

31기 여름방학 세미나

1팀

김동환
노정아
성준혁
이정환
이지원
하희나

INDEX

1. 데이터 확인 및 EDA
2. 데이터 전처리
3. 모델링
4. 최종 모델
5. 한계 및 의의

1

데이터 확인 및 EDA

1

데이터 확인 및 EDA

데이터 확인 - shape, type

```
1 df = pd.read_csv("training_set.csv") # 훈련세트 csv 파일 열기
2 df
```

	ID_code	class	X1	X2	X3	X4	X5	X6	X7a	X7b	...	X105c	X105d	X105e	X105f	X105g	X105h	X105i	X105j	X106	X107
0	train_1	0	3490	0.0	8.000000e+01	212.0	0.0	0.0	0.0	0.0	...	12986.0	7612.0	17044.0	13682.0	19594.0	74564.0	5270.0	0.0	0.0	0.0
1	train_2	0	92	0.0	1.400000e+01	10.0	0.0	0.0	0.0	0.0	...	512.0	120.0	332.0	344.0	964.0	1414.0	0.0	0.0	0.0	0.0
2	train_3	0	10	0.0	1.800000e+01	2.0	4.0	6.0	0.0	0.0	...	132.0	22.0	32.0	24.0	26.0	54.0	0.0	0.0	0.0	0.0
3	train_4	0	390692	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
4	train_5	0	156758	NaN	2.130706e+09	408.0	0.0	0.0	0.0	0.0	...	1652370.0	852932.0	1494660.0	1026644.0	779338.0	480460.0	784792.0	33582.0	0.0	0.0
...
54995	train_54996	0	23910	NaN	8.680000e+02	694.0	0.0	0.0	0.0	5932.0	...	341764.0	81988.0	151692.0	128676.0	112698.0	144182.0	596.0	0.0	0.0	0.0
54996	train_54997	0	170978	NaN	NaN	NaN	NaN	NaN	0.0	0.0	...	1461416.0	522106.0	821572.0	632884.0	715346.0	3099426.0	194594.0	0.0	NaN	NaN
54997	train_54998	0	14	0.0	0.000000e+00	0.0	0.0	0.0	0.0	0.0	...	124.0	22.0	48.0	56.0	12.0	18.0	0.0	0.0	0.0	0.0
54998	train_54999	1	437486	0.0	NaN	NaN	0.0	0.0	0.0	96944.0	...	1088628.0	576962.0	1260296.0	1452080.0	2642866.0	2303888.0	91546.0	0.0	0.0	0.0
54999	train_55000	0	10574	NaN	0.000000e+00	NaN	0.0	0.0	0.0	0.0	...	216636.0	185804.0	155976.0	20684.0	2884.0	274.0	160.0	0.0	0.0	0.0

55000 rows × 172 columns

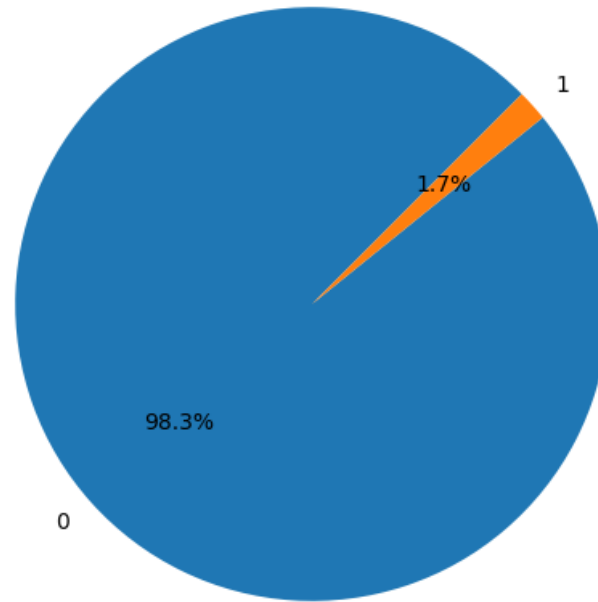
■
■
■
■

① 대용량 데이터 ② 모두 수치형 ③ 변수명이 없는 masked data

1

데이터 확인 및 EDA

데이터 확인 - 종속변수 비율



⋮

종속변수 비율이 약 99:1로, **클래스 불균형**이 심각함을 확인함

데이터 확인 및 EDA

18 JANUARY 2005

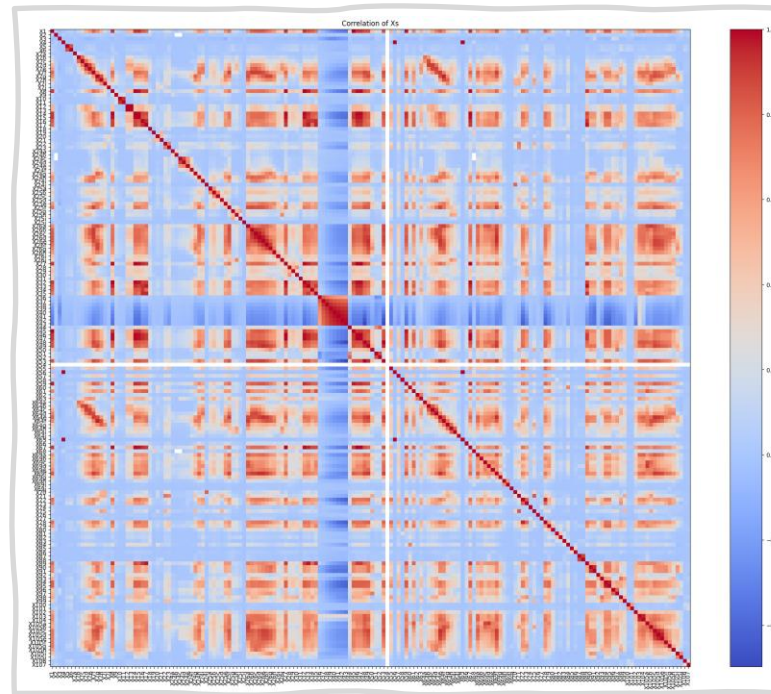


결측치도 많기 때문에 결측치 제거 및 보간에 대한 고려가 필요함

1

데이터 확인 및 EDA

EDA - 상관계수 확인



⋮

변수 간 상관이 높은 조합이 많으며, 이를 이후 변수 선택 시 고려해야 함

EDA - 상관계수 확인



추가적으로 상관계수를 정렬 후 특징을 파악함

① 양의 상관 matrix를 정렬

→ 자기 자신이 아닌데도 상관 계수가 1인 조합이 존재

② 음의 상관 matrix를 정렬

→ 절댓값이 가장 높은 조합도 -0.49에 그침

③ 상관 절댓값 0.5 이상인 경우를 따로 확인

변수 간 상관이 높은 조합이 많으며, 이를 이후 변수 선택 시 고려해야 함

1

데이터 확인 및 EDA

EDA - 정규성 검정

① Shapiro-Wilk Test

	검정통계량	p-value	정규성 만족?
X1	0.384142	0.0	False
X2	NaN	1.0	True
X3	NaN	1.0	True
X4	NaN	1.0	True
X5	NaN	1.0	True
...
X105h	NaN	1.0	True
X105i	NaN	1.0	True
X105j	NaN	1.0	True
X106	NaN	1.0	True
X107	NaN	1.0	True

[170 rows x 3 columns]

```
[ ] 1 normal_1 = result_df_1[result_df_1['검정통계량'].notna]
    2 print(normal_1)
```

	검정통계량	p-value	정규성 만족?
X1	0.384142	0.0	False

② Anderson-Darling Test

	검정통계량	Critical Values	#
X1	10349.044067	[0.576, 0.656, 0.787, 0.918, 1.092]	
X2	NaN	[0.576, 0.656, 0.787, 0.918, 1.092]	
X3	NaN	[0.576, 0.656, 0.787, 0.918, 1.092]	
X4	NaN	[0.576, 0.656, 0.787, 0.918, 1.092]	
X5	NaN	[0.576, 0.656, 0.787, 0.918, 1.092]	
...
X105h	NaN	[0.576, 0.656, 0.787, 0.918, 1.092]	
X105i	NaN	[0.576, 0.656, 0.787, 0.918, 1.092]	
X105j	NaN	[0.576, 0.656, 0.787, 0.918, 1.092]	
X106	NaN	[0.576, 0.656, 0.787, 0.918, 1.092]	
X107	NaN	[0.576, 0.656, 0.787, 0.918, 1.092]	

유의수준

X1	[15.0, 10.0, 5.0, 2.5, 1.0]
X2	[15.0, 10.0, 5.0, 2.5, 1.0]
X3	[15.0, 10.0, 5.0, 2.5, 1.0]
X4	[15.0, 10.0, 5.0, 2.5, 1.0]
X5	[15.0, 10.0, 5.0, 2.5, 1.0]
...	...
X105h	[15.0, 10.0, 5.0, 2.5, 1.0]
X105i	[15.0, 10.0, 5.0, 2.5, 1.0]
X105j	[15.0, 10.0, 5.0, 2.5, 1.0]
X106	[15.0, 10.0, 5.0, 2.5, 1.0]
X107	[15.0, 10.0, 5.0, 2.5, 1.0]

	검정통계량	Critical Values	#
X1	10349.044067	[0.576, 0.656, 0.787, 0.918, 1.092]	

