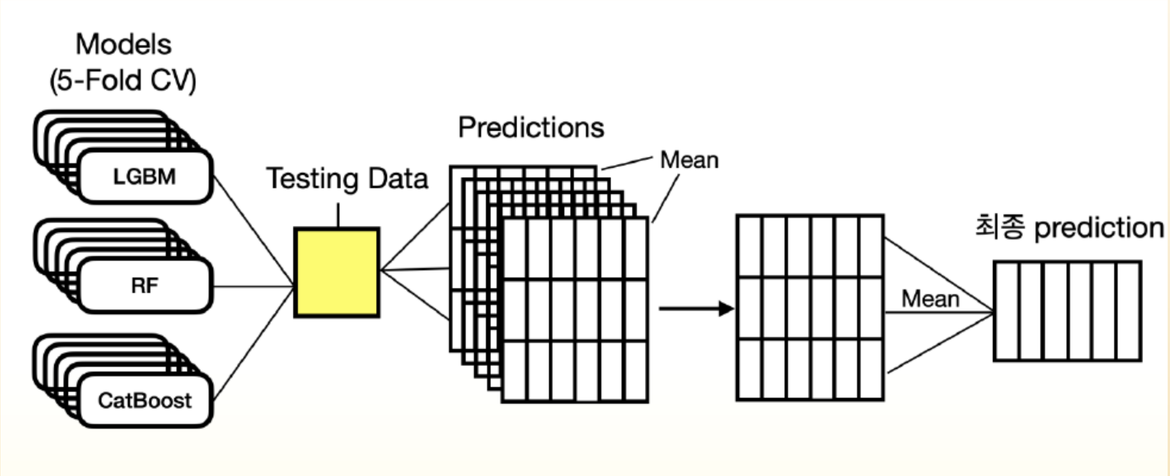
- 각 기본 분류기의 클래스별 예측 확률을 평균내는 방식이다.
- K-Fold 이후 Soft Voting을 거침으로서, 전체 데이터에 대한 학습을 하면서도 트래킹이 가능해졌다.

004 Modeling

Ensemble: 성능 극대화 시도

Ensemble 시도

- 각 모델별로 5-fold를 soft voting한 후, 모델끼리 Soft Voting하는 방법 시도
- 추가로, stacking 실험까지 진행



Stacking Ensemble

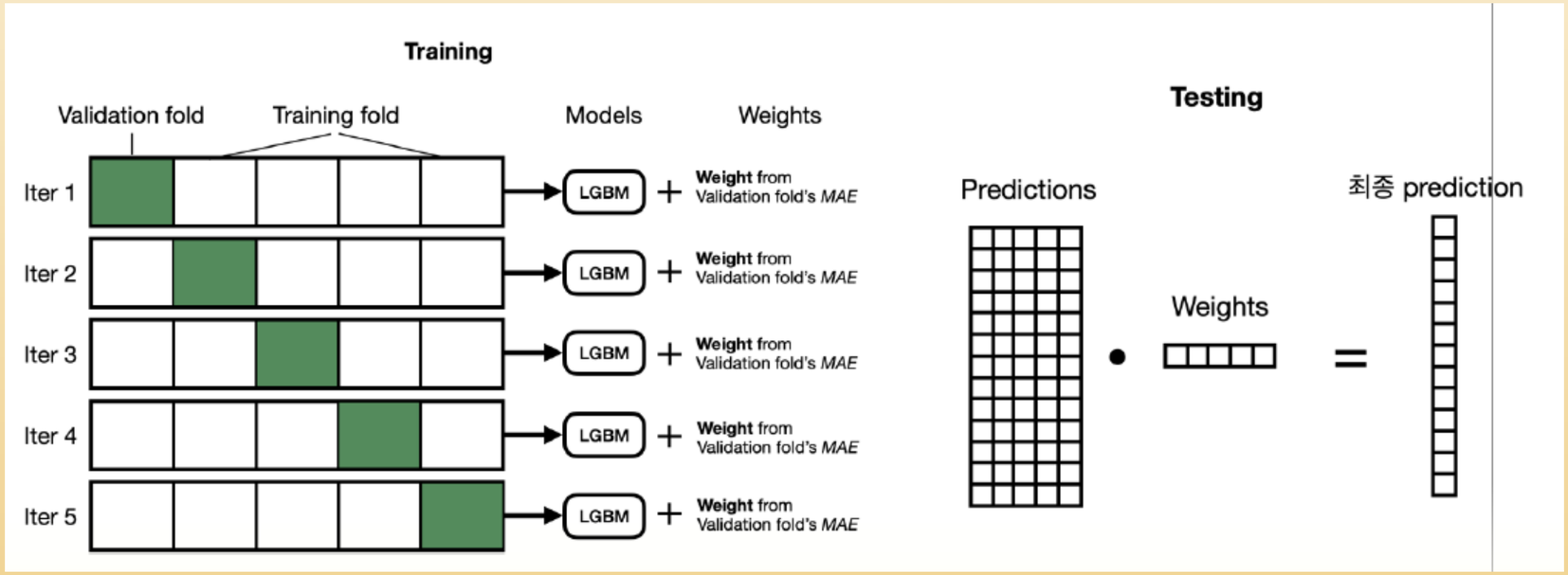
- 실험 배경: LGBM, Random Forest, CatBoost를 기본 모델로, Linear Regression을 메타 모델로 활용해 스태킹 앙상블 모델을 설계하면, 다양한 모델의 구조적 특성을 반영할 수 있어 예측 성능이 극대화될 것이라고 판단했다.
- 실험 과정1: LGBM, Random Forest, CatBoost를 기본 모델로 KFold CV를 적용해 예측값을 softvoting하고, 메타 모델 학습에 사용했다.
- 실험 과정2: 3가지 기본 모델의 KFold CV 예측값에 가중치를 적용해 메타 모델 학습에 사용했다.
- 실험 과정3: LGBM 단일 모델의 KFold CV 예측값에 가중치를 적용해 메타 모델 학습에 사용했다.
- 실험 결과: LGBM 단일 모델로만 CV를 실행한 경우 가장 성능이 좋았다. 이는 사용한 세 모델 모두 트리 기반의 유사한 알고리즘으로, 메타 모델에 다양한 패턴을 제공하지 못해 성능 개선이 이루어지지 않은 것으로 보인다.

