모형진단과 교차검증 (실습)

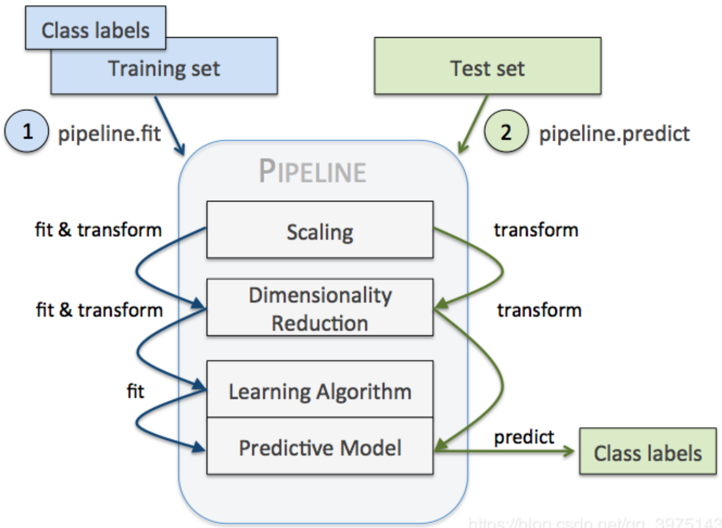
Evaluation & Cross Validation

파이프라인 실습 #1

- **파이프 라인(Pipeline)**
  - 사이킷런의 Pipeline 클래스는 연속된 변환을 순차적으로 처리할 수 있는 기능을 제공하는 유용한 래퍼(Wrapper) 도구

```
pipe_lr = make_pipeline(StandardScaler(),
                        PCA(n_components=2),
                        LogisticRegression(solver='liblinear', random_state=1))

pipe_lr.fit(X_train, y_train)
y_pred = pipe_lr.predict(X_test)
print('테스트 정확도: %.3f' % pipe_lr.score(X_test, y_test))
```



파이프라인 실습 #1

- 데이터셋: 유방암 데이터
- 학습/시험 데이터: X, 학습/시험 데이터 라벨: Y

① 데이터 로드

```
# https://scikit-learn.org/stable/modules/generated/sklearn.datasets.load_breast_cancer.html
# Dimensionality: 30, Classes: 2
# 212(M-유방암)-label('0'), 357(B-정상인) - label('1')
from sklearn.datasets import load_breast_cancer
cancer = load_breast_cancer()
X = cancer.data
Y = cancer.target
```

② 데이터 분할 (학습/테스트)

```
# https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.train_test_split.htm

# 학습 데이터 분할
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, stratify=Y, random_state=1)
```

파이프라인 실습 #1

- 데이터셋: 유방암 데이터
- 학습/시험 데이터: X, 학습/시험 데이터 라벨: Y

③ 파이프라인 모듈 설계

```
# https://scikit-learn.org/stable/modules/generated/sklearn.pipeline.Pipeline.html
# 파이파라인 기능을 이용한 모듈 설계

from sklearn.preprocessing import StandardScaler
from sklearn.decomposition import PCA
from sklearn.linear_model import LogisticRegression
from sklearn.pipeline import make_pipeline

pipeline = make_pipeline(StandardScaler(), PCA(n_components=4), LogisticRegression() )
```

④ 모델 학습 및 평가

```
# 모델 학습 및 테스트
pipeline.fit(X_train,Y_train)
Y_Pred = pipline.predict(X_test)
```

```
0.9736842105263158
```

```
# 모델 평가
from sklearn.metrics import accuracy_score
accuracy_score(Y_test, Y_Pred)
```

```
0.9736842105263158
```

교차검증 실습 #1

- 데이터셋: 유방암 데이터
- 학습/시험 데이터: X, 학습/시험 데이터 라벨: Y

① 데이터 로드

```python
# https://scikit-learn.org/stable/modules/generated/sklearn.datasets.load_breast_cancer.html
# Dimensionality: 30, Classes: 2
# 212(M-유방암)-label('0'), 357(B-정상인) - label('1')
from sklearn.datasets import load_breast_cancer
cancer = load_breast_cancer()
X = cancer.data
Y = cancer.target
```

② 데이터 분할 (학습/테스트)

```python
# https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.train_test_split.htm

# 학습 데이터 분할
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, stratify=Y, random_state=1)
```

교차검증 실습 #1

- 데이터셋: 유방암 데이터
- 학습/시험 데이터: X, 학습/시험 데이터 라벨: Y

③ 파이프라인 모듈 설계

```
# https://scikit-learn.org/stable/modules/generated/sklearn.pipeline.Pipeline.html
# 파이파라인 기능을 이용한 모듈 설계

from sklearn.preprocessing import StandardScaler
from sklearn.decomposition import PCA
from sklearn.linear_model import LogisticRegression
from sklearn.pipeline import make_pipeline

pipeline = make_pipeline(StandardScaler(), PCA(n_components=4), LogisticRegression() )
```

④ 교차검증

```
# https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.cross_validate.html#

from sklearn.model_selection import cross_validate
scores = cross_validate(pipeline, X_train, Y_train, cv=10, return_train_score=True)
```

