데이터 정규화

데이터를 특정 범위나 척도로 변환하여 처리하거나 분석할 때 사용되는 기술

데이터 정규화의 목표는 서로 다른 단위나 범위를 가진 데이터를 동일한 기준으로 맞춤으로써, 데이터 분석이나 머신러닝 모델의 성능을 향상시키는 것

#01. 작업준비

패키지 참조

```
from pandas import read_excel
from sklearn.preprocessing import MinMaxScaler, StandardScaler, RobustSc
```

데이터 가져오기

```
df = read_excel('https://data.hossam.kr/D05/gradeuate.xlsx')
df
```

	합격여부	필기점수	학부성적	병원경력
0	0	380	3.61	3
1	1	660	3.67	3

	합격여부	필기점수	학부성적	병원경력
2	1	800	4.00	1
3	1	640	3.19	4
4	0	520	2.93	4
...	...	...	...	...
395	0	620	4.00	2
396	0	560	3.04	3
397	0	460	2.63	2
398	0	700	3.65	2
399	0	600	3.89	3

400 rows × 4 columns

#02. Min-Max Scaler (Normalization, 정규화)

모든 데이터의 범위를 `0~1`로 변환하는 것.

데이터에서 최소값을 0으로, 최대값을 1로 매핑

$$\text{정규화된값} = (X - Xmin)/(Xmax - Xmin)$$

이 방법은 데이터의 분포를 유지하면서 데이터를 특정 범위로 축소시키는 데에 유용

직접 계산

```
Xmin = df['필기점수'].min()
Xmax = df['필기점수'].max()
df['필기점수_MinMax(1)'] = (df['필기점수'] - Xmin) / (Xmax - Xmin)
df
```

	합격여부	필기점수	학부성적	병원경력	필기점수_MinMax(1)
0	0	380	3.61	3	0.275862
1	1	660	3.67	3	0.758621
2	1	800	4.00	1	1.000000
3	1	640	3.19	4	0.724138
4	0	520	2.93	4	0.517241
...	...	...	...	...	...
395	0	620	4.00	2	0.689655
396	0	560	3.04	3	0.586207
397	0	460	2.63	2	0.413793
398	0	700	3.65	2	0.827586
399	0	600	3.89	3	0.655172

400 rows × 5 columns

파이썬 활용

데이터 정규화

#01. 작업준비

패키지 참조

데이터 가져오기

#02. Min-Max Scaler (Normalization, 정규화)

직접 계산

파이썬 활용

#03. 표준화 (StandardScaler), z-score

그래서 어쩌라구?

직접 계산

파이썬 스타일

#04. RobustScaler

직접계산

파이썬 스타일

```
# 표준화 기능을 제공하는 객체를 생성
scaler = MinMaxScaler()

# 표준화를 적용할 필드를 scaler 객체에게 알려준다.
scaler.fit(df[['필기점수']])

# 표준화 적용
df['필기점수_MinMax(2)'] = scaler.transform(df[['필기점수']])

df
```

	합격여부	필기점수	학부성적	병원경력	필기점수_MinMax(1)	필기점수_MinMax(2)
0	0	380	3.61	3	0.275862	0.275862
1	1	660	3.67	3	0.758621	0.758621
2	1	800	4.00	1	1.000000	1.000000
3	1	640	3.19	4	0.724138	0.724138
4	0	520	2.93	4	0.517241	0.517241
...	...	...	...	...	...	...
395	0	620	4.00	2	0.689655	0.689655
396	0	560	3.04	3	0.586207	0.586207
397	0	460	2.63	2	0.413793	0.413793
398	0	700	3.65	2	0.827586	0.827586

	합격여부	필기점수	학부성적	병원경력	필기점수_MinMax(1)	필기점수_MinMax(2)
399	0	600	3.89	3	0.655172	0.655172

400 rows × 6 columns

#03. 표준화 (StandardScaler), z-score

데이터를 평균이 `0`, 표준편차가 `1`인 표준정규분포를 따르도록 변환

$$\text{정규화된값} = (X - \text{평균}) / \text{표준편차}$$

데이터를 정규분포에 근사시켜서 이상치에 덜 민감하게 만들어 줌

그래서 어쩌라구?

- 값들의 단위가 비슷하다면 MinMax
- 값들의 단위가 상이하다면 Standard
- 잘 모르겠으면 Standard

> 분류 문제에서는 종속변수가 범주형(0, 1)이므로 종속변수는 표준화를 적용하지 않는다.

직접 계산

```
평균 = df['학부성적'].mean()
표준편차 = df['학부성적'].std()
df['학부성적_Standard(1)'] = (df['학부성적'] - 평균) / 표준편차
df
```

	합격여부	필기점수	학부성적	병원경력	필기점수_MinMax(1)	필기점수_MinMax(2)	학부성적_Standard(1)
0	0	380	3.61	3	0.275862	0.275862	0.578348
1	1	660	3.67	3	0.758621	0.758621	0.736008
2	1	800	4.00	1	1.000000	1.000000	1.603135
3	1	640	3.19	4	0.724138	0.724138	-0.525269
4	0	520	2.93	4	0.517241	0.517241	-1.208461
...	...	...	...	...	...	...	...
395	0	620	4.00	2	0.689655	0.689655	1.603135
396	0	560	3.04	3	0.586207	0.586207	-0.919418
397	0	460	2.63	2	0.413793	0.413793	-1.996758
398	0	700	3.65	2	0.827586	0.827586	0.683455
399	0	600	3.89	3	0.655172	0.655172	1.314093

