3. 모델 학습 및 평가

1 사이킷런을 이용한 모델 학습

지도 학습 모델

사이킷런은 다양한 지도 학습 모델 클래스를 지원하며, 모든 클래스의 사용 과정이 거의 비슷합니다.

모델 종류 (패키지명)	분류 모델 클래스	예측 모델 클래스
선형 모델 (linear_model)	• LogisticRegression	• LinearRegression • Ridge • Lasso • ElasticNet
서포트 벡터 머신 (svm)	• SVC	• SVR
k-최근접 이웃 (neighbors)	• KNeighborsClassifier	• KNeighborsRegressor
나이브베이즈 (naïve_bayes)	• GaussianNB • MultinomialNB • BernoulliNB	해당없음
결정 나무 (tree)	• DecisionTreeClassifier	• DecisionTreeRegressor
앙상블 (ensemble)	• RandomForestClassifier • GradientBoostingClassifier • StackingClassifier	• RandomForestRegressor • GradientBoostingRegressor • StackingRegressor
신경망 (neural_network)	• MLPClassifier	• MLPRegressor

예제 데이터 불러오기

모델 학습에 사용할 예제 데이터를 불러옵니다.

예제 데이터 불러오기

```
1  import os
2  import pandas as pd
3  from sklearn.model_selection import train_test_split
4  os.chdir("../../data")
5  df1 = pd.read_csv("classification/sonar.csv")
6  df2 = pd.read_csv("classification/iris.csv")
7  df3 = pd.read_csv("regression/wankara.csv")
8  X1 = df1.drop('y', axis = 1)
9  y1 = df1['y']
10 X1_train, X1_test, y1_train, y1_test = train_test_split(X1, y1, random_state = 2022)
11 X2 = df2.drop('y', axis = 1)
12 y2 = df2['y']
13 X2_train, X2_test, y2_train, y2_test = train_test_split(X2, y2, random_state = 2022)
14 X3 = df3.drop('y', axis = 1)
15 y3 = df3['y']
16 X3_train, X3_test, y3_train, y3_test = train_test_split(X3, y3, random_state = 2022)
```

- df1: 이진 분류 데이터
- df2: 다중 분류 데이터
- df3: 회귀 데이터

모델 인스턴스화

모델 인스턴스화는 사이킷런의 지도 학습 클래스를 이용해 인스턴스를 만드는 것을 의미합니다. 이 과정에서 모델의 하이퍼 파라미터를 설정합니다.

모델 인스턴스화 예시: 최대 깊이가 10인 결정 나무 모델

```
1  from sklearn.tree import DecisionTreeClassifier as DTC
2  model = DTC(max_depth = 10)
3  print(model)
```

```
DecisionTreeClassifier(max_depth=10)
```

- 모델 클래스(DecisionTreeClassifier)와 설정한 하이퍼 파라미터 (max_depth=10)만 출력됨
- 이렇게 만든 인스턴스는 하이퍼 파라미터를 통해 어떻게 학습할지만 설정됐고 아직 학습되지 않았으므로 예측과 학습된 모델 정보 확인 등을 할 수 없음

모델 인스턴스화 예시: 사전을 이용한 입력

```
1  parameter = {"max_depth": 10}
2  model = DTC(**parameter)
3  print(model)
```

```
DecisionTreeClassifier(max_depth=10)
```

모델 인스턴스화 (계속)

모델 인스턴스화는 사이킷런의 지도 학습 클래스를 이용해 인스턴스를 만드는 것을 의미합니다. 이 과정에서 모델의 하이퍼 파라미터를 설정합니다.

이진 분류 모델, 다중 분류 모델, 회귀 모델 생성

```
1  from sklearn.tree import DecisionTreeRegressor as DTR
2  model1 = DTC(max_depth = 10)
3  model2 = DTC(max_depth = 10)
4  model3 = DTR(max_depth = 10)
```

- **라인 2 - 3**: 분류 모델의 유형은 학습된 후에 결정되므로 아직까진 model1과 model2는 차이가 없음
- **라인 4**: DecisionTreeRegressor 클래스로 회귀 나무 인스턴스를 생성했습니다.

fit 메서드

fit 메서드는 특징과 라벨 간 관계를 학습하며, fit 메서드를 사용한 다음에 모델을 활용할 수 있습니다.