유의수준

X1	[15.0, 10.0, 5.0, 2.5, 1.0]
----	-----------------------------

③ Kolmogorov-Smirnov Test

	검정통계량	p-value	정규성 만족?
X1	0.3396	0.0	False
X2	0.772273	0.0	False
X3	0.456036	0.0	False
X4	0.499873	0.0	False
X5	0.487291	0.0	False
...
X105h	0.422036	0.0	False
X105i	0.376727	0.0	False
X105j	0.429782	0.0	False
X106	0.492855	0.0	False
X107	0.491436	0.0	False

[170 rows x 3 columns]

```
1 normal_3 = result_df_3[result_df_3['정규성 만족?'] == True]
2 print(normal_3)
```

Empty DataFrame
Columns: [검정통계량, p-value, 정규성 만족?]
Index: []

대부분의 변수에 결측치가 포함되어 있는데, 정규성 test들은 결측치를 인식할 수 없음

→ 결측치 보간 기준 논의 후 재시도

EDA - 정규성 검정

① Shapiro-Wilk Test

```
from scipy.stats import shapiro # 정규성 검정 (1) Shapiro-Wilk Test

# 각 변수에 대해 정규성 검정
normality_results1 = {}
for column in test_ks20_knn.columns:
    stat, p = shapiro(test_ks20_knn[column])
    normality_results1[column] = {'검정통계량': stat, 'p-value': p,

# 결과 출력
result_df_1 = pd.DataFrame(normality_results1).T
print(result_df_1)
```

	검정통계량	p-value	정규성 만족?
X7e	0.161455	0.0	False
X7g	0.359593	0.0	False
X7h	0.307781	0.0	False
X7i	0.084012	0.0	False
X24h	0.293102	0.0	False
X24i	0.201225	0.0	False
X25a	0.034966	0.0	False
X26e	0.331324	0.0	False
X64d	0.203911	0.0	False
X64f	0.372863	0.0	False
X64g	0.287012	0.0	False
X64h	0.12374	0.0	False
X69a	0.315889	0.0	False
X69b	0.307721	0.0	False
X69g	0.466878	0.0	False
X69h	0.314635	0.0	False
X96	0.388353	0.0	False
X97	0.273025	0.0	False
X105h	0.194295	0.0	False
X105i	0.230839	0.0	False

② Anderson-Darling Test

```
from scipy.stats import anderson # 정규성 검정 (2) Anderson-Darling Test

# 각 변수에 대해 정규성 검정
normality_results2 = {}
for column in test_ks20_knn.columns:
    result = anderson(test_ks20_knn[column], dist='norm')
    normality_results2[column] = {'검정통계량': result.statistic, 'Critical Values': result.critical_values}

# 결과 출력
result_df_2 = pd.DataFrame(normality_results2).T
print(result_df_2)
```

	검정통계량	Critical Values	W
X7e	6367.974374	[0.576, 0.656, 0.787, 0.918, 1.092]	
X7g	3446.37325	[0.576, 0.656, 0.787, 0.918, 1.092]	
X7h	3766.884309	[0.576, 0.656, 0.787, 0.918, 1.092]	
X7i	6093.656297	[0.576, 0.656, 0.787, 0.918, 1.092]	
X24h	4515.063244	[0.576, 0.656, 0.787, 0.918, 1.092]	
X24i	5419.193437	[0.576, 0.656, 0.787, 0.918, 1.092]	
X25a	6923.285317	[0.576, 0.656, 0.787, 0.918, 1.092]	
X26e	4041.339971	[0.576, 0.656, 0.787, 0.918, 1.092]	
X64d	5590.923606	[0.576, 0.656, 0.787, 0.918, 1.092]	
X64f	3416.505664	[0.576, 0.656, 0.787, 0.918, 1.092]	
X64g	3908.799651	[0.576, 0.656, 0.787, 0.918, 1.092]	
X64h	6195.386563	[0.576, 0.656, 0.787, 0.918, 1.092]	
X69a	3568.348427	[0.576, 0.656, 0.787, 0.918, 1.092]	
X69b	4211.7565	[0.576, 0.656, 0.787, 0.918, 1.092]	
X69g	2614.068076	[0.576, 0.656, 0.787, 0.918, 1.092]	
X69h	3018.488161	[0.576, 0.656, 0.787, 0.918, 1.092]	
X96	3967.232878	[0.576, 0.656, 0.787, 0.918, 1.092]	
X97	4611.734983	[0.576, 0.656, 0.787, 0.918, 1.092]	
X105h	5846.165847	[0.576, 0.656, 0.787, 0.918, 1.092]	
X105i	4459.935757	[0.576, 0.656, 0.787, 0.918, 1.092]	

③ Kolmogorov-Smirnov Test

```
from scipy.stats import ks_2samp, norm # (3) Kolmogorov-Smirnov Test

# 정규분포를 따르는 기준 샘플 생성
mean_value = test_ks20_knn.mean()
std_value = test_ks20_knn.std()
sample_from_normal = norm.rvs(loc=mean_value, scale=std_value, size=len(test_ks20_knn))

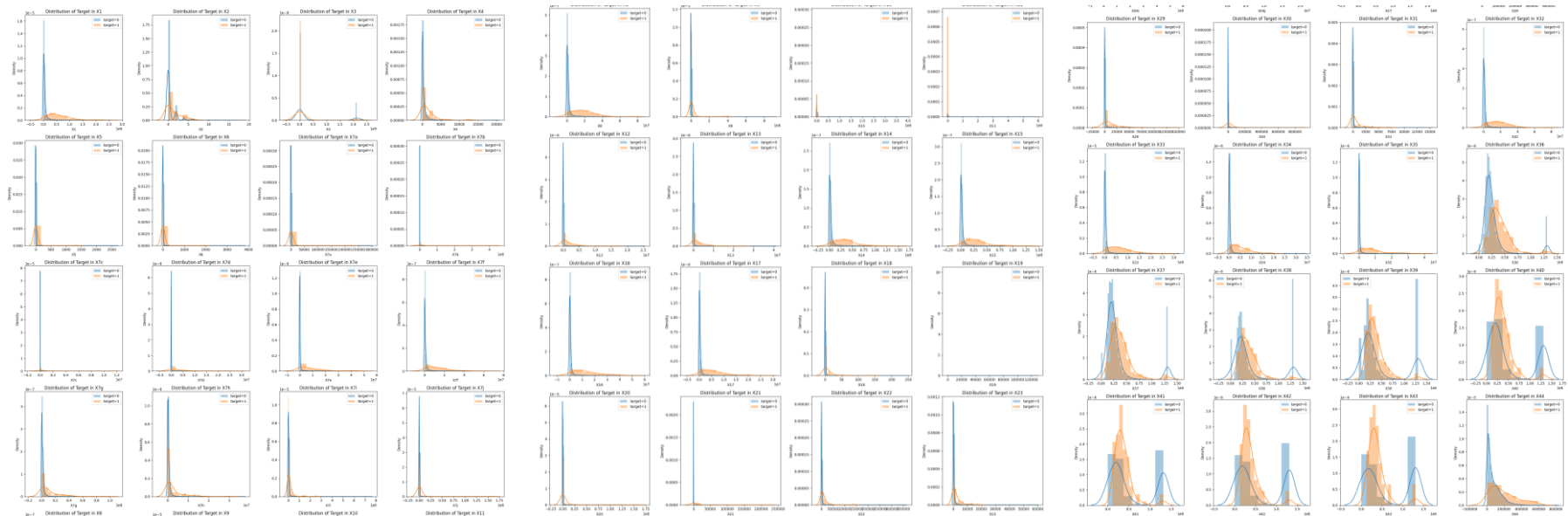
# 각 변수에 대해 정규성 검정
normality_results3 = {}
for idx, column in enumerate(test_ks20_knn.columns):
    stat, p = ks_2samp(test_ks20_knn[column], sample_from_normal)
    normality_results3[column] = {'검정통계량': stat, 'p-value': p,

# 결과 출력
result_df_3 = pd.DataFrame(normality_results3).T
print(result_df_3)
```

	검정통계량	p-value	정규성 만족?
X7e	0.432571	0.0	False
X7g	0.338143	0.0	False
X7h	0.360714	0.0	False
X7i	0.44	0.0	False
X24h	0.376238	0.0	False
X24i	0.408429	0.0	False
X25a	0.470619	0.0	False
X26e	0.355476	0.0	False
X64d	0.410143	0.0	False
X64f	0.333714	0.0	False
X64g	0.366333	0.0	False
X64h	0.435333	0.0	False
X69a	0.323619	0.0	False
X69b	0.361143	0.0	False
X69g	0.302667	0.0	False
X69h	0.333333	0.0	False

결측치 보간 이후 정규성 검정을 시도하여 정규성을 만족하지 않음을 확인하였음!

