후판 공장의 가열 및 압연 공정 중 스케일 생성 영향인자 분석 및 발생량 예측

A반 2조 강지훈

과제 개요

1. 과제 정의: 후판 공정과 스케일 불량

- 주요 내용, 가설 목록, 목표 변수 설명

2. 데이터 전처리

- Feature Engineering
- 이상치 / 결측치 처리

3. EDA로 유의인자 찾기

- TEMP 관련 변수
- DESCALING 관련 변수
- 제품(PT) 관련 변수
- 기타 변수: 작업조, 작업순번, ROLLING_DATE

4. Logistic Regression

- 모델 구현, 회귀 계수, 성능 평가

5. Decision Tree

- 모델 구현, 변수 중요도, 성능 평가

6. Gradient Boosting

- 모델 구현, 변수 중요도, 성능 평가

7. Other Thoughts

- PCA를 이용해 TEMP / PT 변수 축소
- PCA + Gradient Boosting
- 시계열 분석 아이디어

8. Vital Few 도출 및 결론

- 유의 인자 도출, 모델 종합 평가

9. 피드백 / 개선 방향

1. 과제 정의: 후판 공정과 스케일 불량

후판의 정의

후판은 두꺼운 철판을 의미 ➔ 교량, 차량, 구조물, 압력용기, 선박제조 등에 쓰인다.

후판 공정에서 가열과 압연

제강 공정과 연주 공정을 지나 만들어지는 중간재인 slab는 압연 공정까지 이동 도중 온도가 약 740~800도까지 떨어지게 된다.
압연의 적정 온도는 일반적으로 1000~1100도 사이이므로, **적정 온도를 맞춰주기 위해 '가열 공정'이 필요**하다.

가열로 구성

가열로의 종류는 **예열대, 가열대, 균열대**가 있다. 가열대에서 온도를 높여주고, 균열대에서는 slab 내부까지 열이 고르게 퍼지도록 해준다.

스케일 불량이란

스케일은 가열 처리나 압연 처리 중 slab나 billet의 표면에 붙어 있는 산화철(scale)을 의미한다.

- **가열 중에 생기는 스케일을 1차 스케일, 압연 중에 생기는 스케일을 2차 스케일**이라고 한다.
- 스케일은 높은 온도에서 많이 발생하며, 강종마다 스케일 발생 정도가 다르다.

스케일 제거 기법

HSB(Hydraulic Scale Breaker)

- 가열 처리 중 발생하는 스케일을 제거하기 위해, 가열로에서 나온 반제품 표면에 고압수를 뿌려 스케일을 떼어내는 작업이다.

ROLLING DESCALING:

- 조압연기(roughing mill), 사상압연기(finishing mill) 등을 거치는 압연 공정 중에도 수시로 디스케일링 과정을 거친다.

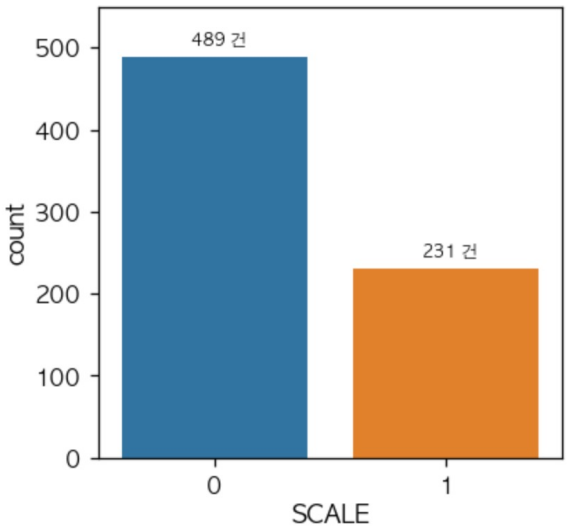
가설 목록

- 가열 온도(FUR_HZ_TEMP)와 압연 온도(ROLLING_TEMP_T5) 높으면 스케일 발생
- 디스케일링 기법인 HSB 적용 안하면 스케일 발생
- 압연 중 디스케일링(ROLLING_DESCALING) 횟수가 적으면 스케일 발생
- 강종마다 스케일 발생률 차이남

1. 과제 정의: 목표 변수 탐색 및 변수 설명

목표 변수 탐색

- 목표 변수는 스케일 불량 여부로 전체 관측치 720건 중 489건(68%)이 양품이고, 231건(32%)이 불량품이다.
- 데이터셋의 설명변수는 총 20개이다.
- 프로젝트의 목표는 이진 분류 모형을 구현하여 목표변수를 높은 성능으로 예측하는 것 + 목표 변수에 대한 유의 변수를 도출하는 것이다.