004 Modeling

Ensemble: 가중치 Voting

가중치 Soft Voting

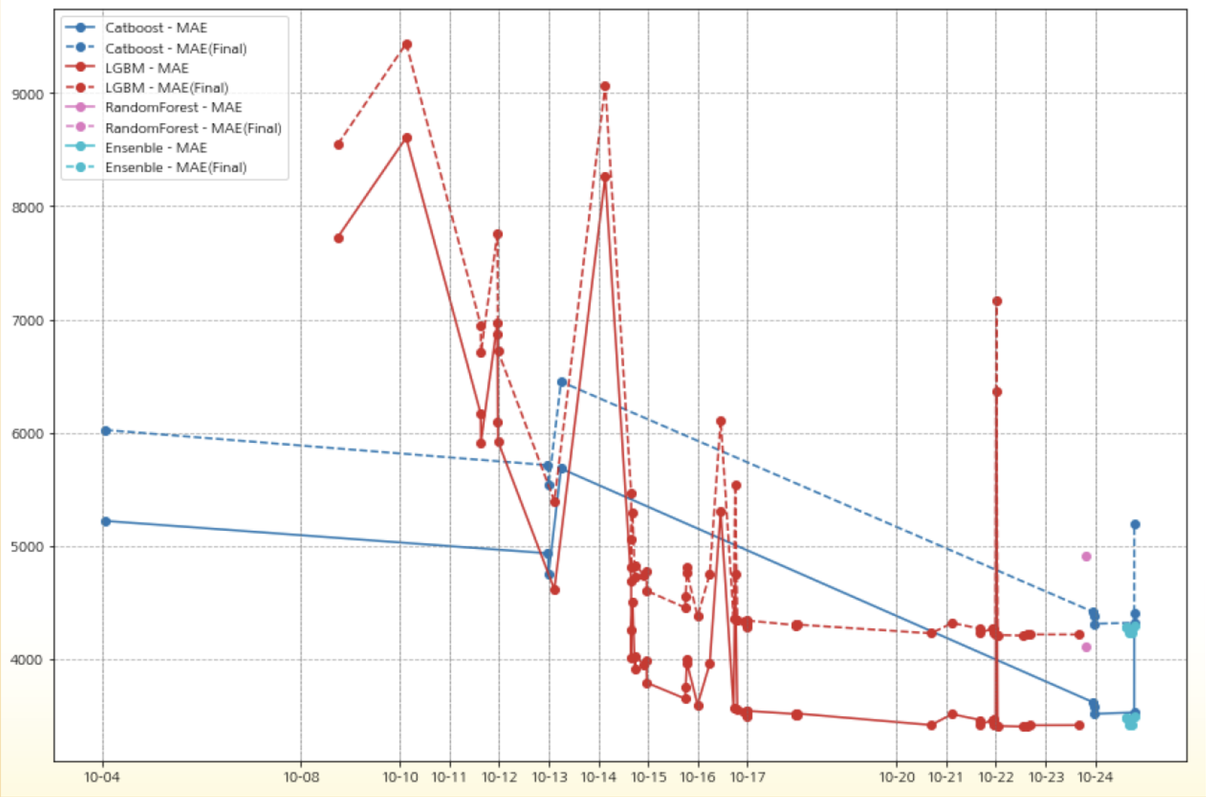
- K-Fold 수행 시, 각 iteration 마다 validation fold의 성능(MAE)을 이용하여 해당 iteration에서 학습된 모델의 가중치를 계산하였다.
- K-Fold 수행 중 각 단계의 Validation Fold만으로 가중치를 만드는 것은 train Data에 과적합될 가능성이 크다고 판단하였다.
- Validation set, Public test set, Private test set은 모두 다른 성질을 가지고 있을 가능성이 많기에, 합리적 추측이라 예상하였다.
- 정말 운이 좋아서 3가지 데이터가 비슷한 분포에 있어 큰 차이가 없다면 유리하지만, 항상 그렇지는 않을 것이라고 생각하였다.
- 이번 대회의 데이터셋이 비슷한 분포를 가지고 있어 가장 좋은 성능을 낸 것으로 추측



