Bagging (1)

#01. 패키지 참조

```
import warnings
warnings.filterwarnings('ignore')

from pandas import read_excel
from sklearn.ensemble import BaggingClassifier, BaggingRegressor
from sklearn.linear_model import LogisticRegression, LinearRegression

from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score, r2_score
```

#02. 분류 문제

1. 데이터 가져오기

```
origin = read_excel('https://data.hossam.kr/G02/breast_cancer.xlsx')
origin.head()
```

	mean radius	mean texture	mean perimeter	mean area	mean smoothness	mean compactness	mean concavity	mean concave points	me symme
0	17.99	10.38	122.80	1001.0	0.11840	0.27760	0.3001	0.14710	0.2419
1	20.57	17.77	132.90	1326.0	0.08474	0.07864	0.0869	0.07017	0.1812
2	19.69	21.25	130.00	1203.0	0.10960	0.15990	0.1974	0.12790	0.2069
3	11.42	20.38	77.58	386.1	0.14250	0.28390	0.2414	0.10520	0.2597
4	20.29	14.34	135.10	1297.0	0.10030	0.13280	0.1980	0.10430	0.1809

5 rows × 31 columns

2. 데이터 전처리

독립/종속 변수 분리

```
x = origin.drop('target', axis=1)
y = origin['target']
x.shape, y.shape
```

```
((569, 30), (569,))
```

데이터 표준화

```
scaler = StandardScaler()
std_x = scaler.fit_transform(x)
std_x[:1]
```

```
array([[ 1.09706398, -2.07333501,  1.26993369,  0.9843749 ,  1.56846633,
         3.28351467,  2.65287398,  2.53247522,  2.21751501,  2.25574689,
         2.48973393, -0.56526506,  2.83303087,  2.48757756, -0.21400165,
         1.31686157,  0.72402616,  0.66081994,  1.14875667,  0.90708308,
         1.88668963, -1.35929347,  2.30360062,  2.00123749,  1.30768627,
         2.61666502,  2.10952635,  2.29607613,  2.75062224,  1.93701461]])
```

훈련/검증 데이터 분할

```
x_train, x_test, y_train, y_test = train_test_split(
    std_x, y, test_size=0.3, random_state=777)
x_train.shape, x_test.shape, y_train.shape, y_test.shape
```

```
((398, 30), (171, 30), (398,), (171,))
```

3. 분류 모델 구현

분류 알고리즘 객체 정의

```
lr = LogisticRegression()   # KNN, DTREE 등 모든 분류 알고리즘 적용 가능
```

Bagging 모델 구현

```
clf = BaggingClassifier(
    base_estimator=lr,
    n_estimators=50, # 부트스트랩 샘플 개수
    max_samples=1,   # 부트스트랩 샘플 비율 => 1이면 학습데이터를 모두 샘플링한다.
    bootstrap=True,  # 복원 추출,  False이면 비복원 추출
    random_state=777,

    # 하나의 예측기에 들어가는 샘플에 대하여 컬럼의 중복 사용여부를 결정
    bootstrap_features=False,
    n_jobs=-1)

clf.fit(x_train, y_train)
print("BaggingClassifier 훈련 정확도: {:.3f}".format(clf.score(x_train, y_train)))

y_pred = clf.predict(x_test)
print("BaggingClassifier 테스트 정확도: {:.3f}".format(accuracy_score(y_test, y_pred)))
```

```
BaggingClassifier 훈련 정확도: 0.611
BaggingClassifier 테스트 정확도: 0.667
```

#03. 회귀문제

1. 데이터 가져오기

```
origin = read_excel('https://data.hossam.kr/E04/boston.xlsx')
origin.head()
```

	CRIM	ZN	INDUS	CHAS	NOX	RM	AGE	DIS	RAD	TAX	PTRATIO	
0	0.00632	18.0	2.31	0	0.538	6.575	65.2	4.0900	1	296	15.3	396.9
1	0.02731	0.0	7.07	0	0.469	6.421	78.9	4.9671	2	242	17.8	396.9
2	0.02729	0.0	7.07	0	0.469	7.185	61.1	4.9671	2	242	17.8	392.8
3	0.03237	0.0	2.18	0	0.458	6.998	45.8	6.0622	3	222	18.7	394.6
4	0.06905	0.0	2.18	0	0.458	7.147	54.2	6.0622	3	222	18.7	396.9

2. 데이터 전처리

독립/종속변수 분리

```
x = origin.drop('MEDV', axis=1)
y = origin['MEDV']
x.shape, y.shape
```

```
((506, 14), (506,))
```

훈련/검증 데이터 분리

```
x_train, x_test, y_train, y_test = train_test_split(
    x, y, test_size=0.3, random_state=777)
x_train.shape, y_train.shape, x_test.shape, y_test.shape
```

```
((354, 14), (354,), (152, 14), (152,))
```

3. 회귀 모델 구현

회귀 알고리즘 객체 정의

```
rg = LinearRegression()
```

Bagging 모델 구현

```
reg = BaggingRegressor(
    base_estimator=rg,
    n_estimators=50, # 부트스트랩 샘플 개수
```

```
    max_samples=1,   # 부트스트랩 샘플 비율 => 1이면 학습데이터를 모두 샘플링한다.
    bootstrap=True,  # 복원 추출,  False이면 비복원 추출
    random_state=777,

    # 하나의 예측기에 들어가는 샘플에 대하여 컬럼의 중복 사용여부를 결정
    bootstrap_features=False,
    n_jobs=-1)

reg.fit(x_train, y_train)
print("BaggingClassifier 훈련 R2: {:f}".format(reg.score(x_train, y_train)))

y_pred = reg.predict(x_test)
print("BaggingClassifier 테스트 R2: {:f}".format(r2_score(y_test, y_pred)))
```

```
BaggingClassifier 훈련 R2: -0.016774
BaggingClassifier 테스트 R2: -0.010032
```