Pipeline

PinkLAB Edu

4 November, 2025

Table of Contents

1 Pipeline?

1.1 지금까지 내용에서 불편함은?

- 단순히 Iris, Wine 데이터를 받아서 사용했을 뿐인데,
  직접 공부하면서 코드를 하나씩 실행해보면 혼돈이 크다는 것을 알 수 있다
- Jupyter Notebook 상황에서 데이터의 전처리와 여러 알고리즘의 반복 실행,
  하이퍼 파라미터의 튜닝 과정을 번갈아 하다 보면 코드의 실행 순서에 혼돈이 있을 수 있다
- 이런 경우 클래스(class)로 만들어서 진행해도 되지만,
- sklearn 유저에게는 꼭 그럴 필요없이 준비된 기능이 있다 → Pipeline

1.2 다시 와인 데이터~

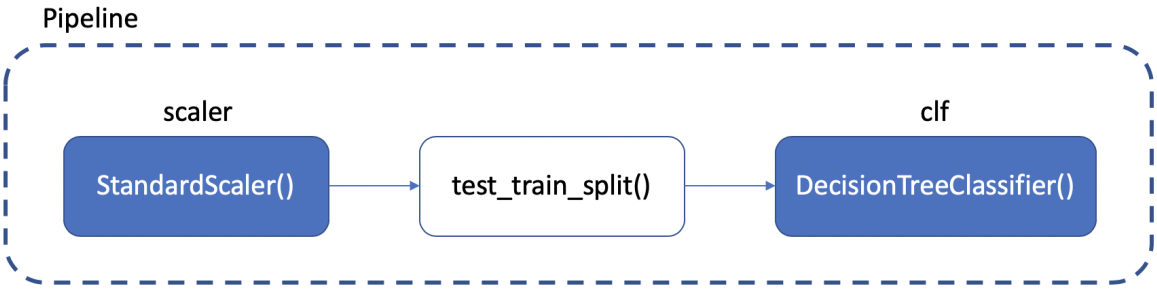
```python
import pandas as pd

red_url = "https://github.com/PinkWink/ML_tutorial/raw/refs/heads" + \
                                        "/master/dataset/winequality-red.csv"

white_url = "https://github.com/PinkWink/ML_tutorial/raw/refs/heads" +\
                                        "/master/dataset/winequality-white.csv"



red_wine = pd.read_csv(red_url, sep = ';')
white_wine = pd.read_csv(white_url, sep = ";")

red_wine['color'] = 1
white_wine['color']= 0

wine = pd.concat([red_wine, white_wine])
```

✓ 1.1s Python

```python
X = wine.drop(['color'], axis = 1)
y = wine['color']
```

✓ 0.0s Python

1.3 레드/화이트 와인 분류기의 동작 Process



- 여기서 test_train_split은 Pipeline 내부가 아니다.

1.4 방금 부분의 Pipeline을 코드로 구현하면?

```
from sklearn.pipeline import Pipeline
from sklearn.tree import DecisionTreeClassifier
from sklearn.preprocessing import StandardScaler

estimators = [('scaler', StandardScaler()),
              ('clf' , DecisionTreeClassifier())]

pipe = Pipeline(estimators)
```

✓ 0.6s Python

- 와우~ 쉽죠?

1.5 pipeline.steps

```
pipe.steps
```

✓ 0.0s Python

```
[('scaler', StandardScaler()), ('clf', DecisionTreeClassifier())]
```

```
pipe.steps[0]
```

✓ 0.0s Python

```
('scaler', StandardScaler())
```

```
pipe.steps[1]
```

✓ 0.0s Python

```
('clf', DecisionTreeClassifier())
```

1.6 스텝별로 객체 호출

```
pipe[0]
```

✓ 0.0s Python

▾ StandardScaler

```
StandardScaler()
```

```
pipe['scaler']
```

✓ 0.0s Python

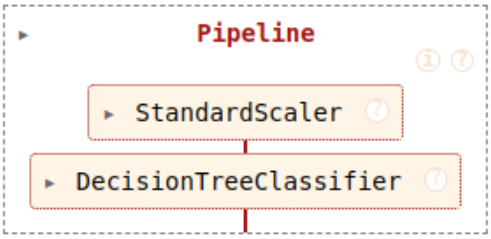
▾ StandardScaler

```
StandardScaler()
```

1.7 set_params

```
pipe.set_params(clf__max_depth = 2)
pipe.set_params(clf__random_state = 13)
```

✓ 0.0s Python

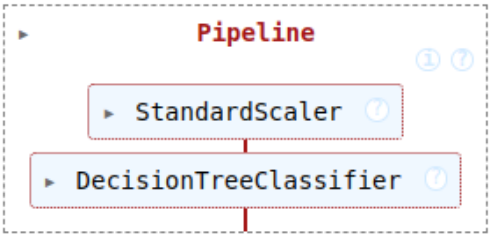


- 스탭이름 “clf” + 언더바 두 개 “- -” + 속성 이름

1.8 Pipeline을 이용한 분류기 구성

```
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2,
                                                    random_state = 13,
                                                    stratify = y)
pipe.fit(X_train, y_train)
```

✓ 0.0s Python



1.9 성과

```
from sklearn.metrics import accuracy_score

y_pred_tr = pipe.predict(X_train)
y_pred_test = pipe.predict(X_test)

print('Train Acc : ' , accuracy_score(y_train, y_pred_tr))
print("Test Acc : " , accuracy_score(y_test, y_pred_test))
```

✓ 0.0s Python

```
Train Acc :  0.9657494708485664
Test Acc :  0.9576923076923077
```

1.10 모델 구조 확인

```
import matplotlib.pyplot as plt
from sklearn import tree

fig = plt.figure(figsize=(15, 8))
_ = tree.plot_tree(pipe['clf'],
                   feature_names = X.columns,
                   class_names = ["W" , "R"],
                   rounded = True,
                   filled=True)
```

✓ 0.1s Python