Machine_Learning_Model_For_Strong_Structures_With_FRP

학번: 1818306

이름: 김주현

Github address: htt ps://github.com/markmark50/Machine_Learning_Model_For_Strong_Structures_With_FRP.git

1. 안전 관련 머신러닝 모델 개발의 목적

a. 현재 순살아파트 등의 부실한 건축물들이 많은 이슈가 되고 있는 것이 떠올라 이를 활용하여 이와 관련된 머신러닝 모델을 만들면 좋을 것 같다고 생각하여 강화 콘크리트 구조물의 내구성과 안전성 항상, 철근 부식 방지를 위해 FRP (Fiber Reinforced Polymers)를 사용해 건축물을 보강하는데에 이용이 될 수 있는 데이터를 찾았고 이를 사용한다면 현재 FRP(Fiber Reinforced Polymers)를 사용해 구조물을 보강하는 것에 대한 설계 지침들이 부족하여 잘활용이 되지 못하고 있는 상황을 이 데이터를 통해 설계 규칙을 개발한다면 FRP(Fiber Reinforced Polymers)를 실질적으로 활용하는데에 많은 도움이 될 것 같아 이에 대한 머신 러닝 모델을 개발하였다.

b. 학습 모델 활용 대상:

b-1. FRP 보강재를 활용하고자 하는 건축가, 건설 업체

c. 데이터의 변수:

c-1.독립 변수: 'cement', 'blast_furnace_slag', 'fly_ash', 'water', 'superplasticizer', 'coarse_aggregate', 'age', 'fine_aggregate'

c-2. 종속 변수: 'concrete_compressive_strength'

c-3. 변수의 예측: 콘크리트 조성 성분, 재령 등의 독립 변수를 사용하여 종속 변수 콘크리트의 강도를 예측한다.

d. 개발의 의의:

d-1. FRP 보강재를 사용한 콘크리트 구조물의 안전성을 향상시켜 안전한 건축 환경을 조성하는데 가치를 두고 있다.

2. 안전 관련 머신러닝 모델의 네이밍의 의미

a. Machine_Learning_Model_For_Strong_Structures_With_FRP 는 FRP(Fiber Reinforced Polymers)보강재를 활용한 강한 구조물을 위한 머신러닝 모델이라는 뜻을 가지고 있고 FRP(Fiber Reinforced Polymers) 보강재의 활용에 따른 구조물을 강도를 예측하고 최적화하여 강한 구조물을 만들기

위한 머신러닝 모델을 의미한다.

3. 개발 계획

a. 데이터에 대한 요약 정리 및 시각화

a-1. 데이터 요약 정리: 'Info' 와 'describe' 매써드를 활용해 데이터의 정보와 기술 통계를 확인할 예정.

a-2. isnull() 매써드를 통해 결측치의 여부를 확인할 예정

a-3. seaborn 라이브러리를 활용하여 산점도 그래프와 상관관계 히트맵을 만들어 데이터를 시각화 할 예정.

b. 데이터 전처리 계획

b-1. fillna() 매써드를 사용해서 결측값을 해당 열의 평균 값으로 대체하고 remove_outliers 를 통해 이상치를 제거할 예정

c. 사용할 머신러닝 모델: CatBoost

c-1. CatBoost 의 이론: CatBoost 는 무작위 순열을 활용한 Target Statistics 추정 및 Ordering Principle 을 적용한 Ordered TS 를 도입하여 overfitti ng을 줄이고, oblivious decision tree 와 feature combination 등의 방법론을 통해 성능을 향상시는 방법으로 실행 된다.

d. 머신러닝 모델의 예측 결과 예상:

d-1. CatBoost 모델은 독립 변수로 받은 : 시멘트, 슬래그 물 등을 학습하여 콘크리트의 압축 강도를 예측할 것이다.

e. 사용할 성능 지표:

e.1. 평균 제곱 오차 Mean Squared Error(MSE): 모델이 예측한 값과 실제 값 간의 차이를 제곱한 뒤 평균을 내는 지표.

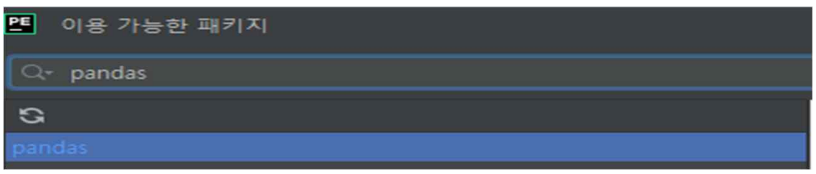
f. 성능 검증 방법:

f.1. K-Fold Cross Validation 을 통해 각 폴드에서 모델을 학습하고 평가해 모델의 성능을 검증할 예정.

