KUBIG
DATA SCIENCE & AI

스마트 공장 제품 품질 상태 분류 AI

ML Team2 | 19기 이지운, 20기 김재훈

KUBIG
DATA SCIENCE & AI

CONTENTS

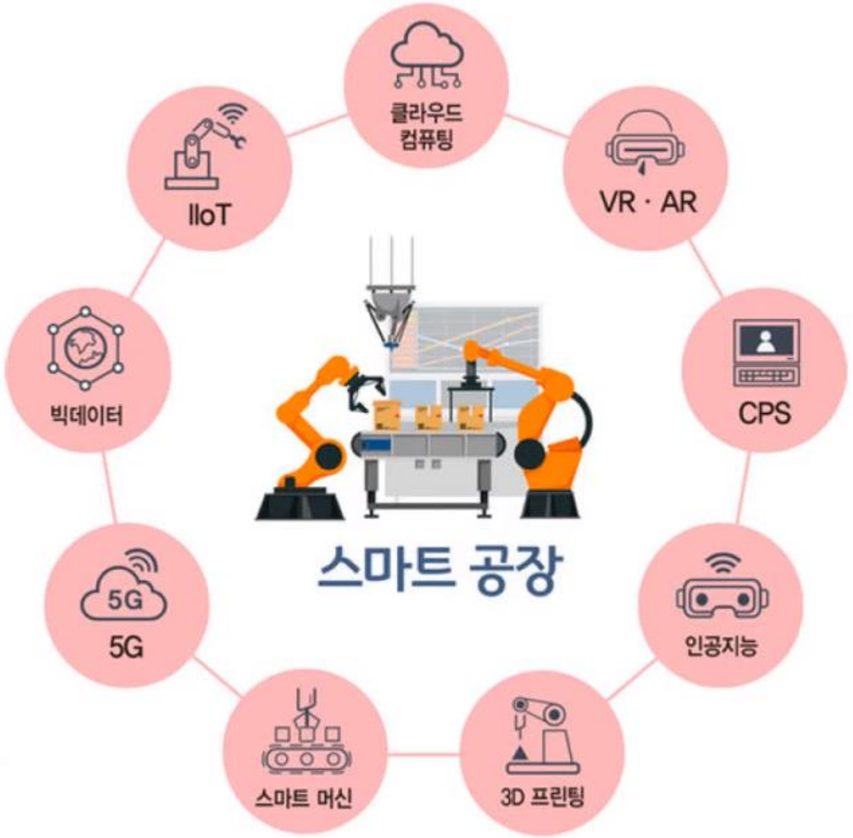
KUBIG
DATA SCIENCE & AI

01. 주제 선정

01. 주제 선정

스마트 공장 제품 품질 상태 분류 AI

- 공정 데이터에서 인사이트를 발굴하고 해석하여 자동화된 프로세스 구현
- 실제 스마트 공장 데이터를 기반으로 제품의 품질 상태 분류하는 AI 모델 개발



KUBIG
DATA SCIENCE & AI

02. EDA, 데이터 전처리

02. 데이터

- PRODUCT_ID : 제품의 고유 ID
- Y_Class : 제품 품질 상태(Target)

  0 : 적정 기준 미달 (부적합) / 1 : 적합 / 2 : 적정 기준 초과 (부적합)
- Y_Quality : 제품 품질 관련 정량적 수치
- TIMESTAMP : 제품이 공정에 들어간 시각
- LINE : 제품이 들어간 공정 LINE 종류

  ('T050304', 'T050307', 'T100304', 'T100306', 'T010306', 'T010305' 존재)
- PRODUCT_CODE : 제품의 CODE 번호 ('A_31', 'T_31', 'O_31' 존재)
- X_1 ~ X_2875 : 공정 과정에서 추출되어 비식별화된 변수

