**AIFactory 제 2 회 USG AI⋅데이터 문제해결 제조혁신 경진대회 코드 설명자료**

**- USG AI 데이터 문제해결 제조혁신 경진대회 압축성형 결과 예측 -**

**대회 참가자 (노정현, 김석진, 정제완, 추호성)**

**2022년 03월 29일**

1. **라이브러리 및 데이터 (Library & Data)**

* 코드 내에 사용할 library import
* 사용할 데이터를 pandas의 read\_csv를 사용해 Dataframe형태로 load

|  |
| --- |
| # 필요한 library import  import pandas as pd # 데이터를 Dataframe 형태로 load하기 위해  import numpy as np # 데이터를 numpy 형태로 변환하기 위해  import seaborn as sns # 데이터 EDA를 위해  import matplotlib.pyplot as plt # 데이터 EDA를 위해  from sklearn.preprocessing import LabelEncoder # 데이터 전처리를 위해  # train, val 데이터 분리 및 데이터 검증을 위해  from sklearn.model\_selection import train\_test\_split, GridSearchCV  from xgboost import XGBRegressor # 데이터 훈련 모델을 위해  # 데이터 불러오기  train = pd.read\_csv('train\_data.csv')  test = pd.read\_csv('test\_data.csv') |

1. **데이터 전처리 (Data Cleansing & Pre-Processing)**

* data에서 object형식인 EQUIP1~4를 label encoding을 진행하여 int값으로 변환
* label 데이터인 OUT열을 따로 분리하여 저장

|  |
| --- |
| # label encoding 진행  le = LabelEncoder()  for i in range(4):  train['EQUIP'+str(i+1)] = le.fit\_transform(train['EQUIP'+str(i+1)])  test['EQUIP'+str(i+1)] = le.fit\_transform(test['EQUIP'+str(i+1)])  # label 데이터 분리  data = train.drop(['OUT'], axis=1)  label = train['OUT'] |

1. **탐색적 자료 분석 (Exploratory Data Analysis)**

* 데이터의 shape, 정보 등을 알기 위해 shape, head, info, describe 메소드를 사용
* corr 메소드를 사용해서 나타난 상관관계를 hitmap으로 출력
* 각 열 별 데이터 분포도를 그래프를 사용해서 출력

|  |
| --- |
| # data shape 출력  print(train\_input.shape)  print(test\_input.shape)  # head 함수를 사용하여 데이터 출력  train\_input.head()  test\_input.head()  # info 함수를 사용하여 데이터 정보 획득, 개수, type, 결측 치 확인 등  train\_input.info()  test\_input.info()  # label encodeing을 진행한 후 다시 데이터 정보 확인  le = LabelEncoder()  for i in range(4):  train\_input["EQUIP"+str(i+1)] = le.fit\_transform(train\_input["EQUIP"+str(i+1)])  test\_input["EQUIP"+str(i+1)] = le.fit\_transform(test\_input["EQUIP"+str(i+1)])  data = train\_input  data.head()  # describe 함수를 사용하여 데이터의 개수, 평균, 분산, 최대 최소 값 등을 출력  data.describe()  # corr 메소드를 사용해 나타난 상관관계를 seaborn 라이브러리와 matplotlib.pyplot 라이브러리를 사용해 hitmap으로 표현  heatmap = sns.heatmap(data.corr(),annot=True,cmap='RdYlGn',linewidths=0.2)  fig=plt.gcf()  fig.set\_size\_inches(10,10)  heatmap.set\_xticklabels(heatmap.get\_xticklabels(), fontsize=20)  heatmap.set\_yticklabels(heatmap.get\_yticklabels(), fontsize=20)  plt.title('correlation between features', fontsize=40)  plt.show()  # 센서별 분포도를 seaborn 라이브러리와 matplotlib.pyplot 라이브러리를 사용해 그래프로 표현  feature = data.columns  plt.figure(figsize=(20,60))  for i in range(len(feature)):  plt.subplot(11,3,i+1)  plt.title(feature[i])  plt.xlim(-3,3)  plt.ylim(0,1)  sns.distplot(data[feature[i]],color='magenta')  plt.show() |

1. **변수 선택 및 모델 구축 (Feature Engineering & Initial Modeling)**

* XGBoost Regressor 사용
* rmse를 정의하여 검증 결과 값 확인
* XGBRegressor의 parameter를 GridSearch, 결과 제출, 검증셋의 결과를 통해 조정

|  |
| --- |
| # train\_test\_split를 사용해 validation 데이터 제작  train = pd.read\_csv('train\_data.csv')  test = pd.read\_csv('test\_data.csv')  le = LabelEncoder()  for i in range(4):  train['EQUIP'+str(i+1)] = le.fit\_transform(train['EQUIP'+str(i+1)])  test['EQUIP'+str(i+1)] = le.fit\_transform(test['EQUIP'+str(i+1)])  data = train.drop(['OUT'], axis=1)  label = train['OUT']  train\_data , val\_data, train\_label, val\_label = train\_test\_split(data, label, random\_state=100)  #RMSE 정의  def rmse(label, pred):  return np.sqrt(np.sum(np.power(label-pred,2))/np.size(pred))  # Grid Search 사용  model = XGBRegressor(learning\_rate=0.1, random\_state=0, n\_jobs=-1)  param = {  'base\_score':[0.4,0.5],  'max\_depth':[9,10,11],  'n\_estimators':[800,850,900]  }  grid = GridSearchCV(model, param\_grid=param, cv=3, verbose = True, n\_jobs=-1, scoring = 'neg\_root\_mean\_squared\_error')  grid.fit(train\_data,train\_label)  print('xgboost best parameters : ', grid.best\_params\_)  print('xgboost best score : ', grid.best\_score\_)  grid\_best = grid.best\_estimator\_  pred = grid\_best.predict(val\_data)  print("xgboost : {}".format(rmse(val\_label, pred)))  # 결과 제출 및 검증을 통해 가장 결과가 좋은 모델 선정  model = XGBRegressor(base\_score=0.4, learning\_rate=0.1, max\_depth=10, n\_estimators=800, random\_state=0, n\_jobs=-1) |

1. **모델 학습 및 검증 (Model Tuning & Evaluation)**

* XGBoost의 fit 함수를 사용해 모델 훈련
* 검증 데이터셋과 예측값 과의 rmse값 계산
* 전체 데이터를 사용해 훈련

|  |
| --- |
| # 모델 훈련  model.fit(train\_data,train\_label)  # 예측 및 검증  pred = model.predict(val\_data)  print(rmse(val\_label,pred))  # 전체 데이터를 사용하여 훈련  model.fit(data,label) |

1. **결과 및 결언 (Conclusion & Discussion)**

* test 데이터를 사용 및 predict 함수를 사용해 예측
* answer\_sample.csv을 불러와 OUT 열에 변경된 값들을 저장 후 index를 제거하여 XGBOOST\_answer.csv파일에 저장
* VAL1 과 VAL2가 역 상관관계를 나타내 이 문제점을 잘 분석하여 모델을 구축하는 것이 핵심
* deep learning, machine learning 기법을 사용하여 다양한 모델을 구축하여 예측 해 보았지만 XGBoost를 사용한 것이 가장 낮은 rmse값을 출력
* 더욱더 정밀하게 예측하기 위해 데이터 증강 및 모델 앙상블 기법을 활용할 필요가 있어 보인다.

|  |
| --- |
| # 예측  pred = model.predict(test)  # 예측 결과를 파일에 저장  submit = pd.read\_csv('answer\_sample.csv')  submit['OUT']=pred  submit.to\_csv('XGBOOST\_answer.csv', index=False) |