Pusan National University
Computer Science and Engineering
Technical Report 2022-09

기계 이상 진단 시스템 개발

저자 1 201724401 강 금석
저자 2 201724422 김 민수

송 길태

목 차

1. 서론 ····· 1

1.1 연구 배경 ····· 1

1.2 기존 문제점 ····· 1

1.3 연구 목표 ····· 1

2. 연구 배경 ····· 1

2.1 연구 배경 항목-1 ····· 1

2.2 연구 배경 항목-2 ····· 1

3. 연구 내용 ····· 1

3.1 연구 내용 절-1 ····· 1

3.1.1 연구 내용 절-1-1 ····· 1

3.1.2 연구 내용 절-1-2 ····· 1

3.2 연구 내용 절-2 ····· 1

3.3 연구 내용 절-3 ····· 1

3.3.1 연구 내용 절-3-1 ····· 1

3.3.2 연구 내용 절-3-2 ····· 1

4. 연구 결과 분석 및 평가 ····· 1

5. 결론 및 향후 연구 방향 ····· 1

6. 구성원별 역할 및 개발일정 ····· 1

7. 참고 문헌 ····· 1

1. 서론

1.1 연구 배경

최근 4차 산업혁명과 함께 많은 최근 4차 산업혁명과 함께 많은 산업에서 IoT 기술 을 활용해 빅데이터 수집과 고성능 컴퓨팅이 가능해졌다. 이를 활용해 기업에서는 다양한 AI기술을 활용해 기계장치 이상 진단에 관련된 연구를 진행하고 있다. 따라서 우리는 선박 엔진의 데이터를 사용해 선박엔진의 부하를 분석하여 선박엔진의 이상을 진단하는 모델을 개발하려 한다.

1.2 기존 문제점

1.2.1 기존에 엔진 정비를 할 때는 TBM(Time Based Maintenance) 방식으로 정비를 한다. 하지만 엔진에 이상이 없음에도 주기적인 정비를 하는 것은 과보전(Over Maintenance)이 되기 쉽고 따라서 보전비가 커진다.

1.2.2 기존 H사에서 엔진의 부하를 예측할 때, 사용한 Feature로는 소기압력, 연소실압력, 터보차저RPM, 배기가스 온도를 사용했다. 하지만 위의 Feature 로 부하 예측을 했을 때 모델 성능 평가 지표인 RMSE의 값이 높았다.

1.3 연구 목표

기존 H사에서 사용하던 엔진 부하를 예측하는 데 사용된 Feature보다 더 높은 상관관계를 가진 Feature를 찾고, 선택하여 더 높은 성능(최소 RMSE)을 가지는 모델을 개발한다. 이를 토대로 엔진의 이상을 예측, 탐지하고 엔진의 유지보수를 TBM 방식에서 더 효율적인 CBM(Condition Based Maintenance) 방식으로 엔진의 최적 상태를 유지하게 한다.

2. 연구 배경

2.1 데이터 수집

H사에서 엔진 데이터를 얻을 수 있었다. 2022년 4월 6일 8시 30분부터 13시 04분까지 엔진에 부하가 걸리고 꺼질 때까지 총 1,524개의 Feature에 대해 1초 간격으로 총 16,441초 동안 얻을 수 있는 연속적인 시계열 데이터다. Feature에는 Analog 값과 사용자가 직접 입력할 수 있는 Parameter값, Characteristic map값, Discrete값이 있다.

A	B	C	D	E	F
Time stamp	5978	5979	5980	5981	5983
	34207 Spe	34208 Spe	34209 Spe	34210 Spe	34213 Spe
	PARAMET	PARAMET	PARAMET	PARAMET	PARAMET
	Cur34207_	Cur34208_	Cur34209_	Cur34210_	Cur34213_
	Characteri	Characteri	Characteri	Characteri	Characteri
2022-04-06 08:30:01	0.40	1.60 s	0.03 s	1.00	0.40
2022-04-06 08:30:02	0.40	1.60 s	0.03 s	1.00	0.40
2022-04-06 08:30:03	0.40	1.60 s	0.03 s	1.00	0.40
2022-04-06 08:30:04	0.40	1.60 s	0.03 s	1.00	0.40
2022-04-06 08:30:05	0.40	1.60 s	0.03 s	1.00	0.40
2022-04-06 08:30:06	0.40	1.60 s	0.03 s	1.00	0.40
2022-04-06 08:30:07	0.40	1.60 s	0.03 s	1.00	0.40
2022-04-06 08:30:08	0.40	1.60 s	0.03 s	1.00	0.40
2022-04-06 08:30:09	0.40	1.60 s	0.03 s	1.00	0.40

그림 1 모델 학습에 사용된 데이터의 일부

2.2 Anaconda

Anaconda는 가상환경을 만들어서 인공지능 모델 학습을 위해 현재 사용자의 개발환경과 독립적으로 개발환경을 제공해준다. 이러한 점 덕분에 프로젝트에 따라 개별적으로 사용되는 패키지를 다르게 관리할 수 있다.