제품 관련 변수

- STEEL_KIND: 강종
- SPEC: 규격
- PT_THK, PT_WDTH, PT_LTH, PT_WGT: 제품의 두께, 너비, 길이, 무게

온도 관련 변수

- FUR_HZ_TEMP, FUR_HZ_TIME: 가열대 온도 / 가열대 시간
- FUR_SZ_TEMP, FUR_SZ_TIME: 균열대 온도 / 균열대 시간
- FUR_TIME, FUR_EXTEMP: 가열로 시간 / 가열로 추출온도
- ROLLING_TEMP_T5: 압연 온도

디스케일링 관련 변수

- HSB: HSB 적용 여부
- ROLLING_DESCALING: 압연 중 디스케일링 횟수

기타

- PLATE_NO: 제품의 id 값
- FUR_NO_ROW, WORK_GR: 작업순번, 작업조
- FUR_NO: 가열로 호기
- ROLLING_DATE: 압연 일시

Feature Engineering

- **PLATE_NO**는 id값이므로 제외
- **SCALE** 변수 정수 라벨링 ➔ 양품: 0, 불량: 1
- **HSB** 변수 정수 라벨링 ➔ 미적용: 0, 적용: 1

- **STEEL_KIND**(강종) 알파벳 첫 글자로 라벨링 ➔ 'T' & 'C'
- **SPEC**(규격) 라벨링
  - 규격은 변수가 굉장히 다양했는데, 모든 변수가 하이픈(-) 혹은 슬래시(/)를 포함하고 있었다.
    예) AB/EH32-TM
  - 그래서 슬래시나 하이픈의 앞에 있는 알파벳만 추출해서 라벨링했다.
    예) AB/EH32-TM ➔ AB
  - 그리고 관측치가 20개 이하인 규격 총 7개 (A283, V42JBN3, A516, API, A709, A131, CCS)는 'UNCOMMON'으로 라벨링했다.
- **ROLLING_DATE**
  - 압연일시(연월일시분초)를 datetime format으로 변경하고, 포맷을 '연-월-일:시'로 변경했다.

이상치 / 결측치 처리

- 결측치(NaN)는 하나도 없었다.

	ROLLING_TEMP_T5
count	720.000
mean	933.921
std	107.864
min	0.000
25%	889.750
50%	951.000
75%	994.250
max	1078.000

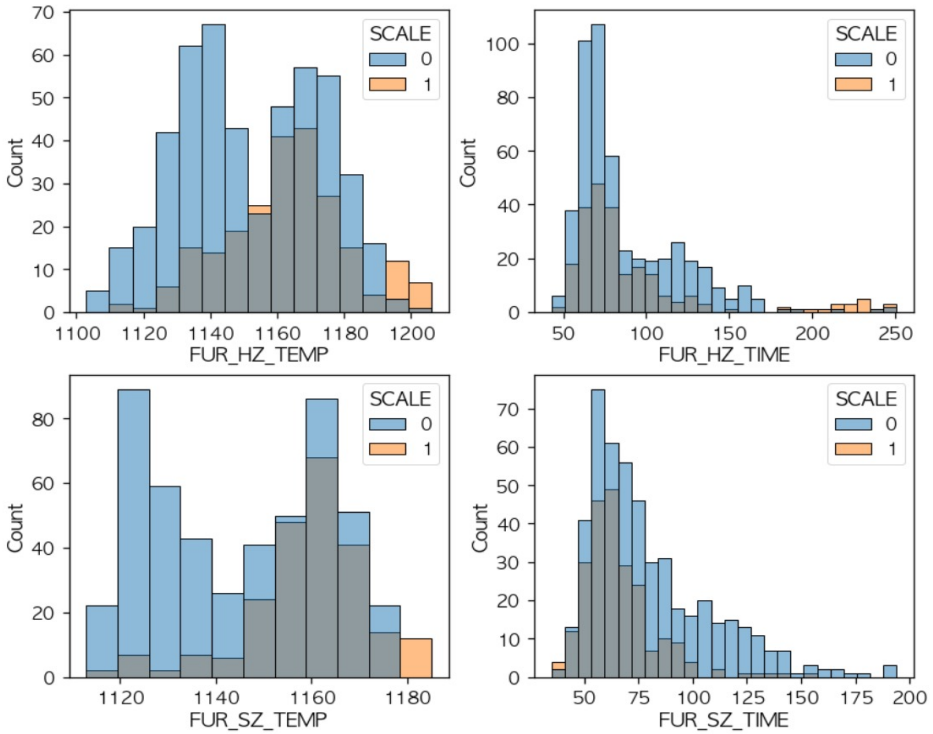
- 다만 **압연온도**에서 이상치가 보였는데, **최솟값이 0인 관측치가 6개** 있었다.
  - 이상치를 채우기 위해 **ROLLING_DATE 변수를 활용**했다. ➔ 1시간 내에 압연온도에 급격한 변화가 없을 것이라 생각해서, **관측치가 해당되는 1시간의 평균값**으로 이상치를 대체했다.

```
# Rolling_Temp 이상치 채우기
idx_outlier = list(df_edited['ROLLING_TEMP_T5']
[df_edited['ROLLING_TEMP_T5']==0].index)

for i in idx_outlier:
    rolling_temp_mean = df_edited[df_edited['ROLLING_DATE'] ==
df_edited['ROLLING_DATE'].iloc[i]].mean()['ROLLING_TEMP_T5']

    df_edited['ROLLING_TEMP_T5'].iloc[i] = rolling_temp_mean
```