02. EDA

```
[ ]  # line에 따른 사용 변수 개수 확인

     line_counts = train_df['LINE'].value_counts()
     line_counts
```

LINE	count
T100304	175
T100306	174
T050304	78
T010306	70
T010305	59
T050307	42

dtype: int64

제품 LINE별 사용 변수 개수 확인

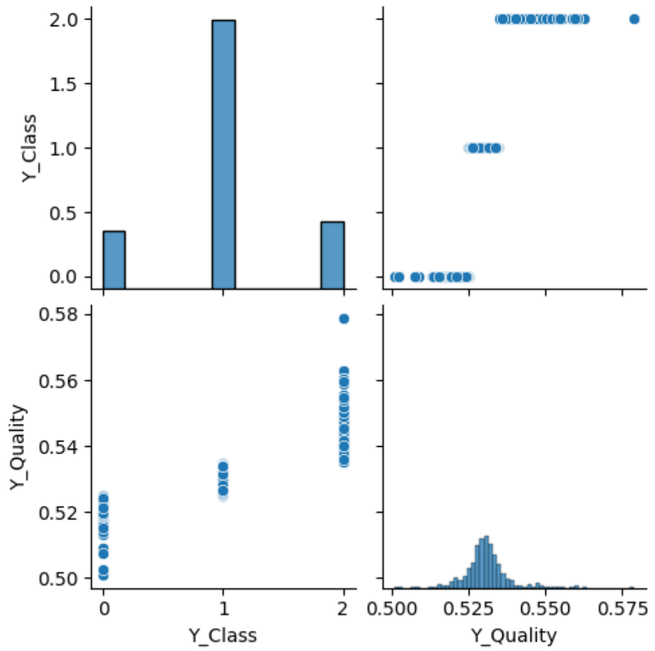
(T100304, T100306) &

(T050304, T010306, T010305, T050307)

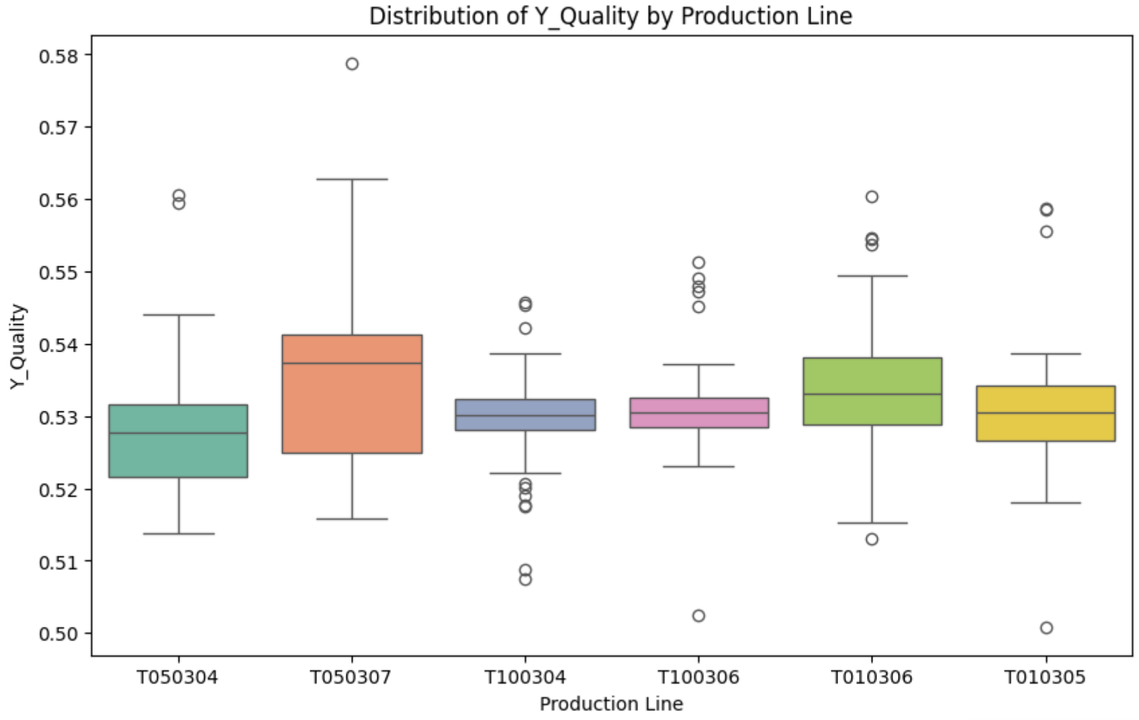
사용 변수 비슷한 LINE 끼리 묶어서 train 진행

02. EDA

	Y_Class	min	max
0	0	0.500856	0.525067
1	1	0.525086	0.534843
2	2	0.534951	0.578841