4. 개발 과정

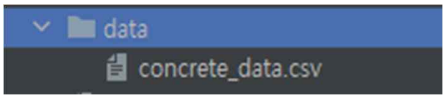
a. 계획 후 실제 학습 모델 개발 과정을 기록 (*개발 과정 캡쳐 필수)

a.1. 사용할 라이브러리 설치



(numpy, matplotlib, seaborn, scikit-learn, catboost 도 동일한 방법으로 설치)

a.2. data 라는 이름을 가진 디렉터리 생성 후 가지고 있는 데이터 첨부



a.3. import 를 통해 사용할 라이브러리들을 가져오고 각각을 네이밍

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from catboost import CatBoostRegressor
from sklearn.model_selection import KFold
from sklearn.metrics import mean_squared_error
```

a.4. data 라는 이름을 가진 디렉터리에 첨부한 데이터의 경로 지정 (상대 경로 사용)

```
# 데이터 파일 경로 지정
file_path = "./data/concrete_data.csv"
```

a.5. 데이터를 불러와 df 변수에 저장

```
# 데이터 불러오기
df = pd.read_csv(file_path)
```

a.6. info, describe 를 활용해 데이터의 정보, 요약 통계를 확인하여 데이터의 전반적인 정보를 파악

```
# 데이터 정보 출력
print(df.info())

# 데이터 요약 통계 출력
print(df.describe())
```

a.7. isnull 을 통해 데이터의 결측값 확인

```
# 결측값 확인
print(df.isnull().sum())
```

a.8. 데이터의 결측값을 평균값으로 대체하고 사분위 범위 기반으로 이상치 제거

```
# 전처리 결측값을 평균값으로 대체
df.fillna(df.mean(), inplace=True)

# 이상치 처리: IQR 기반
def remove_outliers(data, column):
    Q1 = data[column].quantile(0.25)
    Q3 = data[column].quantile(0.75)
    IQR = Q3 - Q1
    lower_bound = Q1 - 1.5 * IQR
    upper_bound = Q3 + 1.5 * IQR
    return data[(data[column] >= lower_bound) & (data[column] <= upper_boun

# 모든 열에 대해 이상치 처리 적용
for column in df.columns:
    df = remove_outliers(df, column)
```

a.9. 데이터를 시각화를 위한 산점도 그래프 출력(한번에 모든 그래프를 출력하게 되면 일부분이 잘려서 보이지 않아 가독성을 높이기 위해 데이터를 2 개의 그룹으로 나눠서 그래프를 생성)

```
# 선택한 열들을 두 그룹으로 나누기
selected_columns_1 = ['cement', 'blast_furnace_slag', 'fly_ash', 'water', 'superplasticizer']
selected_columns_2 = ['coarse_aggregate', 'age', 'concrete_compressive_strength', 'fine_aggregate ']

# 첫 번째 그룹의 산점도 매트릭스
scatter_1 = sns.pairplot(df[selected_columns_1], height=2, plot_kws={'s': 5})
scatter_1.fig.suptitle('Scatterplot Matrix - Part 1', y=1.02)
plt.show()

# 두 번째 그룹의 산점도 매트릭스
scatter_2 = sns.pairplot(df[selected_columns_2], height=2, plot_kws={'s': 5})
scatter_2.fig.suptitle('Scatterplot Matrix - Part 2', y=1.02)
plt.show()
```

a.10. 데이터 시각화를 위한 열 상관관계 히트맵 생성(그래프의 높이는 17 너비는 8 로 설정하였고 히트맵의 색상은 coolwarm 사용)

```
# 열 상관관계 히트맵
plt.figure(figsize=[17, 8])
sns.heatmap(df.corr(), annot=True, cmap='coolwarm')
plt.show()
```

a.11. 독립 변수와 종속 변수 설정(콘크리트의 압축 강도를 제외한 나머지 열들을 선택하여 독립변수(x)로 할당, 콘크리트의 압축 강도 열은 종속 변수(y)로 할당

```
# 독립 변수와 종속 변수 설정
x = df.drop(['concrete_compressive_strength'], axis=1)
y = df['concrete_compressive_strength']
```

a.12. 데이터 검증을 위해 사용할 K-Folf Cross Validation 을 수행하기 위해 데이터셋을 5 개의 폴드로 나누고 데이터를 섞어 난수를 생성해 실험 결과를 재현하기 위해 사용

```
# K-Fold Cross Validation을 위한 설정
kf = KFold(n_splits=5, shuffle=True, random_state=42)
```

a.13. 평균 제곱 오차(MSE)를 저장할 리스트 생성