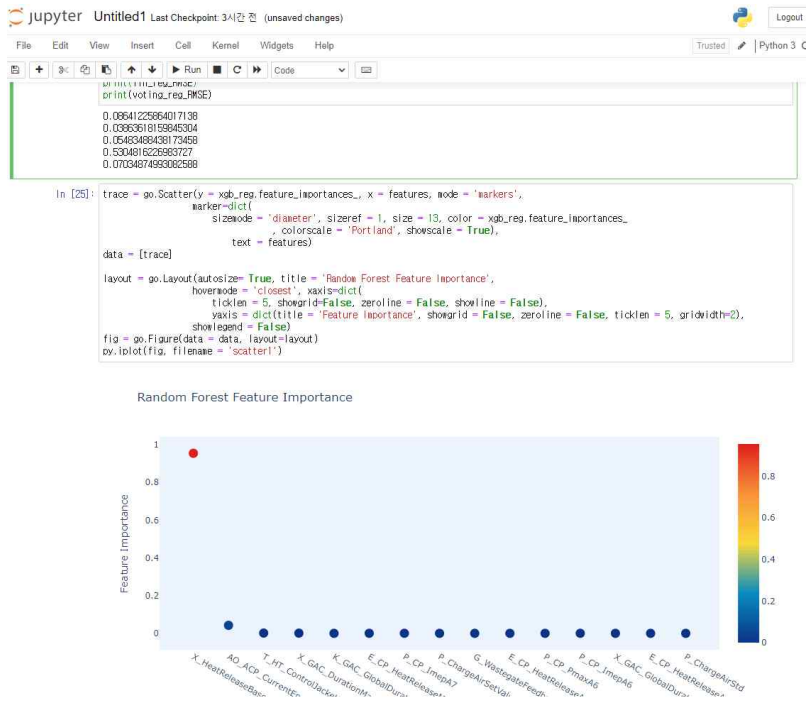


그림 2 실제 아나콘다 개발환경에서 개발하는 모습

3. 연구 내용

3.1 데이터 준비

기존 데이터는 엑셀 파일로 저장돼 있었다. 따라서 처음에 간단한 전처리는 엑셀을 이용했다. 총 1524개의 Feature들 중에서 Discrete값, character map 값, value의 변화가 없는 값, target value와 1대1 매칭되는 모델 학습에 의미가 없는 값을 가지는 Feature는 모델 학습에서 필요한 값들이 아니기 때문에 제외시켰다. 모델 학습에 사용될 데이터를 엑셀 파일로 저장한 것에서 처리속도가 더 빠른 csv 파일로 변환하였다.

그림 3 데이터 준비 단계를 거친 csv 파일들

3.2 Feature selection

데이터 준비단계에서 기존 Raw data의 1524개의 Feature를 418개로 줄였다. 후에 RandomForest 알고리즘과 XGBoost 알고리즘을 사용해 Feature importance를 측정했다.

측정한 후, 두 알고리즘에서 공통적으로 높게 나온 Feature 15개를 선택했다.

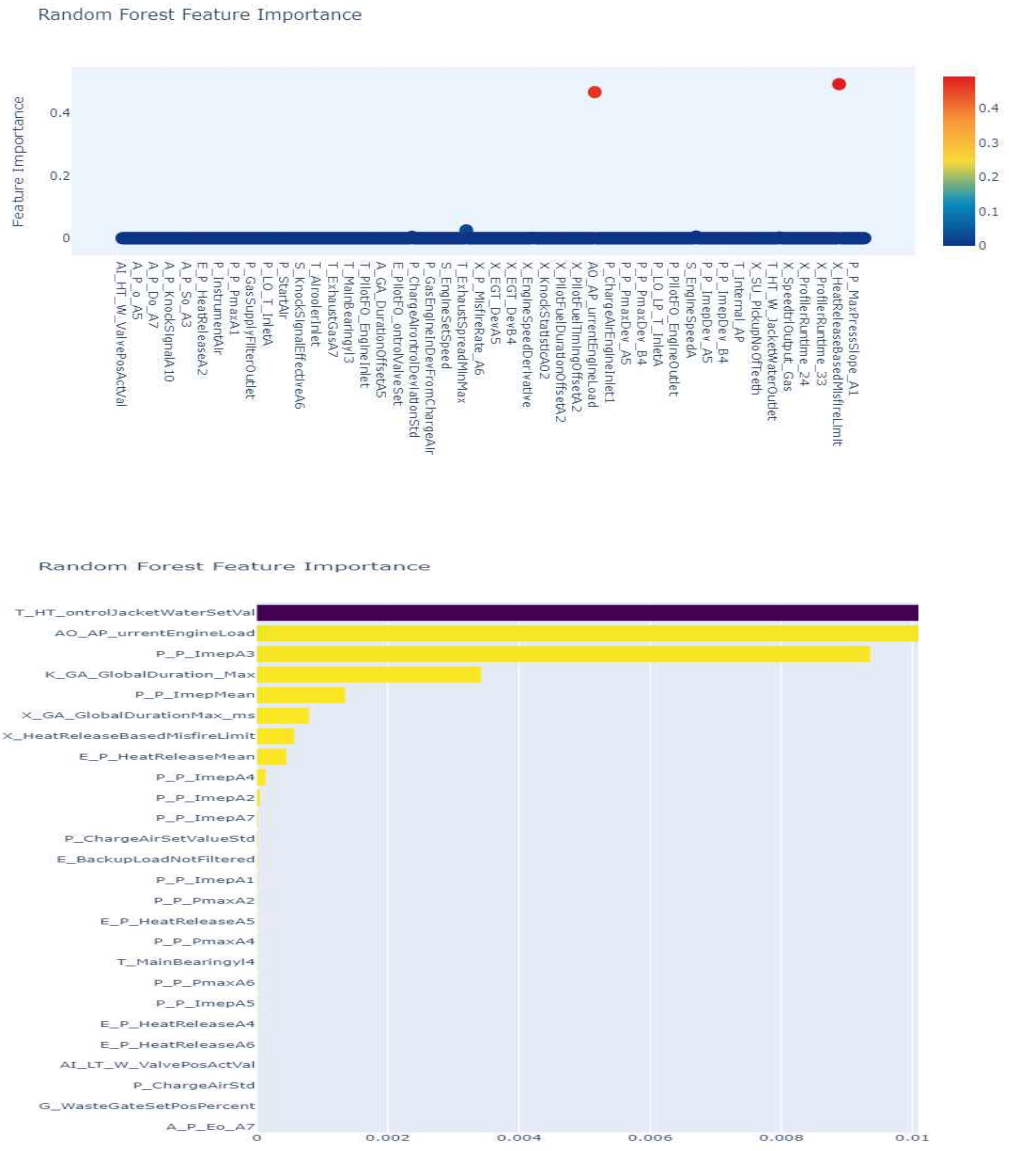
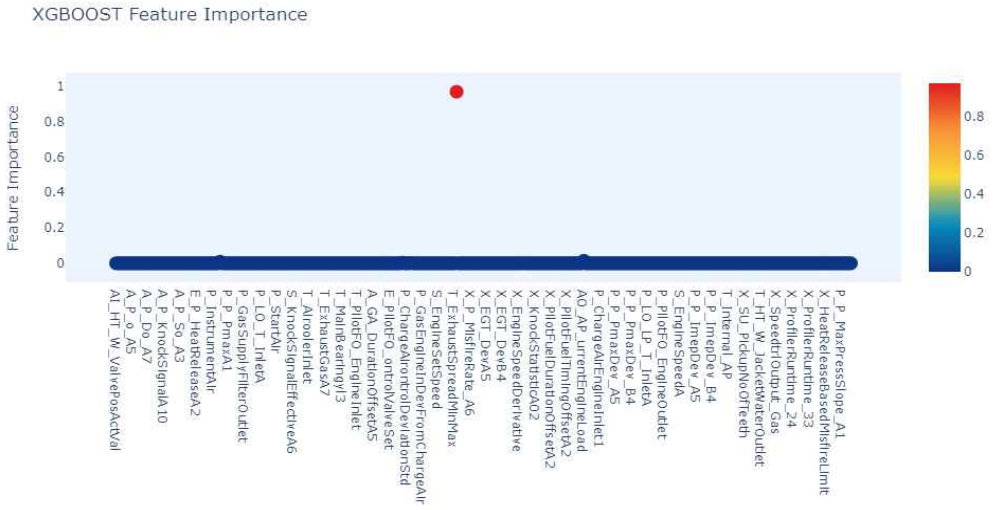