Y_Quality 구간에 따라 Y_Class 결정



LINE별 Y_Quality 분포 확인

02. 전처리

- 결측치는 평균값으로 대체, 열 전체가 결측값인 경우는 drop

```
#train
#X의 평균으로 대체
col_train = ['PRODUCT_ID', 'Y_Class', 'Y_Quality', 'TIMESTAMP', 'LINE', 'PRODUCT_CODE']
train_df_excluded = train_df[col_train]
train_df_filtered = train_df.drop(columns = col_train)
mean_train = train_df_filtered.mean()
train_df_perfect = train_df_filtered.fillna(mean_train)

#결측값만 있는 열 제외
train_df_perfect = train_df_perfect.dropna(axis=1)

#다시 합치기
train_df = pd.concat([train_df_excluded, train_df_perfect], axis = 1)
train_df.head()
```

	PRODUCT_ID	Y_Class	Y_Quality	TIMESTAMP	LINE	PRODUCT_CODE	X_1	X_2	X_3	X_4	...	X_2862	X_2863	X_2864	X_2865	X_2866	X_2867	X_2868	X_2869	X_2870	X_2871
0	TRAIN_000	1	0.533433	2022-06-13 5:14	T050304	A_31	2.409742	95.123209	0.0	45.0	...	189.0	383.0	368.296296	353.0	39.34	40.89	32.56	34.09	77.77	1.0
1	TRAIN_001	2	0.541819	2022-06-13 5:22	T050307	A_31	2.409742	95.123209	0.0	45.0	...	185.6	383.0	367.735849	353.0	38.89	42.82	43.92	35.34	72.55	1.0
2	TRAIN_002	1	0.531267	2022-06-13 5:30	T050304	A_31	2.409742	95.123209	0.0	45.0	...	165.5	383.0	367.320755	353.0	39.19	36.65	42.47	36.53	78.35	1.0
3	TRAIN_003	2	0.537325	2022-06-13 5:39	T050307	A_31	2.409742	95.123209	0.0	45.0	...	165.8	384.0	369.188679	353.0	37.74	39.17	52.17	30.58	71.78	1.0
4	TRAIN_004	1	0.531590	2022-06-13 5:47	T050304	A_31	2.409742	95.123209	0.0	45.0	...	182.6	383.0	367.351852	352.0	38.70	41.89	46.93	33.09	76.97	1.0

5 rows × 2799 columns

- Y_Class를 정수형에서 범주형으로 변환

```
[ ]  #정수형인 Y_Class를 범주형으로 바꾸기
     line1_train_scaled_df['Y_Class'] = line1_train_scaled_df['Y_Class'].astype('category')
     line2_train_scaled_df['Y_Class'] = line2_train_scaled_df['Y_Class'].astype('category')
```

KUBIG
DATA SCIENCE & AI

03. 분석 결과

03. 분석결과

H2O AUTO ML로 분석 진행.

LINE1의 분석결과는 다음과 같다

```
#line1의 bestmodel찾기
from h2o.automl import H2OAutoML

aml = H2OAutoML(max_runtime_secs=120, seed=42, exclude_algos=None, include_algos= None)
aml.train(y='Y_Class', training_frame = line1_train_h2o)
```

Cross-Validation Metrics Summary:

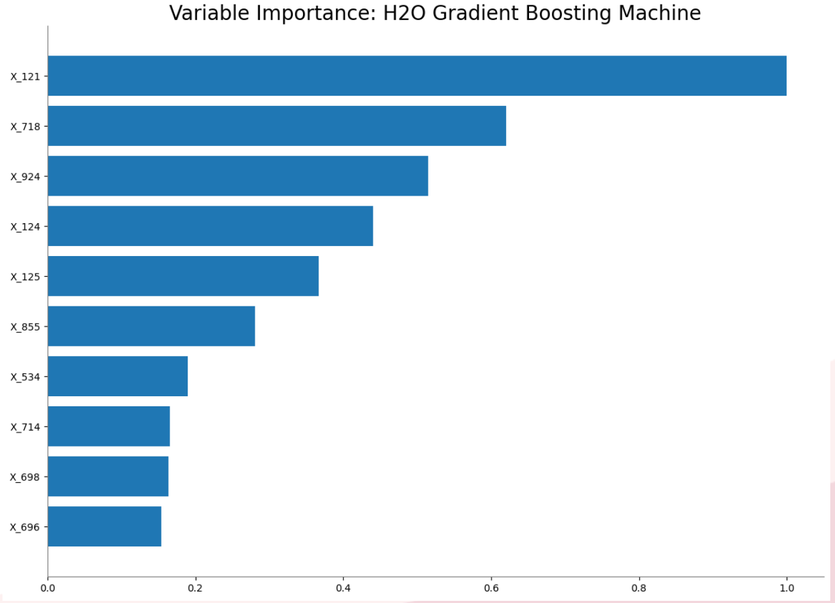
	mean	sd	cv_1_valid	cv_2_valid	cv_3_valid	cv_4_valid	cv_5_valid
accuracy	0.8312215	0.1128975	0.8857143	0.8142857	0.8857143	0.6428571	0.9275362
aic	nan	0.0	nan	nan	nan	nan	nan
auc	nan	0.0	nan	nan	nan	nan	nan
err	0.1687785	0.1128975	0.1142857	0.1857143	0.1142857	0.3571429	0.0724638
err_count	11.8	7.918333	8.0	13.0	8.0	25.0	5.0
loglikelihood	nan	0.0	nan	nan	nan	nan	nan
logloss	0.8979419	0.0633402	0.8657002	0.8916382	0.8697084	1.0086031	0.8540598
max_per_class_error	0.74	0.1474788	0.5	0.75	0.8	0.9	0.75
mean_per_class_accuracy	0.5635265	0.1387913	0.65	0.5214885	0.7121693	0.350641	0.5833333
mean_per_class_error	0.4364736	0.1387913	0.35	0.4785115	0.2878307	0.649359	0.4166667
mse	0.3491975	0.0301175	0.3339824	0.3453604	0.336136	0.4019681	0.3285407
pr_auc	nan	0.0	nan	nan	nan	nan	nan
r2	-1.3201374	0.844624	-1.351313	-0.4232682	-2.4242547	-0.5681875	-1.8336635
rmse	0.5905105	0.0248718	0.5779121	0.5876737	0.5797724	0.6340095	0.5731847

Top-3 Hit Ratios:

k	hit_ratio
1	0.9140401
2	0.9713467
3	1.0

```
ModelMetricsMultinomial: gbm
** Reported on cross-validation data. **

MSE: 0.34925670313824114
RMSE: 0.5909794439219025
LogLoss: 0.8980676629490099
Mean Per-Class Error: 0.48686974789915966
```



03. 분석결과

LINE2도 같은 방법으로 분석 진행

```
#line2의 bestmodel찾기
from h2o.automl import H2OAutoML
aml2 = H2OAutoML(max_runtime_secs=120, seed=42,exclude_algos=None, include_algos= None)
aml2.train(y='Y_Class', training_frame = line2_train_h2o)
```

	mean	sd	cv_1_valid	cv_2_valid	cv_3_valid	cv_4_valid	cv_5_valid
accuracy	0.6984127	0.0527196	0.6111111	0.6944444	0.7222222	0.75	0.7142857
aic	nan	0.0	nan	nan	nan	nan	nan
auc	nan	0.0	nan	nan	nan	nan	nan
err	0.3015873	0.0527196	0.3888889	0.3055556	0.2777778	0.25	0.2857143
err_count	10.8	1.9235384	14.0	11.0	10.0	9.0	10.0
loglikelihood	0.0	0.0	0.0	0.0	0.0	0.0	0.0
logloss	0.7343895	0.0877026	0.8233607	0.8150578	0.7170802	0.6110977	0.7053512
max_per_class_error	0.4695527	0.0744115	0.5555556	0.5	0.5	0.4285714	0.3636364
mean_per_class_accuracy	0.6723113	0.0536259	0.5814815	0.6757576	0.6830065	0.7203007	0.7010101
mean_per_class_error	0.3276887	0.0536259	0.4185185	0.3242424	0.3169935	0.2796992	0.2989899
mse	0.2523771	0.0338718	0.2855563	0.2801105	0.2547937	0.2018969	0.2395283
null_deviance	76.450165	1.7103728	77.93484	78.36156	76.24273	74.257065	75.45462
pr_auc	nan	0.0	nan	nan	nan	nan	nan
r2	0.5369533	0.0329723	0.5045770	0.5191745	0.5165261	0.5660723	0.5784165
residual_deviance	52.593903	6.461169	59.281967	58.684162	51.62978	43.99903	49.374584
rmse	0.5014292	0.0343858	0.5343746	0.5292547	0.5047709	0.4493294	0.4894162

