

LG 자율주행 안테나 경진대회

Team 해커톤

정재윤, Kade Na, 강보근

목차

1. 해커톤 팀 역할분담
2. 데이터 분석 & 파생변수 생성
3. 데이터 구축 & 모델 선정

해커톤 팀 역할분담

정재윤

파생변수 생성, 데이터 구축 및 모델 서치

Kade Na

모델 서치 및 하이퍼파라미터 튜닝

강보근

하이퍼파라미터 튜닝

데이터 분석 & 파생변수 생성

데이터 분석

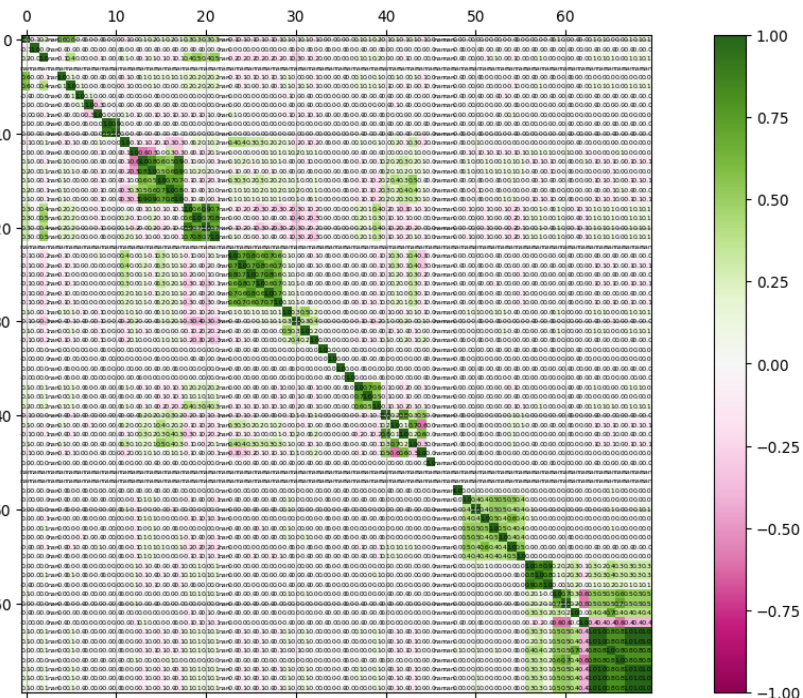
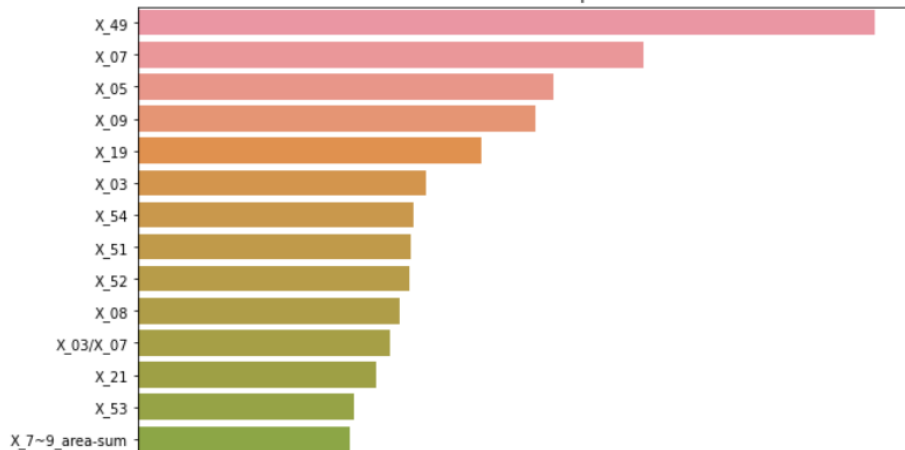
도메인 지식 및 아이디어 수집

파생변수 생성

데이터 Drop

데이터 분석

CATBOOST Feature Importance



Catboost의 Feature Importance, X_value Heatmap, Shapley value 등의 분석을 사용해 전반적인 데이터를 분석함.

