Scaler

PinkLAB Edu

3 November, 2025

Table of Contents

1 label_encoder 4
1.1 간단한 데이터 하나~ 4
1.2 이렇게 생김 4
1.3 label encoder란? 4
1.4 대상이 되는 문자로 된 데이터를 5
1.5 숫자 - 카테고리컬한 데이터로~ 5
1.6 fit 한 후에 transform을 하면 변환됨 5
1.7 한 번에 줄이는 것은 fit_transform 5
1.8 혹시 역변환하고 싶다면 5
2 min-max scaling 6
2.1 min-max scaling이란? 6
2.2 역시 데이터 준비하고 6
2.3 fit~ 6
2.4 그러면 이런 요소를 찾게 된다 7
2.5 transform 시키면 이렇게 7
2.6 역변환 시키면 이렇게 7
2.7 한번에는 이렇게~^^ 7
3 Standard Scaler 8
3.1 표준정규분포 8
3.2 아까부터 사용하던 원 데이터 8
3.3 이번에도 fit~ 8
3.4 평균과 표준편차 8
3.5 transform 9
3.6 fit_transform 9
3.7 inverse_transform 9
4 Robust Scaler 10
4.1 Robust Scaler 10
4.2 데이터를 새롭게… 10
4.3 방금 배운 것 포함 Robust 스케일러까지 10

4.4 이번에는 모두 불러와서 10
4.5 스케일링 된 결과..... 11
4.6 각 스케일러의 차이와 그 중 Robust 스케일러의 특징을 보자 11

1 label_encoder

1.1 간단한 데이터 하나~

```python
import pandas as pd
```

```python
df = pd.DataFrame( {
    'A' : ['a', 'b', 'c', 'a', 'b'],
    'B' : [1, 2, 3, 1, 0]
})

df
```

1.2 이렇게 생김

	A	B
0	a	1
1	b	2
2	c	3
3	a	1
4	b	0

1.3 label encoder란?

```python
from sklearn.preprocessing import LabelEncoder

le = LabelEncoder()
le.fit(df['A'])
```

```
LabelEncoder()
```

1.4 대상이 되는 문자로 된 데이터를

```python
le.classes_
```

```
array(['a', 'b', 'c'], dtype=object)
```

1.5 숫자 - 카테고리컬한 데이터로~

```python
df['le_A'] = le.transform(df['A'])
df
```

	A	B	le_A
0	a	1	0
1	b	2	1
2	c	3	2
3	a	1	0
4	b	0	1

1.6 fit 한 후에 transform을 하면 변환됨

```python
le.transform(['a' , 'b'])
```

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

1.7 한 번에 줄이는 것은 fit_transform

```python
le.fit_transform(df["A"])
```

```
array([0, 1, 2, 0, 1])
```

1.8 혹시 역변환하고 싶다면

```python
le.inverse_transform([1, 2, 2, 2])
```

```
array(['b', 'c', 'c', 'c'], dtype=object)
```

2 min-max scaling

2.1 min-max scaling이란?

$$x' = \frac{x - \min(x)}{\max(x) - \min(x)}$$

2.2 역시 데이터 준비하고

```python
df = pd.DataFrame( {
    'A' : [10, 20 , -10, 0, 25],
    'B' : [1 ,2 ,3, 1, 0]
})

df
```

Python

	A	B
0	10	1
1	20	2
2	-10	3
3	0	1
4	25	0

2.3 fit~

```python
from sklearn.preprocessing import MinMaxScaler

mms = MinMaxScaler()
mms.fit(df)
```

Python

```
MinMaxScaler()
```

2.4 그러면 이런 요소를 찾게 된다

```python
mms.data_max_, mms.data_min_
```

```
(array([25.,  3.]), array([-10.,   0.]))
```

2.5 transform 시키면 이렇게

```python
df_mms = mms.transform(df)
df_mms
```

```
array([[0.57142857, 0.33333333],
       [0.85714286, 0.66666667],
       [0.        , 1.        ],
       [0.28571429, 0.33333333],
       [1.        , 0.        ]])
```