- k-fold를 수행할 때 각 iteration마다 validation fold의 성능(MAE)을 가지고 해당 iteration에서 나오는 모델의 가중치를 계산
- mae는 낮을 수록 좋은 지표이므로, 역수를 취해 아래와 같이 가중치를 계산한 뒤 모든 가중치를 합쳐서 1이 되도록 정규화

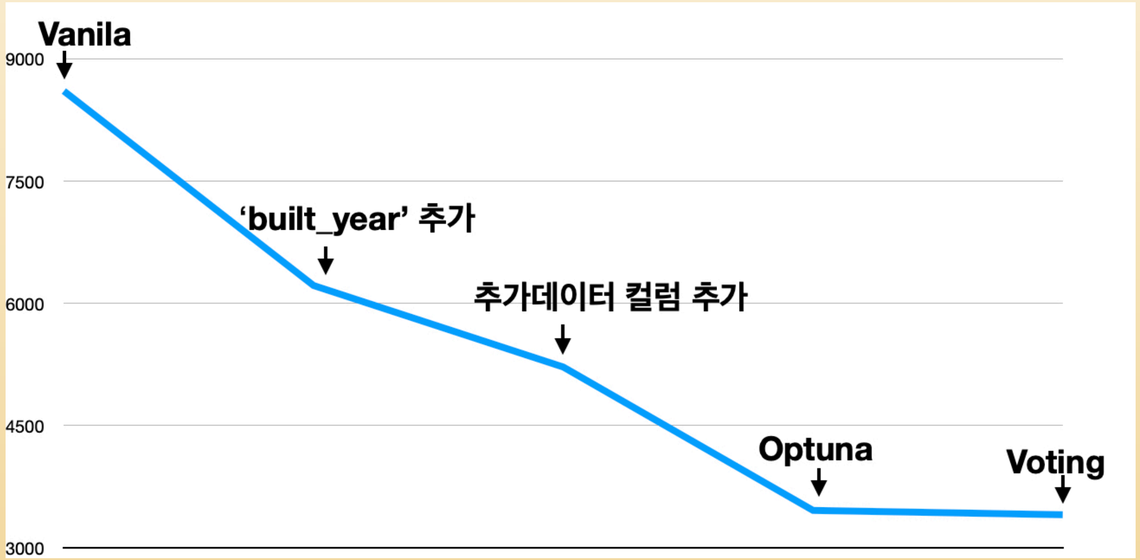
005 결과 해석

결과



대회 기간 중, 모델 별 진행한 실험들의
성능(MAE) 일지를 그래프로 표현

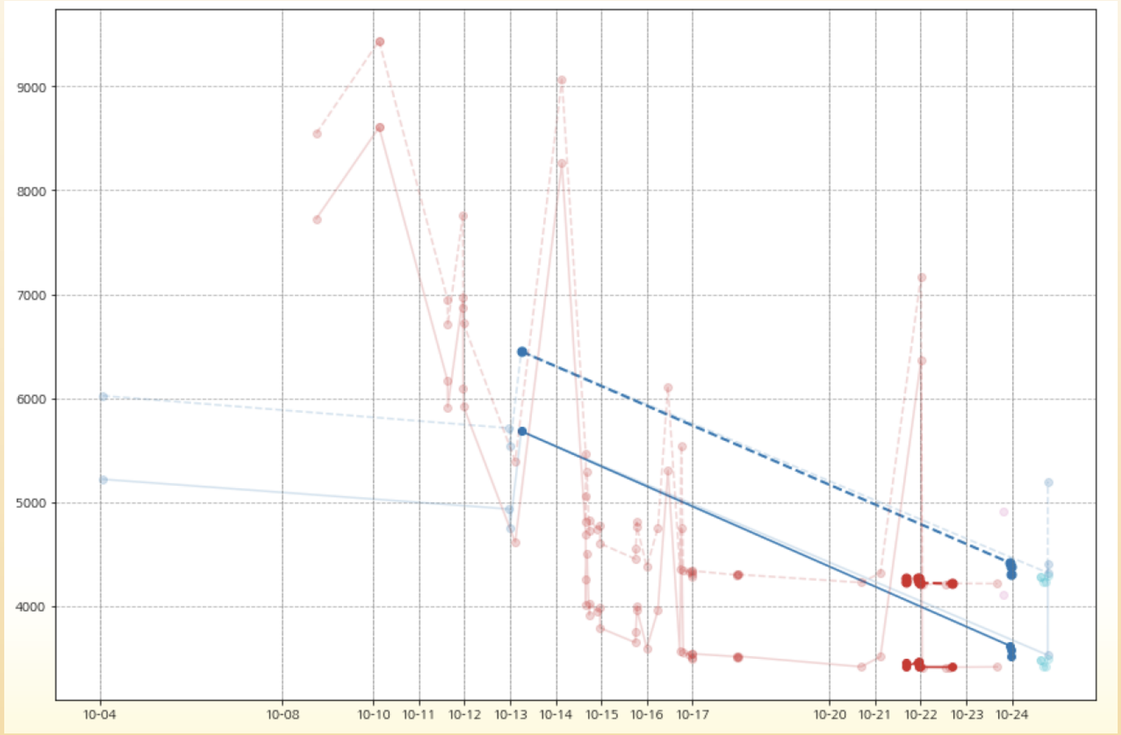
성능 개선 변천사(단일 모델, LGBM)