그림 4 Random Forest를 사용해 얻은 Feature Importance의 산점도와 Bar plot



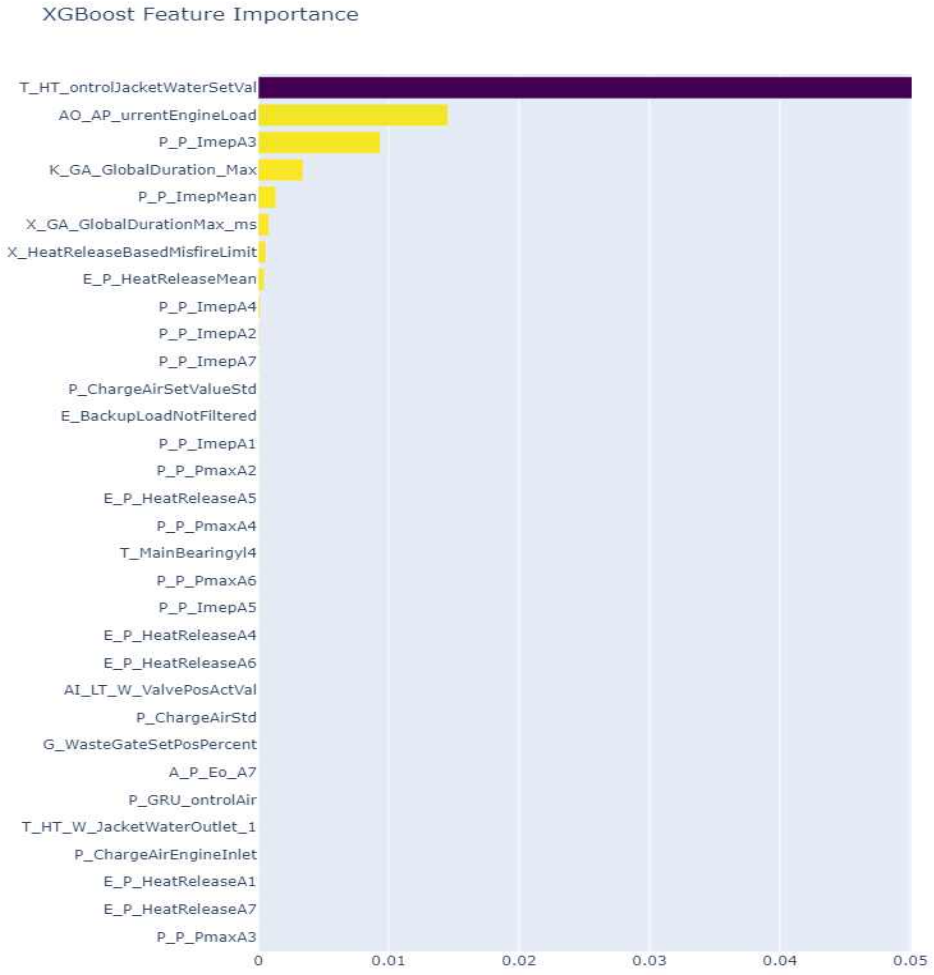


그림 5 Xgboost를 사용해 얻은 Feature Importance의 산점도와 Bar plot

3.3 모델 설계

3.3.1 다양한 인공지능/머신러닝 모델을 적용해본다.