2.6 역변환 시키면 이렇게

```python
mms.inverse_transform(df_mms)
```

```
array([[ 10.,   1.],
       [ 20.,   2.],
       [-10.,   3.],
       [  0.,   1.],
       [ 25.,   0.]])
```

2.7 한번에는 이렇게~^^

```python
mms.fit_transform(df_mms)
```

```
array([[0.57142857, 0.33333333],
       [0.85714286, 0.66666667],
       [0.        , 1.        ],
       [0.28571429, 0.33333333],
       [1.        , 0.        ]])
```

3 Standard Scaler

3.1 표준정규분포

$$z = \frac{X - \mu}{\sigma}$$

3.2 아까부터 사용하던 원 데이터

```
df
```

Python

	A	B
0	10	1
1	20	2
2	-10	3
3	0	1
4	25	0

3.3 이번에도 fit~

```
from sklearn.preprocessing import StandardScaler

ss = StandardScaler()
ss.fit(df)
```

Python

```
StandardScaler()
```

3.4 평균과 표준편차

```
ss.mean_, ss.scale_
```

Python

```
(array([9. , 1.4]), array([12.80624847,  1.0198039 ]))
```

3.5 transform

```python
df_ss = ss.transform(df)
df_ss
```

```
array([[ 0.07808688, -0.39223227],
       [ 0.85895569,  0.58834841],
       [-1.48365074,  1.56892908],
       [-0.70278193, -0.39223227],
       [ 1.2493901 , -1.37281295]])
```

3.6 fit_transform

```python
ss.fit_transform(df)
```

```
array([[ 0.07808688, -0.39223227],
       [ 0.85895569,  0.58834841],
       [-1.48365074,  1.56892908],
       [-0.70278193, -0.39223227],
       [ 1.2493901 , -1.37281295]])
```

3.7 inverse_transform

```python
ss.inverse_transform(df_ss)
```

```
array([[ 10.,   1.],
       [ 20.,   2.],
       [-10.,   3.],
       [  0.,   1.],
       [ 25.,   0.]])
```

4 Robust Scaler

4.1 Robust Scaler

$$\frac{x_i - Q_2}{Q_3 - Q_1}$$

4.2 데이터를 새롭게...

```python
df = pd.DataFrame({
    'A' : [-0.1, 0., 0.1, 0.2, 0.3, 0.4, 1.0, 1.1, 5]
})
```

Python

4.3 방금 배운 것 포함 Robust 스케일러까지

```python
from sklearn.preprocessing import MinMaxScaler, StandardScaler, RobustScaler

mm = MinMaxScaler()
ss = StandardScaler()
rs = RobustScaler()
```

Python

4.4 이번에는 모두 불러와서

```python
df_scaler = df.copy()
```

Python

```python
df_scaler["MinMax"] = mm.fit_transform(df)
df_scaler["Standard"] = ss.fit_transform(df)
df_scaler["Robust"] = rs.fit_transform(df)
```

Python

4.5 스케일링 된 결과

```
df_scaler
```
Python

	A	MinMax	Standard	Robust
0	-0.1	0.000000	-0.656688	-0.444444
1	0.0	0.019608	-0.590281	-0.333333
2	0.1	0.039216	-0.523875	-0.222222
3	0.2	0.058824	-0.457468	-0.111111
4	0.3	0.078431	-0.391061	0.000000
5	0.4	0.098039	-0.324655	0.111111
6	1.0	0.215686	0.073785	0.777778
7	1.1	0.235294	0.140192	0.888889
8	5.0	1.000000	2.730051	5.222222

4.6 각 스케일러의 차이와 그 중 Robust 스케일러의 특징을 보자

```
import seaborn as sns
import matplotlib.pyplot as plt

sns.set_theme(style = 'whitegrid')
plt.figure(figsize = (16, 6))
sns.boxplot(data = df_scaler, orient = "h");
```
✓ 0.1s Python