005 결과 해석

성과 및 의의

Optuna 적용 후, 성능이 눈에 띄게 개선됨을 확인



파란 선: Catboost에 Optuna 적용 후 MAE가 5000 –> 3000으로 감소

빨간 점: LightGBM은 Optuna 적용 후 MAE가 3500보다 커지지 않음

협업 및 다양한 실험을 통한 성능 극대화

- Convention 및 여러 협업 도구를 활용하여 다양한 실험을 원활하게 진행할 수 있었다.
- 다양한 모델링을 시도하였고 동시에 Optuna를 통해 하이퍼 파라미터 최적화하여 성능을 개선하였다는 점에서, 추후에도 다시 적용 가능할 정도로 모델의 성능을 개선시키는 flow에 익숙해졌다. 동시에 주력으로 사용한 Tree 모델들(LGBM, CatBoost, XGBoost)에 대한 이해도를 높일 수 있었던 계기가 되었다.
- 앙상블이 무조건 성능이 오를 것이라고 기대하고 진행하였지만, 예상대로 성능 개선이 이루어지지 못하였다. 학습 데이터와 테스트 데이터가 크게 다르지 않아 효용을 보지 못하였고, 비슷한 구조의 모델들만 앙상블하여 다각적으로 서로의 단점을 보완해줄 수 있는 앙상블의 장점을 살리지 못하였다.

006 회고

팀 회고

프로젝트 초기 목표 달성률

- 협업: 초기 목표(모듈화, 컨벤션, 이슈 및 브랜치 관리)를 모두 이룸. 또한 팀의 공동 목표를 세우고, 진행하고자 하는 방향을 의논하여 맞추어 감.
- 데이터 전처리: 초기 목표(WandB 실험 관리, 다양한 피처 엔지니어링)를 모두 이루긴 하였으나 실험을 통해 제거된 피처가 많았다는 아쉬움이 있었음. 또한, 다양한 feature를 사용하고자 하였으나, 결국 대부분의 feature가 제거되었다는 점에서 아쉬움이 남음.
- 모델링: 다양한 Tree모델(LGBM, CB, XGB, RF)에 다양한 기법(Optuna, KFCV)을 적용해 보았다는 점에서 의미가 있었음. 하지만 시간 상 다양한 NN기반 모델을 적용해보지 못했고, 적용해 본 모델마저 fine-tuning 하지 못해 낮은 성능을 보인 점이 아쉬움.

우수하게 수행된 요소

- 협업 Tool 별로 용도를 나누어 체계적인 협업과 실험관리를 진행함에 따라 피처 실험이 원활하게 진행됨. 수작업 대신 Optuna를 통한 하이퍼파라미터 튜닝으로 모델 성능 효과적으로 향상
- 팀의 강의 진행 상황을 공유하는 것을 비롯하여, 어느 정도 기간을 정해 프로젝트 과정의 진도를 맞추어 진행
- 초기 목표 설계대로 실험을 진행하고 진행 상황을 공유함. Baseline을 공유해 효과적으로 협업하고 필요에 따라 팀원 간 역할 분배가 이루어짐

진행중 발생 이슈 및 개선 방안

- 자치구 정보 부재. 행정동 정보 부재 → cluster 변수 생성
- built_year와 age 변수의 다중공선성 문제 → 둘 중 하나만 사용
- 초기에는 사용이 편리하다는 이유만으로 .ipynb 파일로만 작업하였음 → Github과의 연동성 및 체계적 모듈화를 위해 최종 결과물은 py파일로 모두 변환

006 회고

개인 회고 [김영찬]

"The most effective way to do it, is to do it." – 에밀리아 에어하트

프로젝트 기간 중 두 번의 여행으로 공부의 연속성이 끊겼다. 복귀 후 팀원들과의 격차는 분명했고, 피어세션에서 오가는 생산적인 실험들 속에서 나는 진도 따라잡기에만 급급했다. 이런 상황이 반복되며 슬럼프가 찾아왔다.

