스마트팩토리
반도체 공정 최적화

조 이름 : 다함께힘내보 조
조장 : 이영철
조원 : 박장훈, 정의석, 이석재

Index

01 프로젝트 배경

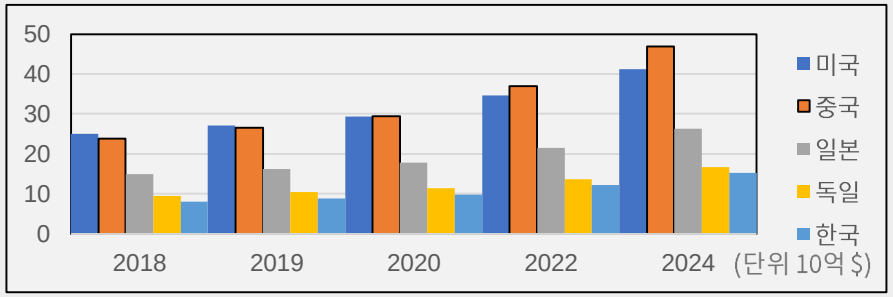
01 프로젝트 배경

스마트 팩토리란?

제품을 조립 및 포장하고 기계를 점검하는 모든 과정이 자동으로 이루어지는 공장 정보통신기술(ICT)의 융합으로 이뤄지는 4차 산업혁명의 핵심.

상호 연결된 기계 네트워크, 통신 메커니즘, 컴퓨터의 계산 능력인 스마트 팩토리는 AI, 머신러닝 같은 고급 기술을 적용해 데이터를 분석하고, 프로세스 자동화를 가속화하며, 시간 경과에 따라 학습하는 사이버 물리 시스템(CPS)이다.

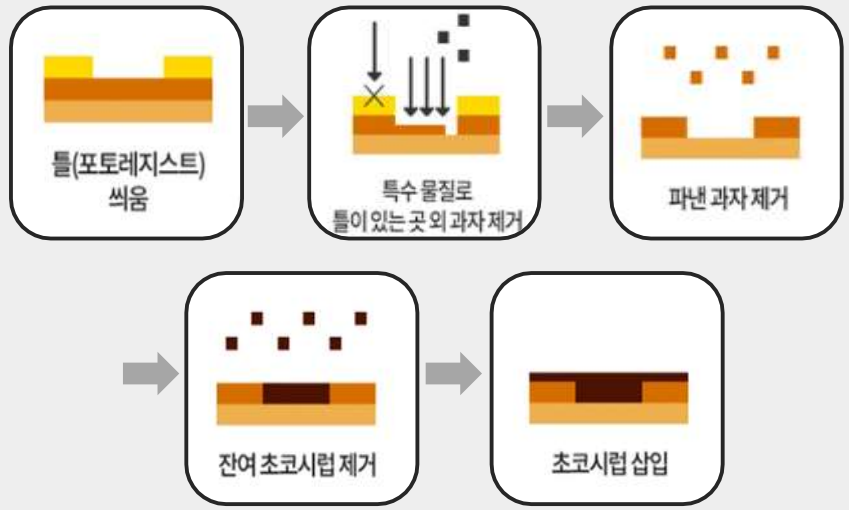
현 5대 제조 강국에서의 스마트 제조 시장 규모가 연평균 약 10% 성장 중이며, 보다 높은 수율/품질을 갖추기 위해 필수적인 요소로 자리잡고 있다.



01 프로젝트 배경

건식 플라즈마 식각(Etching)

*식각 공정 모식도



"완벽한 스마트 공장은 언제?"... 반도체 핵심 공정에 AI도입 늦는 이유

…(중략) 건식 플라즈마 식각 공정은 습식 식각 등 타 공정 보다 훨씬 정교하게 만들 수 있어 반도체/디스플레이 제조 공정의 60 ~ 70%에서 건식 플라즈마 식각 공정이 사용된다.

하지만, 매우 작은 원자 단위의 식각 가스 입자들로 인해 공정 파라미터 변화에 취약하다. 쉽게 예측할 수 있는 범위를 넘어서는 공정인자와 결과 간의 "비선형성 " 을 보여주기 때문.

*2021.08.27기사 발췌

따라서, 이를 ML을 통해 식각 공정 산출물인 인자 간 관계를 찾고 불량을 예측 할 수 있는 시스템 구축을 목표로 주제 선정

02 프로젝트 목표

02 프로젝트 목표 및 과정

목표 1.

40개의 특성과 결과의 상관관계를 파악하고
공정에 필요한 최적의 특성을 선별

목표 2.

약 68만개의 전체 데이터에서 모델링을 통해
불량 예측률 95% 이상 달성

1. EDA 전처리

시각화 분석으로 어떤 피쳐가 불량에 영향을 끼치는지 직관적으로 파악

1. 상관관계도 : 불량에 영향이 큰 피쳐 파악
2. 피쳐 중요도 : 피쳐 중요도 순으로 비교
3. 히스토그램 : 정상과 불량의 히스토그램 양상 비교

2. 모델 선정

분류 모델의 성능 비교

1. Logistc Regression
2. Random Forest
3. DecesonTree
4. SVC
5. XGBoost
6. LightGBM

3. 모델 최적화 및 성능 향상

모델의 최고 성능을 위한 최적화 과정

1. 결측치 대치법
2. 핵심 피쳐만 예측에 적용
3. 하이퍼 파라미터 최적화

4. 모델 평가 및 결론

모델의 성능과 유효성을 평가

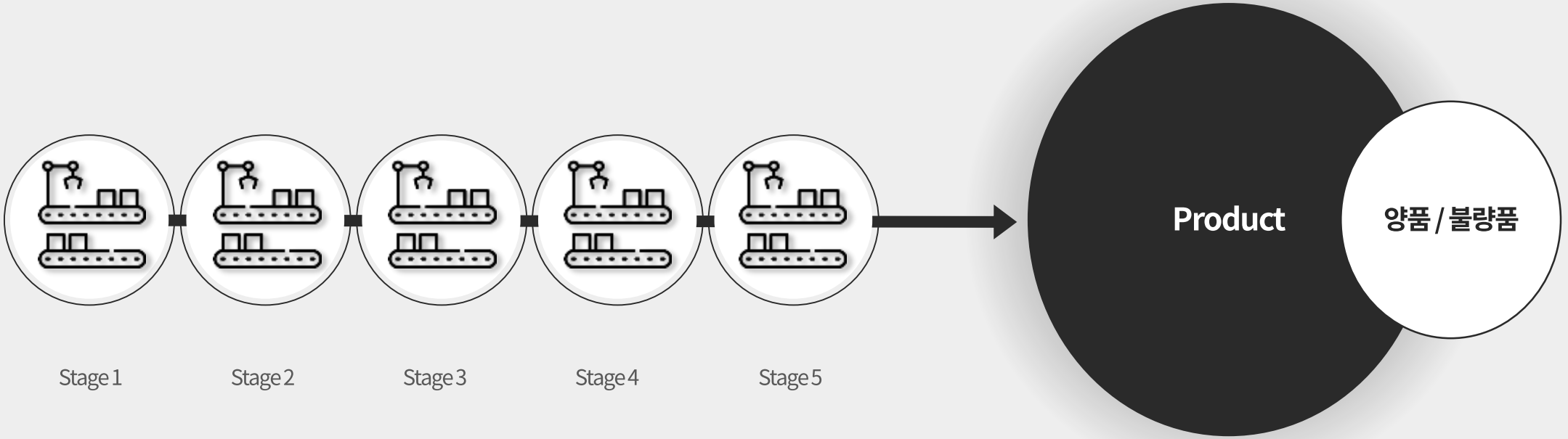
1. 모델 성능 평가 : 최적의 성능에서 불량 예측률 97% 달성 확인
2. 유효성 검증 : 과대적합에 대한 유효성 검증

03 데이터 셋

03 데이터 셋

총 5 개의 stage와 stage 별 8 개의 Feature로 구성.

* 8개 Feature 종류 - 온도, 습도, 유량, 점도, 밀도, 산소 농도, 질소 농도, 이산화탄소 농도



03 데이터 셋

Feature	Feature Definition	Feature type
Stage #_temp	공정 온도	모두 **연속형 데이터** **Deviation Feature**: 표준값 대비 차이
Stage #_humidity	공정 습도	
Stage #_flow_deviation	공정 input의 유량 차이	
Stage #_viscosity_deviation	공정 input의 점도 차이	
Stage #_density_deviation	공정 input의 밀도 차이	
Stage #_o2_deviation	공정 반응에서 생성된 산소 농도 차이	
Stage #_n_deviation	공정 반응에서 생성된 질소 농도 차이	
Stage #_co2_deviation	공정 반응에서 생성된 이산화탄소 농도 차이	

* 총 40 columns X 16998 rows, 모든 Feature마다 정상(0) or 불량(1) Label 존재

04 EDA

04 EDA Processing 과정

분포 확인

1) 데이터 분포 확인을 통해
진행 과정 회의

결측치 대체

1) 0으로 대체
2) 보간법 대체
3) 최빈값 대체
4) 중앙값 대체
5) MICE (다중대치법)