3.3.1.1 Linear Regression

단순 선형 회귀는 식으로 $y = Xw + b$나타난다. 머신러닝에서는 독립 변수 x에 곱해지는 W값을 가중치(weight), 상수항에 해당하는 b를 편향(bias)이라고 부른다.

3.3.1.2 MLP Regression

Nueral Network를 기반으로 여러 개의 퍼셉트론 뉴런을 여러 층으로 쌓은 다층신경망 구조, 즉 단층 퍼셉트론 형태에서 입력층과 출력층 사이에 하나 이상의 은닉층을 가지고 있는 신경망이다. 은닉층에 존재하는 노드는 기본 선형 회귀 모델과 동일하게 $y = Xw + b$로 이루어져 있다. 이러한 선형 분리를 할 수 있는 모델을 여러 개 모아서 비선형 분리를 하는것이 MLP Regression이다.

3.3.1.3 XGBoost Regression

여러 개의 약한 Decision Tree를 조합해서 사용하는 앙상블 기법 중 하나다. 약한 예측모형들의 학습 에러에 가중치를 두고, 순차적으로 다음 학습 모델에 반영해 강한 예측모형을 만드는 것이다. 회귀영역에서 뛰어난 예측 성능을 발휘한다.

3.3.1.4 Random Forest Regression

Random Forest는 다수의 결정 트리들을 학습하는 Ensemble 기법 중 하나이다. 훈련 과정에서 구성한 다수의 Decision Tree로부터 부류(분류) 또는 평균 예측치(회귀 분석)를 출력함으로써 동작한다. Random Forest는 검출, 분류, 그리고 회귀 등 다양한 문제에 활용되고 있다.

3.3.1.5 OLS Regression

최소자승법(Ordinary Least Squares)은 기존 선형 회귀 방식에서 잔차제곱합(Residual Sum of Squares)을 최소화하는 가중치 벡터를

구하는 방법이다. 기존 선형 회귀 $\hat{y}=Xw$에 상수항이 결합된 선형 모형이다. 여기서 잔차 벡터는 $e=y-\hat{y}=y-Xw$이고 잔차 제곱합은 $RSS=e^Te=(y-Xw)^T(y-Xw)=y^Ty-2y^TXw+w^TX^TXw$이다.
OLS는 RSS를 가장 작게 하는 가중치 벡터를 가지므로 위의 식을 미분하여 RSS의 gradient 벡터를 구하면 다음과 같다.

$$\frac{dRSS}{dw} = -2X^Ty+2X^TXw$$

잔차가 최소가 되는 조건은 위 식의 벡터가 0 벡터가 되므로 다음 식이 성립한다.

$$\frac{dRSS}{dw}=0$$

3.3.1.6 Voting Regression

Voting Regression은 앙상블 회귀방법 중 하나로 Voting의 단어 뜻 그래도 여러 가지 회귀 알고리즘을 조합하여 투표를 통해 결정하는 방식이다. Random Forest나 Boosting 알고리즘들은 모두 Decision 트리 알고리즘을 기반으로 모델을 형성하는데 이는 weak learner로써 과적합을 할 수 있는 단점이 존재하지만 앙상블 모델로 보완할 수 있다.

3.3.2 모델 학습

이번 기계 이상진단 모델을 설계할 때 다양한 회귀모델을 사용하기 위해 scikit-learn 라이브러리와 statsmodel.api 라이브러리를 사용하였다.

scikit-learn에서는 앙상블 모델인 Xgboost와 Voting Regression와 Nueral Network를 이용한 Multi Layer Perceptron Regression, Decision Tree 기반 알고리즘인 Random Forest Regression, 일반적인 선형회귀 방식인 Linear Regression 모델을 사용하였다.

statsmodel.api에서는 최소자승법 선형회귀 방식인 OLS Regression 모델을 사용하였다.

이외에도 csv파일을 사용하기 위해 pandas 라이브러리와 RMSE측정을 위

해 numpy 라이브러리, 모델의 성능비교를 시각화 하기 위해 matplotlib와 plotly 라이브러리를 사용하였다.