'과연 이 길이 진정 내가 원하는 것일까?' 이 질문의 답을 찾아 여러 사람들에게 조언을 구했다. 그러다 올해 1월 친구와의 대화와 우연히 본 한 장의 사진에서 답을 찾았다.

"더 이상 재미를 쫓지 말고, 그저 매일 하라. 결과는 따라올 것이다."

이 말을 실천하기로 했다. 끝까지 이해되지 않는 강의는 잠시 미뤄두고 프로젝트에 집중했다. 데이터 처리 연구가 이미 많이 진행된 상태였기에, 코드 모듈화와 새로운 모델 실험을 맡았다.

하지만 여전히 어려움은 있었다. 자신감 부족으로 모듈화한 코드를 발표하지 못했고, 데이터와 실험 기록 확인이 미흡해 부적절한 모델을 사용하기도 했다. 그렇게 우여곡절 끝에 경진대회는 끝이 났다.

비록 실패의 경험이 더 많았지만, 팀에 기여하고 싶은 열망과 현재의 부족한 실력을 개선하고자 하는 의지가 생겼다. 이것들을 발판 삼아 다음 경진대회를 준비하려 한다. 지금은 오직 그 생각뿐이다.

006 회고

개인 회고 [박광진]

이번 프로젝트는, 개인적으로 관심있고 자신있는 도메인 분야에 관련한 프로젝트여서 여러가지 Feature를 활용한 성능개선에 개인적 목표를 두고 이번 프로젝트를 진행하였다. 하지만 결과는, 기대만큼 성능 개선을 이루지 못하여 아쉬웠지만 추후에 추가적으로 진행해보고픈 마음을 담아 개인 회고를 작성해 본다.

우선, 저번 프로젝트에서 겪은 시행착오를 바탕으로 팀 협업 Flow는 매우 많이 개선되었음을 느꼈다. '개발자스럽게 Github 사용하기' 강의를 통해 팀원 모두가 Github을 통한 코드 공유 및 버전 관리에 많이 능숙해졌고, Wandb를 익히고 사용해보며 실험내용 로깅 및 타 팀원의 실험 기록 트래킹을 더욱 원활히 할 수 있었다. 개인적으로 Github 및 기타 Tool을 하나도 다룰 줄 몰랐던 사람으로서, 많이 성장함을 느낄 수 있었다.

도메인 지식을 첨가한 Feature Engineering을 수행하며, 3가지 Feature를 생성하고 성능을 실험해 보았다. 기존 Feature를 가공한 것도 있었고, 도메인 지식만을 활용하여 아예 새로운 Feature를 생성해본 것도 있었다.

- 기존 Feature 가공: age_categorical(column 'age'), is_transfer_station(subwayInfo.csv)
  - age_categorical: 아파트 전세가격의 생애주기를 고려하여 만들어본 Feature. 주기 Term에 따라 6단계로 분류하여, 0~5까지 6개로 분류한 Feature

    > 0 - age = 0-1: 신규 입주 아파트 - 일반적으로 전세계약은 2년단위 + 입주장은 전세공급이 많음 → 2년간은 일반적인 주변 시세보다 저렴
    > 1 - age = 2-5: 신축 아파트_전성기 - 전성기 (전세값 고점)
    > 2 - age = 6-10: 신축 아파트_인기유지 - 일반적으로 관리 상태 양호, 인기 유지 (고점 근방 유지)
    > 3 - age = 11-20: 준신축 아파트 (전세값 하락세 도래)
    > 4 - age = 21-29: 구축 아파트
    > 5 - age > 30: 재건축 아파트 - 재건축 가능 연한(=30년) 도래 (전세값 최저점)

  - is_transfer_station: 아파트 근처의 지하철역이, 환승역일 때가 환승역이 아닐 때보다 선호도가 높다는 가정으로 만든 Feature. 환승역일때 vs 비환승역일때를 이진 분류하여 가공
- 새로운 Feature 생성: seasonal_month
  - seasonal_month: 일반적으로 이사수요가 많다고 알려진 시기를 고려하여,
    수요가 많은달 vs 적은달을 이진 분류한 Feature

    > 이사 수요가 많은 시기(이사 시기 기준)
    > **2월** - 기업들의 인사 발령 및 자녀 학교 교육을 위한 이사 수요
    > **5월** - 날씨 좋을 때 이사를 하려고 함(일반 이사 수요)
    > **10월** - 자녀들의 학교 배정을 위한 이사 수요(학교 배정을 위한 주소 확인이 10월에 이루어짐)
    > ⇒ 보통 수요가 있는 시기의 2~3달 전에 계약이 이루어짐

하지만, 적용해본 성능은 썩 좋지 못하여서 아쉬웠고 다 끝난 지금 돌아보니 수정해서 적용해보고픈 것이 떠오른다. 관심있는 도메인인 만큼, 복습하면서 꼭 추후에 적용해봐야 겠다.