3. EDA로 유의인자 찾기: TEMP 관련 변수

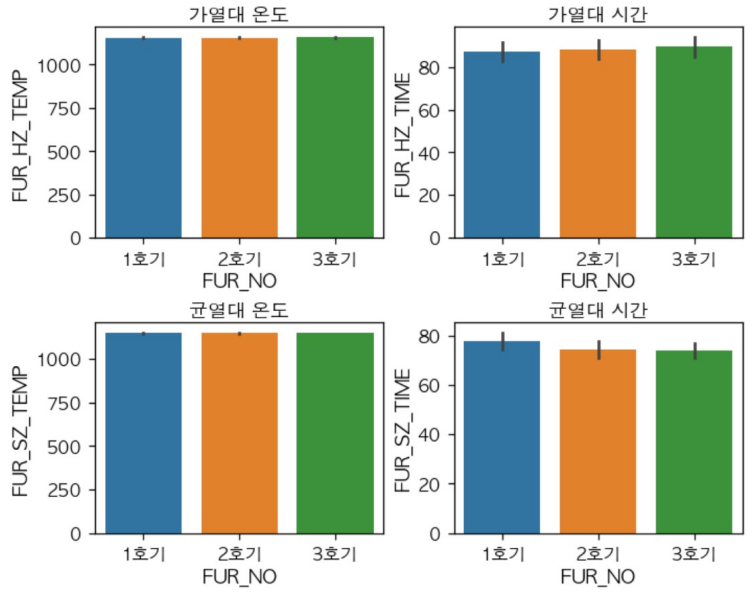
가열대 온도/시간 & 균열대 온도/시간

- 아래 네 가지 그래프가 일관적으로 말하는 건 **"온도가 높으면 불량률이 높다"**이다.
  - 가열대(HZ) 온도와 균열대(SZ) 온도는 높을수록 불량률이 높고, 가열대와 균열대에 머무는 시간이 적을수록(즉, 온도가 높을수록) 불량률이 높다.



가열로 호기별 온도 / 불량률 차이

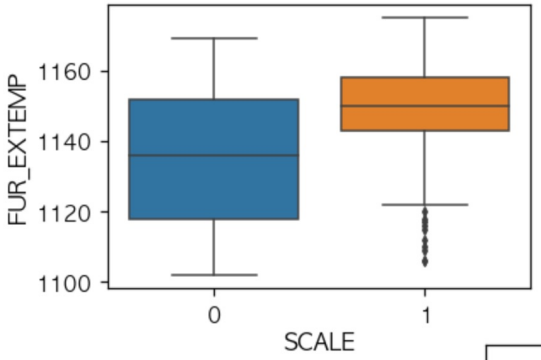
- **가열로 호기별 가열대/균열대 온도 및 시간**에는 차이가 없었다. ➔ 가열대온도, 균열대온도, 추출온도를 가열로 호기별로 집단을 나누고 **ANOVA 분석**을 했는데 모두 p-value가 0.05 이상이었다.
- **가열로 호기별 불량률**에 차이가 있는지 **카이제곱 동질성 검정**을 수행한 결과 p-value는 0.23으로 집단 간 불량률에 차이가 없다고 할 수 있다.

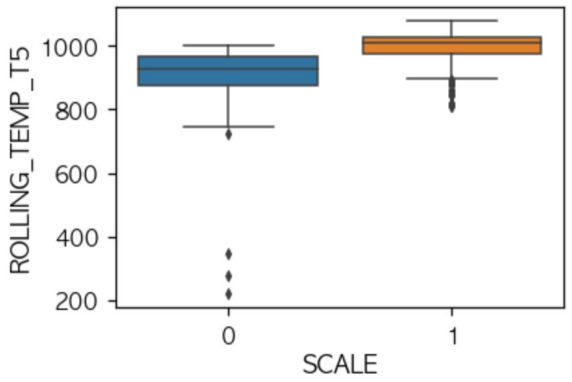


3. EDA로 유의인자 찾기: TEMP / DESCALING 관련 변수

FUR_EXTEMP (가열로 추출온도) & ROLLING_TEMP_T5 (압연 온도)

- 가열로에서 추출했을 때 중간재의 온도와 압연 시 중간재의 온도 모두 불량률과 관련이 있어 보인다.
  - 여기서도 **"온도가 높을 수록 불량률이 높은 것 같다"**는 가설이 지지된다.





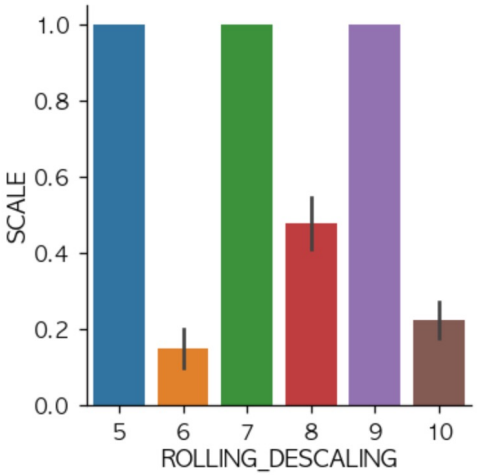
HSB

- 가열로 추출 시 수행하는 디스케일링 공정인 **HSB가 적용되지 않은 관측치는 33개 모두 불량**이었다.
  - **HSB 적용**한 687개의 관측치 중 198개가 불량이었다 (**29%**).