시각화

1) Heatmap Correalation
2) Feature Importance
3) Histogram

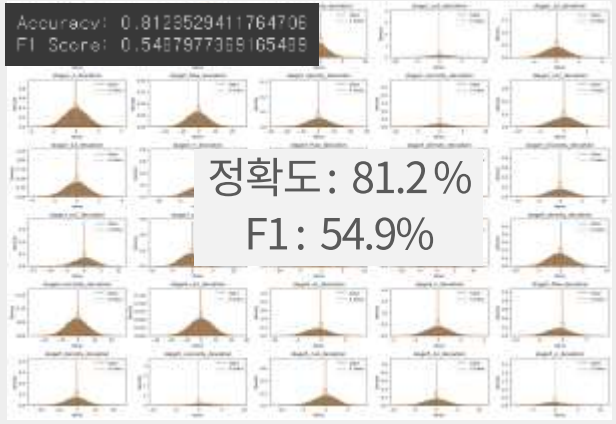
Para. 선택

1) 불량 영향을 준다고
판단되는 요소 출력
2) 몇 개의 Feature를 사용할지
결정

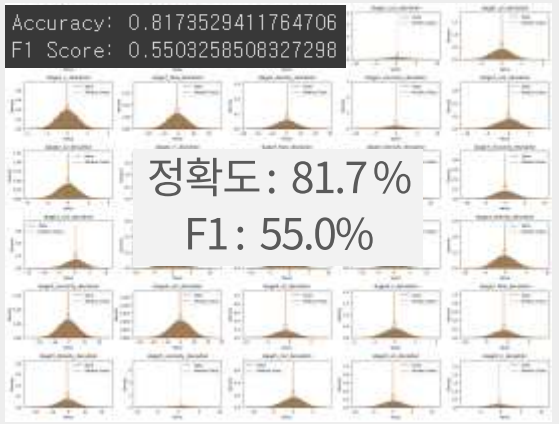
04 EDA Processing 결측치 처리

Stage1	Null	Stage2	Null	Stage3	Null	Stage4	Null	Stage5	Null
Flow	5810	Flow	1366	Flow	2913	Flow	3290	Flow	2907
Density	925	Density	2506	Density	866	Density	1708	Density	2710
Viscosity	1407	Viscosity	5622	Viscosity	2904	Viscosity	1086	Viscosity	7250
O_2	2138	O_2	1411	O_2	1717	O_2	3010	O_2	3331
N	917	N	2352	N	1246	N	2389	N	5542
CO_2	6348	CO_2	2239	CO_2	2629	CO_2	1096	CO_2	2217

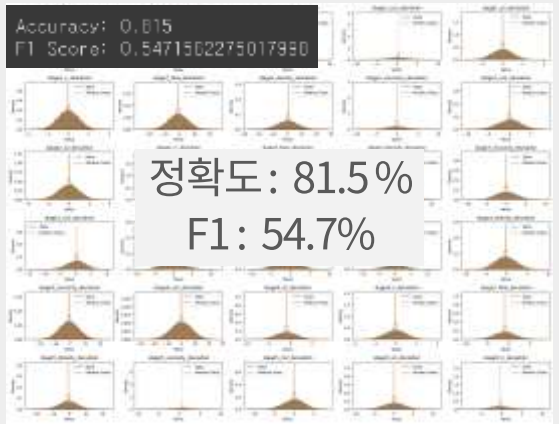
04 EDA Processing 결측치 처리



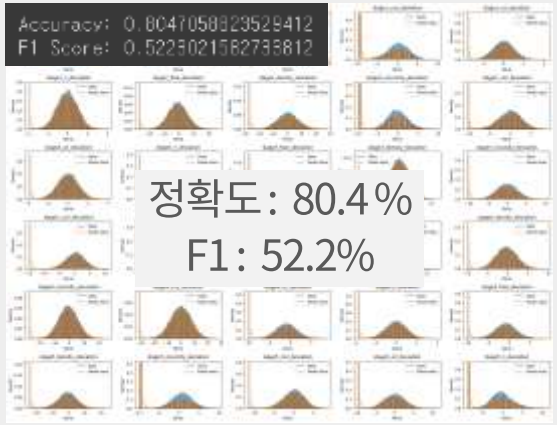
*0으로 결측치 처리



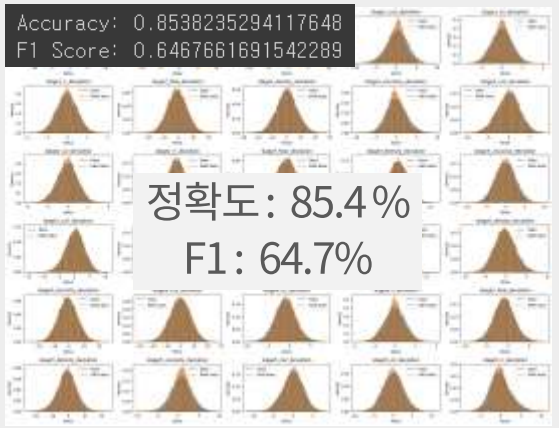
*평균값으로 결측치 처리



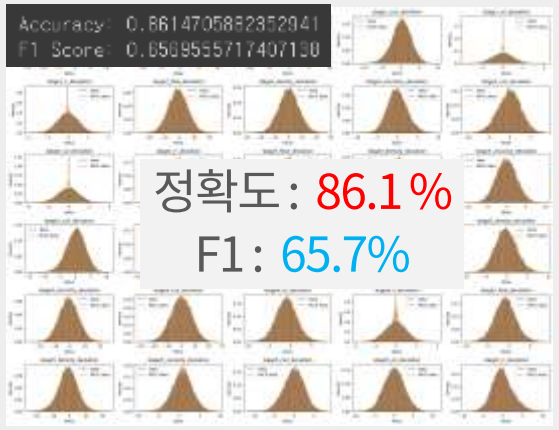
*중앙값으로 결측치 처리



*최빈값으로 결측치 처리



*KNN으로 결측치 처리



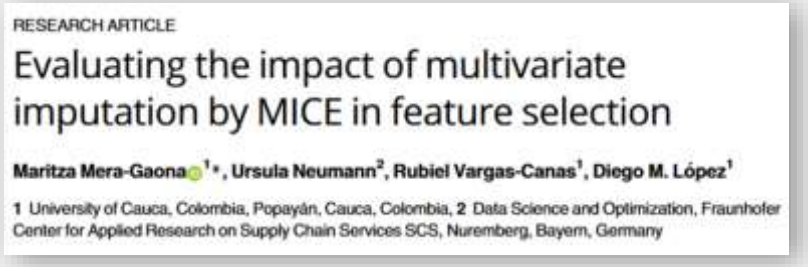
*MICE로 결측치 처리

04 EDA Processing 결측치 처리

MICE(Multiple Imputation by Chained Equations)

MICE는 multiple-imputation의 기법 중 하나로,
확률모형을 이용해 결측값을 여러 번 대체시켜서 여러 개의 데이터 세트를 획득한 다음 각각의 자료 세트마다 모형 학습 및 예측을 진행. 이렇게 여러 개의 데이터 세트의 모형 예측 값들의 평균 등을 구해 최종 추정치로 결정하는 방법이다.

그 중 PMM(Predictive Mean Matching)이란 결측치들을 명시적 모형(선형회귀 등)으로 예측값을 구한 후, 그 예측값과 가장 가까운 완비데이터(Complete Cases)의 값으로 대체하는 방법을 말한다.



RESEARCH ARTICLE

Evaluating the impact of multivariate imputation by MICE in feature selection

Maritza Mera-Gaona[1]*, Ursula Neumann[2], Rubiel Vargas-Canas[1], Diego M. López[1]

1 University of Cauca, Colombia, Popayán, Cauca, Colombia, 2 Data Science and Optimization, Fraunhofer Center for Applied Research on Supply Chain Services SCS, Nuremberg, Bayern, Germany

Method	Description	Scale Type
pmm	Predictive mean matching	Any*
midastouch	Weighted predictive mean matching	Any
sample	Random sample from observed values	Any
cart	Classification and regression trees	Any
rf	Random forest imputation	Any
mean	Unconditional mean imputation	Numeric
norm	Bayesian linear regression	Numeric
norm.boot	Normal imputation with bootstrap	Numeric
norm.nob	Normal imputation ignoring model error	Numeric
norm.predict	Normal imputation, predicted values	Numeric
quadratic	Imputation of quadratic terms	Numeric
ri	Random indicator for nonignorable data	Numeric
logreg	Logistic regression	Binary*
logreg.boot	Logistic regression with bootstrap	Binary
polr	Proportional odds model	Ordinal*
polyreg	Polytomous logistic regression	Nominal*
lda	Discriminant analysis	Nominal

* MICE에서 선택할 수 있는 확률모형

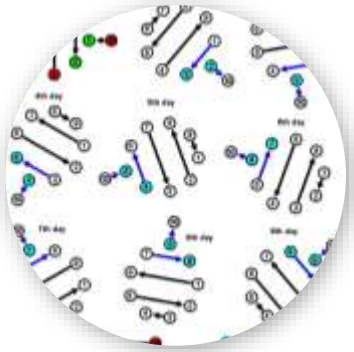
04 EDA Processing 결측치 처리

M개의데이터셋생성

특정 알고리즘에 따라 결측치를 대체 값으로 바꾼 m개의 데이터셋을 생성. 여기서 알고리즘이란 선형회귀 알고리즘을 뜻하며, default함수는 pmm이다.

데이터셋분석

M개의 데이터셋을 분석하고, 그 결과에서 모수 추정치와 표준오차를 계산.



