# LG Aimers
# -제품 이상 여부 판별 프로젝트-

불량췤!:
김혜윤, 노연수, 박종혁, 박민정, 양병욱

# 목 차

# 1. 데이터 전처리

# 1. 데이터전처리

## a) Null값 처리

train

40506 rows × 469 columns

test

17361 rows × 468 columns

row의 절반보다 결측치의 개수가 많으면 변수 drop

train

40506 rows × 185 columns

test

17361 rows × 184 columns

# 1. 데이터전처리

## a) Null값 처리

train

40506 rows × 185 columns

test

17361 rows × 184 columns

NULL 값 확인

```
HEAD NORMAL COORDINATE X AXIS(Stage1) Collect Result_Dam     12766
HEAD NORMAL COORDINATE X AXIS(Stage1) Collect Result_Fill1   12766
HEAD NORMAL COORDINATE X AXIS(Stage1) Collect Result_Fill2   12766
dtype: int64
```

3개의 column이 12766개의
Nulll 값을 가짐.

# 1. 데이터전처리

## a) Null값 처리

```
HEAD NORMAL COORDINATE X AXIS(Stage1) Collect Result_Dam      12766
HEAD NORMAL COORDINATE X AXIS(Stage1) Collect Result_Fill1    12766
HEAD NORMAL COORDINATE X AXIS(Stage1) Collect Result_Fill2    12766
dtype: int64
```

```
Unique values in column 'HEAD NORMAL COORDINATE X AXIS(Stage1) Collect Result_Dam':
[nan '550.3' 'OK' '162.4' '549' '549.5' '550' '548.5']

Unique values in column 'HEAD NORMAL COORDINATE X AXIS(Stage1) Collect Result_Fill1':
[nan '838.4' 'OK' '837.7' '837.9' '838.2' '837.5']

Unique values in column 'HEAD NORMAL COORDINATE X AXIS(Stage1) Collect Result_Fill2':
[nan '835.5' 'OK' '305']
```

numeric값과 character값 동시에 존재

# 1. 데이터전처리

## a) Null값 처리

| | | | | |
|---|---|---|---|---|
| OK | 162.7 | 465.7 | 552 | 1271.8 |
| OK | 161.7 | 464.7 | 551 | 1271.8 |
| OK | 162.7 | 465.7 | 552 | 1271.8 |
| OK | 550.6 | 463.9 | 161.5 | 377.6 |
| OK | 549.4 | 462.7 | 160.3 | 377.6 |
| OK | 550.6 | 463.9 | 161.5 | 377.6 |
| OK | 550.3 | 463.6 | 161.2 | 377.6 |
| OK | 550.6 | 463.9 | 161.5 | 377.6 |
| 550.3 | 463.8 | 160.8 | 377.3 | 377.3 |
| 549 | 462.5 | 159.5 | 377.5 | 377 |
| 549 | 462.5 | 159.5 | 377.5 | 377 |
| 549 | 462.5 | 159.5 | 377.5 | 377 |
| 549.5 | 463 | 160 | 377.5 | 377 |
| 550 | 463.5 | 160.5 | 377.3 | 377.3 |
| 550 | 463.5 | 160.5 | 377.3 | 377.3 |
| 550 | 463.5 | 160.5 | 377.3 | 377.3 |
| 550.3 | 463.8 | 160.8 | 377.3 | 377.3 |
| 550 | 463.5 | 160.5 | 377.3 | 377.3 |
| 549.5 | 463 | 160 | 377.5 | 377 |

| Stage3 Line4 Distance Speed | THICKNESS 1 | THICKNESS 2 | THICKNESS 3 | WorkMode | |
|---|---|---|---|---|---|
| 4000 | 0 | 0 | 0 | 7 | |
| 4000 | 0 | 0 | 0 | 7 | |
| 4000 | 0 | 0 | 0 | 7 | |
| 4000 | 0 | 0 | 0 | 7 | |
| 4000 | 4000 | 0 | 0 | 0 | 7 |
| 4000 | 4000 | 0 | 0 | 0 | 7 |
| 4000 | 4000 | 0 | 0 | 0 | 7 |
| 4000 | 4000 | 0 | 0 | 0 | 7 |
| 4000 | 4000 | 0 | 0 | 0 | 7 |
| 4000 | 4000 | 0 | 0 | 0 | 7 |
| 4000 | 4000 | 0 | 0 | 0 | 7 |
| 4000 | 4000 | 0 | 0 | 0 | 7 |