EDA - 분포 확인

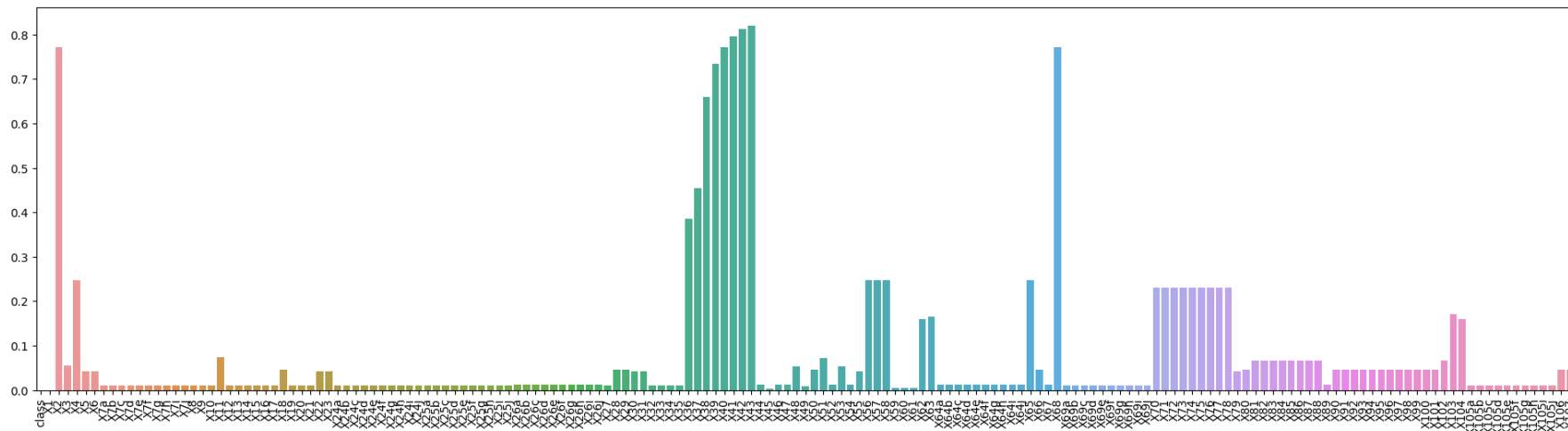


타깃별 분포를 확인한 결과, **대부분 한 쪽으로 치우쳐 있는 분포**를 보임
 특히 대부분의 변수에 **0값이 많음**

2

데이터 전처리

데이터 전처리 - 결측치 처리



...



결측치 비율이 20% 이상인 변수는 모두 제거

2 데이터 전처리

데이터 전처리 - 결측치 처리

MICE

데이터프레임에 있는 다른 모든 변수로 누락값을 예측하여
데이터셋을 여러 개 만들고 하나의 결과로 통합하는 다중 대체법

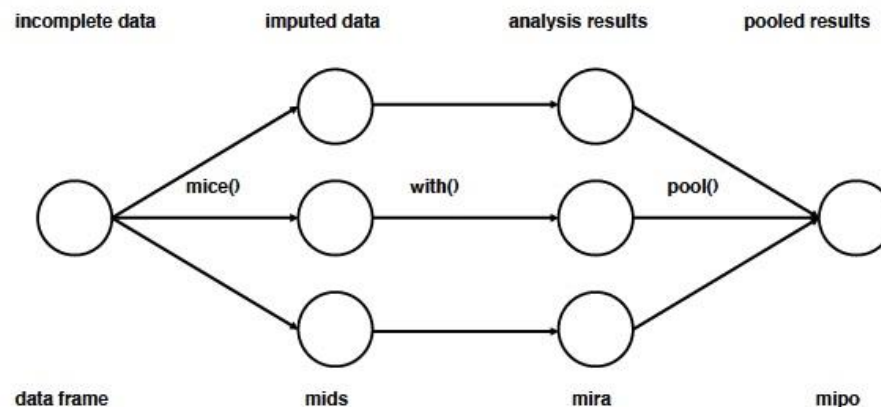


Figure 1: Main steps used in multiple imputation.



MICE로 전체 데이터의 결측치 보간을 시도

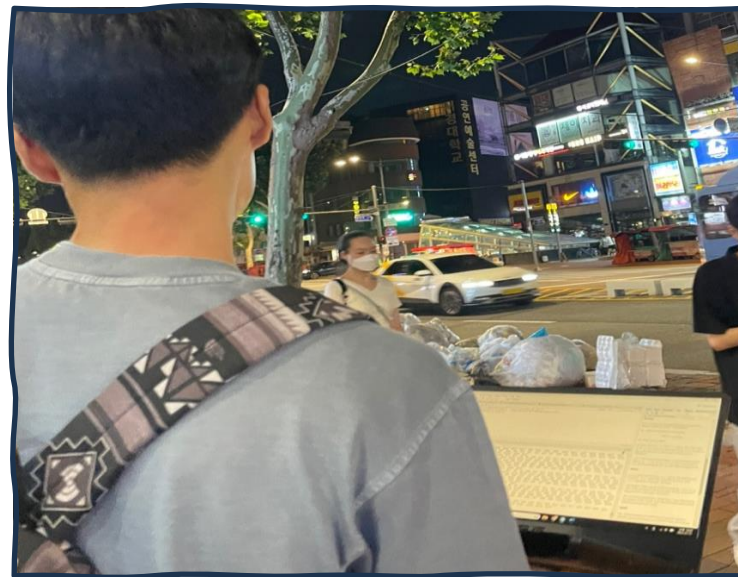
2 데이터 전처리

데이터 전처리 - 결측치 처리

12:00



22:00



시무룩한 뉘모습...



그러나... 수행 시간이 너무 오래 걸림

→ MICE 이외에 간단한 보간법을 (knn, mean, medoid) 병렬로 진행

변수선택을 먼저 해서 데이터를 좀 줄인 뒤에 보간해보자!

데이터 전처리 - 변수 선택

변수 선택

변수가 너무 많으면 계산량이 늘어나 학습 시간이 길어져 비효율적임
→ 변수 선택을 진행하고자 함

- 1) PCA를 통한 차원 축소
- 2) 타깃별 분포가 상이한 변수 선택
- 3) 트리모델의 특성중요도 활용

데이터 전처리 - 변수 선택

변수 선택

변수가 너무 많으면 계산량이 늘어나 학습 시간이 길어져 비효율적임
→ 변수 선택을 진행하고자 함

1) PCA를 통한 차원 축소

2) 타깃별 분포가 상이한 변수 선택

3) 트리모델의 특성중요도 활용

변수 간의 상관관계가 존재하는
다차원의 데이터를 효율적으로
저차원의 데이터로 요약하는 차원축소 기법



Data의 분산을 가장 잘 설명하는
30개의 주성분을 선택함

데이터 전처리 - 변수 선택

변수 선택

변수가 너무 많으면 계산량이 늘어나 학습 시간이 길어져 비효율적임
→ 변수 선택을 진행하고자 함

1) PCA를 통한 차원 축소

2) 타깃별 분포가 상이한 변수 선택

3) 트리모델의 특성중요도 활용

0과 1의 분포 차이가 커야 학습 시 용이할 것



두 집단 간 분포 차이를 분석하는 방법인 K-S 검정,
Mann-Whitney 검정 사용, 각 설명 변수별
검정값의 threshold를 기준으로 변수 선택



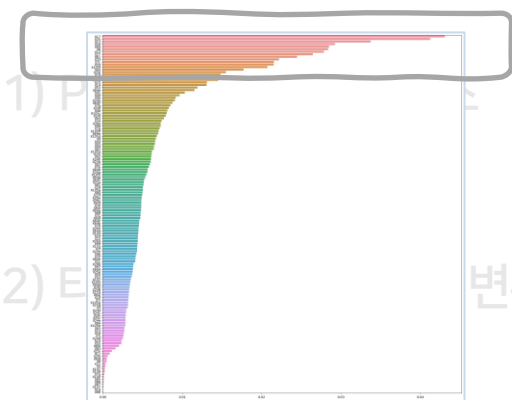
K-S검정: 20개, Mann-Whitney 검정: 71개

2 데이터 전처리

데이터 전처리 - 변수 선택

변수 선택

변수가 너무 많으면 계산량이 늘어나 학습 시간이 길어져 비효율적임
→ 변수 선택을 진행하고자 함



3) 트리모델의 특성중요도 활용

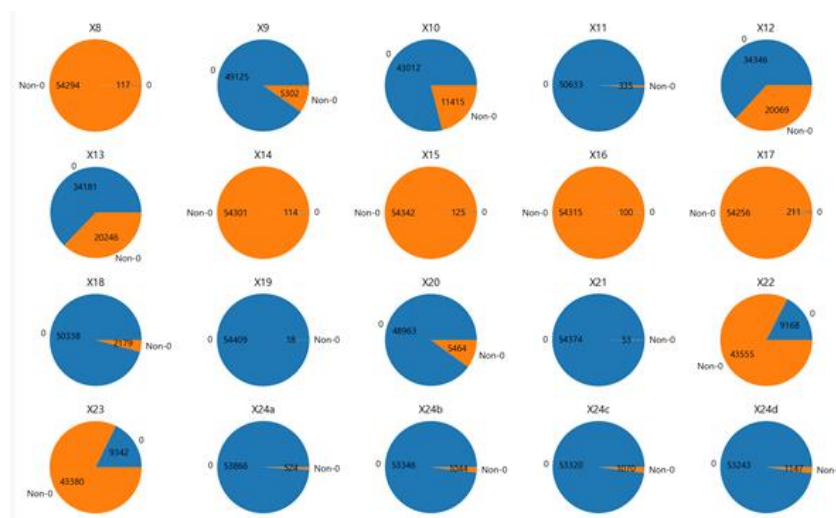
특성 중요도

= 각 노드의 타겟값 불순도 척도를
감소시키는 정도가 큰 특성



특성 중요도가 크게 감소하는 구간까지
선택해 총 13개의 설명 변수 선택

데이터 전처리 - 샘플링



여전히 클래스 불균형이 심해서 **데이터 왜곡** 예상

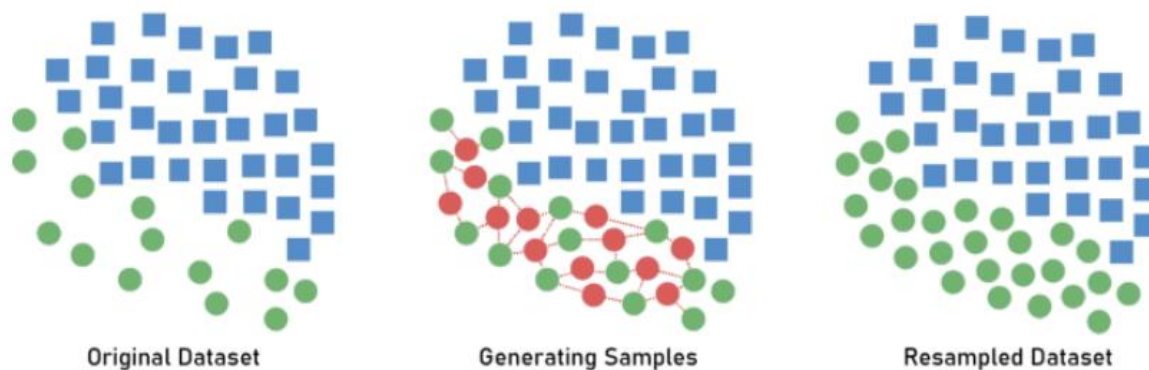


샘플링을 통해 데이터 왜곡을 줄이자!