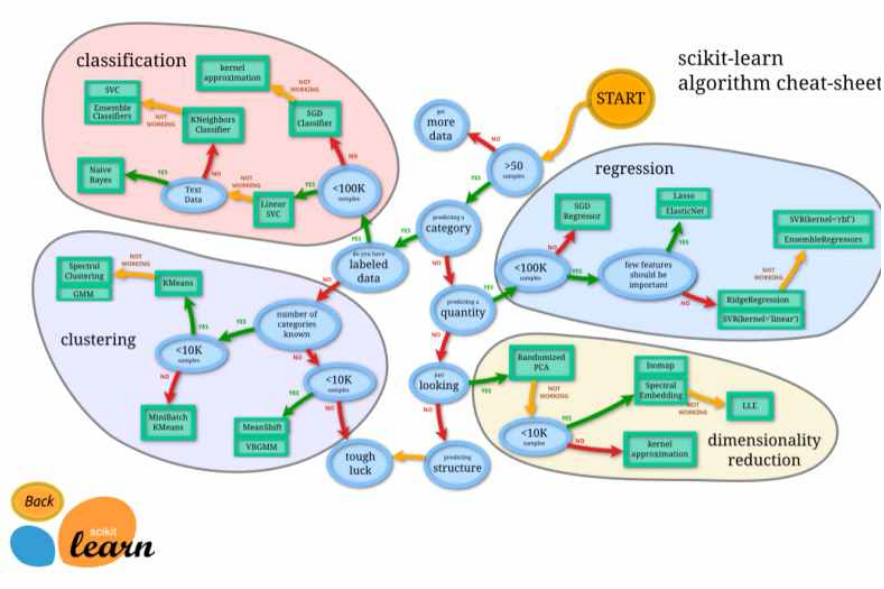


그림 6 scikit-learn algroithm cheat-sheet

```
In [119]: from statsmodels.graphics.tsaplots import plot_acf, plot_pacf
          from statsmodels.tsa.arima_process import ArmaProcess
          from statsmodels.regression.linear_model import yule_walker
          from statsmodels.tsa.stattools import adfuller
          import matplotlib.pyplot as plt
          import pandas as pd
          import numpy as np
          import plotly.graph_objs as go
          import plotly.tools as tls
          import plotly.offline as py
          py.init_notebook_mode(connected = True)

          %matplotlib inline
```

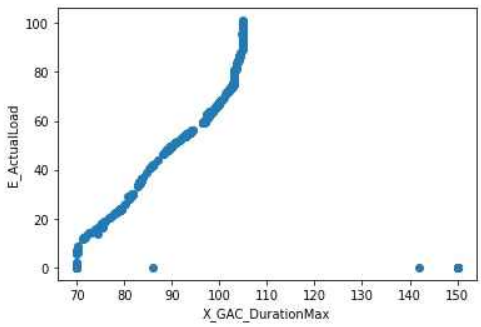
```
In [137]: from sklearn.model_selection import train_test_split
          from sklearn.metrics import mean_squared_error
          from sklearn.metrics import r2_score

          X_train, X_test, Y_train, Y_test = train_test_split(df, target, test_size = 0.1, shuf

          features = X_train.columns.values

          #X_train = np.nan_to_num(X_train)
          #X_test = np.nan_to_num(X_test)
```

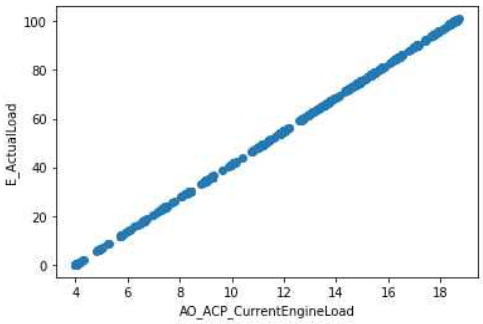
```
In [138]: from sklearn.ensemble import RandomForestRegressor
          from sklearn.ensemble import VotingRegressor
          from sklearn.linear_model import LinearRegression
          from sklearn.neural_network import MLPRegressor
          import statsmodels.api as sm
          import xgboost as xgb
```

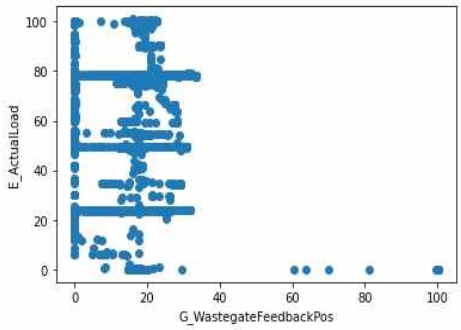
그림 7 모델 학습에 사용된 라이브러리를 import한 모습

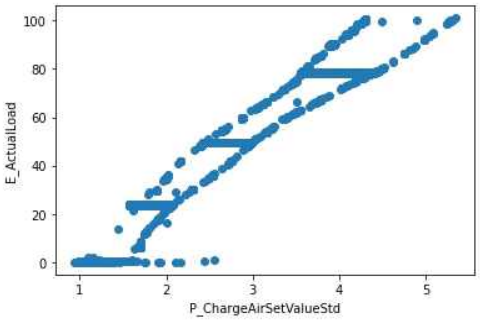
4. 연구 결과 분석 및 평가

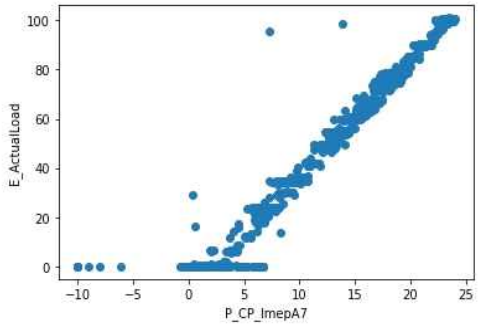
4.1 선택한 Feature의 value와 Target value의 상관관계