```
# 평균 제곱 오차(MSE)를 저장할 리스트
mse_scores = []
```

a.14. 먼저 for 반복문을 통해 각 폴드에 대해 반복하게 하였고 x 와 y train 의 훈련셋과 x, y test 의 검증셋을 생성
CatBoostRegressor 모델을 생성해 모델을 학습
eval.set 을 통해 검증셋으로 모델을 평가하며 학습 과정을 모니터링
x_test 에 대해 예측을 수행하여 y_pred 에 저장
MSE 값을 계산하고 이를 mse_scores 리스트에 추가
cat.score 를 통해 모델의 스코어 계산

```
# CatBoost 모델 학습 및 평가
for train_index, test_index in kf.split(x):
    x_train, x_test = x.iloc[train_index], x.iloc[test_index]
    y_train, y_test = y.iloc[train_index], y.iloc[test_index]

    cat = CatBoostRegressor(loss_function='RMSE', random_state=42)
    cat.fit(x_train, y_train, eval_set=(x_test, y_test), plot=False)

    y_pred = cat.predict(x_test)
    mse = mean_squared_error(y_test, y_pred)
    mse_scores.append(mse)

    cat_score = cat.score(x_test, y_test)
    print(f'CatBoost Score: {cat_score}')
```

a.15. K-Folf Cross Validation 의 각 폴드에서 계산된 MSE 를 출력

```
# K-Fold Cross Validation 결과 출력
print(f'Mean Squared Error for each fold: {mse_scores}')
print(f'Mean Squared Error (average): {np.mean(mse_scores)}')
```

a.16. 머신러닝 모델 학습 후 산점도 그래프를 통해 결과를 시각화
(색의 차별화를 위해 예측값은 빨강, 실제값은 파랑을 적용)

```
# 예측값과 실제값 비교 시각화 (예측값: 빨강, 실제값: 파랑)
plt.figure(figsize=(10, 6))
plt.scatter(y_test, y_pred, color='red', alpha=0.5, label='Predicted')
plt.scatter(y_test, y_test, color='blue', alpha=0.5, label='Actual')
plt.xlabel("Concrete Compressive Strength")
plt.ylabel("Values")
plt.title("CatBoost Model - Actual vs Predicted")
plt.legend()
plt.show()
```

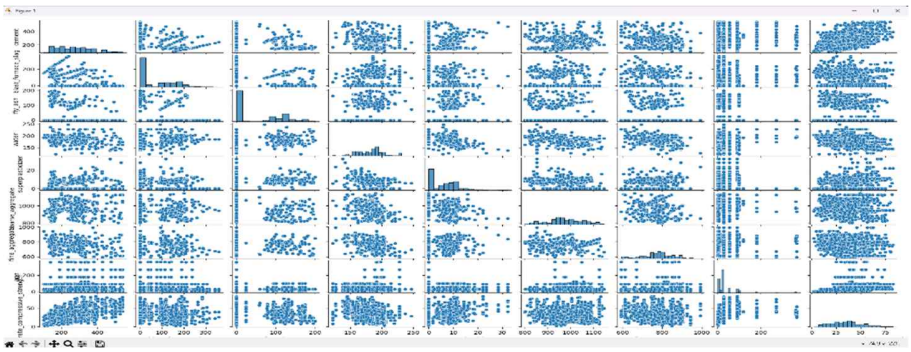
b. 함수의 동작 설명

b.1. remove_outliers 함수: 주어진 데이터프레임 data 와 열 이름 column 을 받아 해당 열에 대해 사분위 범위 기반의 이상치를 제거한

새로운 데이터프레임을 반환하고 이 함수를 모든 열에 대해 for 문을 사용하여 모든 열의 이상치를 처리

c. 에러 발생 지점 및 해결 과정

c.1. 산점도 그래프에서 열에 데이터가 겹쳐서 표기되고 너무 많은 양의 자료가 나와 데이터를 전혀 알아볼 수가 없었다.



이를 해결하기 위해 열들을 두개의 그룹으로 나누어서 산점도 그래프를 2 번으로 나누어 제작 하도록 설정하였다.

c.2. 데이터 fine_aggregate 가 존재하지 않는다는 오류