데이터 확인결과 일부 값들이 밀려있는 것을 확인함

# 1. 데이터전처리

## a) Null값 처리

밀려있는 부분 처리 후 Null값 확인

```
# 결측치가 있는 열만 추출
missing_values = train_data.isna().sum()
columns_with_missing_values = missing_values[missing_values != 0].index

# 결측치가 있는 열만 추출하여 새로운 데이터프레임 생성
train_data_missing = train_data[columns_with_missing_values]

# 결과 출력 (선택 사항)
print("Columns with missing values:")
print(columns_with_missing_values)
```

```
Columns with missing values:
Index([], dtype='object')
```

Null값 없음

# 1. 데이터전처리

## b) EDA

column 중 같은 값임에도 불구하고 각각 인식되는 값들이 존재

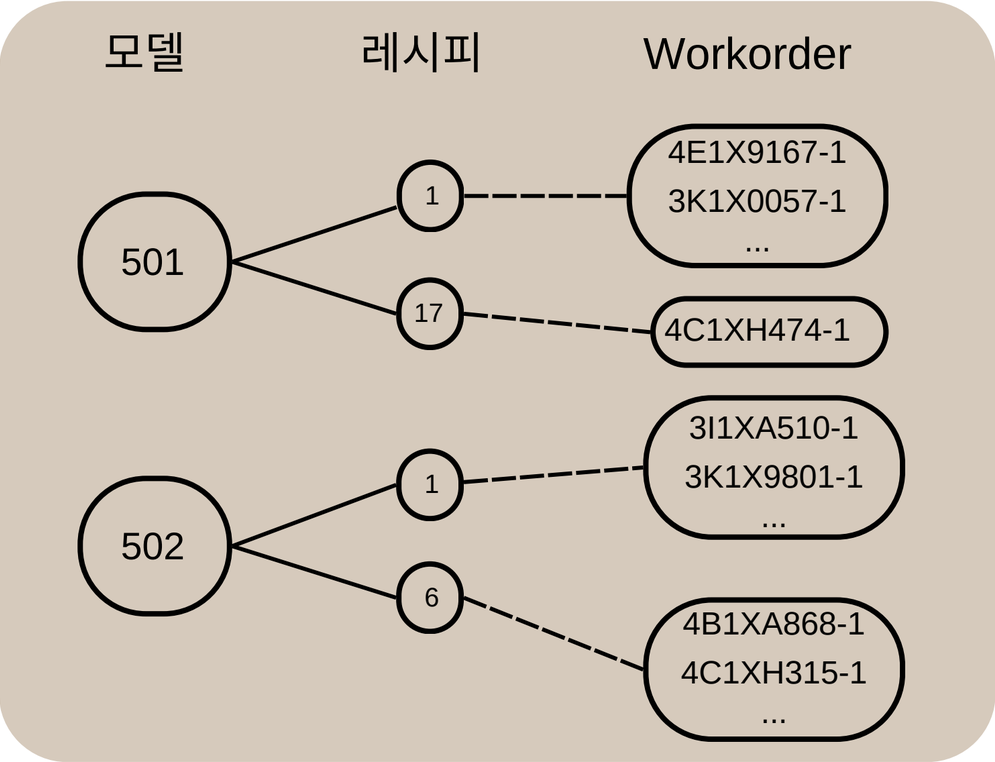
| HEAD NORMAL COORDINATE X AXIS(Stage1) Collect Result_Fill2 | count |
|---|---|
| **835.5** | 12868 |
| **835.5** | 12158 |
| **305.0** | 11005 |
| **305** | 3579 |
| **304.8** | 896 |

숫자형과 문자열이 섞여있는 열을 찾고 숫자형으로 변환

# 1. 데이터전처리

## b) EDA

범주형 처리 : 겉보기에는 수치형 데이터로 보이지만, Model.Suffix 및 Workorder 컬럼의 고유값에 따라 대부분의 **측정값이 패턴을 형성**함




단순히 수치형 데이터로 분석하는 대신, 컬럼 간 패턴을 고려하여 모든 값을 **범주형으로 변환**

데이터를 범주형으로 처리함으로써 **숨겨진 패턴**과 **규칙성**을 더 명확하게 식별

# 1. 데이터전처리

b) EDA

1 ) 중복값을 갖는 컬럼

한 개의 컬럼만 남기고 제거

2 ) 단일값을 갖는 컬럼

무의미하다고 판단하여 제거

3 ) 각 columns들이 숫자형 값을 가지지만  범주형태를 띄는 변수가 다수 존재

machine tact time 제외하고 모두 범주형변수 처리

# 1. 데이터전처리

## b) EDA - 시간 관련파생변수 생성

### 1) durations

→ Dam 공정부터 AutoClave 공정까지의 총 소요 시간 변수
Collect Date_Dam - Collect Date_AutoClave

### 2 ) 각 공정에서의 Collect date 변수

→ Dam 공정 과정에서의 데이터 수집 시간 중 년, 월, 일만 추출

### 3 ) 각 공정 간의 duration

→ 공정 과정 각각의 소요 시간 변수
Dam에서 Fill1, Fill1에서 Fill2, Fill2에서 AutoClave의 Collect time 빼기