SMOTE, Random Over Sampling 진행

샘플링 - SMOTE



Over Sampling 기법 중 하나

낮은 비율의 클래스 데이터를 **KNN 알고리즘**을 활용해 새롭게 생성



과적합 발생 가능성이 비교적 적음

2 데이터 전처리

샘플링 - SMOTE



SMOTE는 소수의 클래스의 데이터 간의 거리만을 고려해 데이터를 생성하기 때문에 기존 데이터와 겹치거나 노이즈 발생가능

Original Dataset

Generating Samples

Resampled Dataset

고차원 데이터에서는 효율적이지 않을 가능성이 커,

현재 데이터에서는 성능 개선을 기대할 수 없었음

낮은 비율의 클래스 데이터를 KNN 알고리즘을 활용해 새롭게 생성



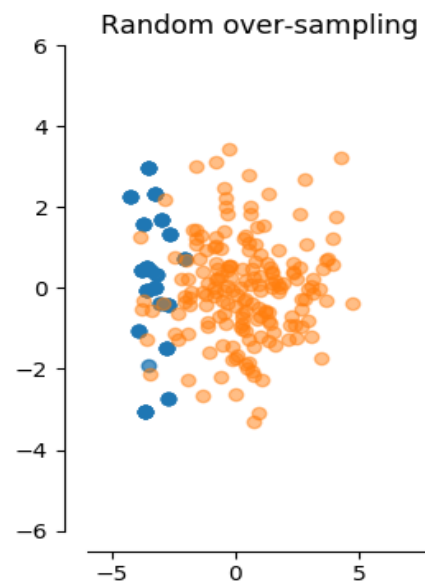
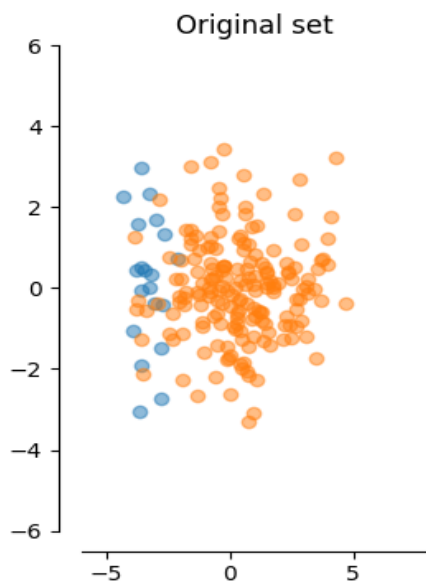
과적합 발생 가능성이 비교적 적음



샘플링 - Random Over Sampling

랜덤 오버 샘플링 (Random Over Sampling)

임의적으로 **소수의 클래스**의 데이터를 **복제**하여 관측치의 수를 늘리는 방법



전처리 이후 Dataset



train_v2: raw_train에서 결측률 > 0.2인 열 제거, 결측행 제거

1

변수 선택법

1. PCA
2. KS & 맨휘트니 검정
3. 특성 중요도

↓
Raw Data

↓
Train_PCA
Train_ks20
Train_manh
Train_feaimp

2

결측치 처리

1. knn
2. median
3. mice

↓
Train_PCA_knn
Train_PCA_median
...
Train_fi_mice

3

데이터 증강

1. SMOTE
2. RandomOver

↓
Train_PCA_knn_aug
Train_PCA_median_up
...
Train_fi_mice_up

전처리 이후 Dataset

이외에도 더 있음..

Ver.	train	test
_v5_knn	v5에서 knn으로 결측치 보간한 데이터	동일작업 진행한 test set
_v5_knn_aug	v5_knn 에서 SMOTE로 업샘플링한 데이터	v5_knn과 동일
_v5_knn_up	v5_knn 에서 Random Over Smampling한 데이터	v5_knn과 동일
_ks20_knn	ks20에서 knn으로 결측치 보간한 데이터	동일작업 진행한 test set
_v6	v4에서 PCA로 변수선택된 데이터	수동으로 생성함 (계수 기반)
_v7_knn	v7에서 knn으로 결측치 보간한 데이터	동일작업 진행한 test set
_v7_knn_aug	v5_knn 에서 SMOTE로 업샘플링한 데이터	v7_knn과 동일
_manh71	맨휘트니 변수선택 & 중앙값으로 결측치 보간 & 상관 0.7 이상 제거	v7_knn과 동일
_ks12_knn1	ks20에서 상관관계 높은 변수 제외한 후, knn으로 결측치 보간	동일작업 진행한 test set

경우에 따라 사용할 데이터셋을 다양하게 완성!



전처리 이후 Dataset



train_v2: raw_train에서 결측치 0.2인 열 제거, 결측행 제거



1

변수 선택법

1. PCA
2. KS 검정 & 맨휘트니
3. 특성 중요도

Raw Data

Train_PCA
Train_ks20
Train_manh
Train_feaimp

2

결측치 처리

1. knn
2. median
3. mice

Train_PCA_knn

Train_PCA_median

...

Train_fo_mice

3

데이터 증강

1. SMOTE
2. RandomOver

Train_PCA_knn

Train_PCA_median

...

Train_fo_mice

각종 데이터셋으로 모델 성능을 비교함!

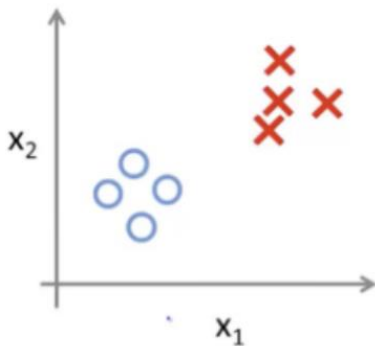
3

모델링

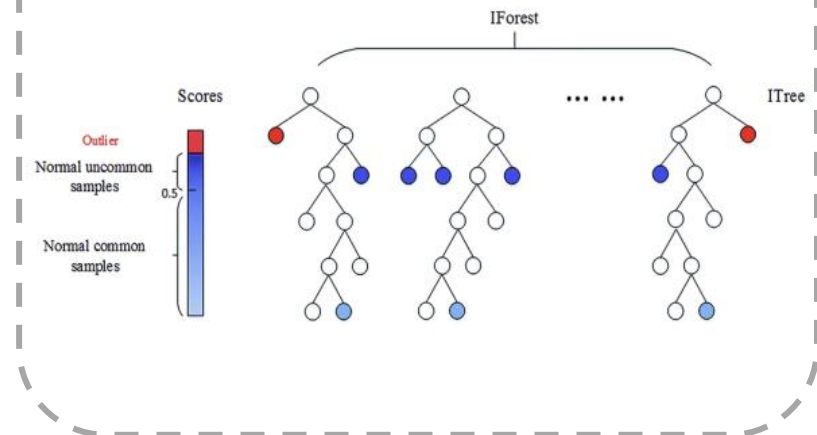
모델링 - outline

이진분류

Binary classification:



이상치 탐지



0은 정상, 1은 이상치인 게 아닐까?

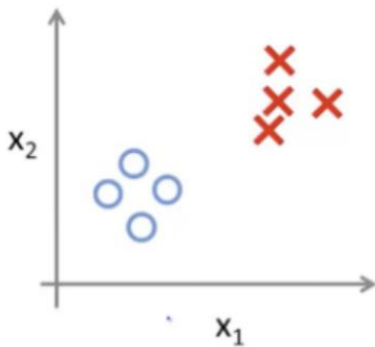
데이터의 특성을 고려해,

이진 분류 task와 이상치 탐지 task를 모두 수행해보기로 결정함

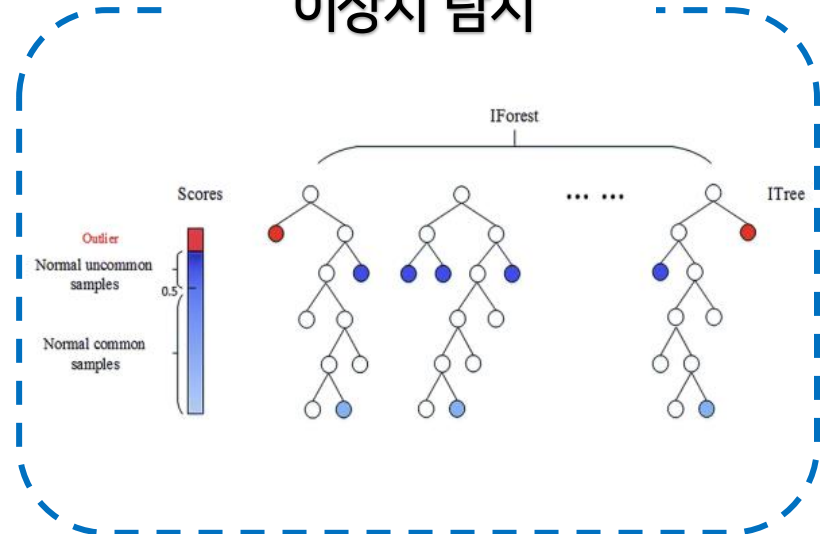
모델링 - outline

이진분류

Binary classification:



이상치 탐지



0은 정상, 1은 이상치인 게 아닐까?