대입과결합

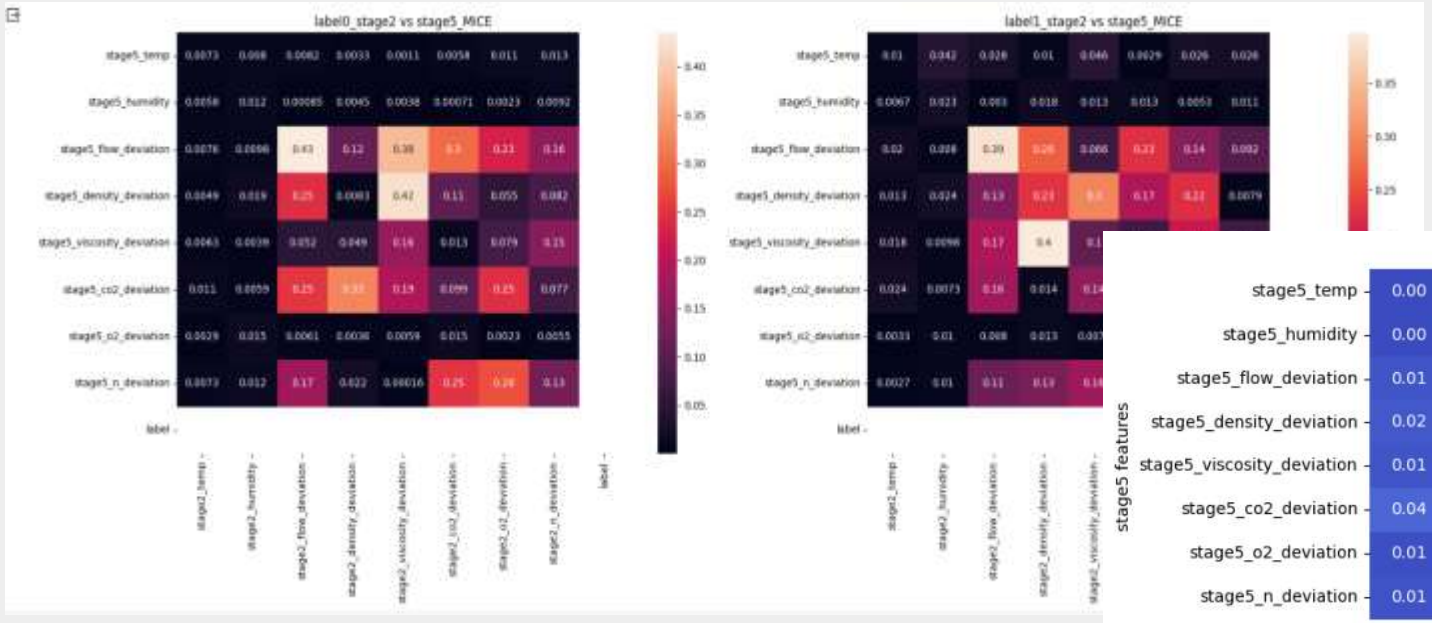
각 데이터셋의 결과를 Round Robin 규칙에 의해 결합. Round Robin 규칙은 모든 변수에 동등한 기회를 부여하며, 예측한 모든 값을 결측치에 대입해보고 결정.

반복

원하는 횟수만큼 위 과정을 반복하며 결측치 대체 값의 변화가 없을때 까지 해당 과정을 반복.