# 1. 데이터전처리

## b) EDA

최종 모델링 데이터셋 구축

train

40506 rows × 121 columns

test

17361 rows × 120 columns

# 2. 모델링

# 2. 모델링

1.Model : CatBoostClassifier

대부분의 변수가 범주형이기 때문에 범주형 데이터 처리에 뛰어난

CatboostClassifier를 최종 모델로 선택

2.Tuning parameter

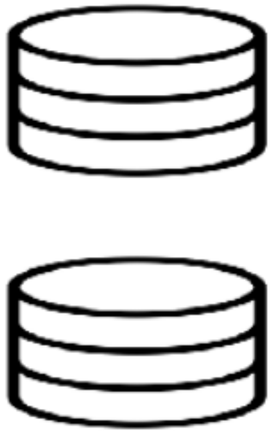
: iterations, depth, learning_rate, class_weights , l2_leaf_reg

클래스 불균형이 매우 심한 데이터이기 때문에 class_weights도 함께 튜닝

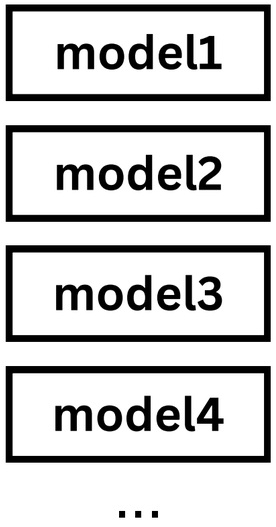
추후 test data에서의 과적합을 방지하고자 l2_leaf_reg 파라미터도 사용

# 2. 모델링

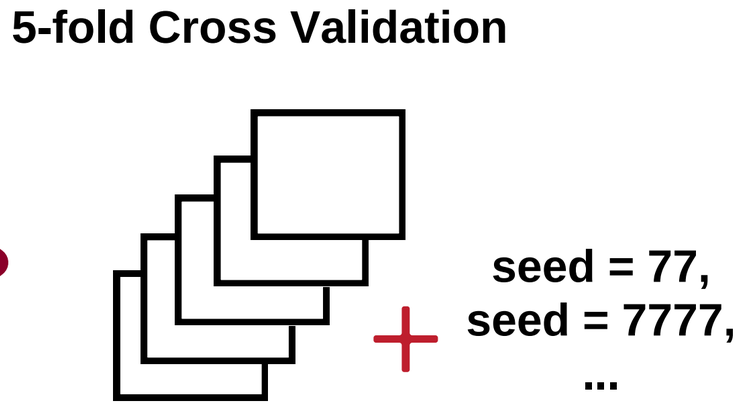
## 3. Modeling



Train dataset을
train-validation
7:3으로 split




**optuna**를 사용해
validation set에서의
F1 Score, Recall, Precision이
0.27 이상인 파라미터 조합 저장




StratifiedCV 및
data seed 변경을 통해
가장 안정적인 성능을 보이는
**4개의 모델 선정**

# 2. 모델링

```
params1 = {'iterations': 561,
'depth': 7,
'learning_rate': 0.05857910198594344,
'class_weights': {0:1, 1:7.1},
'l2_leaf_reg': 9}

params2 = {'iterations': 827,
'depth': 6,
'learning_rate': 0.048934679784453365,
'class_weights': {0:1, 1:8.3},
'l2_leaf_reg': 12}

params3 = {'iterations': 950,
        'depth': 9,
        'learning_rate': 0.05184429466563718,
        'class_weights': {0:1, 1:7.4},
        'l2_leaf_reg': 3}

params4 = {'iterations': 869,
        'depth': 5,
        'learning_rate': 0.07797672575412745,
        'class_weights': {0:1, 1:7.4},
        'l2_leaf_reg': 6}
```

4개의 학습된 모델로

Soft Voting을 이용한 앙상블 모델 생성

precision과 recall의 값이 balaced된

하이퍼 파라미터 선정

최종예측

| target | |
|---|---|
| Normal | 16630 |
| AbNormal | 731 |

# 감사합니다