모델 인스턴스화 예시: 최대 깊이가 10인 결정 나무 모델

```
model1.fit(X1_train, y1_train)
model2.fit(X2_train, y2_train)
model3.fit(X3_train, y3_train)
```

3. 모델 학습 및 평가

2 사이킷런을 이용한 모델 평가

predict 메서드

predict 메서드는 학습한 모델을 사용해 새로 입력된 특징 벡터의 라벨을 예측합니다. 이때 새로 입력된 데이터의 구조는 학습 데이터의 구조와 반드시 같아야 합니다.

predict 메서드 예시

```
1  y1_pred = model1.predict(X1_test)
2  y2_pred = model2.predict(X2_test)
3  y3_pred = model3.predict(X3_test)
4  display(y1_pred[:5])
```

```
array([1, 1, 0, 1, 1], dtype=int64)
```

- **라인 1**: predict는 예측한 라벨을 ndarray로 반환하며 이 배열의 각 요소는 같은 위치에 있는 X1_test의 요소에 대한 예측 결과로, y_pred1[i]는 X1_test[i]를 예측한 결과임

predict_proba 메서드

predict_proba 메서드는 각 샘플이 특정 클래스에 속할 확률을 계산합니다. 출력은 ndarray로 i행 j열 요소는 i번째 샘플이 j번째 클래스에 속할 확률을 나타냅니다.

predict 메서드 예시

```
1  y1_prob = model1.predict_proba(X1_test)
2  y2_prob = model2.predict_proba(X2_test)
3  display(y2_prob[:5])
```

```
array([[0., 0., 1.],
       [0., 0., 1.],
       [1., 0., 0.],
       [1., 0., 0.],
       [1., 0., 0.]])
```

- **라인 1**: predict는 예측한 라벨을 ndarray로 반환하며 이 배열의 각 요소는 같은 위치에 있는 X1_test의 요소에 대한 예측 결과로, y_pred1[i]는 X1_test[i]를 예측한 결과임

이진 분류 모델 평가

분류 모델을 평가하는 데 사용하는 함수로는 metrics 모듈의 accuracy_score, precision_score, recall_score, f1_score 등이 있습니다. 이 모듈에 속한 모든 평가 함수는 순서대로 실제 라벨과 예측한 라벨을 입력받습니다.

이진 분류 모델 평가 예시

```
1  from sklearn.metrics import *
2  acc = accuracy_score(y1_test, y1_pred)
3  pre = precision_score(y1_test, y1_pred)
4  rec = recall_score(y1_test, y1_pred)
5  f1 = f1_score(y1_test, y1_pred)
6  print(acc, pre, rec, f1)
```

```
0.8076923076923077  0.7391304347826086  0.8095238095238095  0.7727272727272727
```

다중 분류 모델 평가

다중 분류 모델은 average 인자를 “micro”, “macro”, “weighted” 등으로 설정해야 합니다.

잘못된 다중 분류 모델 평가 예시

```
1    f1_score(y2_test, y2_pred)
```

```
ValueError                    Traceback (most recent call last)
<ipython-input-42-91a6632949be> in <module>
----> 1 f1_score(y2_test, y2_pred)

(중략)
ValueError: Target is multiclass but average='binary'. Please choose another average setting, one of [None,
'micro', 'macro', 'weighted'].
```

다중 분류 모델 평가 (계속)

다중 분류 모델은 average 인자를 “micro”, “macro”, “weighted” 등으로 설정해야 합니다.

다중 분류 모델 평가 예시

```
1  macro_f1 = f1_score(y2_test, y2_pred, average = "macro")
2  micro_f1 = f1_score(y2_test, y2_pred, average = "micro")
3  print(macro_f1, micro_f1)
```

```
0.9375   0.9473684210526315
```

회귀 모델 평가

MAE와 MSE는 각각 metrics 모듈의 mean_absolute_error와 mean_squared_error 함수로 계산할 수 있습니다.

회귀 모델 평가 예시

```
1 mae = mean_absolute_error(y3_test, y3_pred)
2 mse = mean_squared_error(y3_test, y3_pred)
3 rmse = mse ** 0.5
4 print(mae, mse, rmse)
```

```
1.5658031900382647   3.9832470919211196   1.9958073784614385
```