데이터의 특성을 고려해,

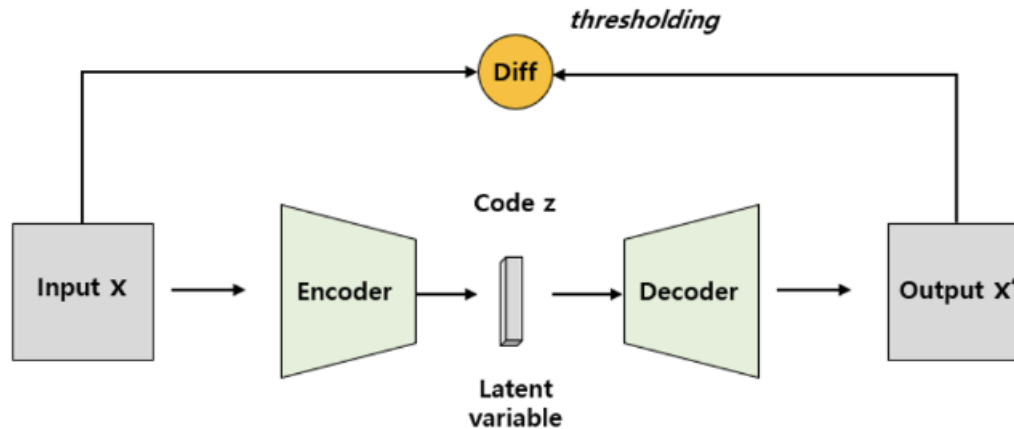
이진 분류 task와 이상치 탐지를 먼저 시도!

이상치 탐지 - Auto Encoder

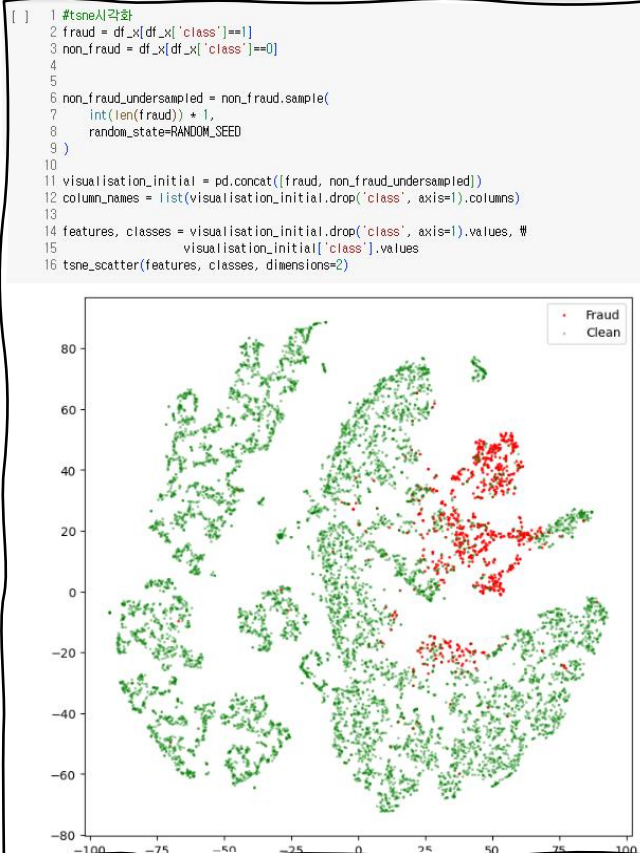
Auto Encoder

정상 데이터에 대해서만 학습을 수행하고, 이후 테스트 데이터에 대해
Input X와 output X'의 차이를 기준으로 이상치 여부 판단함

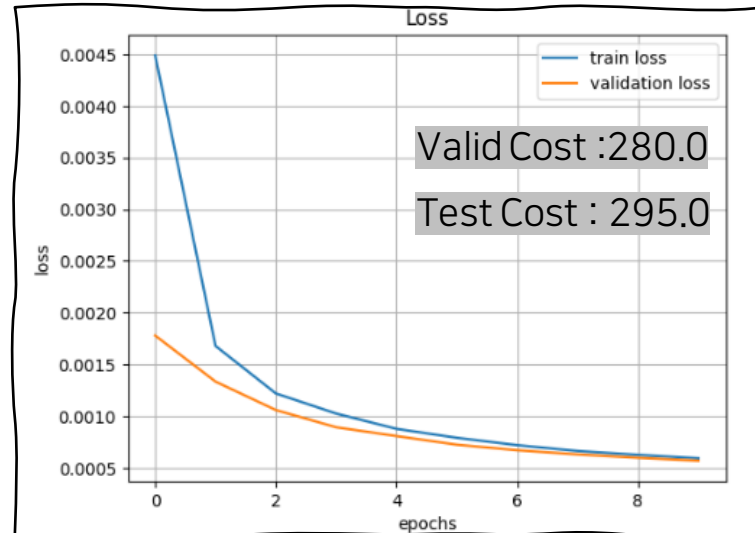
Input과 Output의 차이는 threshold를 기준으로 함



이상치 탐지 – Auto Encoder



이상치 탐지 결과를 tSNE로 시각화



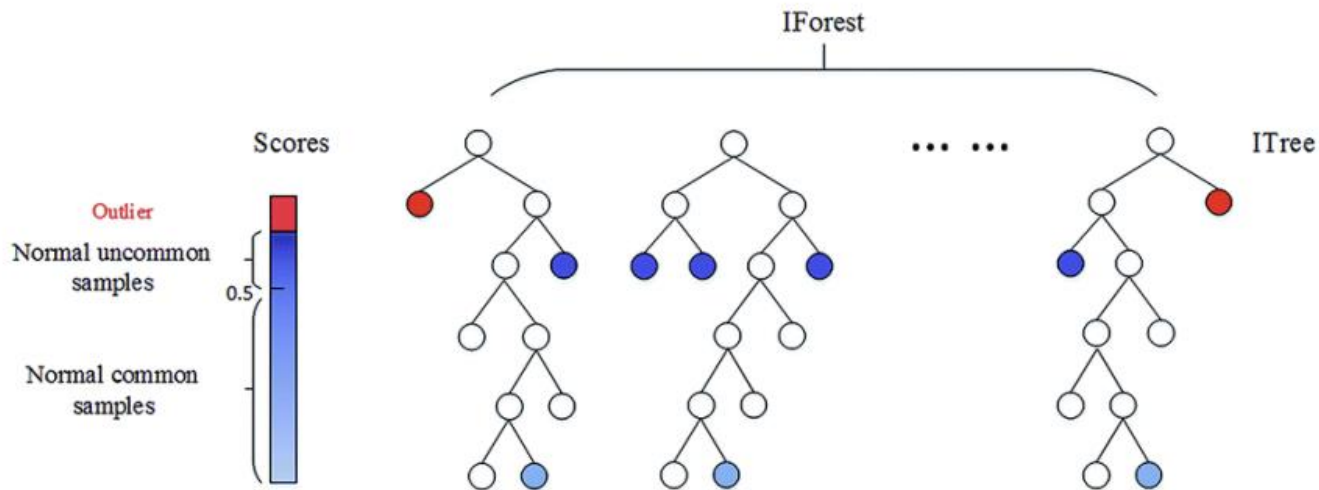
⋮

Valid/Test Cost가 매우 좋았지만
Kaggle에 제출 시 점수가 형편없었음

이상치 탐지 - Isolation Forest

Isolation Forest

Decision Tree에서 파생된 모델로, 얇은 위치에서 고립된 관측값을
비정상 데이터로 판단하는 머신러닝 모델



이상치 탐지 - Isolation Forest



Isolation Forest

Decision Tree에서 파생된 모델로, 비정상 데이터를

데이터의 불균형이 심해서 이상치 탐지를
Tree의 가장 가까운 노드에서 고립시켜 만드는 모델

먼저 해보았지만 성능이 기대만큼 좋지 않았음

(Kaggle score 10000점 ~ 50000점대)

IForest



CATBoost 시도해보자!