- age와 built_year가 다중 공선성의 위험이 있어서 제거 하였는데, 그렇다면 age_categorical을 built_year를 활용하여 만들어 보았으면 어떨까
- 집 근처 역이 아무리 환승역이어도, 강남3구 근처에 위치한 아파트 전세가격보다 비쌀까? 그렇다면 우리가 힘주어 가공해 보았던 cluster 변수와 같이 섞어서 사용해 보았다면?(같은 권역에 위치할 때, 환승역세권 전세값은 비쌀 수 있으니)

https://cat-jury-a5e.notion.site/_-12da5ea423548028b9cbdec3ee238736?pvs=4

006 회고

개인 회고 [박세연]

1. EDA 및 추가 데이터 활용: 이번 데이터셋은 컬럼 수가 적고, 행이 엄청 많았다. 결측치도 없었다. 무엇보다 EDA를 수행한 후 추가 컬럼을 만들기 위해 데이터를 가공할 때 효율적인 코드를 작성하지 않으면 시간이 매우 오래 걸렸다. 이런 문제들을 직면하면서 잔머리도 굴리고 병렬처리도 하면서 계산 비용이 적은 효율적인 코드를 작성할 수 있어서 매우 뜻 깊었다. 추가 데이터 중 금리 데이터를 잘 활용하고 싶었는데, “금리가 오르면 집 값이 떨어질 것이고 금리가 내리면 집값이 오를 것이다.” 라고 생각해서 그래프를 그려보았는데, 실제로 금리에 따라 집값의 변동이 일어나는 부분이 보이는 것 같아 계약년월에 맞는 금리 정보를 추가한 뒤 실험했을 때 성능 향상은 미미하였다. 그래서 사용하지 못했다. 지하철 데이터를 보고 생각이 든 것은, 가장 가까운 지하철과의 거리를 재보는 것이었다. 또한 공원 데이터에는 면적 정보가 나오기에, 특정 반경 이내의 공원의 면적의 합에서 총 면적을 나누어 밀도를 계산하고자 했다. 그리고 학교 데이터는 학교의 종류(초, 중, 고) 정보가 있었다. 처음에는 가장 가까운 종류별 학교 거리를 추가했고, 이후에는 특정 거리 이내의 학교의 종류별 개수를 셌다. 문득 학교 종류별 개수를 셀 때 기준 거리를 다르게 해보면 좋을 것 같아 많이 실험해 본 결과, 1, 5, 5km일 때가 가장 좋은 성능이 보였다. “가장 가까운” 아이템을 찾기 위해 최근접 이웃을 찾는 알고리즘이 필요했다. 그래서 KDTree를 이용했다. 이 자료구조를 이용하면 가장 가까운 지하철과의 유클리드 거리가 반환된다. 여기서 111정도를 곱하면 km와 비슷해지지만 나는 추가로 하버사인 연산을 했다. 그러면 사실 연산량이 너무 늘어서 오래 걸린다. 그래서 나는 kdtree로 각 아파트 데이터와 가장 가까운 몇 개의 지하철/공원/학교 데이터만 가져와서 그 데이터들과만 하버사인 연산을 진행했다. 이렇게 컬럼을 추가하고 정말 많은 실험을 통해 최적의 컬럼을 추리게 되었다. 기존 데이터에서도 계약 일자를 쓰지 않으면 성능이 매우 올랐다. 이런 과정에서 성능에 초점을 맞추어야 하는 것인지, 아니면 “내 생각에 이 피처는 유의미하다”라고 느끼면 그냥 놔두는 게 맞는지? 헷갈렸다.
2. 앙상블 기법: 성능을 올리는 방법을 생각하다, 5-fold CV 후 생긴 5개의 모델의 가중치를 계산하여 테스트 시 반영해보는 실험을 했다. 각 단계의 validation fold에 대한 MAE의 역수를 취해 가중치를 계산하고, 각 가중치의 모든 가중치의 합을 나누어 합이 1이 되도록 했다. 테스트때 각 모델들의 예측과 가중치를 내적하여 최종 결과를 계산하는 방법을 사용했다. 실험을 진행하면서도, validation data에 과적합 되는 방식이라는 것을 알고 있었다. 운이 좋게도 성능이 잘 나오는 것처럼 보였지만, 위험이 큰 방법론이라고 생각했다.
3. 협업: 깃을 이용해서 좋았지만, 컨벤션을 정했음에도 자꾸 까먹고 다르게 작성한 부분이 있어 좀 아쉽다. 하지만 Discussion, Issue, Pull request도 잘 활용하고, 기능을 개발 할 때 브랜치를 만들어서 작업하는 등 모두 이전에 배웠었던 거지만 희미해져 있었고, 실제로 잘 활용하지 않았던 기능들에 익숙해지는 계기가 되었다. 또한 Wandb를 이용하며 실험을 기록할 수 있어 좋았고, 노션도 자세한 내용을 기록하면서 활용도가 높아진 것 같아 좋았다.
4. 나는 여태까지 모델의 성능을 높이는 것을 가장 우선적으로 생각했는데, 결국 대회를 하는 목적이 좋은 성능을 위한 것이 아닌지? 아니면 연구 과정을 담아내는 것이 더 중요한 지? 자꾸 헷갈린다. 도메인 지식으로 만든 컬럼이 좋은 성능을 내면 그건 근거가 있는 부분이 아닌 것일지, 그렇다면 어떻게 근거를 들 수 있는지? 많은 생각이 드는 대회였다.