400 rows × 7 columns

파이썬 스타일

```
scaler = StandardScaler()
#scaler.fit(df[['학부성적']])
#df['학부성적_Standard(2)'] = scaler.transform(df[['학부성적']])
```

```
df['학부성적_Standard(2)'] = scaler.fit_transform(df[['학부성적']])
df
```

	합격여부	필기점수	학부성적	병원경력	필기점수_MinMax(1)	필기점수_MinMax(2)	학부성적_Standard(1)	학부성적_Standard(2)
0	0	380	3.61	3	0.275862	0.275862	0.578348	0.579072
1	1	660	3.67	3	0.758621	0.758621	0.736008	0.736929
2	1	800	4.00	1	1.000000	1.000000	1.603135	1.605143
3	1	640	3.19	4	0.724138	0.724138	-0.525269	-0.525927
4	0	520	2.93	4	0.517241	0.517241	-1.208461	-1.209974
...	...	...	...	...	...	...	...	...
395	0	620	4.00	2	0.689655	0.689655	1.603135	1.605143
396	0	560	3.04	3	0.586207	0.586207	-0.919418	-0.920570
397	0	460	2.63	2	0.413793	0.413793	-1.996758	-1.999259
398	0	700	3.65	2	0.827586	0.827586	0.683455	0.684310
399	0	600	3.89	3	0.655172	0.655172	1.314093	1.315739

400 rows × 8 columns

#04. RobustScaler

이상치가 존재할 경우 사용하는 방법.

이상치(outliers)에 영향을 최소화하여 데이터를 스케일링하는 방법

이상치가 포함된 데이터를 표준화(Standardization)하거나 정규화(Normalization)할 때, 이상치의 영향으로 전체 데이터의 분포가 왜곡됨

RobustScaler는 이 문제를 해결하기 위해 중앙값과 사분위수를 사용하여 데이터를 스케일링 함

$(X - median)/iqr$

직접계산

```
중앙값 = df['병원경력'].median()
iqr = df['병원경력'].quantile(0.75) - df['병원경력'].quantile(0.25)
df['병원경력_Robust(1)'] = (df['병원경력'] - 중앙값) / iqr
df
```

	합격여부	필기점수	학부성적	병원경력	필기점수_MinMax(1)	필기점수_MinMax(2)	학부성적_Standard(1)	학부성적_Standard(2)
0	0	380	3.61	3	0.275862	0.275862	0.578348	0.579072
1	1	660	3.67	3	0.758621	0.758621	0.736008	0.736929
2	1	800	4.00	1	1.000000	1.000000	1.603135	1.605143
3	1	640	3.19	4	0.724138	0.724138	-0.525269	-0.525927
4	0	520	2.93	4	0.517241	0.517241	-1.208461	-1.209974

	합격여부	필기점수	학부성적	병원경력	필기점수_MinMax(1)	필기점수_MinMax(2)	학부성적_Standard(1)	학부성적_Standard(2)
...	...	...	...	...	...	...	...	...
395	0	620	4.00	2	0.689655	0.689655	1.603135	1.605143
396	0	560	3.04	3	0.586207	0.586207	-0.919418	-0.920570
397	0	460	2.63	2	0.413793	0.413793	-1.996758	-1.999259
398	0	700	3.65	2	0.827586	0.827586	0.683455	0.684310
399	0	600	3.89	3	0.655172	0.655172	1.314093	1.315739

400 rows × 9 columns

파이썬 스타일

```
scaler = RobustScaler()
scaler.fit(df[['병원경력']])
df['병원경력_Robust(2)'] = scaler.transform(df[['병원경력']])
df
```

데이터 정규화

#01. 작업준비

패키지 참조

데이터 가져오기

#02. Min-Max Scaler (Normalization, 정규화)

직접 계산

파이썬 활용

#03. 표준화 (StandardScaler), z-score

그래서 어쩌라구?

직접 계산

파이썬 스타일

#04. RobustScaler

직접계산

파이썬 스타일

	합격여부	필기점수	학부성적	병원경력	필기점수_MinMax(1)	필기점수_MinMax(2)	학부성적_Standard(1)	학부성적_Standard(2)
0	0	380	3.61	3	0.275862	0.275862	0.578348	0.579072
1	1	660	3.67	3	0.758621	0.758621	0.736008	0.736929
2	1	800	4.00	1	1.000000	1.000000	1.603135	1.605143
3	1	640	3.19	4	0.724138	0.724138	-0.525269	-0.525927
4	0	520	2.93	4	0.517241	0.517241	-1.208461	-1.209974
...	...	...	...	...	...	...	...	...
395	0	620	4.00	2	0.689655	0.689655	1.603135	1.605143
396	0	560	3.04	3	0.586207	0.586207	-0.919418	-0.920570
397	0	460	2.63	2	0.413793	0.413793	-1.996758	-1.999259
398	0	700	3.65	2	0.827586	0.827586	0.683455	0.684310
399	0	600	3.89	3	0.655172	0.655172	1.314093	1.315739

400 rows × 10 columns