Top-3 Hit Ratios:

k	hit_ratio
1	0.9608939
2	0.9944134
3	1.0

```
ModelMetricsMultinomialGLM: glm
** Reported on cross-validation data. **

MSE: 0.25784988944379994
RMSE: 0.5077892175340079
LogLoss: 0.7456402479813465
Null degrees of freedom: 178
Residual degrees of freedom: -5413
Null deviance: 382.25080361034
Residual deviance: 266.93920877732205
```

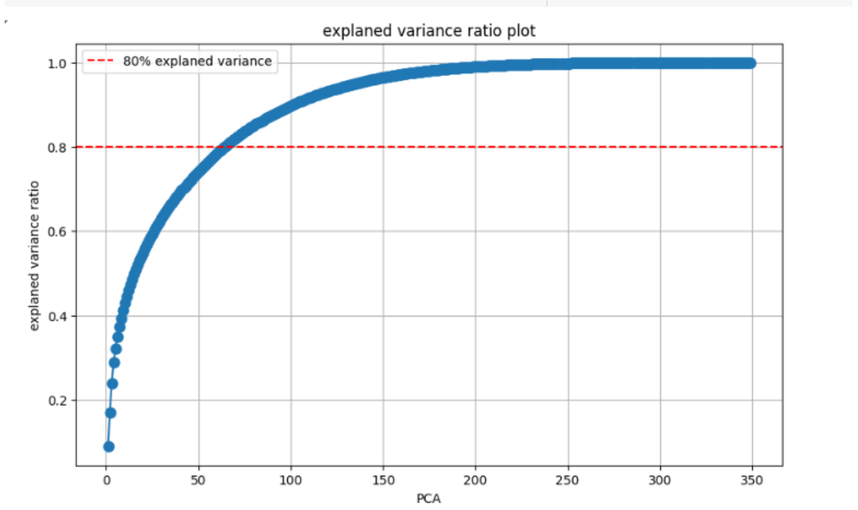
Variable Importances:

variable	relative_importance	scaled_importance	percentage
X_1382	0.10867089778184891	1.0	0.008881968664826446
X_1373	0.09981679916381836	0.9185237372768864	0.00356568036600815 2
X_360	0.09040562808513641	0.8319212404651422	0.0032294921870892 29
X_368	0.0902697741985321	0.8306711000008844	0.0032246391809803 49
X_354	0.09019683301448822	0.8299998882456424	0.0032220335579790 36
X_318	0.08810710161924362	0.8107699799822577	0.0031473836566730897
X_367	0.08773277699947357	0.8073254090123787	0.0031340119401042487
X_2793	0.08542750775814056	0.7861121008646839	0.003051662542597 59
X_380	0.07838988304138184	0.7213512047976762	0.0028002627733593 84
X_953	0.07563228160142899	0.6959754924750581	0.0027017550532753296
---	---	---	---
X_1121	0.0007482141954824328	0.006885138622710501	2.672789238634848e-05
X_1506	0.0007482141954824328	0.006885138622710501	2.672789238634848e-05
X_1094	0.0006450068904086947	0.005935415125616354	2.3041095530379612e-05
X_1505	0.0006450068904086947	0.005935415125616354	2.3041095530379612e-05
X_1148	0.0006373618962243199	0.005865065157589756	2.2767999158928816e-05
X_1507	0.0006373618962243199	0.005865065157589756	2.2767999158928816e-05
X_1589	0.0005591925582848489	0.005145743429923607 5	1.9975614752200003e-05
X_1409	0.0005573137896135449	0.005128454820832 74	1.9908500913450824e-05
X_1590	0.000555527803953737	0.005112020009891974 5	1.9844701492366425e-05
X_1665	0.0005252673290669918	0.004833560224389037 5	1.876372933062973e-05

[1863 rows x 4 columns]