ROLLING_DESCALING

- 압연 중 디스케일링 횟수(ROLLING_DESCALING)은 조금 특이한 모습을 보였다.
  - 디스케일링을 **5번, 7번, 9번 하면 100% 불량이었고, 6번 했을 때 불량률이 제일 낮았다.**

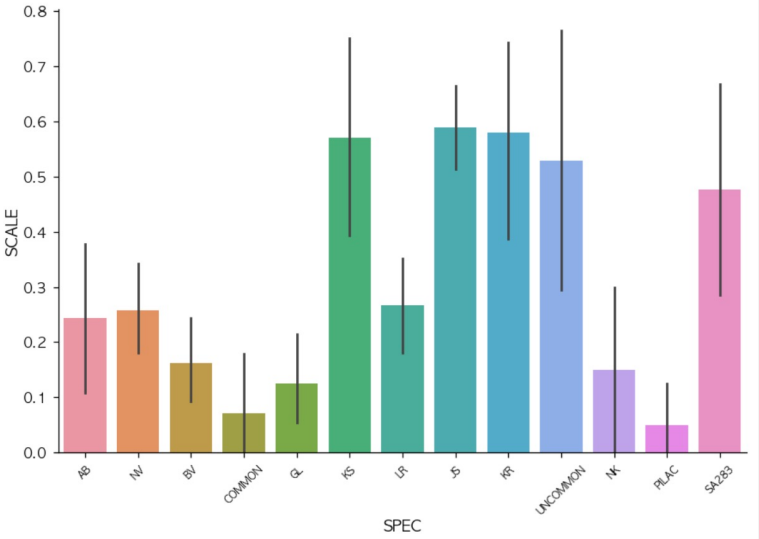
SCALE	0	1
HSB		
0	0	33
1	489	198



3. EDA로 유의인자 찾기: 제품 관련 변수

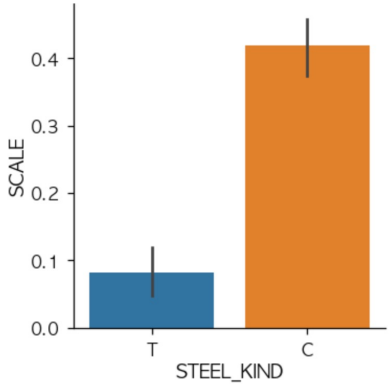
SPEC (규격)

- **규격 간에 불량률 차이가 두드러졌다**. 데이터 수가 많지 않다보니 규격마다 표본 개수가 많지 않아 오차선(막대 위 검은 선)이 긴 경우도 있지만, **오차를 고려해도 규격 간 불량률 차이가 분명히 존재**한다.
- 불량률이 가장 낮은 규격은 **COMMON과 PILAC**이고, 가장 불량률이 높은 규격은 **KS, JS, KR, UNCOMMON, SA283** 등이다.



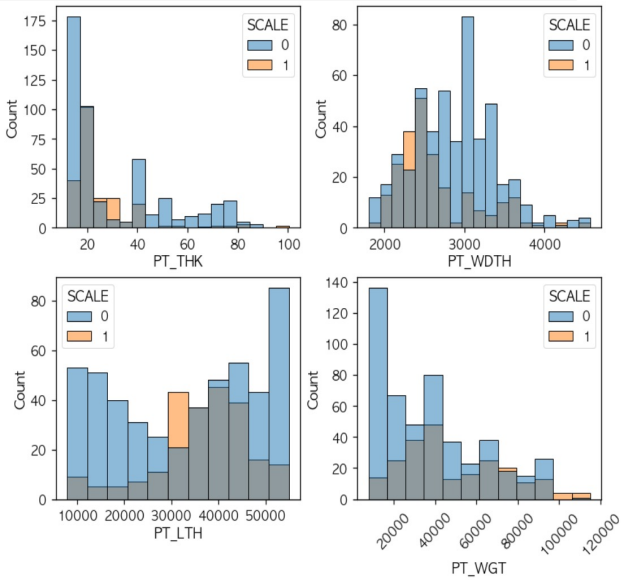
STEEL_KIND (강종)

- **강종 T와 C 사이에 불량률 차이가 명확**해 보인다.



PT_X

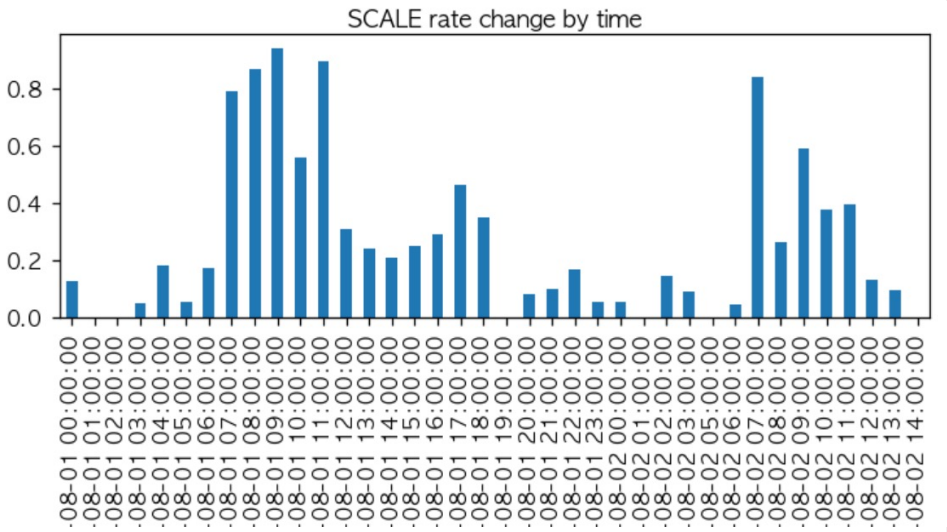
- **두께(THK)가 40 이하인 제품에서만 불량** 발생했다.
- **WDTH가 3000 이하일 때 불량**이 자주 발생.
- **LTH가 40000일 때 불량률 높음**
- **WGT가 낮은 경우 양품 비율이 높고**, WGT가 올라갈 수록 불량률 높아짐