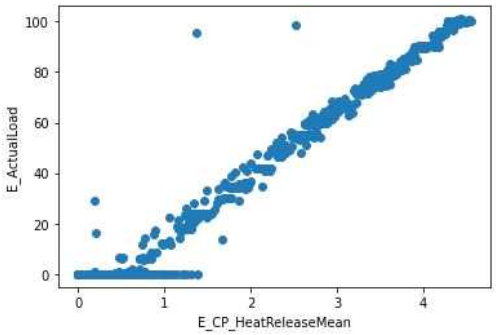


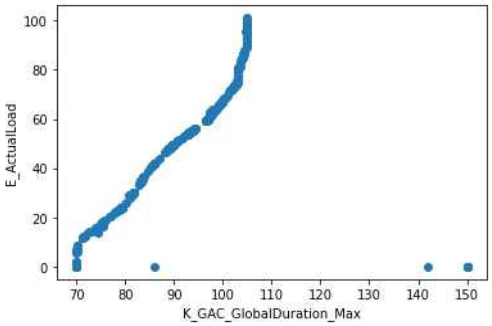


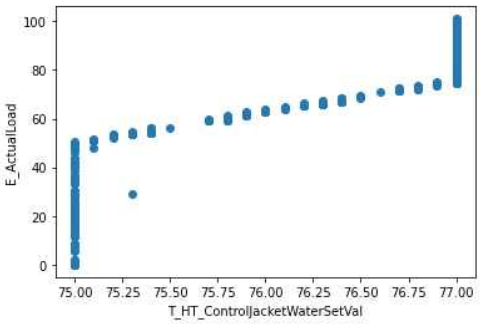












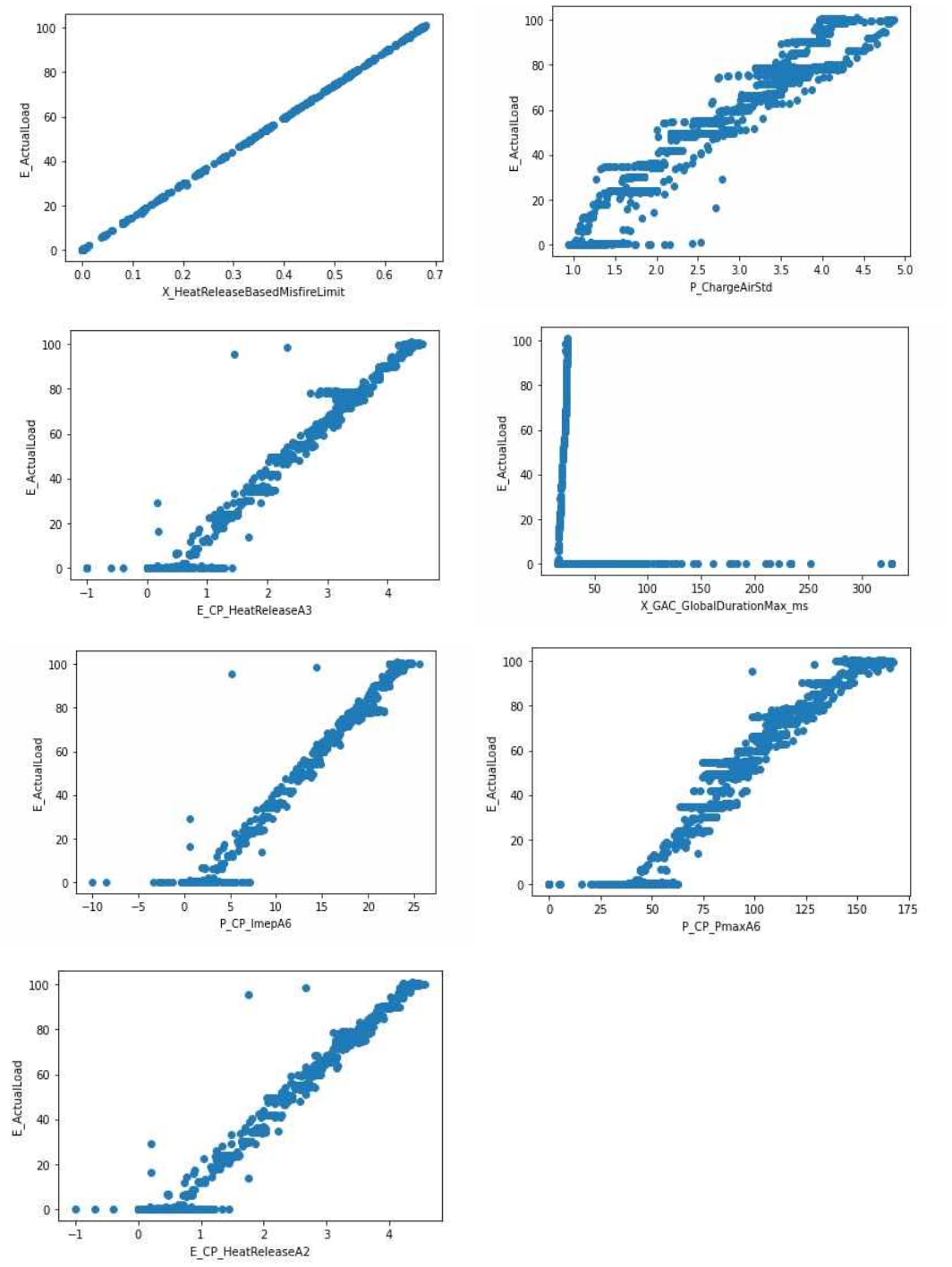


그림 9 학습된 Feature와 Target간의 상관관계

위의 그래프를 보면 feature와 Target value는 대부분 비례하는 관계를 가지고 있음을 알 수 있다.