03. PCA

LINE1, LINE2에 대해 차원 축소 진행



Cross-Validation Metrics Summary:

	mean	sd	cv_1_valid	cv_2_valid	cv_3_valid	cv_4_valid	cv_5_valid
accuracy	0.742319	0.0655216	0.714286	0.685714	0.814286	0.685714	0.811594
aic	nan	0	nan	nan	nan	nan	nan
auc	nan	0	nan	nan	nan	nan	nan
err	0.257681	0.0655216	0.285714	0.314286	0.185714	0.314286	0.188406
err_count	18	4.63681	20	22	13	22	13
loglikelihood	nan	0	nan	nan	nan	nan	nan
logloss	1.27851	0.425295	1.4418	1.7074	0.793281	1.59363	0.85644
max_per_class_error	0.977778	0.0496904	1	0.888889	1	1	1
mean_per_class_accuracy	0.32779	0.0457044	0.277778	0.368012	0.301587	0.307692	0.38388
mean_per_class_error	0.67221	0.0457044	0.722222	0.631988	0.698413	0.692308	0.61612
mse	0.229593	0.0623024	0.261375	0.274939	0.167319	0.287249	0.157083
pr_auc	nan	0	nan	nan	nan	nan	nan
r2	-0.430632	0.329048	-0.840137	-0.133053	-0.704494	-0.120637	-0.354838
rmse	0.475387	0.0670852	0.511248	0.524346	0.409046	0.535956	0.396337

Scoring History:

	timestamp	duration	training_speed	epochs	iterations	samples	training_rmse	training_logloss	training_r2	training_classification_error	training_auc	training_pr_auc
	2024-08-25 04:59:27	0.000 sec		0	0	0	nan	nan	nan	nan	nan	nan
	2024-08-25 04:59:27	5.065 sec	31000 obs/sec	0.799427	1	279	0.662159	2.18765	-1.55229	0.495702	nan	nan
	2024-08-25 04:59:27	5.203 sec	19845 obs/sec	8.0745	10	2818	0.325986	0.406632	0.38141	0.131805	nan	nan

Variable Importances:

variable	relative_importance	scaled_importance	percentage
PC57	1.0	1.0	0.021601813001359334
PC30	0.935310602188110 4	0.935310602188110 4	0.020204404726656352
PC7	0.902515053749084 5	0.902515053749084 5	0.019495961421999494
PC39	0.871537148952484 1	0.871537148952484 1	0.018826782515409 42
PC51	0.856427371501922 6	0.856427371501922 6	0.018500383928430234
PC33	0.839112102985382 1	0.839112102985382 1	0.018126342735867 6
PC18	0.831825911998748 8	0.831825911998748 8	0.017968947800682 16
PC16	0.823415815830230 7	0.823415815830230 7	0.017787274475926 38
PC27	0.821676671504974 4	0.821676671504974 4	0.017749705805429818
PC14	0.817875921726226 8	0.817875921726226 8	0.017667602719444356
---	---	---	---
PC23	0.622879385948181 2	0.622879385948181 2	0.013455324017654138
PC46	0.622302532196044 9	0.622302532196044 9	0.013442862930771 36
PC54	0.611677050590515 1	0.611677050590515 1	0.013213333264079322
PC61	0.604869365692138 7	0.604869365692138 7	0.013066274927932416
PC48	0.590036392211914 1	0.590036392211914 1	0.012745855808558481
PC40	0.562258958816528 3	0.562258958816528 3	0.012145812886693644
PC32	0.542861342430114 7	0.542861342430114 7	0.011726789204842235
PC55	0.534363627433776 9	0.534363627433776 9	0.011543223154552496
PC12	0.531277239322662 4	0.531277239322662 4	0.011476551575726582
PC15	0.455990076065063 5	0.455990076065063 5	0.009850212353633 12

[65 rows x 4 columns]

KUBIG
DATA SCIENCE & AI

04. 개선점

04. 개선점

<프로젝트 의의>

- 비슷한 특성을 가진 제품 LINE별로 묶어 데이터 분석을 시도했던 점
- H2O AUTO ML을 사용하여 LINE별 가장 최적화된 모델을 도출해낸 점

<프로젝트의 한계점>

- LINE1, LINE2를 통합한 모델을 만드려고 하였으나 각각의 모델만 구축하는 선에서 그친 점
- H2O 이외의 AUTO ML 사용을 시도하였으나 실패한 점

KUBIG
DATA SCIENCE & AI

Thank You