

15

인공신경망(회귀)

[인공신경망 회귀분석_실습]

#이번 시간에는 hidden_layer_sizes를 조정하여 다층퍼셉트론 회귀분석으로 학습해보겠습니다.

00:00

```

jupyter 15-5_인공신경망(회귀) (autosaved)

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

15.5 인공신경망(회귀)_실습

[예제] kc_house_data 데이터의 train 데이터로 price를 예측하는 인공신경망 모델 만들기

In [1]: 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 [2]: from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
X_train_scaled = pd.DataFrame(scaler.fit_transform(X_train), columns=X_train.columns)
X_test_scaled = pd.DataFrame(scaler.transform(X_test), columns=X_train.columns)

In [4]: from sklearn.neural_network import MLPRegressor
mlpr = MLPRegressor()
mlpr.fit(X_train_scaled, y_train)

C:\Users\jang\Anaconda3\envs\MLP_Class\lib\site-packages\sklearn\neural_network\_multilayer_perceptron.py:617: ConvergenceWarning: Stochastic Optimizer: Maximum iterations (200) reached and the optimization hasn't converged yet.
% self.max_iter, ConvergenceWarning)

```

[인공신경망 회귀분석_실습]

#다층 퍼셉트론 학습을 하기 위한 매개변수 설정

02:10

```

jupyter 15-5_인공신경망(회귀) (unsaved changes)

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

C:\Users\jang\Anaconda3\envs\MLP_Class\lib\site-packages\sklearn\neural_network\_multilayer_perceptron.py:617: ConvergenceWarning: Stochastic Optimizer: Maximum iterations (200) reached and the optimization hasn't converged yet.
% self.max_iter, ConvergenceWarning)

Out[3]: MLPRegressor()

In [4]: print("train data R2 : ", mlpr.score(X_train_scaled, y_train))
print("test data R2 : ", mlpr.score(X_test_scaled, y_test))

train data R2 : -1.2479688921978869
test data R2 : -1.2394995907025992

In [8]: mlpr = MLPRegressor(hidden_layer_sizes=(64, 64, 64, 64), activation='relu', max_iter=1000, random_state=2021)
mlpr.fit(X_train_scaled, y_train)

Out[8]: MLPRegressor(hidden_layer_sizes=(64, 64, 64, 64), max_iter=1000, random_state=2021)

In [9]: print("train data R2 : ", mlpr.score(X_train_scaled, y_train))
print("test data R2 : ", mlpr.score(X_test_scaled, y_test))

train data R2 : 0.8040855205565576
test data R2 : 0.7462992409208402

In [12]: mlpr.coef_[0][1:]

Out[12]: array([-0.1688061,  0.05893822, -0.37564003,  0.04167213, -0.00292468,
                -0.04128374,  0.01470665,  0.08436326,  0.09713633, -0.33673569,
                -0.06519876, -0.1914487,  0.06291696,  0.20230849, -0.21202029,
                0.04819812,  0.00919401,  0.36065995, -0.11779025, -0.15627477,
                0.28452225,  0.04353277,  0.03580989,  0.45141057,  0.00162786,
                -0.361352,  0.17931096,  0.07106107,  0.04838185, -0.10301388,
                0.0696264, -0.11077809,  0.08570706, -0.20538489,  0.07668678,
                -0.03455234,  0.14784271,  0.01936299, -0.10131567,  0.25462604,
                0.04486164, -0.09219818, -0.30425389,  0.09340591,  0.11362098,
                -0.14354025, -0.35052719,  0.43308387, -0.18418875, -0.03668598,
                0.14868336,  0.4600734,  0.04613761,  0.0068348,  0.00787454])

```