3. EDA로 유의인자 찾기: 기타

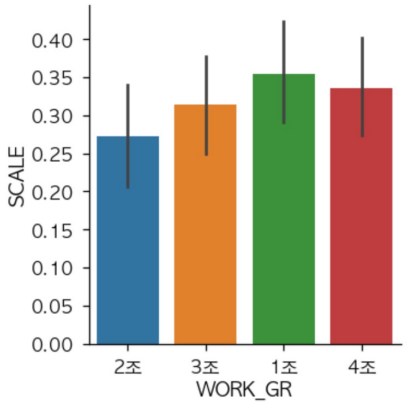
ROLLING_DATE

- 불량률을 시계열로 나열하면 특이한 패턴이 보인다.
  - **"불량률이 오전 7시~11시 사이에 높다"**
- 불량률이 진짜 시간이랑 관련이 있을까? 아니면 특정 시간대에 공정 상에 무언가 변화가 있는 걸까?



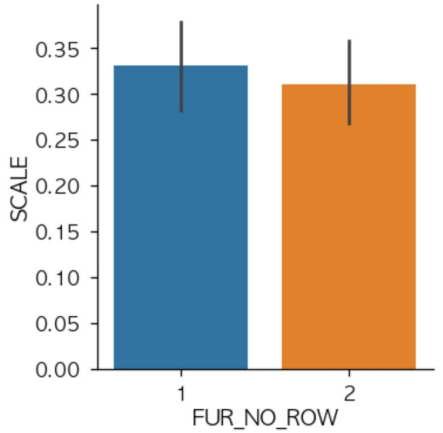
WORK_GR (작업조)

- 작업조 간 불량률 차이가 있는지 알기 위해 **카이제곱 동질성 검정**을 실시했다.
- 결과는 **p-value가 0.4**로 집단 간 불량률 차이가 없다.



FUR_NO_ROW (작업 순번)

- 작업순번 간 불량률 차이가 있는지 알기 위해 **카이제곱 동질성 검정**을 실시했고, **p-value는 0.6**이었다.
  순번 간 불량률 차이는 없었다.



3. EDA로 유의인자 찾기: EDA 결론

EDA를 통해 도출한 Vital Few

- **FUR_EXTEMP(추출온도), ROLLING_TEMP_T5(압연온도)**, **가열대/균열대 온도와 시간** 등 TEMP에 관련된 변수들이 불량률과 관련이 큰 것 같다.

- 가열로에서 추출된 후 적용하는 **디스케일링 기법인 HSB 적용 유무**와 **압연 중 디스케일링 횟수인 ROLLING_DESCALING**도 불량률과 상관성을 보인다.

- **STEEL_KIND(강종)과 SPEC(규격)**도 그래프 상 불량률과 분명한 상관성을 보인다.

EDA를 통해 도출한 Trivial Many

- **FUR_NO(가열로 호기)**는 ANOVA 검정 결과 호기별 TEMP 관련 값에 차이도 없고, 카이제곱 검정 결과 불량률과의 관계도 없었다.

- **FUR_NO_ROW(작업순번)과 WORK_GR(작업조)**도 카이제곱검정 결과 불량률과 집단 간의 동질성이 있다는 귀무가설이 채택됐다.

기타

- **ROLLING_DATE(압연일시)**의 경우 직접 모델링에 변수로 넣지 않고, 데이터셋을 탐색하거나 분할하거나 비교할 때 사용해보기로 했다.

4. Logistic Regression

모델 구현

- 데이터셋에서 **FUR_NO, FUR_NO_ROW, WORK_GR, ROLLING_DATE 를 drop**하고; **SPEC과 STEEL_KIND를 더미변수화**했다.

- Test size=0.3으로 스플릿하고, Logistic Regression 모델을 구현했다.

```
df_selected = df_edited.drop(['FUR_NO', 'FUR_NO_ROW',
'WORK_GR', 'ROLLING_DATE'], axis=1)

df_selected = pd.get_dummies(df_selected, columns=['SPEC',
'STEEL_KIND'])

X_lr = df_selected.drop('SCALE', axis=1)
y_lr = df_selected['SCALE']

X_lr_train, X_lr_test, y_lr_train, y_lr_test = \
        train_test_split(X_lr, y_lr, test_size=0.3,
stratify=y_lr, random_state=0)
```

executed in 26ms, finished 02:05:32 2021-09-01

```
lr = LogisticRegression(random_state=0)
lr = lr.fit(X_lr_train, y_lr_train)
```

executed in 37ms, finished 02:05:33 2021-09-01

회귀 계수

- 회귀 계수에 exp를 씌워서 beta를 odds ratio로 만들어줬다.
- 가장 **계수가 높은 변수는 "압연온도"**로, 압연온도가 **1단위 증가할 때마다 스케일 불량일 확률이 약 3% 증가**한다.
- 압연온도는 섭씨 1도씩 변동하는 변수가 아니라, 평균 938에 표준편차는 78인 변수이기 때문에.. **3%는 꽤 큰 숫자**다.
  예) 압연 온도가 78만큼 증가하면 불량률 2.34배 증가

성능 평가

- Class imbalance 문제가 있는 동시에 positive class가 '불량'이나 '질병'처럼 예측하지 못할 때 cost가 큰 상황이면 **"재현율"**이 중요한 지표다.
- 로지스틱 모형에서는 재현율이 성능지표 중 꽤 낮은 편에 속한다.