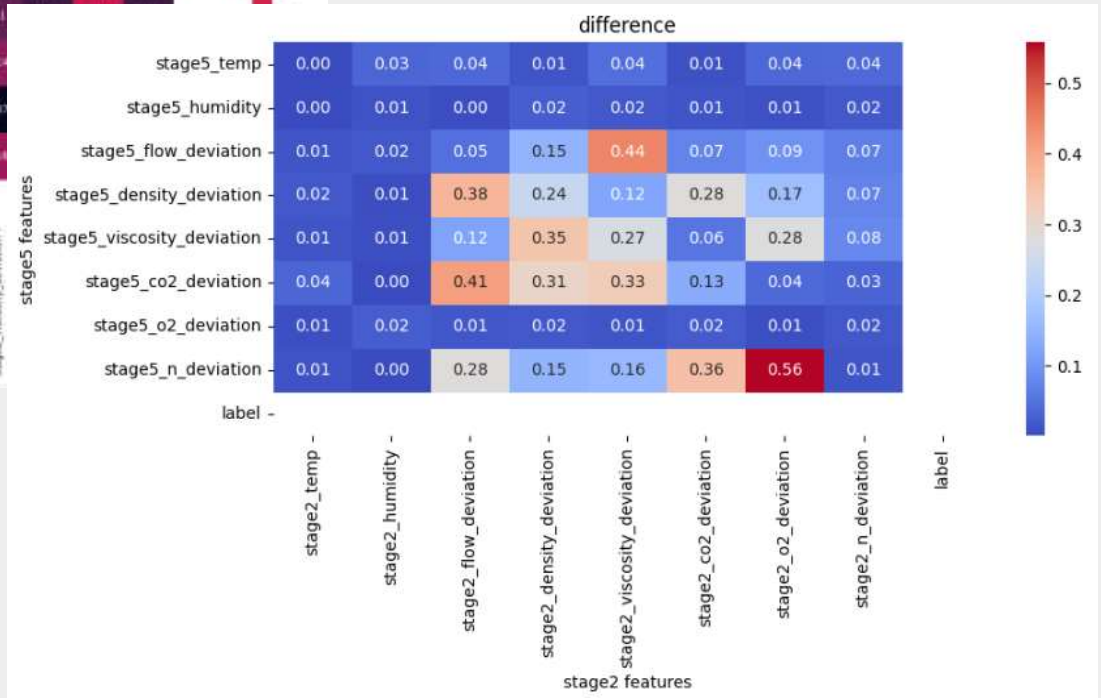
04 EDA Processing Heatmap

불량 Stage 상관관계

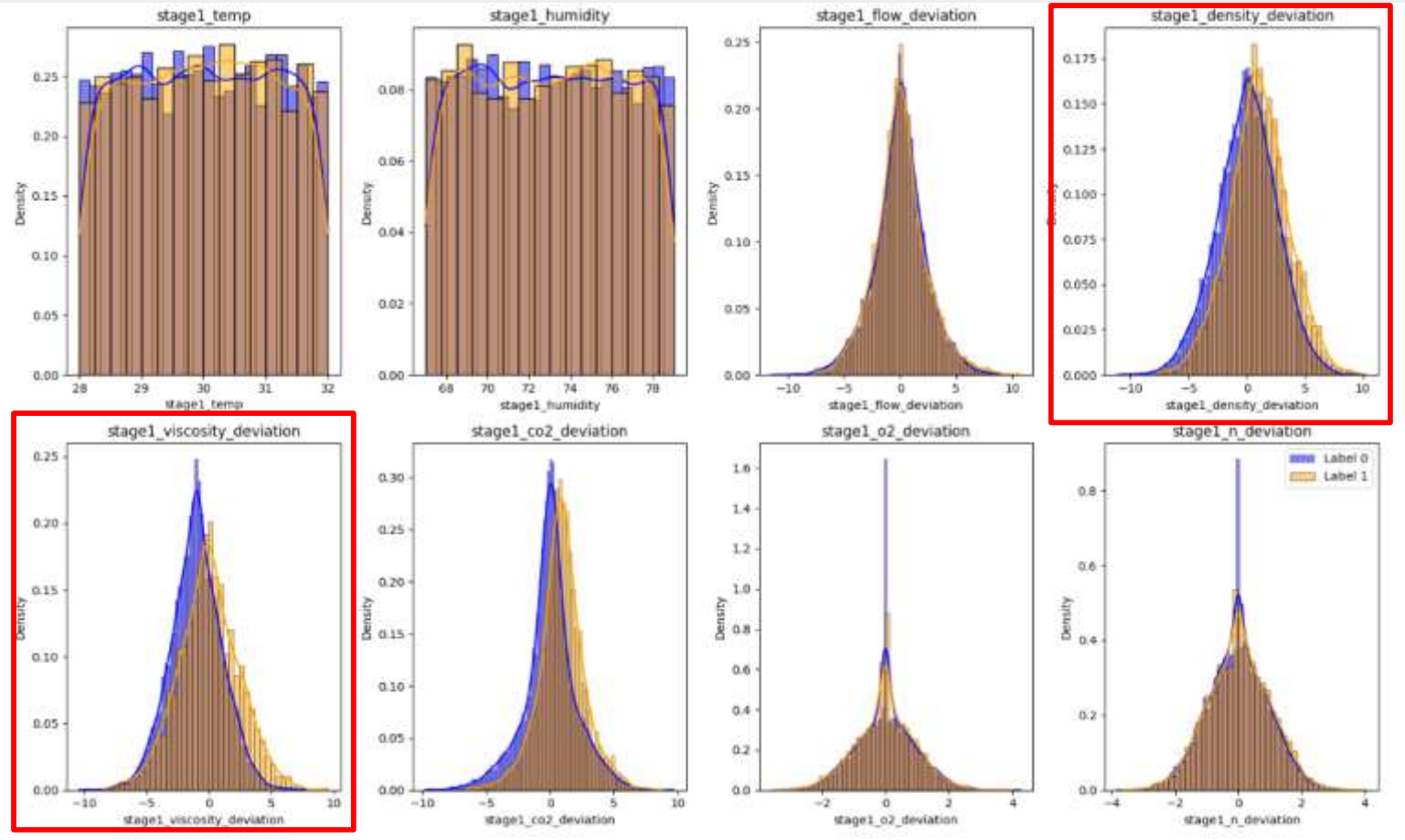


정상 Stage 간 상관관계

해당 Stage 정상-불량 상관관계 차이



04 EDA Processing Histogram

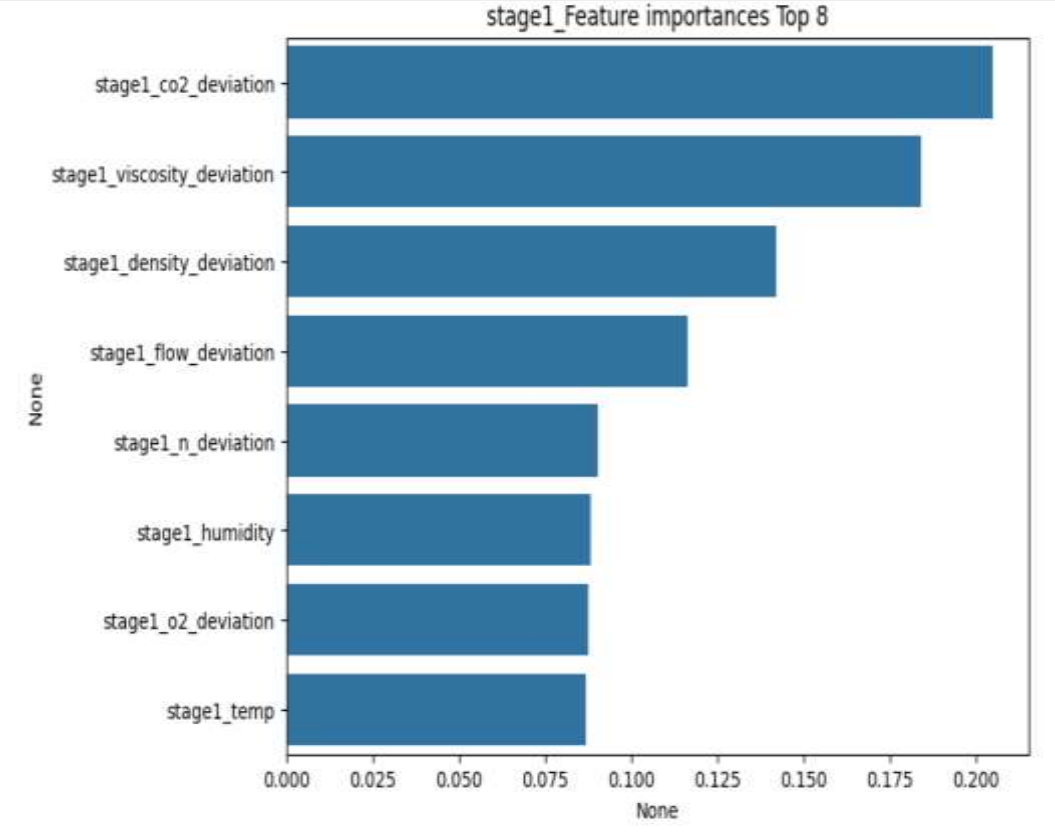


히스토그램

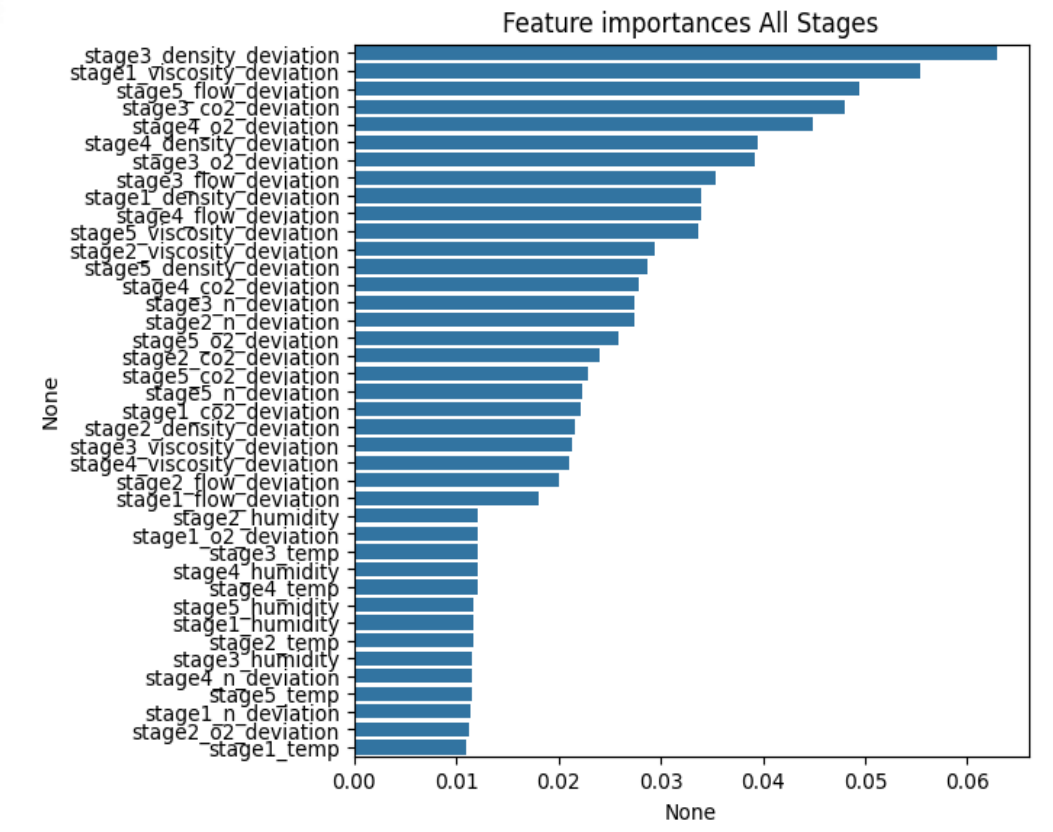
```
# 각 스테이지별로 겹치는 부분의 넓이 비율을 출력
for stage_num, stage_overlap_ratios_sorted in enum
    print(f"스테이지 {stage_num} 겹치는 부분의 넓이
    for feature_name, overlap_ratio in stage_overl
        print(f"  특성 {feature_name}: {overlap_ra
```

```
특성 stage3_density_deviation: 71.20%
특성 stage5_flow_deviation: 79.97%
특성 stage4_density_deviation: 82.38%
특성 stage3_flow_deviation: 82.68%
특성 stage4_o2_deviation: 82.90%
특성 stage1_viscosity_deviation: 83.64%
특성 stage3_n_deviation: 84.55%
특성 stage4_flow_deviation: 85.06%
특성 stage1_density_deviation: 85.60%
특성 stage2_n_deviation: 85.64%
특성 stage5_viscosity_deviation: 86.00%
특성 stage1_co2_deviation: 87.42%
특성 stage3_o2_deviation: 87.75%
```

04 EDA Processing Feature Importance

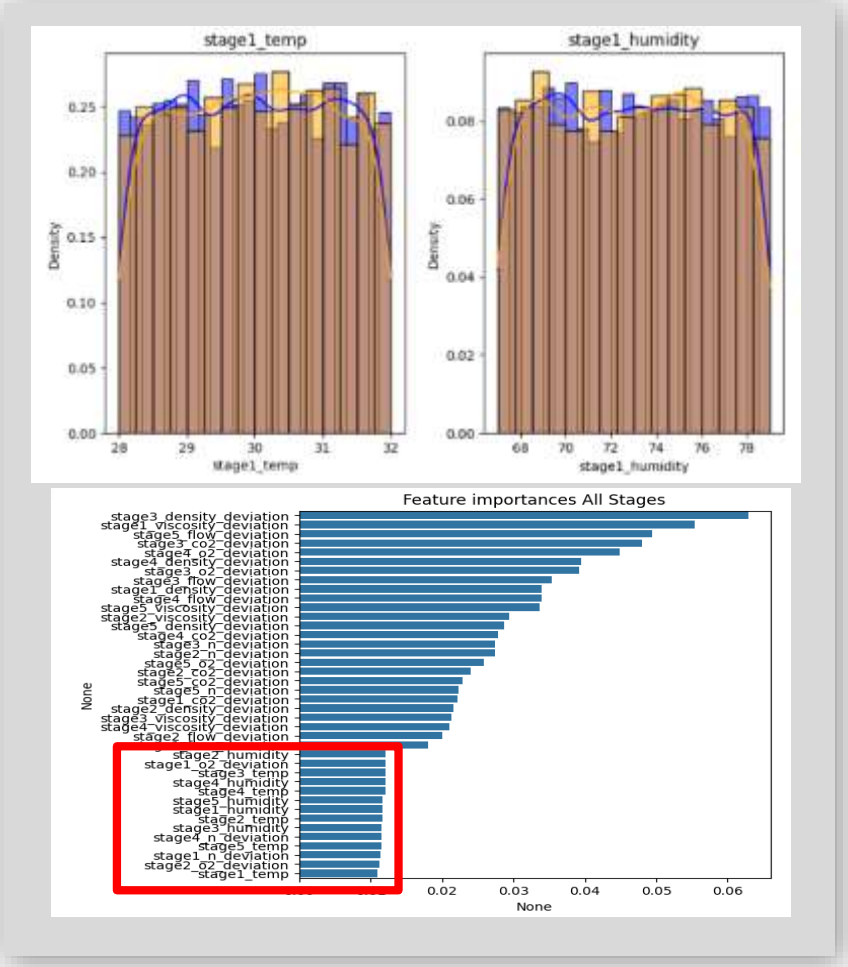


개별 Stage Feature Importance



전체 Feature Importance

04 EDA Processing Feature Selection



1) 히스토그램 상, 1 / 0 분포 차이가 없음
2) Feature Importance 상으로도 결과에 영향을 크게 주는 요인이 아님

→ 온도, 습도 등 일부 Feature 제거

Feature	Feature
stage1_flow	stage3_viscosity
stage1_CO_2	stage3_flow
stage1_viscosity	stage4_flow
stage1_density	stage4_density
stage2_n	stage4_O_2
stage2_CO_2	stage4_CO_2
stage2_flow	stage5_viscosity
stage2_viscosity	stage5_n
stage3_density	stage5_O_2
stage3_O_2	stage5_CO_2
stage3_CO_2	stage5_density

* 최종 Feature Selection

05 Model Selection

05 Model Selection

모델 선정

- Logistic Regression, Decision Tree, Random Forest, SVC, XGBoost, LightGBM을 통해 최적의 모델 선정.