4.2 모델 성능 평가

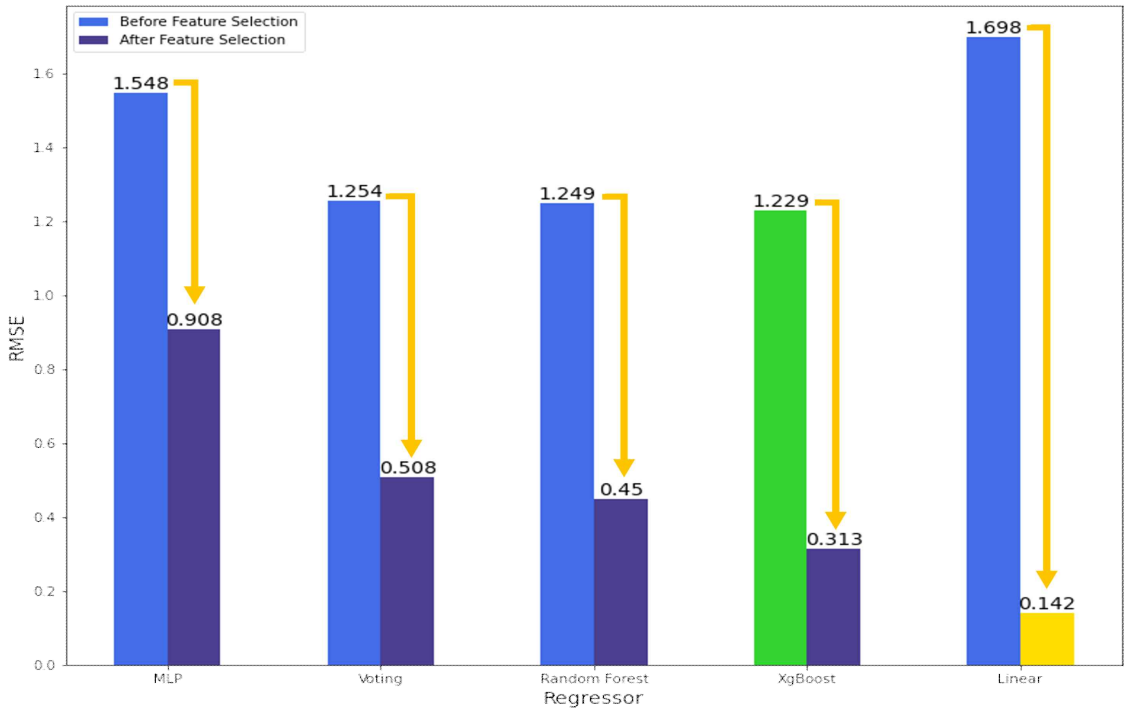


그림 10 Feature Selection 전과 후 전체적인 모델의 RMSE 비교 그래프

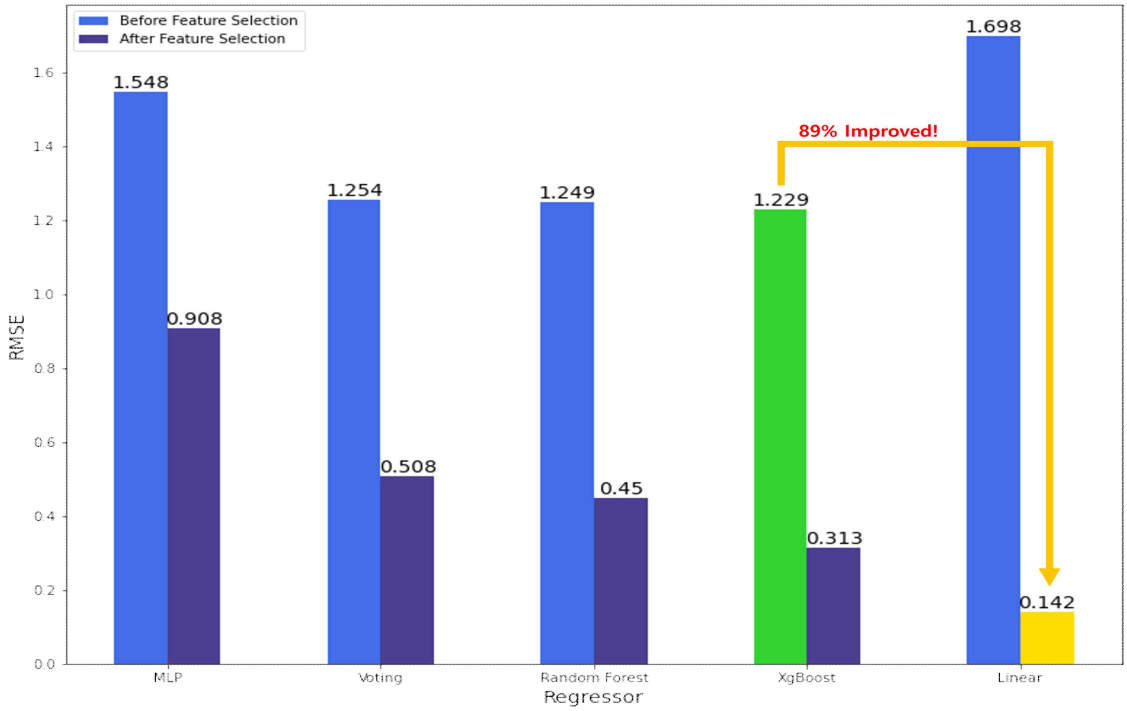


그림 11 Feature Selection 전과 후의 가장 좋은 성능을 가지는 모델 간의 비교

위의 그래프를 보면 Feature Selection을 하기 전에는 XgBoost Regression이 RMSE가 1.229로 가장 좋은 성능을 보여주고 있다. 이후 Random Forest와 XgBoost의 Feature Importance를 기반으로 Feature Selection을 하였다.