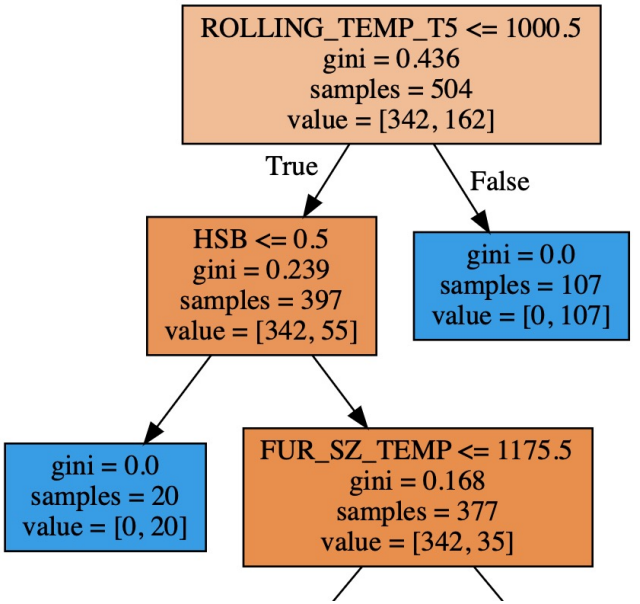
	LR feat	LR exp(coef)
0	ROLLING_TEMP_T5	1.027
1	FUR_HZ_TIME	1.005
2	STEEL_KIND_C	1.000
3	SPEC_NV	1.000
4	SPEC_KR	1.000
5	FUR_HZ_TEMP	0.996
6	FUR_TIME	0.995
7	FUR_SZ_TIME	0.993
8	FUR_EXTEMP	0.993
9	FUR_SZ_TEMP	0.992

	Logistic Regression
Train Acc	0.833
Test Acc	0.856
AUC	0.818
Precision	0.817
Recall	0.710
F1 Score	0.760

모델 구현

- 로지스틱 회귀와 데이터셋과 같은 train/test set으로 Decision Tree를 구현했다.
  하이퍼파라미터로는 max_depth=6만 설정했다.

- Graphviz를 보면 **첫 번째 node의 split 기준이 "압연온도" 1000.5도**이다.
  - 1000.5도가 넘으면 **162개의 불량 중 107개가 검출**된다.
  - 실제로 데이터를 살펴보면.. **압연온도가 1000.0도 이상인 152개 데이터 모두가 스케일 불량**이다. 총 불량 데이터가 231개니까, 약 65%가 압연온도만으로 검출되는 셈이다.



성능평가

- 로지스틱 회귀 모형보다 **전반적으로 높은 성능**을 보여준다.
- 다만 다른 지표에 비해 **재현율은 낮다**.

	Decision Tree
Train Acc	0.978
Test Acc	0.958
AUC	0.939
Precision	0.984
Recall	0.884
F1 Score	0.931

변수중요도

- 두 번째 변수 중요도는 **HSB**다.
- 전체 720개 데이터 중 **HSB 미적용 데이터는 총 33건인데, 모두 다 불량**이다.

	DT feat	DT imp
0	ROLLING_TEMP_T5	0.630073
1	HSB	0.157444
2	FUR_SZ_TEMP	0.138516
3	ROLLING_DESCALING	0.045840
4	FUR_TIME	0.018698
5	FUR_SZ_TIME	0.009429
6	SPEC_KS	0.000000

6. Gradient Boosting

모델 구현

- 로지스틱 회귀와 데이터셋과 같은 train/test set으로 Decision Tree를 구현했다.

- 하이퍼파라미터 튜닝을 위해서 GridSearchCV를 사용했다.

```
gb = GradientBoostingClassifier(random_state=0)

params = {'max_depth': [i for i in range(1, 10)],
          'n_estimators': [i for i in range(100,600,100)],
          'min_samples_split': [i * 2 for i in range(2, 15,
2)]}
gb_grid = GridSearchCV(gb, param_grid=params, cv=3,
refit=True)

gb_grid.fit(X_gb_train, y_gb_train)

gb = gb_grid.best_estimator_
```

변수 중요도

- 마찬가지로 **"압연온도"가 가장 높은 변수 중요도**를 기록했다.

- 그 다음이 **HSB, FUR_SZ_TEMP로, Decision Tree와 같은 순서**다.

	GB feat	GB imp
0	ROLLING_TEMP_T5	0.571326
1	HSB	0.142413
2	FUR_SZ_TEMP	0.125546
3	ROLLING_DESCALING	0.095960
4	PT_THK	0.020607
5	FUR_TIME	0.016763
6	FUR_SZ_TIME	0.008086
7	SPEC_KR	0.007616
8	FUR_HZ_TIME	0.003691
9	PT_WDTH	0.002429

성능 평가

- 성능의 경우 GB가 로지스틱 회귀나 Decision Tree와 비교해서는 **월등한 모습**을 보여줬다.
- 다만 로지스틱 회귀나 Decision Tree에 비해서 **해석력(interpretability)이 다소 떨어지는 블랙박스 모델이라는 단점**이 있다.