모델 선정 기준

- F1-score를 최우선으로 하고 추가로 Precision, Recall, Confusion-Matrix를 참고하여 평가를 통해 프로젝트 데이터에 가장 최적의 성능을 내는 모델 선정.

변인 통제

- 모델의 성능으로만 평가하기 위해 모델 평가에 사용된 데이터셋은 MICE로 결측값 처리, StandardScaler로 표준화, PCA로 Feature Extraction된 데이터셋을 사용.

05 Model Selection

LogisticRegression

```
model = RandomForestClassifier(random_state=42)
model.fit(X_train, y_train)
y_pred = model.predict(X_test)
accuracy = accuracy_score(y_test, y_pred)
report = classification_report(y_test, y_pred)
```

```
Accuracy: 0.87
Classification Report:
              precision    recall  f1-score   support

           0       0.89      0.96      0.92      2717
           1       0.77      0.51      0.61       683

    accuracy                           0.87      3400
   macro avg       0.83      0.73      0.77      3400
weighted avg       0.86      0.87      0.86      3400

F1-score: 0.6102292768959435
```

DecisionTree

```
model = DecisionTreeClassifier(random_state=42)
model.fit(X_train, y_train)
y_pred = model.predict(X_test)
accuracy = accuracy_score(y_test, y_pred)
report = classification_report(y_test, y_pred)
f1 = f1_score(y_test, y_pred)
```

```
Accuracy: 0.8544117647058823
Classification Report:
              precision    recall  f1-score   support

           0       0.91      0.91      0.91      2717
           1       0.64      0.64      0.64       683

    accuracy                           0.85      3400
   macro avg       0.77      0.77      0.77      3400
weighted avg       0.85      0.85      0.85      3400

F1-score: 0.6373626373626373
```

RandomForest

```
model = RandomForestClassifier(random_state=42)
model.fit(X_train, y_train)
y_pred = model.predict(X_test)
accuracy = accuracy_score(y_test, y_pred)
report = classification_report(y_test, y_pred)
f1 = f1_score(y_test, y_pred)
```

```
Accuracy: 0.92
Classification Report:
              precision    recall  f1-score   support

           0       0.91      1.00      0.95      2717
           1       0.97      0.62      0.76       683

    accuracy                           0.92      3400
   macro avg       0.94      0.81      0.85      3400
weighted avg       0.92      0.92      0.91      3400

F1-score: 0.7571428571428571
```

SVC

```
model = SVC(kernel='linear', random_state=42)
model.fit(X_train, y_train)
y_pred = model.predict(X_test)
accuracy = accuracy_score(y_test, y_pred)
report = classification_report(y_test, y_pred)
f1 = f1_score(y_test, y_pred)
```

```
Accuracy: 0.8847058823529412
Classification Report:
              precision    recall  f1-score   support

           0       0.88      0.98      0.93      2717
           1       0.89      0.49      0.63       683

    accuracy                           0.88      3400
   macro avg       0.89      0.74      0.78      3400
weighted avg       0.89      0.88      0.87      3400

F1-score: 0.6294896030245747
```

XGBoost

```
model = XGBClassifier(random_state=42)
model.fit(X_train, y_train)
y_pred = model.predict(X_test)
accuracy = accuracy_score(y_test, y_pred)
report = classification_report(y_test, y_pred)
f1 = f1_score(y_test, y_pred)
```

```
Accuracy: 0.9500294117647
Classification Report:
              precision    recall  f1-score   support

           0       0.96      0.99      0.97      2717
           1       0.95      0.84      0.89       683

    accuracy                           0.96      3400
   macro avg       0.96      0.91      0.93      3400
weighted avg       0.96      0.96      0.96      3400

F1-score: 0.8909657320872275
```

LightGBM

```
model = LGBMClassifier(random_state=42)
model.fit(X_train, y_train)
y_pred = model.predict(X_test)
accuracy = accuracy_score(y_test, y_pred)
report = classification_report(y_test, y_pred)
f1 = f1_score(y_test, y_pred)
```

```
Accuracy: 0.9541176470588235
Classification Report:
              precision    recall  f1-score   support

           0       0.95      0.99      0.97      2717
           1       0.95      0.81      0.88       683

    accuracy                           0.95      3400
   macro avg       0.95      0.90      0.92      3400
weighted avg       0.95      0.95      0.95      3400

F1-score: 0.8765822784810127
```