006 회고

개인 회고 [박재현]

목표

모든 실험에서 가설을 설정하고 검증하는 방식을 사용할 것

성능 저하가 일어나면 그 원인을 파악하고 새로운 실험 계획을 수립 것

이를 통해 최고 성능을 달성할 것

실험과정

LGBM 모델을 baseline으로 설정하고, 하이퍼파라미터 최적화와 피처 엔지니어링을 병행하였다. Leave-One-Out Permutation Importance 방식을 활용해 새로운 피처를 추가하거나 기존 피처를 하나씩 제거하면서 성능 변화를 기록하였다. 새로운 피처는 도메인 지식을 포함한 가정을 세워서 만들어보았고 성능 향상이 일어나면 추가하고 향상이 없으면 그 원인을 분석해, 시도할 수 있는 다음 실험 계획을 세다. 피처 선택 기준으로는 Feature Importance를 사용했으며, RFE를 참고하였다. MAE를 중심 메트릭으로, RMSE와 R^2 값을 보조 지표로 활용했으며, 피처 제거 시, RMSE 대비 MAE 감소가 미미한 경우 이상치에 민감한 것으로 판단하였다.

LGBM 최적화가 이루어진 때부터 리더보드 기준 1위를 기록했으며, 최종 제출까지 이를 유지했다. 이후, 팀원들과 논의를 거쳐 동일한 피처 집합을 CatBoost와 Random Forest 모델에도 적용하였다. 개별 모델별로 최적의 피처를 선택하는 것이 이상적이지만, 모델별 최적의 피처 선택은 우선순위에서 다소 밀려, 동일 피처 집합을 기준으로 실험을 진행하였다.

모델링

트리 기반 모델 외에도, SegRNN과 같은 딥러닝 기반 시계열 모델을 시도하였다. 초기에는 데이터가 시계열 특성을 지닌다고 판단해 paperswithcode에서 SOTA 모델인 하나인 SegRNN을 선택해 논문을 읽어보고 구현해봤으나, 성능은 기대에 미치지 못다. 이는 데이터에 시계열 정보가 부족하고, 범주형 데이터가 많았기 때문이라고 보았다. 이번 실험을 통해 데이터의 특성을 충분히 이해하고 나서 적절한 모델을 선택하는 것이 중요하다는 점을 깨달았다. 다음 프로젝트에서는 데이터 유형별로 최적화된 피처와 모델을 선택하도록 할 계획이다.

앙상블

앙상블 방식은 배깅, 스태킹, 부스팅 순으로 실험하였으며, CatBoost, Random Forest, LGBM을 선정해 Soft Voting과 Stacking을 적용하였다. 추가적으로 5-fold CV와 소프트 보팅을 활용하여, 각 모델별 예측값을 통합해 봤으며, 모델별 MAE 값을 기준으로, 성능이 우수한 모델에 가중치를 높이는 방식도 시도하였다. 그러나, 앙상블 실험 모두 k-fold 소프트 보팅을 활용한 lgbm 단일 모델보다 성능이 낮았다. 그 이유는 앙상블에 사용한 모델이 모두 트리 기반이라 유사한 방식으로 작동하여 데이터의 다양한 측면을 충분지 학습하지 못했기 때문이라고 해석했다. 여러 트리 기반 모델을 사용하기보단 트리 기반 모델 중 가장 성능이 좋았던 lgbm모델만 사용한 것이 더 효과적이었던 것 같다. 다음 프로젝트에서 앙상블을 사용한다면 트리 기반 모델 외에 다양한 모델을 조합해봐야겠다. 검증 과정은 k-fold를 활용하였으나 시계열 정보가 포함되어 데이터 유출이 있었을 것 같다. 다음 프로젝트에 사용할 수 있다면 TimeSeriesSplit 검증도 도입해볼 예정이다.