	Gradient Boosting
Train Acc	1.000
Test Acc	0.986
AUC	0.978
Precision	1.000
Recall	0.957
F1 Score	0.978

7. Other Thoughts: PCA

PCA 모델링

- 데이터셋에 중요한 범주형 변수들(dummy)이 있기 때문에, **모든 설명변수 대상으로 PCA를 진행하지 않았다.**
- PCA를 활용할 만하다고 생각한 부분은:
  - **가열대온도/시간, 균열대온도/시간, 가열로시간/추출온도** 라는 6가지 변수들을 **주성분으로 축소** &
  - **PT_WDTH, PT_LTH, PT_WGT, PT_THK** 라는 4가지 변수들도 **주성분으로 축소**하는 것이다.
  - 위의 변수들은 서로 **성격도 비슷하고 상관관계도 높기 때문에**, 만약 변수를 축소할 수만 있다면 **모델의 효율성**을 더욱 높일 수 있을 것이라 생각했다.

Explained Variance Ratio

- TEMP 데이터와 PT 데이터를 각각 주성분으로 축소해본 결과, explained variance ratio는 다음과 같았다.

```
TEMP's explained variance ratio: [0.58  0.212 0.117 0.07
1 0.019 0.001]

PT's explained variance ratio: [0.825 0.175 0.    0.   ]
```

- 따라서 **TEMP는 주성분 4개(90% 확보), PT는 주성분 2개**로 결정했다.

성능 평가

- Decision Tree는 PCA로 변수를 축소한 뒤 평가했을 때 미세하게 성능이 감소했는데,
- Gradient Boosting은 성능이 증가했다.
- 재현율도 0.971까지 높아졌다.

	GB + PCA
Train Acc	1.000
Test Acc	0.991
AUC	0.986
Precision	1.000
Recall	0.971
F1 Score	0.985

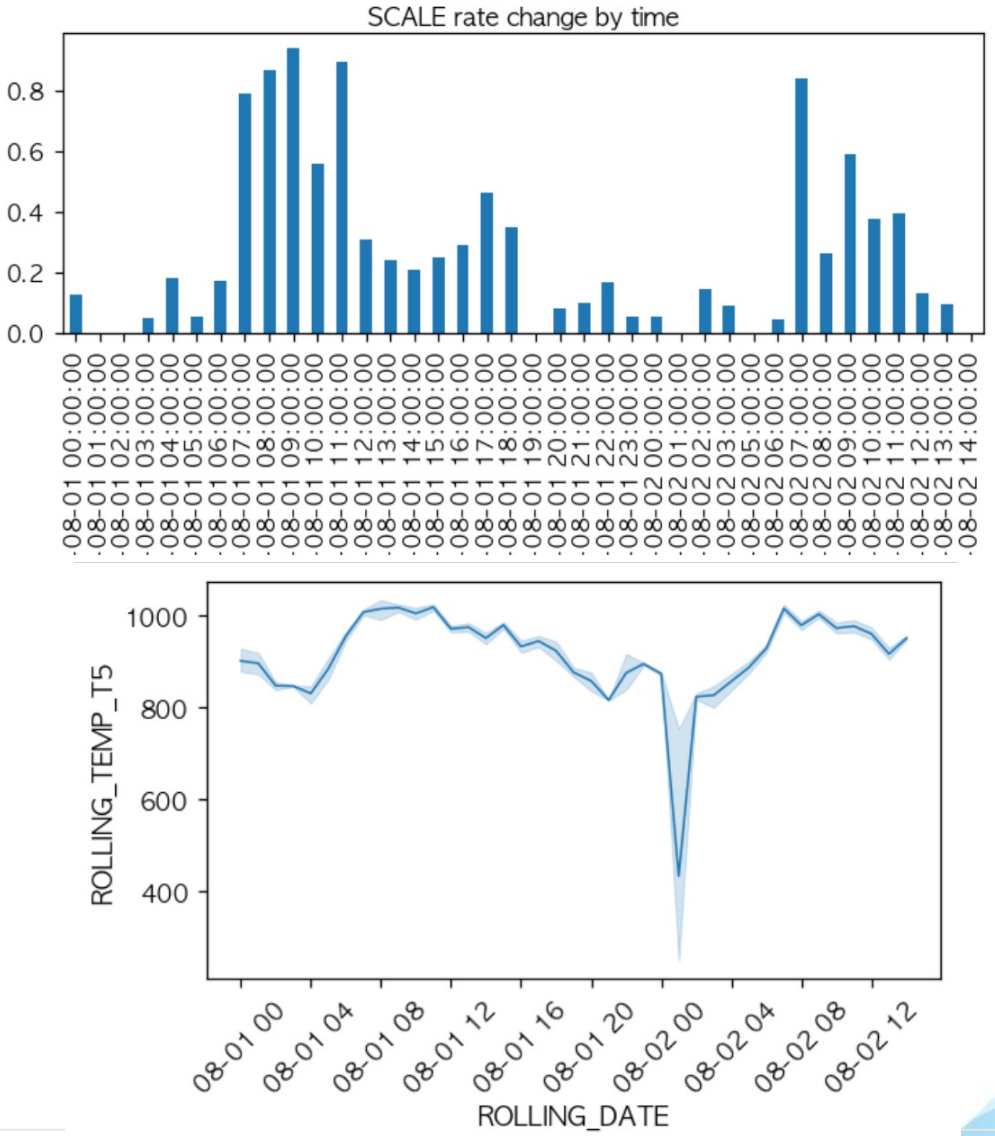
7. Other Thoughts: ROLLING_DATE

ROLLING_DATE와 불량률의 관계

- 앞선 EDA에서 살펴봤듯이, **불량률이 오전 7시~ 오전 11시 사이에 높은 패턴**을 보였다.

