Chapter 07. Pipeline

Pipeline?

zero-base /

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

zero-base /

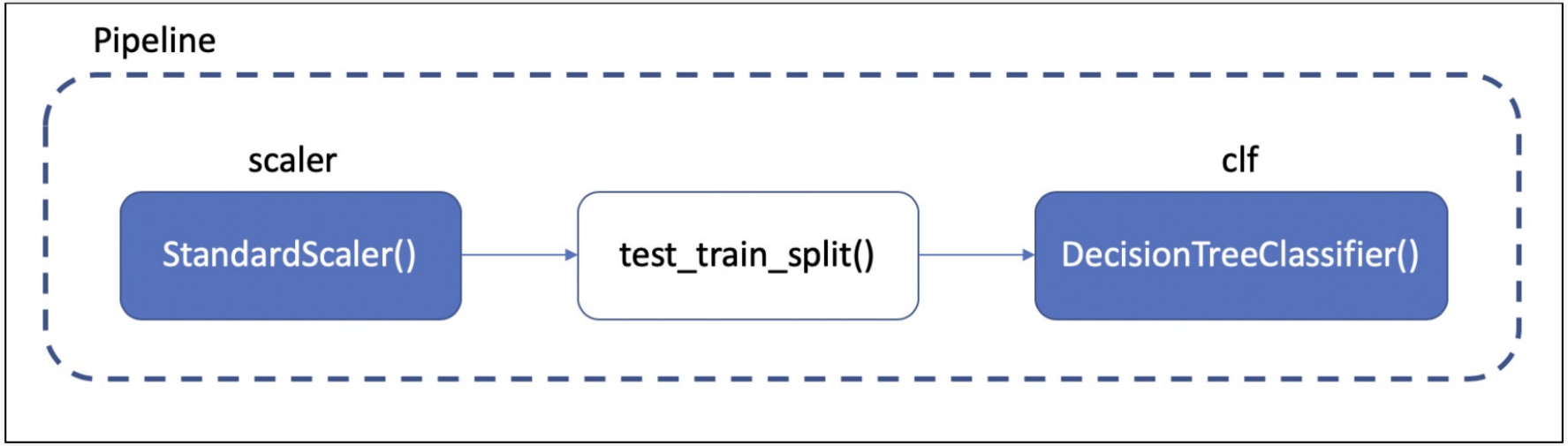
다시 와인 데이터~

```
import pandas as pd

red_url = 'https://raw.githubusercontent.com/PinkWink/ML_tutorial'+\
                                          '/master/dataset/winequality-red.csv'
white_url = 'https://raw.githubusercontent.com/PinkWink/ML_tutorial'+\
                                          '/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])
```

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

zero-base /

레드/화이트 와인 분류기의 동작 Process



- 여기서 test_train_split은 Pipeline 내부가 아니다.

zero-base /

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)
```

- 와우~ 쉽죠?

pipeline.steps


```
pipe.steps
```

```
[('scaler', StandardScaler(copy=True, with_mean=True, with_std=True)),
 ('clf',
  DecisionTreeClassifier(ccp_alpha=0.0, class_weight=None, criterion='gini',
                         max_depth=None, max_features=None, max_leaf_nodes=None,
                         min_impurity_decrease=0.0, min_impurity_split=None,
                         min_samples_leaf=1, min_samples_split=2,
                         min_weight_fraction_leaf=0.0, presort='deprecated',
                         random_state=None, splitter='best'))]
```

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

```
('scaler', StandardScaler(copy=True, with_mean=True, with_std=True))
```

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

```
('clf',
 DecisionTreeClassifier(ccp_alpha=0.0, class_weight=None, criterion='gini',
                        max_depth=None, max_features=None, max_leaf_nodes=None,
                        min_impurity_decrease=0.0, min_impurity_split=None,
                        min_samples_leaf=1, min_samples_split=2,
                        min_weight_fraction_leaf=0.0, presort='deprecated',
                        random_state=None, splitter='best'))
```

zero-base /

스텝별로 객체 호출

```
pipe[0]
```

```
StandardScaler(copy=True, with_mean=True, with_std=True)
```

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

```
StandardScaler(copy=True, with_mean=True, with_std=True)
```

zero-base /

set_params

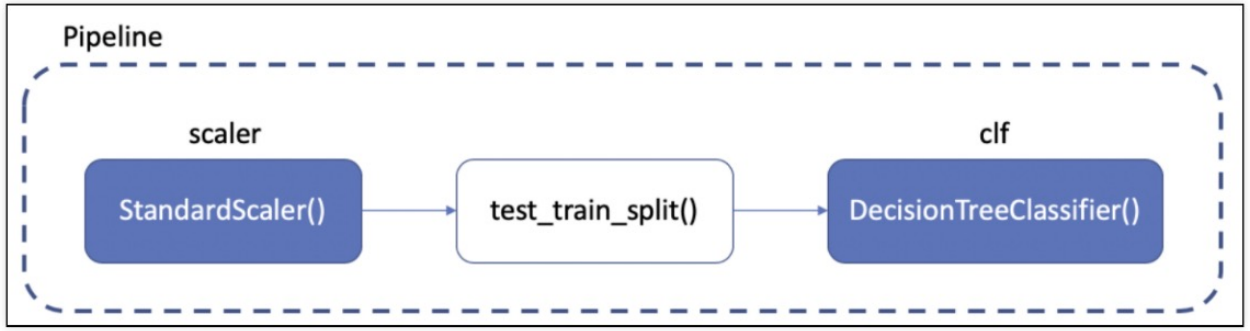
```
pipe.set_params(clf__max_depth=2)
pipe.set_params(clf__random_state=13)
```

```
Pipeline(memory=None,
         steps=[('scaler',
                 StandardScaler(copy=True, with_mean=True, with_std=True)),
                ('clf',
                 DecisionTreeClassifier(ccp_alpha=0.0, class_weight=None,
                                        criterion='gini', max_depth=2,
                                        max_features=None, max_leaf_nodes=None,
                                        min_impurity_decrease=0.0,
                                        min_impurity_split=None,
                                        min_samples_leaf=1, min_samples_split=2,
                                        min_weight_fraction_leaf=0.0,
                                        presort='deprecated', random_state=13,
                                        splitter='best'))],
         verbose=False)
```

- 스탭이름 “clf” + 언더바 두 개 “- -” + 속성 이름

zero-base /

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)
```

zero-base /

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))
```

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

zero-base /

모델 구조 확인

```
from graphviz import Source
from sklearn.tree import export_graphviz

Source(export_graphviz(pipe['clf'], feature_names=X.columns,
                       class_names=['W', 'R'],
                       rounded=True, filled=True))
```