좋았던 점

1. 팀 협업을 위해 Git Convention을 마련하고 이를 준수하며 프로젝트를 진행한 점
2. Baseline을 통일하여 실험 공유가 원활히 이루어진 점
3. Wandb를 활용하 팀원의 실험 결과를 체계적으로 정리하여 참고할 수 있었던 점
4. 모듈화를 통해 실험을 자동화한 점

아쉬운 점과 개선 사항

모델링 과정에서 데이터의 특성을 충분히 이해하지 못한 점이 아쉬다. 데이터 구조와 분포를 면밀히 분석하고 그에 맞는 모델링 전략을 수립하는 것이 중요한 것 같다. 데이터 유출이 발생할 부분도 추가적으로 공부해야겠다. 또, 프로젝트가 끝나고 트리 기반 모델을 공부하다가 catboost 모델은 ordered target encoding이 시간 순서에 의존한다는 점을 알게 되었다. 딥러닝 모델 도입을 시도했으나 성능 향상에 한계가 있었으며, 다음 프로젝트에서는 다양한 모델을 시도해봐야겠다. 또, 앙상블 실험에서도 데이터의 특성을 고려해 다양한 모델을 조합하는 방식으로 앙상블의 장점을 극대화해야겠다. 마지막으로, 실험 및 파일 관리를 위한 팀 단위의 협업 도구의 필요성을 느꼈다.

006 회고

개인 회고 [배현우]

결론부터 말하자면 아쉬운점은 있었어도 목표를 초과달성했던 성공적인 프로젝트였다

이번 프로젝트엔 팀적으로도 그랬지만 개인적으로 특히 저번 프로젝트에서 아쉬웠던 부분들을 개선하려고 집중하였다. 협업에 관련한 부분으로는 명확한 베이스라인 없이 각기 다른 코드로 실험을 하니 팀원 간 공유나 코드의 적용이 무척 어려웠던 점, 깃헙을 프로젝트 후반에 도입하여 제대로 써보지 못한 점이 있었다. 배울 시간이 없어 완디비, 옵튜나, 스윕 같은 툴들도 사용하지 넘어갔었다. 마지막으로 끝까지 노트북 파일로 실험을 진행하였는데, 두 분의 멘토님들이 모두 파이썬 파일 구조로 변경하여 최종 결과물을 만들어 보라고 하셔서 그 부분도 도전했다. 지금 나열된 내가 해결하려고 했던 부분들이 다 엔지니어링 측면 인것 같아 역시 나는 엔지니어의 성향이 있구나가 느껴진다.

이번에는 프로젝트를 시작하며 깃헙 트리 전략을 짜고, 템플릿을 작성하고 이슈와 풀리퀘스트를 활용하였다. 전과 비교하여 변경 사항 추적이 매우 쉬워졌고, 팀원이 진행 중인 실험의 코드를 쉽게 찾아 갈 수 있어 협업의 능률이 상승하였다.

베이스라인 코드의 부재로 오는 실험 결과 공유의 어려움과 시간 낭비를 없애고자 프로젝트 초기에 베이스라인을 작성하였다. WandB 역시 공식 문서와 예시 코드를 보며 베이스라인 코드에 추가한 뒤 팀 계정을 생성하여 실험 기록을 남기었더니 실험 공유 부분에서 엄청난 편의와 효율의 향상이 있었다.

중간에 모델 별 성능 향상을 위해 옵튜나와 스윕도 문서를 보고 적용하여 진행해보았는데, 옵튜나의 파라미터 탐색 알고리즘이 더 우수한 것 같아 옵튜나로 하이퍼 파라미터 서치를 진행하였다.

하이퍼 파라미터 서칭 과정도 지나고 프로젝트의 마지막 단계까지의 그림이 머릿속에 그려진 후로는 노트북 파일 기반의 실험에서 벗어나 파이썬 파일로 구조를 짜는 과정도 진행하였다. 전 기수들, 다른 팀 그리고 널리 쓰이는 템플릿의 프로젝트 파일 구조를 참고하여 우리의 파일 구조를 설계 하였는데 막막했지만 결과적으로 잘 설계가 이루어진 것 같다.

위에 까지가 전 프로젝트에서 아쉬웠던 부분을 개선했던 부분이고, 프로젝트를 진행하며 하나의 모델에 대한 이해를 높여 더욱 잘 사용한 경험과 검증 데이터 없이 전체 데이터로 학습할 때의 불확실성을 해결하려고 KFold 교차 검증을 도입 했던 적이 있다.

아쉬웠던 부분들을 정말 많이 개선하고 새로 배운 부분들도 있지만 아쉬운 부분들 역시 남게 되었는데, 다른 팀의 프로젝트 발표를 보고나니 더욱 크게 느껴졌다. 한 모델을 집중해서 사용하느라 여러 모델을 골고루 깊이 사용해보지 못했고, 다른 모델들로는 점수를 높이기가 힘들게 느껴졌다. 또한 이러한 연유 때문인지 아니면 방법의 문제였던 건지 이 모델들로 앙상블을 진행 했을 때에 성능이 오히려 낮아지는 결과를 얻었다. 나의 역량과 주어진 시간으로는 이번 대회에서는 여기 까지 해볼 수 있었던 것 같고 다음 기회에 모델링과 앙상블 부분도 마저 열심히 배워보고 싶다.

006 회고

개인 회고 [조유솔]

https://velog.io/@u_sol/Naver-AI-Tech-두번째-프로젝트를-마치며