이상치 탐지 - Isolation Forest



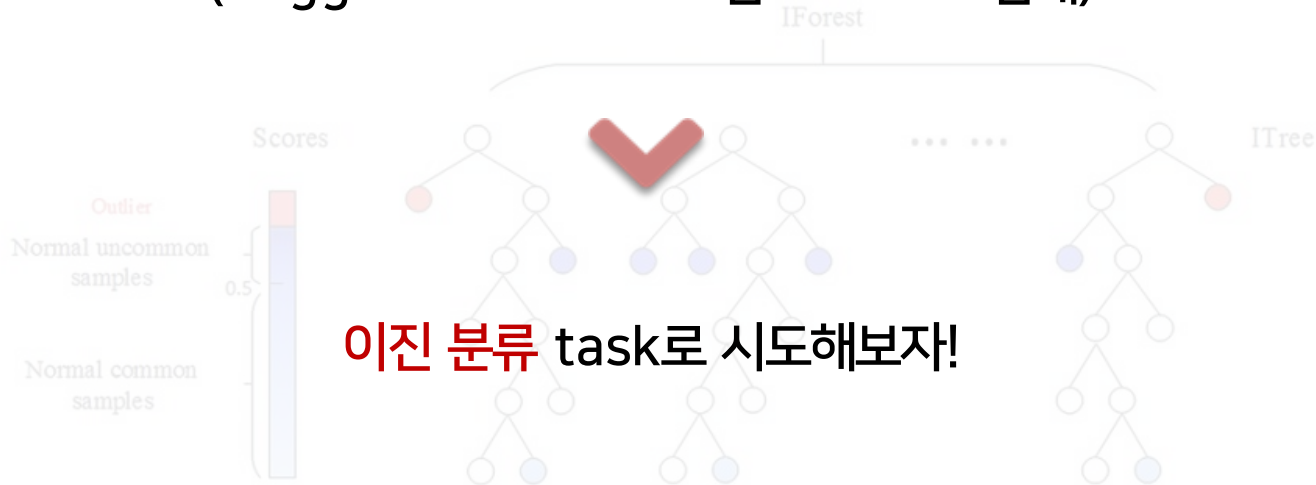
Isolation Forest

Decision Tree에서 파생된 모델로, 비정상 데이터를

데이터의 불균형이 심해서 이상치 탐지를
Tree의 가장 가까운 노드에서 고립시켜 만드는 모델

먼저 해보았지만 성능이 기대만큼 좋지 않았음

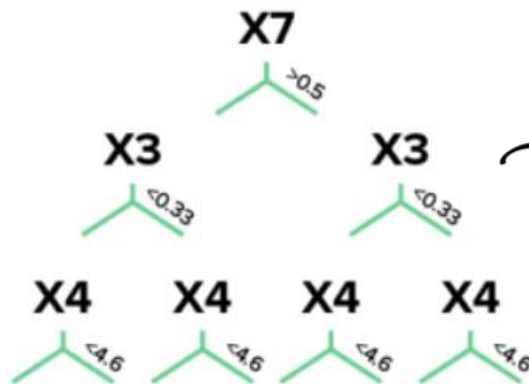
(Kaggle score 10000점 ~ 50000점대)



이진분류 - CATBoost classifier

CATBoost classifier

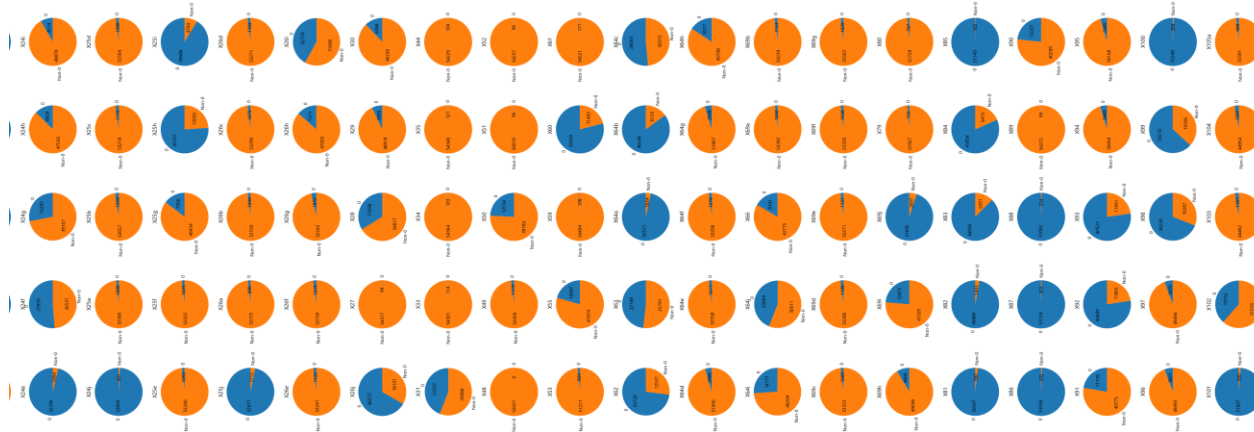
Overfitting을 방지하는데 집중한 트리 기반의 **boosting** 모델로,
일반적으로 범주형 변수가 많을 경우 학습성능이 좋다고 알려져 있음



대칭적으로 트리를 형성함 (level-wise)

CatBoost

이진분류 - CATBoost

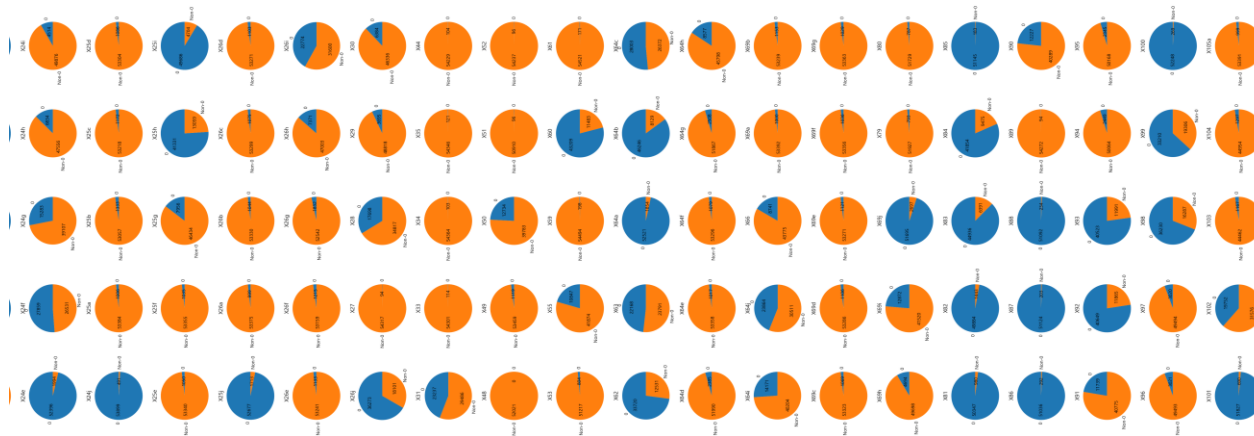


데이터셋의 변수를 범주형 변수로 바꾸어 시도해보았지만 성능이 좋지 않았고,

CATBoost에서도 직접 범주변수를 나누는 것보다

모델 내에서 자체적으로 학습했을 때의 성능이 더 잘 나옴

이진분류 - CATBoost



대체적으로 트리를 형성함 (level-wise)
데이터셋의 변수를 범주형 변수로 바꾸어 시도해보았지만 성능이 좋지 않았고,
그러나... 더 잘 나온 결과도

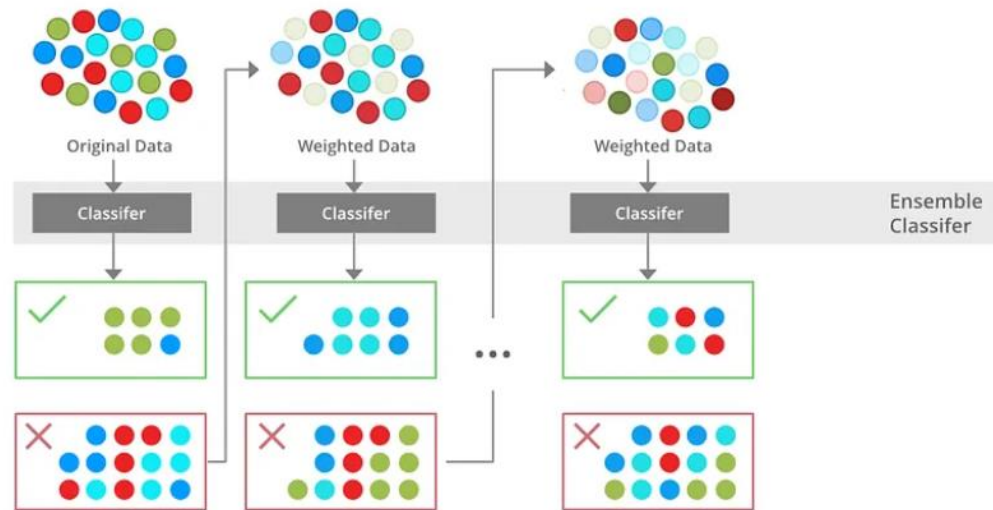
CATBoost에서도 직접 범주변수를 나누는 것보다
Kaggle Score는 10000점 대로, 성능이 크게 개선되지 않았음



이진 분류 - XGBoost

XGBoost

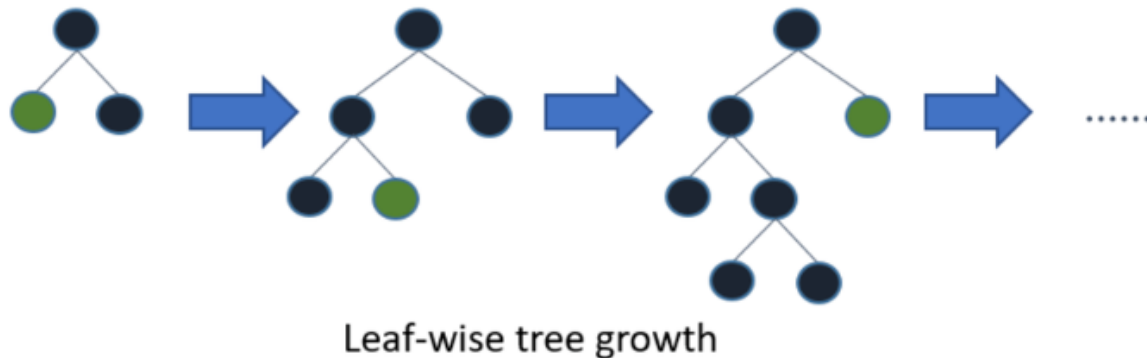
Gradient boosting을 병렬학습이 지원되도록 구현한 라이브러리
Regression, Classification 모두 지원함