교차검증 실습 #1

- 데이터셋: 유방암 데이터
- 학습/시험 데이터: X, 학습/시험 데이터 라벨: Y

⑤ 과적합 여부 분석

```
import numpy as np

print('CV Validation Accuracy scores: ', scores['train_score'])
print('CV Validation Accuracy: %.3f +/- %.3f' %(np.mean(scores['train_score']), np.std(scores['train_score'])))
```

```
CV Validation Accuracy scores:  [0.96577017 0.96577017 0.96577017 0.96577017 0.96821516 0.96585366
 0.97073171 0.96829268 0.97560976 0.96585366]
CV Validation Accuracy: 0.968 +/- 0.003
```

```
import numpy as np

print('CV Validation Accuracy scores: ', scores['test_score'])
print('CV Validation Accuracy: %.3f +/- %.3f' %(np.mean(scores['test_score']), np.std(scores['test_score'])))
```

```
CV Validation Accuracy scores:  [0.97826087 0.97826087 0.95652174 1.         0.95652174 0.97777778
 0.93333333 0.95555556 0.91111111 1.        ]
CV Validation Accuracy: 0.965 +/- 0.027
```

교차검증 실습 #1

- 데이터셋: 유방암 데이터
- 학습/시험 데이터: X, 학습/시험 데이터 라벨: Y

⑥ 최적 모델 탐색

```python
from sklearn.model_selection import GridSearchCV

parameters = {}
gs = GridSearchCV(pipeline, parameters, scoring='accuracy',cv=10)
gs.fit(X_train,Y_train)
```

```python
best = gs.best_estimator_
```

```python
gs.cv_results_
```

```
{'mean_fit_time': array([0.00662093]),
 'mean_score_time': array([0.00051041]),
 'mean_test_score': array([0.9647343]),
 'params': [{}],
 'rank_test_score': array([1], dtype=int32),
 'split0_test_score': array([0.97826087]),
 'split1_test_score': array([0.97826087]),
 'split2_test_score': array([0.95652174]),
 'split3_test_score': array([1.]),
 'split4_test_score': array([0.95652174]),
 'split5_test_score': array([0.97777778]),
 'split6_test_score': array([0.93333333]),
 'split7_test_score': array([0.95555556]),
 'split8_test_score': array([0.91111111]),
 'split9_test_score': array([1.]),
 'std_fit_time': array([0.00164769]),
 'std_score_time': array([3.95498696e-05]),
 'std_test_score': array([0.02665336])}
```

교차검증 실습 #1

- 데이터셋: 유방암 데이터
- 학습/시험 데이터: X, 학습/시험 데이터 라벨: Y

⑦ 최적 모델 평가

```
from sklearn.metrics import accuracy_score

Y_train_Pred = best.predict(X_train)
accuracy_score(Y_train, Y_train_Pred)

```

```
0.967032967032967
```

```
from sklearn.metrics import accuracy_score

Y_test_Pred = best.predict(X_test)
accuracy_score(Y_test, Y_test_Pred)

```

```
0.9736842105263158
```

최적 모델 찾기 실습 #1

- 데이터셋: 유방암 데이터
- 학습/시험 데이터: X, 학습/시험 데이터 라벨: Y

① 데이터 로드

```python
# https://scikit-learn.org/stable/modules/generated/sklearn.datasets.load_breast_cancer.html
# Dimensionality: 30, Classes: 2
# 212(M-유방암)-label('0'), 357(B-정상인) - label('1')
from sklearn.datasets import load_breast_cancer
cancer = load_breast_cancer()
X = cancer.data
Y = cancer.target
```

② 데이터 분할 (학습/테스트)

```python
# https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.train_test_split.htm

# 학습 데이터 분할
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, stratify=Y, random_state=1)
```

최적 모델 찾기 실습 #1

- 데이터셋: 유방암 데이터
- 학습/시험 데이터: X, 학습/시험 데이터 라벨: Y

③ 파이프라인 모듈 설계

```python
# https://scikit-learn.org/stable/modules/generated/sklearn.pipeline.Pipeline.html
# 파이파라인 기능을 이용한 모듈 설계

from sklearn.preprocessing import StandardScaler
from sklearn.svm import SVC
from sklearn.pipeline import Pipeline
from sklearn.model_selection import GridSearchCV

# 파라미터 Parsing
estimators = [('normalization', StandardScaler()), ('clf', SVC())]
pipe = Pipeline(estimators)
```

④ GridSearch 파라미터 세팅

```python
# https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.ParameterGrid.html#sklearn.model_selection.ParameterGrid

from sklearn.model_selection import ParameterGrid
grid = [{'clf__kernel': [['linear']], 'clf__C': [[0.001], [0.01],[0.1],[1],[10], [100], [1000]]},
        {'clf__kernel': [['rbf']], 'clf__gamma': [[0.001], [0.01],[0.1],[1],[10], [100], [1000]], 'clf__C': [[0.001], [0.01],[0.1],[1],[10], [100], [1000]]}]

grid_param = ParameterGrid(grid)
list(grid_param)
```

최적 모델 찾기 실습 #1

- 데이터셋: 유방암 데이터
- 학습/시험 데이터: X, 학습/시험 데이터 라벨: Y

⑤ GridSearch 정의 및 최적 모델 탐색

```
gs = GridSearchCV(pipe, grid_param, scoring='accuracy', cv=10, n_jobs=1)
```

```
gs.fit(X_train,Y_train)
```

```
print(gs.best_score_)
```

```
0.9758454106280192
```

```
print(gs.best_params_)
```

```
{'clf__C': 10, 'clf__gamma': 0.001, 'clf__kernel': 'rbf'}
```

⑥ 최적 모델 평가

```
best_model = gs.best_estimator_
Y_test_pred = best_model.predict(X_test)
```

```
from sklearn.metrics import accuracy_score

Y_test_Pred = best_model.predict(X_test)
accuracy_score(Y_test, Y_test_Pred)
```

```
0.9824561403508771
```