05 Model Selection

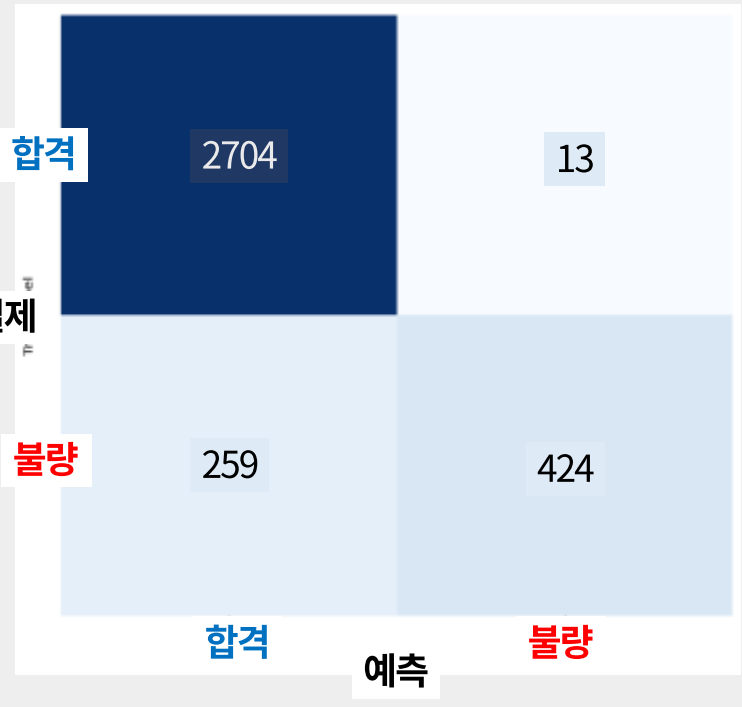
LogisticRegression

예측률 : 약 87%
F1 : 약 61%

DecisionTree

예측률 : 약 85%
F1 : 약 63%

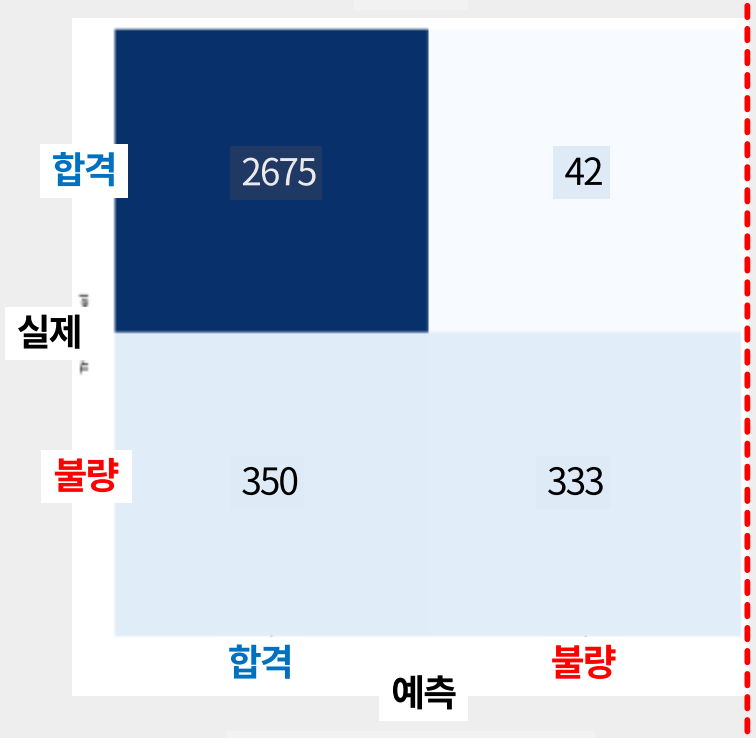
RandomForest



예측률 : 약 92%
F1 : 약 75%

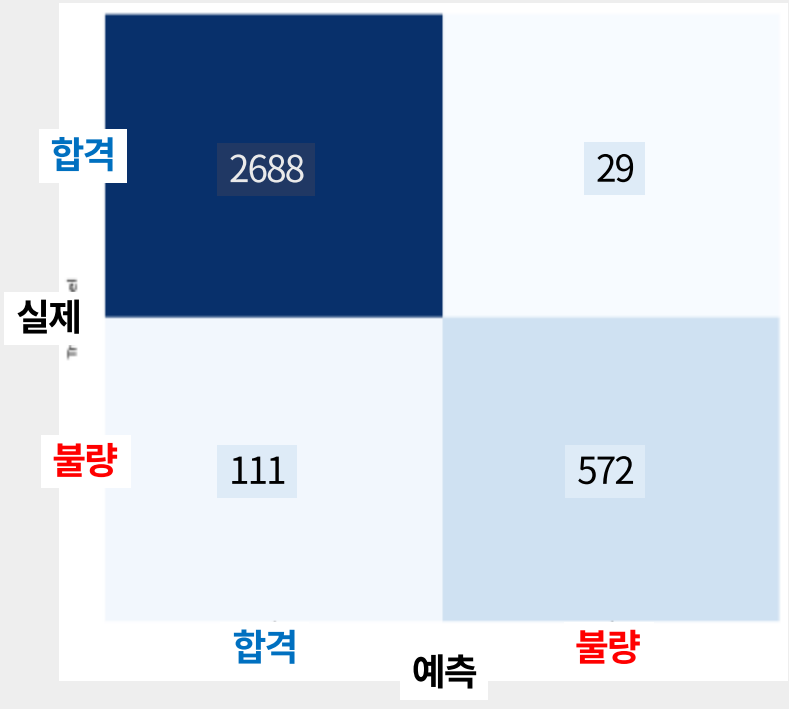
05 Model Selection

SVC



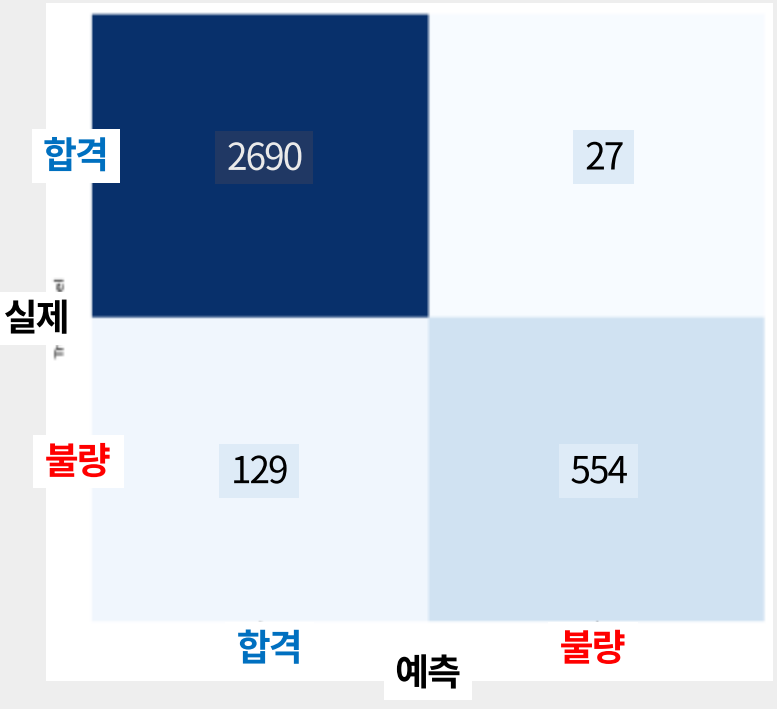
예측률 : 약 88%
F1 : 약 62%

XGBoost



예측률 약 95%, F1 약 89%으로
가장 적합한 모델이라 판단

LightGBM



예측률 : 약 95%
F1 : 약 87%

06 Modeling

06 Modeling Feature Selection

All

```
Accuracy: 0.9588235294117647
Classification Report:
              precision    recall  f1-score   support

           0       0.96      0.99      0.97      2717
           1       0.95      0.84      0.89       683

    accuracy                           0.96      3400
   macro avg       0.96      0.91      0.93      3400
weighted avg       0.96      0.96      0.96      3400

0.8909657320872275
```

30ea

```
Accuracy: 0.9541176470588235
Classification Report:
              precision    recall  f1-score   support

           0       0.95      0.99      0.97      2717
           1       0.95      0.81      0.88       683

    accuracy                           0.95      3400
   macro avg       0.95      0.90      0.92      3400
weighted avg       0.95      0.95      0.95      3400

F1-score: 0.8765822784810127
```

25ea

```
Accuracy: 0.965
Classification Report:
              precision    recall  f1-score   support

           0       0.97      0.99      0.98      2717
           1       0.96      0.86      0.91       683

    accuracy                           0.96      3400
   macro avg       0.96      0.93      0.94      3400
weighted avg       0.96      0.96      0.96      3400

0.908108108108108
```

22ea

```
Accuracy: 0.9635294117647059
Classification Report:
              precision    recall  f1-score   support

           0       0.96      0.99      0.98      2717
           1       0.96      0.85      0.90       683

    accuracy                           0.96      3400
   macro avg       0.96      0.92      0.94      3400
weighted avg       0.96      0.96      0.96      3400

0.903875968992248
```

20ea

```
Accuracy: 0.9673529411764706
Classification Report:
              precision    recall  f1-score   support

           0       0.97      0.99      0.98      2717
           1       0.96      0.87      0.91       683

    accuracy                           0.97      3400
   macro avg       0.96      0.93      0.95      3400
weighted avg       0.97      0.97      0.97      3400

0.9148119723714505
```

06 Modeling Feature Selection

Feature_selection	Accuracy	F1	Note.
40개	95%	89%	전체 Feature
20개	96.4%	90.4%	특성중요도로 추출한 상위 20개
22개	**96.7%**	**91.5%**	**특성중요도로 추출한 상위 22개**
25개	96.5%	90.8%	특성중요도로 추출한 상위 25개
30개	96.4%	90.7%	특성중요도로 추출한 상위 30개

* MICE, DataScaleing를 거친 데이터프레임을 활용

06 Modeling Parameter Optimization

파라미터 선정 방법

- GridSearch, Bayesian Optimization, RandomSearch를 통해 최적의 파라미터 선정

  * GridSearch로 산출한 파라미터가 최적 파라미터라 결론 지을 수 없기 때문에, 다양한 방법으로 최적파라미터 선정

Bayesian Optimization

- Surrogate model과 Accquision function으로 구성되어 미지의 목적함수(blank-box function)을 최대화 혹은 최소화하는 최적해를 찾는 기법

RandomSearch

- 찾고자 하는 파라미터 값의 범위를 지정하여, 설정한 iter값 만큼 Random하게 조합/반복하여 최적의 파라미터를 찾는 기법

06 Modeling Parameter Optimization

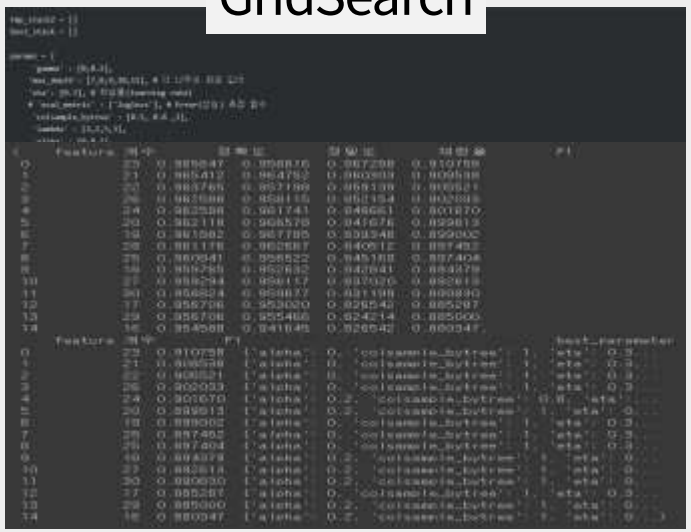
- eta(learning_rate) – 각 트리의 학습 단계에서 사용되는 학습률.
- n_estimators(num_boost_rounds) – 부스팅 반복 횟수. 트리의 개수를 결정
- max_depth – 트리의 최대 깊이
- min_child_weight – 분할 시, 각 리프 노드에 포함되어야 하는 최소한의 샘플 가중치 합.
- gamma – 정보 획득시 사용되는 최소 손실 감소.
- colsample_bytree – 각 트리를 구성할 때 사용할 특성 비율.
- alpha(reg_alpha) – L1 정규화 항의 가중치.
- lambda(reg_lambda) – L2 정규화 항의 가중치.
- objective – 학습 목적 함수.
- eval_metric – 모델의 평가 지표.

06 Modeling Parameter Optimization

Parameter_adjustment	Accuracy	F1	Note.
null	96.7%	91.5%	* 자세한 파라미터는 세부 모델링 참조
GridSearchCV	96.6%	91.1%	* 자세한 파라미터는 세부 모델링 참조
Bayesian Optimization	96.2%	90.4%	* 자세한 파라미터는 세부 모델링 참조
RandomSearchCV	**97.3%**	**93.0%**	* 자세한 파라미터는 세부 모델링 참조

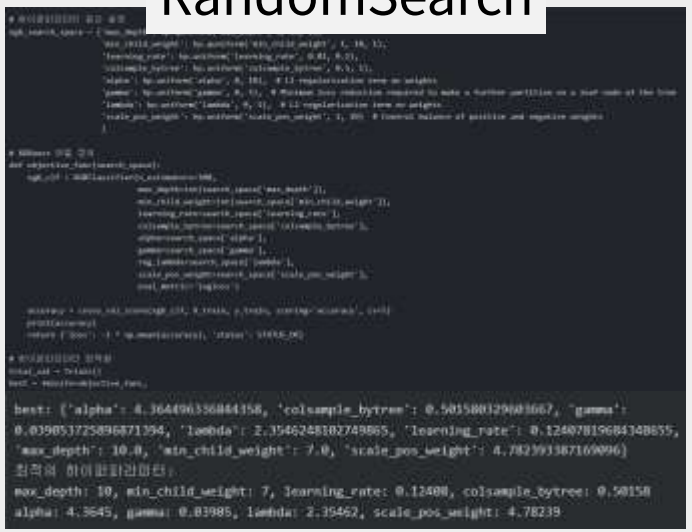
06 Modeling Parameter Optimization

GridSearch



Bayesian Optimization

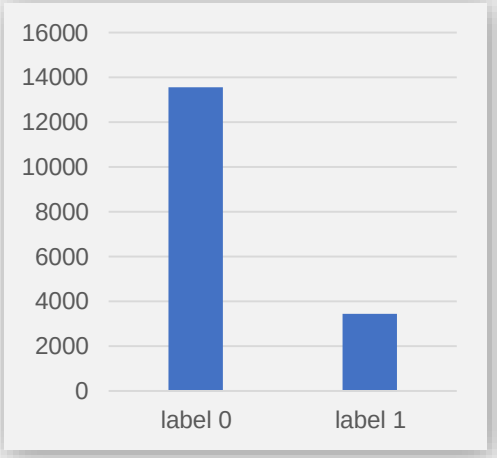
RandomSearch



06 Modeling SMOTE

SMOTE (Synthetic Minority Over-sampling Technique)