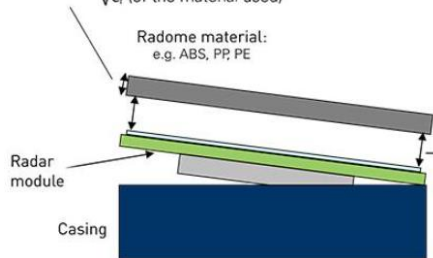
도메인 지식 및 아이디어 수집



그림 2: 왼쪽 그림은 “잘못된 배치”로 레이돔이 표면이 고르지 않고 안테나에 평행하게 배치되지 않았습니다. 오른쪽 그림은 “올바른 배치”로 레이돔의 거리가 균일할 뿐 아니라 배치와 크기가 올바릅니다. (이미지 출처: InnoSenT)

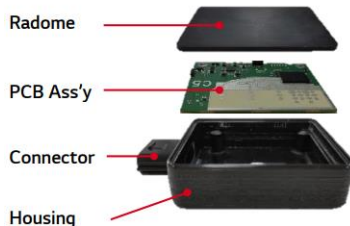
Material thickness of radome:

$$\text{Thickness} = \frac{6.2 \text{ mm (approx. value for 24.125 GHz)}}{\sqrt{\epsilon_r \text{ (of the material used)}}}$$



Distance between antenna surface and radome shall be constant.

As approximate value for 24.125 GHz you can use 6.2 mm.



도메인 지식을 통한 데이터 이해를 위해 Radar 센서 제작 과정 및 주의사항에 대한 조사를 진행함.

레이돔의 거리 균일도에 따라서 전자기파의 분산과 결과물의 결과가 달라질 수 있다는 도메인 지식을 습득함.

=> X 41~X 44 diff 파생변수 생성

이 외에도 습득한 도메인 지식을 기반으로 파생변수를 생성해보거나, feature drop을 시도해봄.

파생변수 생성

1. X_03/X_07 (무게 / 면적)

무게 / 면적을 통해 방열 재료 1의 **면적대비 무게**를 구해 새로운 변수를 생성함. (2와 3은 무게의 데이터 부족)

2. X_41~X_44 diff

X_41~X_44중 max값과 min값을 찾아 max - min을 통해 **레이돔 치수의 최대 차이**에 대한 변수를 생성함.

3. X_01, 02, 05, 06 sum

단계별 누름량을 모두 더해 **총 가해진 누름량**에 대한 변수를 생성함.

4. X_07~09 sum

방열 재료의 면적을 모두 더해 **총 방열재료의 면적**에 대한 변수를 생성함

파생변수 생성

5. X_03/X_19~22

방열재료 무게 / 스크류 삽입 깊이의 합을 통해 무게와 관련된 스크류 삽입 깊이의 관계에 대한 변수를 생성함.

6. X_12 / X_24, X_25

커넥터 위치 좌표와 커넥터핀 치수의 관계를 알기 위해서 변수를 생성함.

7. X_49_7_19_3_8

Heatmap에서 importance가 높은 데이터들을 train 데이터의 평균값으로 나눈 합에 대한 변수를 생성함.

데이터 Drop

```
drop_list = ["X_04", "X_23", "X_47", "X_48"]
```

- X_04, X_23, X_47, X_48 변수는 검사 통과 여부에 대한 변수이며, 학습에 필요가 없기 때문에 제외함.

```
drop_x = ["X_02", "X_10", "X_11", "X_34", "X_35", "X_36", "X_37", "X_45"]
```

Catboost의 select_features 함수를 사용했으며, RecursiveByShapValues 알고리즘 방법으로 Shap value 기반 가장 중요도가 낮은 Feature들을 골라내서 한번 더 분석하고, Drop하였습니다.

- X_02, X_45 : select feature 함수에서 제거되었으며, Feature importance가 낮아 Drop함
- X_10, X_11 : **NAN value**가 너무 많아 복구하더라도 좋은 성능을 보이지 못해서 Drop함
- X_34~X_37 : **스크류 체결 시 분당 회전수**는 select feature 함수에서 제거되었으며, Heatmap에서도 y_value와 유의미한 상관관계를 보이지 못해 Drop함