이진 분류 - LGBM

LGBM

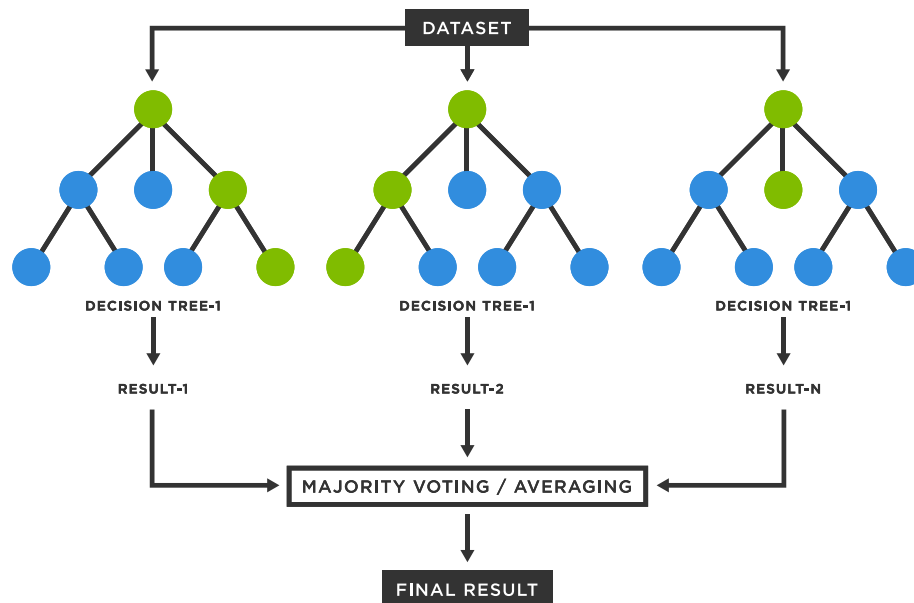
Gradient Boosting 알고리즘을 기반으로 하며, 효율적인 분할 알고리즘으로 결과의 정확도에 초점을 맞추기 때문에 모델의 성능이 좋음



이진 분류 - Random Forest

Random Forest

분류, 회귀 분석 등에 사용되는 **앙상블 학습 방법**의 일종으로,
다수의 결정 트리의 분류를 집계해서 최종적으로 분류함

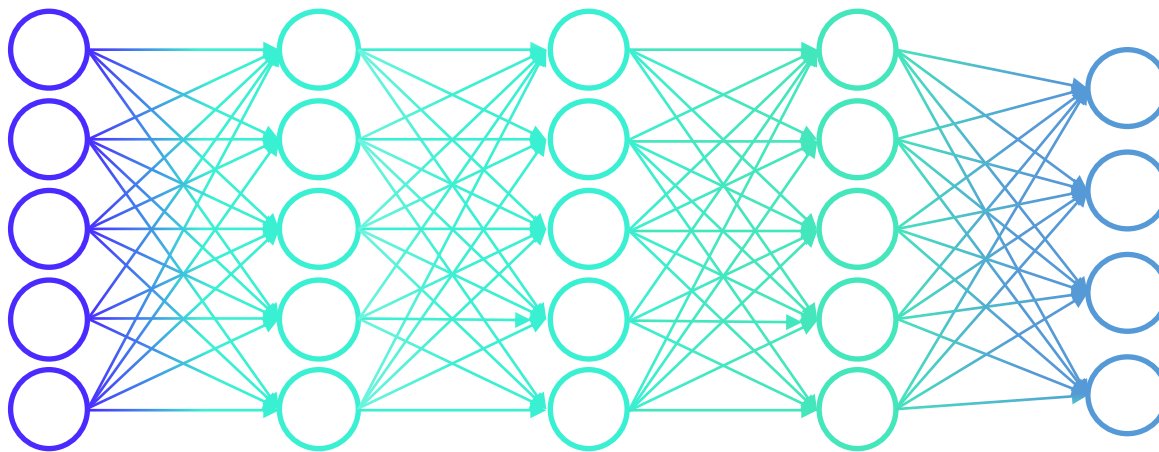


이진 분류 - MLP(Multi Layer Perceptron)

MLP(Multi Layer Perceptron)

가장 기본적인 딥러닝 모델

생물의 뉴런 구조를 모방하여 비선형성 부여

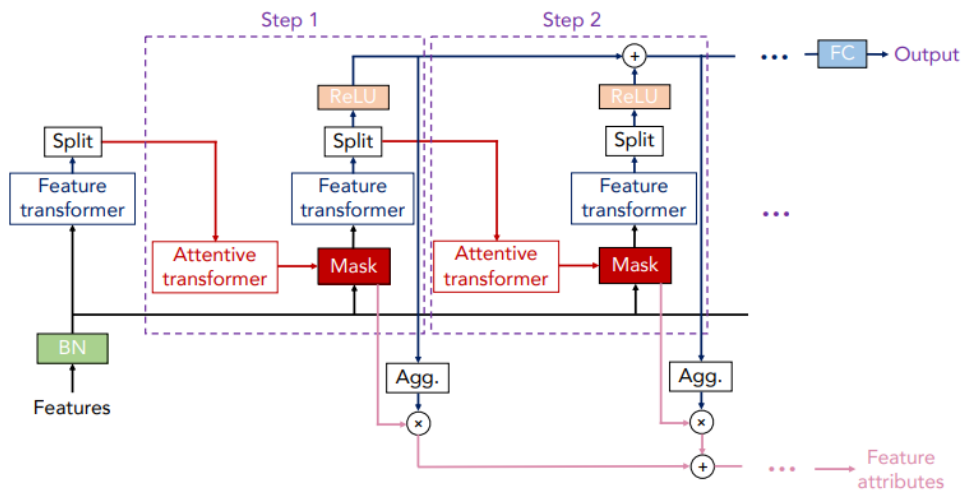


이진 분류 - TabNet

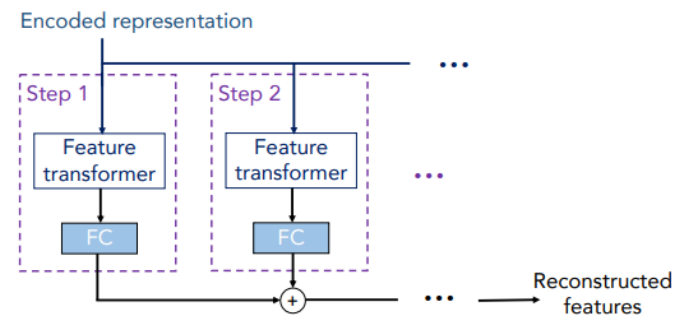
TabNet

정형데이터 학습에 적합한 딥러닝 모델

입력된 데이터에서 Feature를 masking하며 여러 step을 걸쳐 학습



(a) TabNet encoder architecture



(b) TabNet decoder architecture

이진 분류 - TabNet

TabNet

정형데이터 학습에 적합한 딥러닝 모델

입력된 데이터에서 Feature를 masking하며 여러 step을 걸쳐 학습



성능 좋았던 단일 모델 TOP3

**LGBM
Regressor**

데이터셋

v5_mice153
MICE 최대 iteration
적용 후 보간

하이퍼파라미터

num_leaves : 357,
Learning_rate : 0.0058,
n_estimators : 1792,
min_child_samples : 45

.....
Kaggle score : 7540

TabNet

데이터셋

v2 랜덤포레스트 feature
importance 출력 후
상위 9개 변수 선택

하이퍼파라미터

n_a : 8, n_d : 8,
n_steps : 3,
Gamma : 1.3,
n_independent : 2,
n_shared : 2

.....
Kaggle score : 5005

**LGBM
Classifier**

데이터셋

v5_knn_up
KNN 결측치 보간 후
샘플링 진행

하이퍼파라미터

max_depth : 7,
n_estimators : 100,
나머지는 기본값
그대로 사용


.....
Kaggle score : 6565

4

최종 모델

모델 최적화 전략

$$\text{Cost Function} = 250 \times \text{FN} + 5 \times \text{FP}$$

		관측값(Y)	
		Y = 1	Y = 0
예측값(\hat{Y})	$\hat{Y} = 1$	TP	FP
	$\hat{Y} = 0$	FN 	TN

⋮

cost function에 대한 이해 : **FN을 줄이는 것이 핵심!**

4

최종 모델

모델 최적화 전략

$$\text{Cost Function} = 250 \times \text{FN} + 5 \times \text{FP}$$

		관측값(Y)	
		Y = 1	Y = 0
예측값(\hat{Y})	$\hat{Y} = 1$	TP	FP
	$\hat{Y} = 0$	FN	TN

⋮

실제 값은 1인데, 0으로 잘못 예측한 것(FN)을 줄이는 데 집중!

4

최종 모델

모델 최적화 전략

best1
0
0
0 1
...
1
0
0



best2의 정보를 참고해
target을 갱신

best2
0
0
1
...
1
0
0



test 데이터에 대한 **예측값끼리 Voting**을 시도

= 여러 모델의 예측을 결합하여 오차를 줄임

4 최종 모델



모델 최적화 전략

1로 바꿨는데 오히려 틀릴 수도 있는 것 아닌가?