- 합성 소수 샘플링 기술로 다수 클래스를 샘플링하고 기존 소수 샘플을 보간하여 새로운 소수 인스턴스를 합성해내는 방법
- 데이터 셋의 label 0 : label 1 비율이 대략 4 : 1 수준으로 불균형이 심함
- 따라서, SMOTE를 통한 모델 최적화를 수행
- 데이터 수를 줄이기 보단, 늘려 비율을 맞추기 위해 Over-Sampling 선택

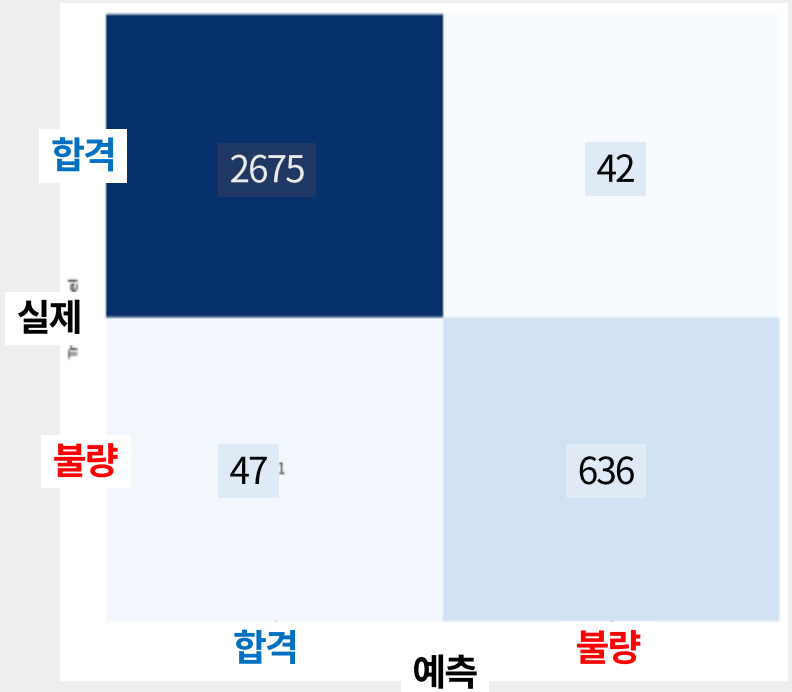


06 Modeling SMOTE

적용 전

예측률 : 97.3%
F1 : 93.0%

적용 후

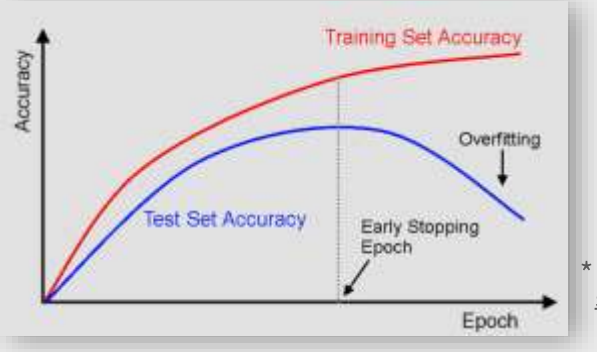


예측률 : 97.4%,
F1 : 93.5%

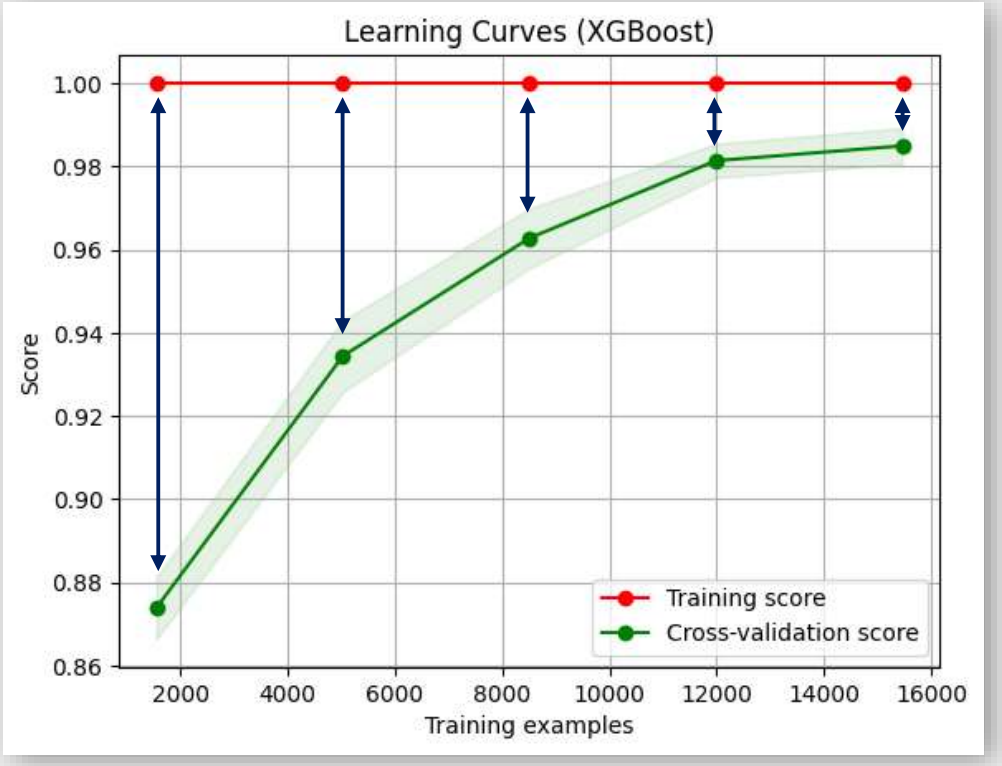
06 Modeling SMOTE

K-Fold 교차검증으로 과대적합 검증

- Learning Curve를 통해 시각화
- Train_score와 Validation_score의 오차가
  지속적으로 감소하는 추세를 통해서
  과대적합 모델이 아님을 검증



* 만약 과대적합이라면,
증가>감소추세로 바뀌는 구간 존재

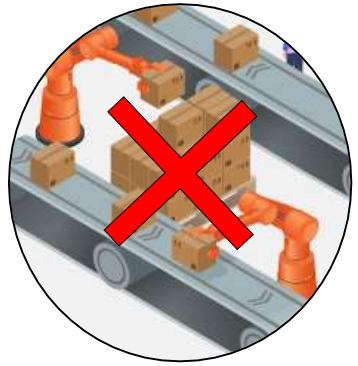


07 결론

07 결론 기대 효과

품질비용절감

97%의 높은 정확도로 데이터 기반불량 검출 자동화를 통해 별다른 시스템을 적용하지 않아도 품질 비용을 절감할 수 있다.



유지보수용이

부가적인 품질 테스터가 불필요 하며, 관리자의 유지보수와 공장 내 공간활용에 보다 용이하다.

인사이트제공

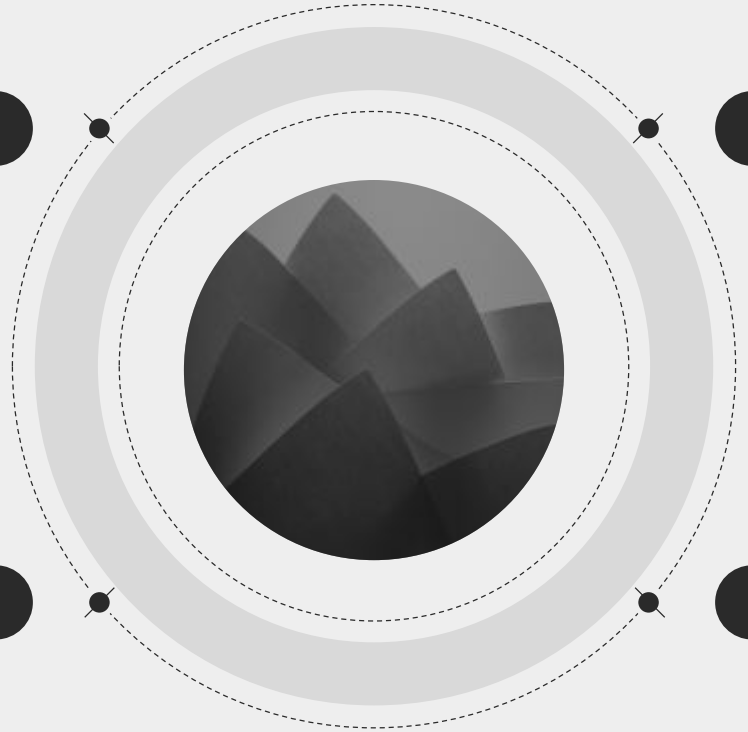
관리자에게 현재 상태를 보다 빠르게 파악할 수 있게 해주며, 이슈 원인파악에 도움을 준다.

즉각적인대응

스스로 학습하는 ML을 적용하여, 즉각적인 대응을 가능케 하며, 빠른 적응성을 통해 다양한 상황에 대처할 수 있다.