- 모든 모델에서 ROLLING_TEMP_T5(압연온도)가 가장 유의한 변수로 채택됐기 때문에, 시간대별 압연온도의 변화를 살펴봤는데...
    - **오전 7~11시에 압연온도가 높아지는 듯한 패턴**을 보였다.

- "오전에 불량률이 높다"는 가설을 세운다기 보다는, **"왜 오전에 압연온도가 높을까?"**가 궁금하다.
- 오전에 주로 생산하는 강종이나 규격이 높은 압연온도를 필요로 해서 압연온도를 조정한 것일까? 아니면 오전 시간대에 생산량이 많아져서 공정 설비가 과열되나? 이 부분에 대한 도메인 지식이 없어서 잘 모르겠다



8. Vital Few 도출 및 결론

Vital Few

- 가장 중요한 변수: **압연온도**
- 기타 유의인자: **HSB, FUR_SZ_TEMP**(균열대온도), **ROLLING_DESCALING**(압연 중 디스케일링 횟수) 등
- **GB+PCA 모델에서도 TEMP 변수를 추출한 주성분 4개**가 모두 어느 정도 변수 중요도를 차지하고 있다.

	Logistic Regression	Decision Tree	Gradient Boosting	GB + PCA
Train Acc	0.833	0.978	1.000	1.000
Test Acc	0.856	0.958	0.986	0.991
AUC	0.818	0.939	0.978	0.986
Precision	0.817	0.984	1.000	1.000
Recall	0.710	0.884	0.957	0.971
F1 Score	0.760	0.931	0.978	0.985

결론

- 스케일 **불량을 정확히 예측하지 못했을 때 발생하는 비용**(압연 동력 증가, 불량품 양산)이 크다고 생각해 **"재현율"**을 중요하게 생각했다.

- 그러나 "재현율"을 기준으로 모델을 선정할 필요 없이, **GB+PCA 모델이 모든 성능 지표에서 최고점**을 기록했다.

	LR feat	LR exp(coef)	DT feat	DT imp	GB feat	GB imp	GB+pca feat	GB+pca imp
0	ROLLING_TEMP_T5	1.027	ROLLING_TEMP_T5	0.630	ROLLING_TEMP_T5	0.571	ROLLING_TEMP_T5	0.606
1	FUR_HZ_TIME	1.005	HSB	0.157	HSB	0.142	HSB	0.147
2	STEEL_KIND_C	1.000	FUR_SZ_TEMP	0.139	FUR_SZ_TEMP	0.126	ROLLING_DESCALING	0.115
3	SPEC_NV	1.000	ROLLING_DESCALING	0.046	ROLLING_DESCALING	0.096	temp1	0.063
4	SPEC_KR	1.000	FUR_TIME	0.019	PT_THK	0.021	temp3	0.057
5	FUR_HZ_TEMP	0.996	FUR_SZ_TIME	0.009	FUR_TIME	0.017	temp2	0.005
6	FUR_TIME	0.995	SPEC_KS	0.000	FUR_SZ_TIME	0.008	temp4	0.003
7	FUR_SZ_TIME	0.993	SPEC_LR	0.000	SPEC_KR	0.008	STEEL_KIND_C	0.002
8	FUR_EXTEMP	0.993	SPEC_NK	0.000	FUR_HZ_TIME	0.004	pt1	0.001
9	FUR_SZ_TEMP	0.992	PT_THK	0.000	PT_WDTH	0.002	SPEC_KR	0.001

9. 피드백 / 개선 방향

피드백

- 일반적으로 **PCA를 사용하면 설명가능성(explainability)가 떨어지는 것이 단점**이다. 하지만 이번 프로젝트에서 구현한 PCA는 서로 성격이 비슷한 **"TEMP 관련 변수", "PT 관련 변수"로 묶다보니, PCA의 단점도 어느 정도 상쇄**했다.

- 튜닝하지 않은 모델도 test score가 꽤 높게 나온다고 느꼈는데, 그 이유는 **데이터셋이 자체적으로 '불량(양성) 데이터를 업샘플링한 것이기 때문'**이라고 생각한다.
    - 만약 실제 상황처럼 불량 데이터가 1%, 0.1% 처럼 **희소한 상황에서도 모델이 잘 동작할지** 궁금하다.

개선 방향

- **ROLLING_DATE를 활용한 시계열 분석**을 해보면 더 많은 인사이트를 얻을 수 있을 것 같다.
    - 진짜 압연온도와 ROLLING_DATE는 관련이 있을까?
    - 왜 오전에 불량률이 높았던 걸까?

- **강종(STEEL_KIND)과 규격(SPEC) 변수에 대해 더 높은 이해**를 가지고 다룬다면, 예측력을 더 높일 수 있을 것이다.
    - 강종과 규격 모두 불량률과 굉장히 유의한 관계가 있는 데이터 같은데, 이를 적극적으로 사용하지 못한 것 같다.

감사합니다.