데이터 구축 & 모델 선정

Validation Set 구축 전략

- 6 Stratified Fold based on label

모델 선정

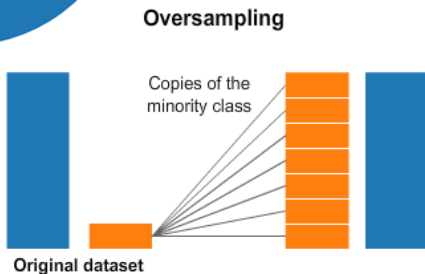
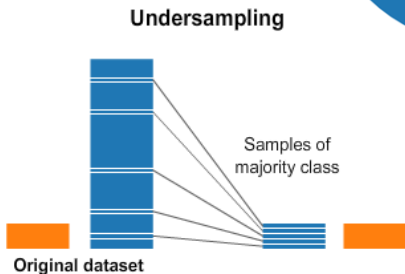
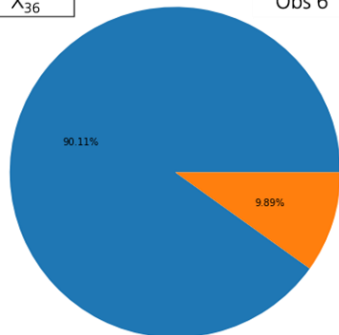
- Gradient Boosting-based ML Algorithms
 - CatBoost
 - LGBM
 - XGBoost
- Hyperparameter Tuning (optuna)
- 3 model ensemble

Validation Set 구축 전략

	X_1	X_2	X_3
Obs 1	X_{11}	X_{21}	X_{31}
Obs 2	X_{12}	X_{22}	X_{32}
Obs 3	X_{13}	X_{23}	X_{33}
Obs 4	X_{14}	X_{24}	X_{34}
Obs 5	X_{15}	X_{25}	X_{35}
Obs 6	X_{16}	X_{26}	X_{36}

→ Swap Noise

	X_1	X_2	X_3
Obs 1	X_{11}	X_{26}	X_{31}
Obs 2	X_{12}	X_{22}	X_{33}
Obs 3	X_{15}	X_{22}	X_{33}
Obs 4	X_{11}	X_{24}	X_{31}
Obs 5	X_{15}	X_{25}	X_{35}
Obs 6	X_{12}	X_{26}	X_{36}

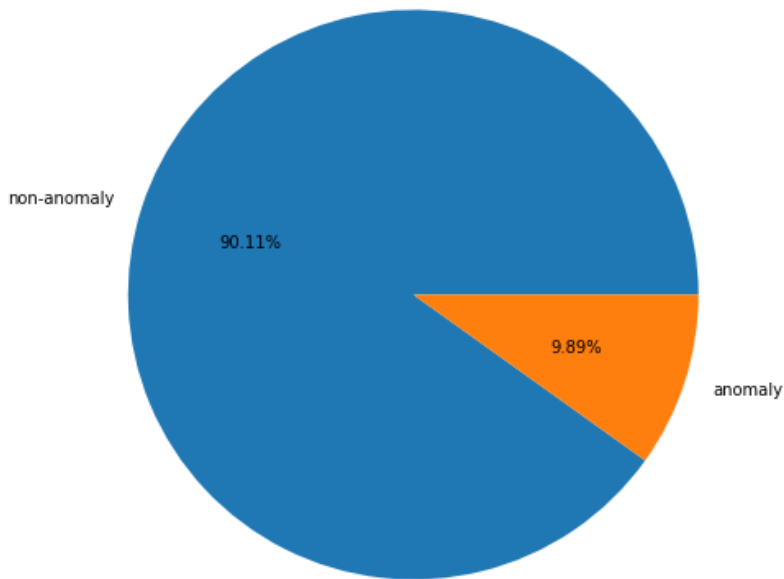


우선 Train 데이터의 수가 39607개로 적은 편에 속했으며, 정상/불량에 대한 label값을 부여하고 나면 불량은 전체 데이터의 약 10%에 해당합니다.

따라서 **class imbalance** 문제를 해결하기 위해서 **swap noise**, **undersampling**, **oversampling** 등의 다양한 기법을 사용해서 정상/불량의 비율을 맞춰주려고 시도해 보았습니다.

하지만 데이터의 개수가 적고, 특히 불량 데이터가 정상 데이터와의 분포가 거의 비슷해서 분류가 어려웠으며 특성 파악이 어려워 이러한 기법을 시도했을때 성능이 항상 떨어지는 결과가 나오게 되었습니다.

Validation Set 구축 전략



다시 한번 Train Dataset의 정상/불량 데이터의 분포를 보게 되면 불량에 해당하는 데이터는 전체 10%에 달합니다.

또한 Train Dataset의 개수가 39607개밖에 없으므로 전체 데이터를 사용해 모델이 적은 수의 **biased한 데이터**로도 잘 학습할 수 있도록 계획을 세워야 했습니다.