해당 Feature를 가지고 동일한 parameter로 모델을 학습한 결과 모든 회귀 모델에서 더 나은 RMSE 지표를 보여주며 평균 66.6%의 성능이 개선되었다.

Feature Selection 이전에 가장 좋은 성능을 보여주는 XgBoost Regression과 Feautre Selection 이후의 가장 좋은 성능을 보여주는 Linear Regression간의 성능 비교는 Linear Regression이 XgBoost Regression에 비해 약 89%의 성능이 개선되었음을 보여준다.

Feature Selection 전과 후의 가장 좋은 모델이 다른 이유로는 이전 Feature들은 Target value와의 관계가 비선형적이어서 XgBoost가 가장 높은 성능을 보였다. 하지만 Random Forest와 XgBoost를 기반으로 한 Feature 들은 이전 Feature들 보다 더 높은 연관성을 가지고 있으므로 모든 모델에서 성능이 개선되었다. Linear Regression이 가장 성능이 높게 나온 이유로는 Feature Selection을 통해 선택된 Feature를 보면 Target Value(엔진 부하)와 선형적인 관계를 갖는 Feature가 대부분이기 때문이다.

5. 결론 및 향후 연구 방향

우리는 H사에서 실제로 사용하는 엔진 데이터를 가지고 엔진 부하를 예측할 때, 기존에 엔진 부하를 예측할 때 사용하던 feature들 외에 엔진 부하와 좀 더 상관관계가 있는 feature를 찾고, 엔진 부하를 예측할 때 더 좋은 성능을 갖는 Regression 모델을 개발하고자 했다.

RAW 데이터를 얻기에는 수월했지만, 얻은 데이터가 크고 feature의 수가 1524개로 해당 raw 데이터 파일에는 직접 값을 지정하는 parameter value와 value의 값이 변화가 없이 일정한 값을 가지는 feature도 있었다. 아무런 전처리를 하지 않고 학습을 할 경우 학습에 매우 오랜 시간이 걸리고 모델의 성능이 떨어지고 dimension이 매우 높아져 좋지 않은 모델이 된다. 이를 보완하기 위해 먼저 excel에서 class feature와 1대1로 mapping되는 종속 feature를 제외 처리하고, Feature selection을 통해 학습을 위한 데이터의 feature를 선별하였다.

이로써 feature의 수가 매우 많은 데이터를 이용하여 모델을 학습하기 전에 먼저 데이터를 전처리하고 모델 학습에 사용할 feature를 selection 한 후 선택된 feature로 생성된 다양한 회귀모델이 기존 feature로 학습된 회귀모델과의 성능을 RMSE를 기준으로 비교해보았으며, 이를 통해 이번 과제에서 사용된 회귀모델들의 성능이 얼마나 향상되었는지 확인했다.

추후 기회가 된다면 최근에 좋은 성능을 내는 Attention 기반의 Transformer 등 최신 AI 알고리즘을 적용할 수 있다면 더 좋은 성능을 가진 모델을 만들 수 있을 것으로 기대한다.

6. 구성원별 역할 및 개발 일정

이름	역할
강 금석	● 데이터 시각화 ● 회귀 모델 생성 ● RMSE를 이용하여 모델의 성능 측정 및 평가
김 민수	● XgBoost과 Random Forest를 이용하여 Feature Importance 분석 ● 데이터 Balance 조절 ● 모델 parameter tuning
공통	보고서 작성 발표 및 시연 준비

표 1 구성원별 역할

6월				7월				8월				9월			
6	13	20	27	4	11	18	25	1	8	15	22	5	12	19	26
데이터 분석															
데이터 전처리															
	기본 분류 모델 생성														
			모델 최적화 및 입력 데이터 수정												
						중간 보고서 작성									
							모델 최적화 및 입력 데이터 수정								
														최종 성능 측정	
														최종 보고서 작성 및 발표 준비	

표 2 주차 별 개발 일정

7. 참고 문헌

<논문지>

[1] Bao. Shiyuan, Yang. Zhifang, Guo. Lin, Yu. Juan, Da., Wei, "One-segment linearization modeling of electricity-gas system optimization" *Journal of ELSEVIER : Energy,* 15 April 2020 197

[2]Brandsæter, A.; Vanem, E.; Glad, I.K.In, "Efficient on-line anomaly detection for ship systems in operation" *Journal of ELSEVIER:* Expert Systems with Applications. (Expert Systems with Applications, 1 May 2019, 121:418-437)

[3]Listou Ellefsen, A.; Han, P.; Cheng, X.; Holmeset, F.T.; AEsoy, V.; Zhang, H;,"Using Fault-Type Independent Spectral Anomaly Detection" *Journal of IEEE,* IEEE Transactions on Instrumentation and Measurement, 69(10):8216-8225 Oct, 2020

<WEB Site>

[1] scikit-learn-library [Online]. Available: https://scikit-learn.org/stable/index.html

[2] statsmodel-library[Online]. Available: https://www.statsmodels.org/stable/index.html

[3] plotly-library [Online]. Available: https://plotly.com/python/