데이터프레임 구조

강창현 | aaakch0316@gmail.com

데이터프레임 구조 확인_p84

> 데이터를 받자마자 아래의 코드를 친다.

- df.head() **
- df.tail() **
- df.shape : (행, 열)
- df.info() : 자료형 확인, NaN값 확인, 갯수 **
- df.dtypes : 자료형 확인
- df.describe() : 산술데이터(평균, 표준편차, 최대값, 최소값, 중간값 등)
- df.describe(include='all') : 문자열데이터 : unique(고유값 갯수), top(최빈값), freq(빈도수)

테이터프레임 구조 확인 코드

```
import pandas as pd
df = pd.read_csv('./sample.csv', header=None)
df.columns = ['A', 'B', 'C', 'D']
df.head() or df.tail()
df.shape
df.info()
df.describe()
```

데이터 개수 확인

- count()
- df['열'].value_counts() : df의 특정 열이 가지고 있는 고유값 확인
- value_counts() = value_counts(dropna=False) // NaN값도 포함.
- value_counts(dropna=True) // NaN값 제외 ***

통계 함수 적용

- df.mean() # 평균
- df.median() # 중간값
- df.max() # 최대값
- df.min() # 최소값
- df.std() # 표준편차
- df.corr() # 상관계수를 계산 (-1 ~ 1)
- df[['mpg', 'weight']].corr()

실습

```
import seaborn as sns
df = sns.load_dataset('titanic')
```

- 데이터 구조확인
- 어제 배운내용 확인
- 코로나 데이터 확인

변수간 스케일 조정(scaling)

> 1에서 1000 사이의 값이 있다면 컴퓨터는 큰값이 더 중요하다고 생각 할 수 있다. 따라서 스케일을 동일하게 맞춰서 비교하는 것이 필요하다.

스케일의 네 가지 방법

Standard Scaler: (정규분포를 따를 때만 써야한다.)평균치를 제거하고 표준편차로 나눠서 스케일을 다 맞추는 것이다.

```
sklearn.preprocessing.StandardScaler().fit()
sklearn.preprocessing.StandardScaler().transform()
sklearn.preprocessing.StandardScaler().fit_transform()
```

Min-Max Scaler: 모습은 그대로 가져가면서 scale만 맞춘다. 즉, 정규분포가 아니거나 표준편차가 매우 작을 때 효과적이다. 그러나 이 알고리즘의 치명적인 단점은 outlier가 있을 경우이다. max값으로 하나가 너무 크다면 전체 값 자체가 너무 작아진다. 이러한 문제점을 피하고자 아래의 Robust Scaler알고리즘을 만들었다.

```
sklearn.preprocessing.MinMaxScaler().fit()
sklearn.preprocessing.MinMaxScaler().transform()
sklearn.preprocessing.MinMaxScaler().fit_transform()
```

Robust Scaler: 최소-최대 스케일러와 유사하지만 최소/최대 대신에 IQR(Interquartile Range) 중 25%값/75%값을 사용하여 변환한다. 이상치(Outlier)에 영향을 최소화하였기에 이상치가 있는 데이터에 효과적이고 적은 데이터에도 효과적인 편이다.

```
sklearn.preprocessing.RobustScaler().fit()
sklearn.preprocessing.RobustScaler().transform()
sklearn.preprocessing.RobustScaler().fit_transform()
```

Normalizer: 모든 X값의 제곱 합을 루트로 나눠줌으로서 반지름 1인 원 내부의 범위로 바꿔준다. 각 변수들의 값은 원점으로부터 반지름 1만큼 떨어진 범위 내로 변환된다.

```
sklearn.preprocessing.Normalizer().fit()
sklearn.preprocessing.Normalizer().transform()
sklearn.preprocessing.Normalizer().fit_transform()
```