		관측값(Y)	
		$Y = 1$	$Y = 0$
예측값(\hat{Y})	$\hat{Y} = 1$	① TP	② FP
	$\hat{Y} = 0$	FN	TN

1개만 맞춰도 250 줄일 수 있음!

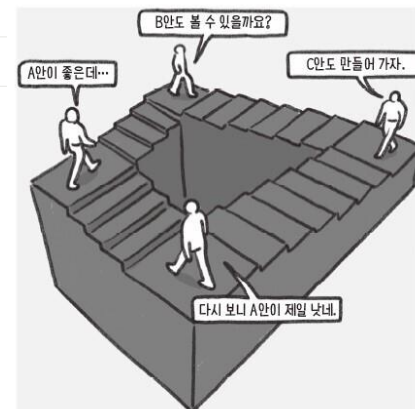
FN에 과도한 가중치가 부여된 confusion matrix 특성상
test 데이터에 대한 예측값끼리 Voting을 시도
cost 값이 높아질 위험이 적음!
= overfitting을 줄 수 있는 단일 모델의 단점을 완화!

최종 모델

변수 명	모델	보유한 1 개수	Kaggle Score 🏆
best1	TABNet : v2	1514	5005
best2	TABNet : v7_knn_2	1356	5610
best3	LGBMClassifier	714	6565
best23	2←3	1388	5175
best13	1←23 = 1←3	1530	4545 🏆
best13_2+3	13←2+3 = best1_2+3	1596	4700
best12	1←2	1582	5165
best4	RandomForest지원	765	8230
best14	1←4	1543	4565
best13_4	13←4	1552	4595
best5	LGBMClassifier	541	9445
best15	1←5	1535	
best13_5	13←5	1548	4605
best6	TABNet : v5_mice153	1516	5200

best6_3	6←3	1527	
best6_4	6←4	1542	5080
best6_5	6←5		
best_iso	IsolationForest	508	17000
best_reg	lgbmregressor	812	7585
best_reg_mice	LGBMRegressor	796	?
reg_mice←xgb		868	
best1_reg	1←reg	1593	
best13_reg_mice	13←reg_mice	1598	4450 🏆
beset13_reg	13←reg	1604	4460 🏆
13_6	13←6	1727	5035
xgb	xgbregressor	614	8320
1_xgb	1←xgb	1523	
13_xgb	13←xgb	1537	

무수한 시도 ... (1) 모델 간 voting



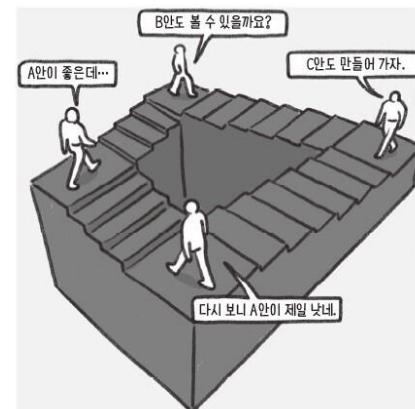
최종 모델

n_a	8	8	8	8	16	16	16	32	32	32
n_d	8	16	32	64	16	32	64	16	32	64
n_steps	3	-	-	-	-	-	-	-	-	-
gamma	1.3	-	-	-	-	-	-	-	-	-
n_independent	2	-	-	-	-	-	-	-	-	-
n_shared	2	-	-	-	-	-	-	-	-	-
train cost	13875	18900	19000	19230	19180	-	-	-	-	-
valid cost	4125	4095	4185	4310	4075	-	-	-	-	-
등수		2	4	폐기	1	-	-	-	-	-

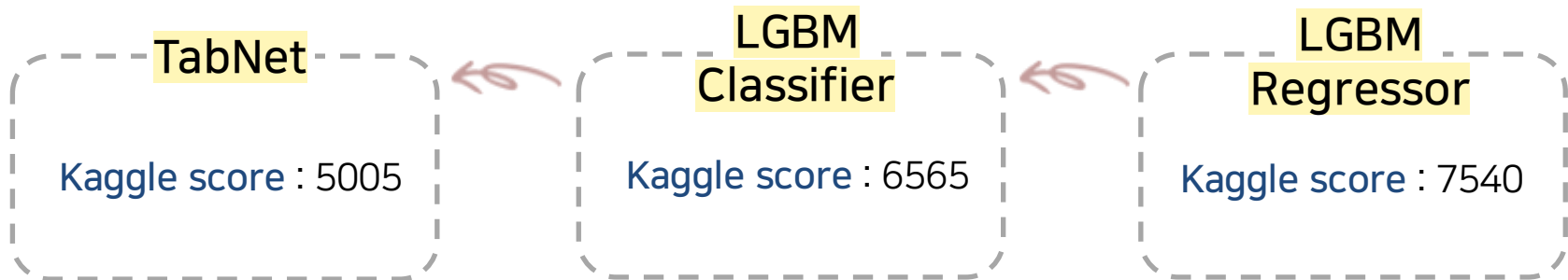
n_a	8	-	-	-	-	-	-	-	-	-
n_d	8	64	64	64	64	64	64	64	64	64
n_steps	3	2	2	2	5	5	5	7	7	7
gamma	1.3	1.0	1.5	2.0	1.0	1.5	2.0	1.0	1.5	2.0
n_independent	2	-	-	-	-	-	-	-	-	-
n_shared	2	-	-	-	-	-	-	-	-	-
train cost	13875	16150	17595	17930	18755	21335	22770	21120	20735	22060
valid cost	4125	4425	3910	4445	3720	4845	5205	4670	4915	5450
등수		폐기	2	폐기	1	폐기	폐기	폐기	폐기	폐기

n_steps	3	-	-	-	-	-	-	-	-	-
gamma	1.3	-	-	-	-	-	-	-	-	-
n_independent	2	1	1	1	3	-	-	-	-	-
n_shared	2	1	3	5	1	3	5	1	3	5
train cost	13875	19555	18965	20005	20155	19890	20755	26065	22165	20510
valid cost	4125	4080	4140	4325	4460	4300	5000	6145	5310	4845
등수		1	2	폐기	폐기	폐기	폐기	폐기	폐기	폐기

무수한 시도 ... (2) 단일 모델 성능 최적화



최종 모델



12

그냥 서유신



4450.00000

7

20h



Your Best Entry!

Your most recent submission scored 4450.00000, which is an improvement of your previous score of 4460.00000. Great job!

Tweet this

⋮

TabNet + LGBMClassifier + LGBMRegressor 로
voting한 경우가 가장 좋은 점수를 획득함!

5

한계 및 의의

한계

- ✓ 시간상의 문제로 변수 선택과 결측치 보간을 정교하게 하지 못함
- ✓ 단일 모델의 성능이 임계치 이상으로 올라가지 못한 아쉬움이 있음
- ✓ OPTUNA 파라미터 튜닝을 했으나, 과적합이 의심되어 성능이 좋지 않았음

의의

- ✓ cost function의 특성을 파악하고, 그에 맞게 좋은 score를 획득함
- ✓ 모델 결합으로 오차를 줄여나가서, 단일 모델의 한계를 극복함
- ✓ 딥러닝, 머신러닝 등 다양한 모델을 활용하고자 노력함

일주일이 정말 삭제가 됐습니다. 날씨도 엄청 덥고 하루종일 데이터만 보느라 짜증날만했는데, 다들 즐겁고 재밌게 열심히 해주셔서 고마운 마음입니다.

응 애!



김동환

일주일이 이렇게 빨리 지날 줄은 몰랐어요... 우리 친절한 똑똑이 팀원들 덕분에 정말 많이 배우며 즐겁게 끝냈습니다! 1팀 짱 너무 수고했오요<33 뒤풀이 때 보아요

재밌었다



노정아

아는 게 많이 없어서 걱정됐는데 훌륭한 팀원분들 덕에 또 많이 배울 수 있었습니다! 친해지고 싶었던 다른 팀원분들과 함께 할 수 있어서 영광이었고, 2학기도 같이 재밌게 했으면 좋겠습니다 ㅎㅎ 다들 4일 내내 너무 고생 많으셨습니다!!!

내가해냄



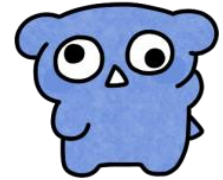
성준혁

오티에서 팀이 발표난 순간부터 달려온 피셋 방세! 다들 바쁜 일정 속에서 일요일부터 회의하고 매일 학관에 모여 데이터 분석 진행하느라 정말 수고하셨습니다. 다같이 이야기하며 분석하는 과정에서 새로운 것을 알 수 있었고, 타팀 간 정보를 공유하는 동시에 경쟁하는 과정도 재밌었습니다. 캐글 점수를 높이기 위해 접하지 못한 모델을 구현하고 팀원들의 코드와 저의 코드를 공유하는 과정에서 정말 많이 성장하는 한 주가 되었던 것 같습니다. 모두 수고하셨습니다.

멋있고 따뜻한 사람들과 함께 방학 세미나를 할 기회가 생겨서 의미 있는 일주일인 것 같습니다 !! 짧은 시간이었지만 함께 배우고 웃으면서 보내는 시간이 즐거웠고, 고생 정말 많았습니다아 뒷풀이 때 즐겁시다 우리 ^~^

방학세미나를 통해 1주일동안 무언가에 집중하면서 뜻깊은 경험을 한 것 같습니다! 웃수저 방세1팀과 함께한 시간들 너무 즐거웠어요~ 2학기 때도 모두 파이팅입니당 누가 뭐래도 1팀이 최고..~~

(진짜모름)



이정환



이지원



하희나

감사합니다