07 결론 추가 목표

현시스템의 한계점은 공정이 모두 마무리 된 이후, 불량 판별이 가능함
이러한 과정에서 불필요한 재화 소모와 추가 개선 사항을 포함하여
추가목표를 선정하였다.



Upper Lower line 특정

불량이 발생하는 각 feature의 범위를 특정하여, 불량품을 선별

Affected item 선별

Affected item에 대해 후속 공정 설비 parameter 조정으로 불량 가능성 감소.

새로운 Feature 도출

보다 쉽게 불량을 판별할 수 있는 새로운 feature를 도출

절감 비용 계산

실제로 절감되는 비용 계산하여 성과를 수치로 표시

07 결론 평가

잘한 점

1. 다양한 모델링 경험

EDA과정에서 다양한 모델링과 해석 방법을 직접 적용해보며 보다 논리적이고
효율성을 높이기 위해 노력하는 과정에서 분석에 어느정도 능숙해졌다.

2. 데이터 기반 의사결정

특성별 중요도, 상관성 분석, 데이터 시각화 등 을 통해 데이터에 의거하여
단계를 나눠 수행하였다.

아쉬운 점

1. 딥러닝 과정

딥러닝에 대한 이해와 적용이 미숙하여 결론을 내리지 못하였다. 프로젝트가 끝나서라도
스스로 공부하고 적용해보는 과정을 가져볼 예정이다.

2. 초기 목표 미달성

불량 범위 선정과 후공정 파라미터 조정을 통한 불량 개선 목표를 달성하지 못하였다.
DL과 DL 외의 방법으로 가능할 수 있게끔 조사와 시도해 보고 싶다.

07 조원 별 느낀 점

박장훈

모델링 과정을 직접 수행해보며, 보다 능숙해진 점이 좋았다. 그리고 데이터를 기반으로 해석하는 것에 대한 여러 관점이 더 생긴 것 같아서 이후에 큰 도움이 될 것 같다.

이석재

모델 학습 시 사용되는 파라미터, 지식이 부족하여 초반에 분석 결과에 대한 의미를 찾는 것이 쉽지 않았습니다. 팀원들끼리 의견을 공유하며 부족한 부분을 하나씩 채워나가며 EDA, 모델링, 데이터 해석 등 여러 방면에서 능숙해질 수 있었습니다.

이영철

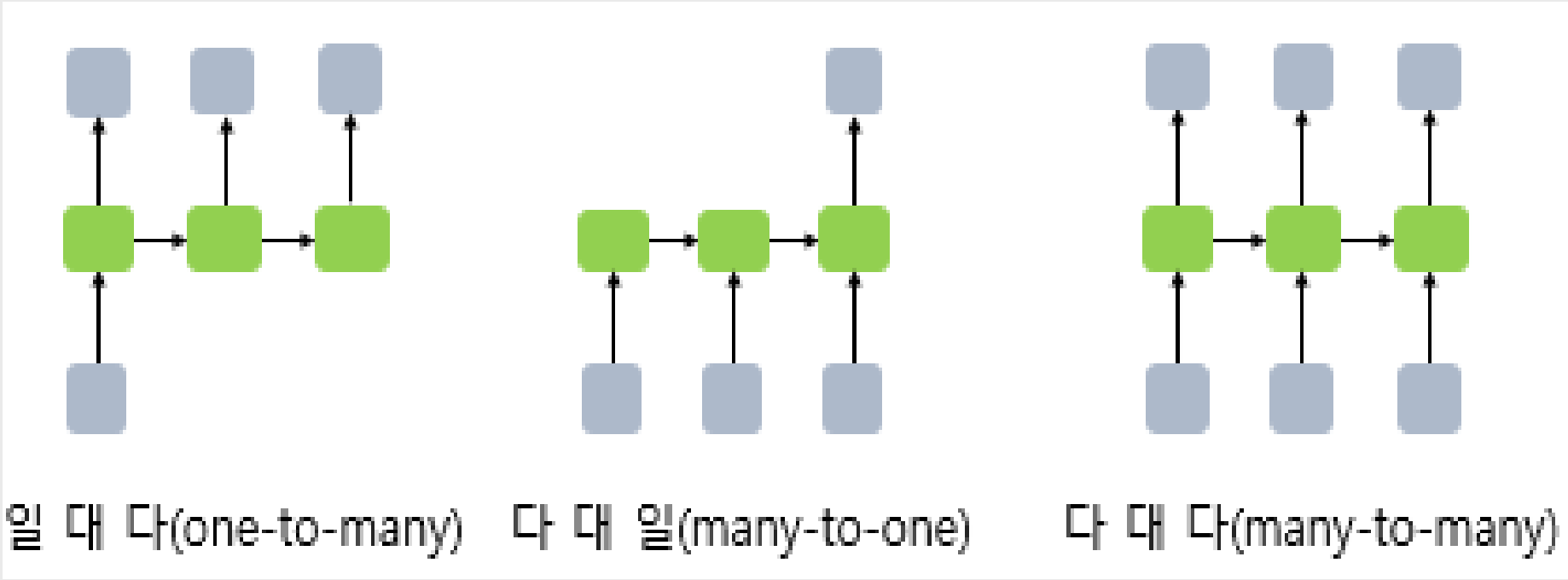
이론으로만 배웠던 데이터 분석을 프로젝트로 진행하며 데이터를 해석하는 방법과 코딩으로만 데이터를 분석하는 것이 아니라는 것을 깨달았습니다. 배경지식과 EDA과정을 기반으로 데이터를 해석, 분류하고 원하는 방향으로 재조립하며 모델링하는 과정에 대해 조금이나마 알 수 있는 시간이었습니다.

정의석

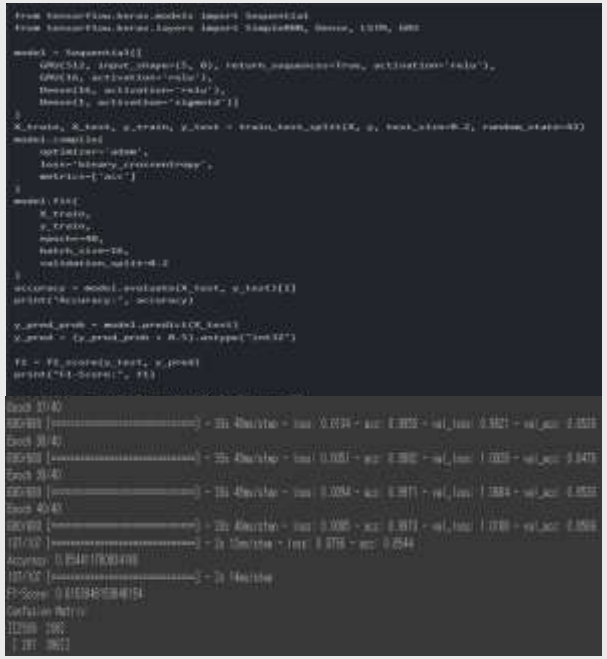
막막해 보이던 데이터가 한개 한개의 과정을 거치면서 다듬어지고, 또 그 과정을 통해 해석 불가능 할 것 같던 데이터들이 그래프와 데이터로 출력되고 그 결과를 해석해나가는 과정을 거치면서 매우 재미있고 더 다양한 것도 시도해보고 싶은 마음이 생겼습니다.

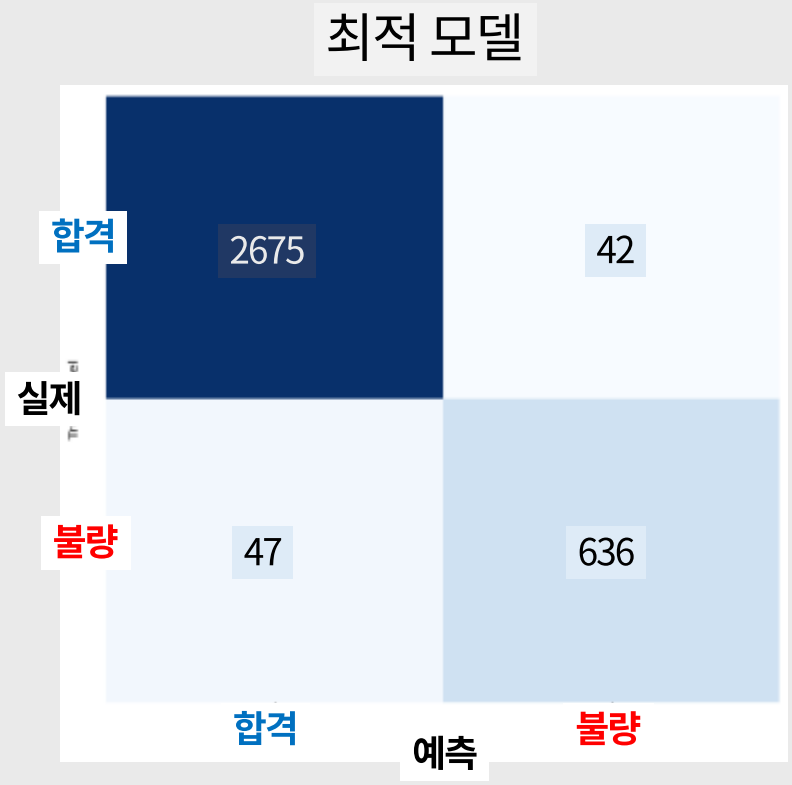
※ 추가 모델링(RNN)

※ RNN(Recurrent Neural Network)



※ RNN(Recurrent Neural Network)





예측률 : 97.4%,
F1 : 93.5%

예측률 : 85.4%,
F1 : 61.3%

THANK YOU