

13

## 부스팅(회귀)

### [부스팅-회귀개념]

# 배경과 달리 순차적 학습을 진행합니다.

- 과적합을 막기 위해 매개변수를 적절하게 지정해주어야 합니다.

00:00

jupyter 13-7\_부스팅(회귀) (unsaved changes)

File Edit View Insert Cell Kernel Navigate Widgets Help Trusted Python 3 (ipykernel)

1 부스팅 (회귀)

- 예측력이 약한 모형들을 결합하여 강한 예측모형을 만드는 방법
- 부스팅을 병렬로 수행(각 모델을 독립적으로 구축)하는 배경과 달리 순차방식으로 학습을 진행함

- 훈련 단계에서 알고리즘은 각 결과 모델에 가중치를 할당하므로 분류 결과가 좋은 데이터는 높은 가중치를, 분류 결과가 좋지 않은 데이터는 낮은 가중치를 할당받음
- 다음 부스팅단계에서 추출된 학습이 높아짐
- 배경에 비해 모델의 장점을 최적화하고 train 데이터에 대해 오류가 적은 결합모델을 생성할 수 있다는 장점이 있음
- train 데이터에 과적합할 위험이 있음

분류의 경우

### [부스팅-회귀 실습]

02:00

jupyter 13-7\_부스팅(회귀) (unsaved changes)

File Edit View Insert Cell Kernel Navigate Widgets Help Trusted Python 3 (ipykernel)

```
sklearn.ensemble.AdaBoostRegressor(base_estimator = None, n_estimators = 50, learning_rate=1.0)

base_estimator: 부스팅에서 수행할 분류기 (None이면 DecisionTreeClassifier/DecisionTreeRegressor를 수행)
n_estimators: 부스팅이 종료되는 최대 분류기의 수 (int, default=50)
```

```
In [32]: import pandas as pd
df = pd.read_csv('../data/kc_house_data.csv')
df = df.drop(['id', 'date'], axis=1)

X = df.drop('price', axis=1)
y = df['price']

X = pd.get_dummies(data = X, columns=['waterfront'])

from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=2021)
```

```
In [5]: from sklearn.ensemble import AdaBoostRegressor # Ada 부스팅
reg = AdaBoostRegressor()
reg = reg.fit(X_train, y_train)
y_pred = reg.predict(X_test)
```

```
In [8]: reg.get_params()
Out[8]: {'base_estimator': None,
         'learning_rate': 1.0,
```

### [과소적합을 해결하기 위한 그리드서치]

03:19

jupyter 13-7\_부스팅(회귀) (unsaved changes) Logout

File Edit View Insert Cell Kernel Navigate Widgets Help Trusted Python 3 (ipykernel)

```

In [3]: reg.get_params()
Out[3]: {'base_estimator': None,
        'learning_rate': 1.0,
        'loss': 'linear',
        'n_estimators': 50,
        'random_state': None}

In [4]: print("train 정확도 : ", reg.score(X_train, y_train))
        print("test 정확도 : ", reg.score(X_test, y_test))
train 정확도 : 0.317637322546489
test 정확도 : 0.284365990104997

In [5]: np.arange(0.01, 0.1, 0.02)
NameError                                Traceback (most recent call last)
~\AppData\Local\Temp\ipykernel_4700\1583018696.py in <module>
----> 1 np.arange(0.01, 0.1, 0.02)

NameError: name 'np' is not defined

In [26]: from sklearn.model_selection import GridSearchCV
import numpy as np
param_grid = {'learning_rate': np.arange(0.01, 0.1, 0.02)}
model_grid_boost = GridSearchCV(AdaBoostRegressor(), param_grid, cv=5)
model_grid_boost.fit(X_train, y_train)

Out[26]: GridSearchCV(cv=5, estimator=AdaBoostRegressor(n_estimators=100),
                    param_grid={'learning_rate': array([0.01, 0.03, 0.05, 0.07, 0.09])})

```