따라서 불량에 해당하는 데이터도 모델이 잘 학습할 수 있고, 데이터를 사용해 여러번 학습할 수 있도록 label 비율 **Stratified KFold**를 사용해 데이터셋 구축시 불량 label에 대한 비율을 강제하고, 여러번 학습할 수 있도록 했습니다.

CatBoost(Categorical Boosting)

- An algorithm for gradient boosting on decision trees
- Ordered boosting
- Uses a breadth-first search algorithm to create trees(**level-wise**)

Advantages

1. Great quality without parameter tuning
2. Improved accuracy

LGBM(Light GBM)

- A gradient boosting framework that uses tree based learning algorithms
- Utilizes **leaf-wise** algorithm on the tree (Memory and time efficiency with a deeper tree)

Advantages

1. Faster training speed and higher efficiency.
2. Lower memory usage.
3. Better accuracy.
4. Support of parallel, distributed, and GPU learning.
5. Capable of handling large-scale data.

XGBoost

- An optimized distributed gradient boosting library designed to be highly efficient, flexible and portable
- Gradient Boosting Machine + Optimization(Tuned Hyperparameters) + Algorithm
- Regularization to avoid overfitting
- Built-in Cross Validation
- Parallelisation
- Cache Optimization

Advantages

1. Better accuracy.
2. Support of parallel, distributed, and GPU learning.
3. It Works well in small to medium dataset

HyperParameter Tuning (optuna)

- For hyperparameter tuning, we used **Optuna**, an automatic hyperparameter optimization software framework
- By setting: (for xgb)
 - random_state
 - a range of learning_rate
 - n_estimators
 - a range of max_depth
 - a range of subsample
 - a range of colsample_bytree
- Multiple trials were ran to find the best/minimum value with the hyperparameters
- For algorithm, we use **TPESampler** (Sampler using Tree-structured Parzen Estimator)

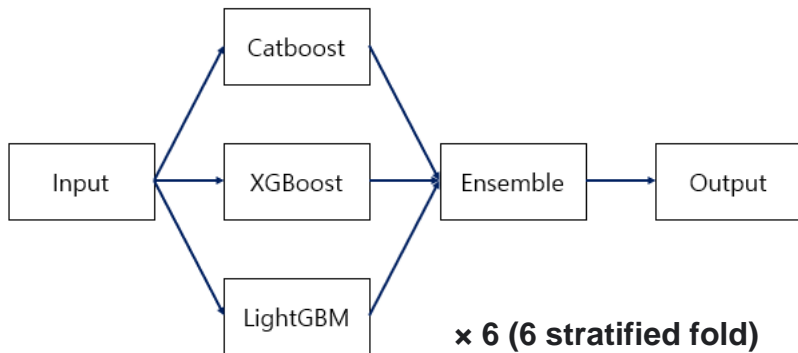
Output Example

- [I 2022-08-10 14:07:34,663] Trial 23 finished with value: 1.941010155721771 and parameters: {'learning_rate': 0.006010848429163341, 'max_depth': 8, 'subsample': 0.7921365050833529, 'colsample_bytree': 0.8569759398897375}. Best is trial 23 with value: 1.941010155721771.
- [I 2022-08-10 00:53:46,864] Trial 31 finished with value: 1.9409859593627379 and parameters: {'learning_rate': 0.006139295456352811, 'max_depth': 8, 'subsample': 0.7555884047040881, 'colsample_bytree': 0.8638977924043896}. Best is trial 31 with value: 1.9409859593627379.

3 model ensemble

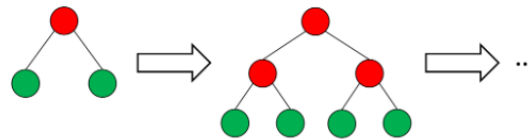
Use Ensemble for

- Higher accuracy
- Reduce the single-model error
- Maintaining the model's generalization

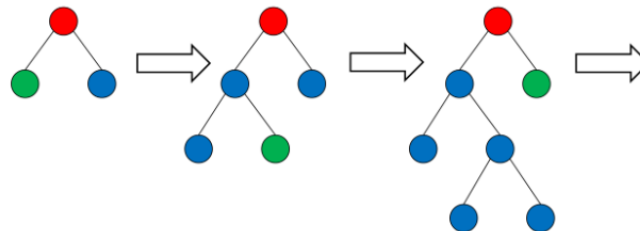


- Use two different tree growing policy
 - Levelwise - Catboost, Xgboost
 - Leafwise - LightGBM

Level-wise:



Leaf-wise:



THANK YOU TO ALL!

Any Questions?