

# 데이터 전처리 p171

강창현 | [aaakch0316@gmail.com](mailto:aaakch0316@gmail.com)

# 목차

| 전처리의 시작은 누락데이터(NaN) 제거에서 시작한다.

1. NaN 확인
2. NaN 제거 및 치환
3. 중복 데이터 확인
4. 중복 데이터 제거

# 1단계\_누락 데이터 처리\_P172

- head()
- info() // NaN값 확인
  - df['열'].value\_counts(dropna=False) # unique한 값으로 NaN값의 개수도 포함된다.
  - df.isnull() # NaN을 True 반환
  - df.notnull() # NaN을 False 반환
  - df.isnull().sum(axis=0) # 각 열의 True를 합한다.

```
# for 반복문으로 각 열의 NaN 개수 계산하기
missing_df = df.isnull()
for col in missing_df.columns:
    missing_count = missing_df[col].value_counts()
    try:
        print(col, ':', missing_count[True])
    except:
        print(col, ':', 0)
```

## 2단계\_누락 데이터 제거

CASE1 : 누락데이터가 80%이상인 열을 찾았다. 선택하시오!!!!!! 열을 지운다 vs 안지운다

### dropna 메소드

```
# NaN 값이 500개 이상인 열을 모두 삭제  
df_thresh = df.dropna(axis=1, thresh=500)
```

CASE2 : 중요한 변수의 열에 NaN값이 30%가 있다. NaN을 지운다 vs NaN을 다른 값을 채운다.

`dropna(subset=[ ], how='any', axis=0)`

- `how='any'` : (디폴트 값) NaN 값이 어느 하나라도 존재하면 삭제한다.
- `subset`을 하나의 열만 넣으면, `how='any'`에 의해서 열 중에서 NaN 값이 있는 모든 행(`axis=0`)을 삭제한다.
- `how='all'` : 모든 데이터가 NaN 값일 경우에만 삭제 한다.

```
# 특정 열에 NaN값이 있는 모든 행을 삭제한다.
```

```
df_na = df.dropna(subset=['A'], how='any', axis=0)
```

## 2단계\_누락 데이터 치환

평균값, 최빈값, 중간값

CASE1 : 데이터의 품질을 올리기 위해 누락 데이터를 삭제했다. good!!!! vs bad!!!!

CASE2 : 데이터의 품질을 포기하고 누락 데이터를 최대한 살렸다. good!!! vs bad!!!

`fillna()` 메소드

- `inplace=True` 옵션을 추가해야 원본 객체를 변경한다.

CASE3 : 데이터가 너무 없다. 누락 데이터를 어떻게 바꿀까? 평균값 vs 최빈값 vs 중간값

부동산, 키, 이상치 탐지(outlier data detection//IQR)

`mean()` 메소드 , `median()` 매소드

```
# 평균 나이 계산하기  
mean_age = df['age'].mean(axis=0)  
df['age'].fillna(mean_age, inplace=True)  
  
# 중간값  
median_age = df['age'].median(axis=0)  
df['age'].fillna(median_age, inplace=True)
```

`value_counts()` 메소드 , `idxmax()` 메소드

```
# 최빈값  
most_freq = df['A'].value_counts(dropna=True).idxmax()  
df['A'].fillna(most_freq, inplace=True)
```

```
df['A'].fillna(method='ffill', inplace=True)
```

## 중복데이터 확인\_p184

동일한 대상이 중복되는 경우에 중복 데이터를 찾아서 삭제 해야한다.

`df.duplicated()` 메소드

`df['열'].duplicated()` 메소드

- 이전과 중복이면 True
- 처음나오면 False
- 0행값은 False

# 중복데이터 제거

`df.drop_duplicates()` 메소드

- 0행과 1행이 중복된다면, 아래 값인 1행이 삭제된다.

`df.drop_duplicates(subset=['A', 'B'])` 메소드

- A,B열을 기준으로 중복된 값을 제거한다.

# 실습

기초3\_문제