```
  File "C:\Users\mark5\venv\Lib\site-packages\pandas\core\frame.py", line 3899, in __getitem__
    indexer = self.columns._get_indexer_strict(key, "columns")[1]
              ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
  File "C:\Users\mark5\venv\Lib\site-packages\pandas\core\indexes\base.py", line 6114, in _get_indexer_strict
    self._raise_if_missing(keyarr, indexer, axis_name)
  File "C:\Users\mark5\venv\Lib\site-packages\pandas\core\indexes\base.py", line 6178, in _raise_if_missing
    raise KeyError(f"{not_found} not in index")
KeyError: "['fine_aggregate'] not in index"
```

이 오류의 발생 이유를 오래 동안 찾지 못하였지만 데이터를 다시 확인해본 결과 데이터가 fine_aggregate 뒤에 한칸의 공백이 존재하였고 이를 맞추어 한칸의 공백을 부여하니 오류가 해결되었다.

d. 학습 모델의 성능 평가

d.1. 최적의 성능 지표:

d.1.1. 첫 번째 학습: 4.030869613

d.1.2. 두 번째 학습: 3.389734469

d.1.3. 세 번째 학습: 4.583356044

d.2. 반복 횟수:

d.2.1. 첫번째 학습: 999 회

d.2.2. 두 번째 학습 998 회

d.2.3. 세 번째 학습 999 회

d.3. 정확도:

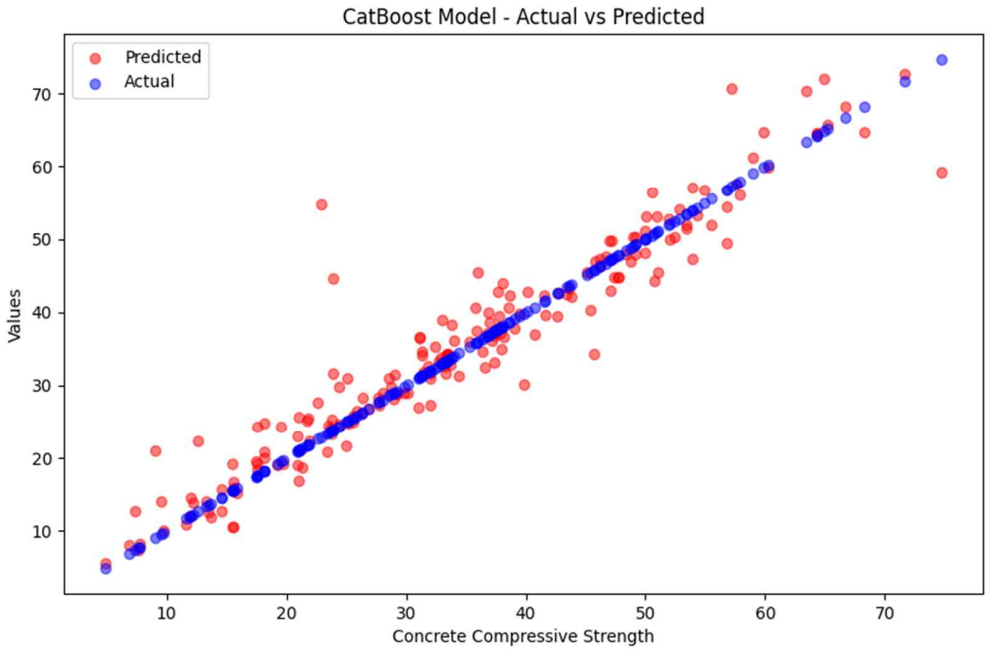
d.3.1. 첫번째 학습: 0.9404152214789132

d.3.2. 두 번째 학습: 0.9637904713221469

d.3.3. 세 번째 학습: 0.9091163766706

d.4. 평균 MSE 값: 17.696527607981796

e. 결과 시각화



5. 개발 후기

a. Machine_Learning_Model_For_Strong_Structures_With_FRP 를 개발하게 되면서 지금 시점에서 어떤 안전 관련 머신러닝 모델을 만들어야 사람들이 관심을 가지고 이를 활용해 볼 수 있을지에 대하여 많은 고민을 하였습니다. 그 결과 FRP (Fiber Reinforced Polymers)를 사용해 건축물을 보강하는 데에 이용이 될 수 있는 데이터 찾았고 이에 대한 머신 러닝 모델을 개발했습니다. 이번 머신 러닝 모델을 개발하면서 데이터를 시각화하는 부분에 있어

가독성이 높은 자료를 만드는 부분에 대한 어려움을 느꼈습니다. 이러한 문제를 해결하기 위해 그래프의 글자 크기 조절, 산점도 그래프의 점 크기 조절 등의 여러 방안들을 고려하여 프로그래밍을 해 보았고 결과적으로 이에 대한 해결책으로 그래프를 그룹화하고 그룹을 2 개로 나누어 데이터를 식각화하는 해결책으로 고안해 내었습니다.
이처럼 다양한 방안들을 생각해 보고 이를 직접 적용해 출력되는 결과들을 눈으로 확인하면서 문제를 해결하는 데에 창의성과 ChatGPT 와 같은 딥러닝 모델을 활용하여 디버깅하는 것에 대한 중요성을 크게 깨달았습니다. 이런 개발 경험을 통해 프로그래밍과 머신러닝에 대한 다양한 지식을 얻어내었고 추가로 여러 문제와 직면하더라도 다양한 해결책을 고안해 내고 이를 통하여 최선의 해결책을 찾는 마인드 셋을 확립할 수 있는 계기가 된 것 같습니다.