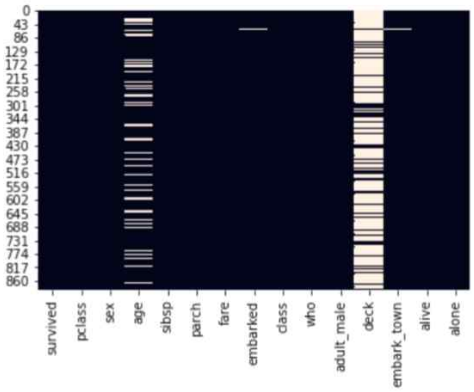
결측값을 시각화하는 6가지 방법

1. Seaborn 라이브러리를 이용한 결측값 시각화

1.1 sns.heatmap(데이터셋.isnull(), cbar=False)

```
In [1]: #결측값시각화
        import pandas as pd
        import seaborn as sns
        import matplotlib.pyplot as plt
        import numpy as np

        df=sns.load_dataset('titanic')

        sns.heatmap(df.isnull(), cbar=False)
```

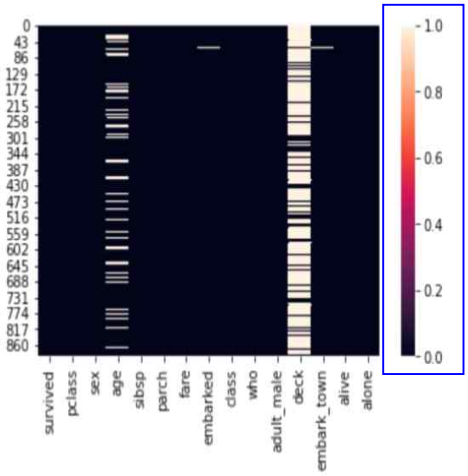
```
Out[1]: <AxesSubplot:>
```



- 위 heatmap을 통해 알 수 있는 정보는 아래와 같습니다.
- 1) 'age' 컬럼은 변동성이 있는 결측치가 듬성듬성 있다.
- 2) 'embarked', 'alive' 컬럼은 초기 행에 소수의 결측치가 있다.
- 3) 'deck' 컬럼은 거의 모든 열이 결측치로 구성되어 있다.
- 'embarked'와 'embark_town' 컬럼을 보면, 동일한 위치에 결측치가 형성되어있음을 볼 수 있는데, 이를 통해 '둘 사이 관련성의 존재'에 대한 가능성을 엿볼 수 있습니다. 이처럼 결측치가 있는 2개 이상의 열에 상관관계가 있으면 heatmap을 통해 직관적으로 파악할 수 있다는 점이 장점입니다.

1.2 sns.heatmap(데이터셋.isnull(), cbar=True)

```
In [2]: #결측값시각화
        sns.heatmap(df.isnull(), cbar=True)
Out[2]: <AxesSubplot:>
```

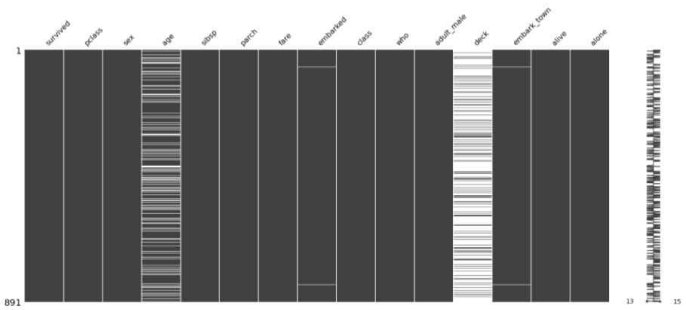


- 위와 다르게 cbar=True 옵션을 통해 각 색깔이 어느 정도의 값을 가지는지 확인할 수 있지만, 본 데이터셋에서는 특정 행이 중복되지 않기 때문에 모두 같은 색으로 표시되었습니다.

2. Missingno 라이브러리를 이용한 결측값 시각화

2.1 msno.matrix(데이터셋)

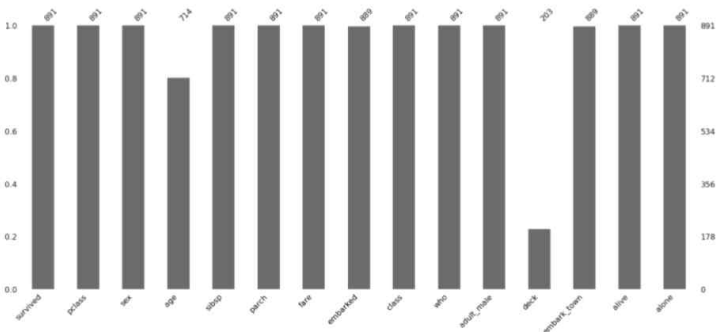
```
In [3]: #결측값 시각화2 #missingno 라이브러리 설치
        import missingno as msno

        msno.matrix(df)
Out[3]: <AxesSubplot:>
```



- 데이터가 방대한 경우, msno.matrix(데이터셋.sample(n))처럼 샘플링 개수를 지정해 요약한 데이터를 시각화할 수 있습니다. 또한, 이 그래프는 최대 50개의 레이블이 지정된 열만을 요약해서 표시할 수 있기 때문에 그 이상의 경우 적절한 조정 과정을 거쳐야 합니다.
- 시계열 데이터(시간에 따라서 달라지는 데이터)를 다룰 경우, 추가로 시간의 주기성(년, 월 단위 등)을 지정할 수 있는 키워드 매개 변수 옵션이 존재합니다(본 자료에서는 다루지 않을 예정입니다).

2.2 msno.bar(데이터셋)

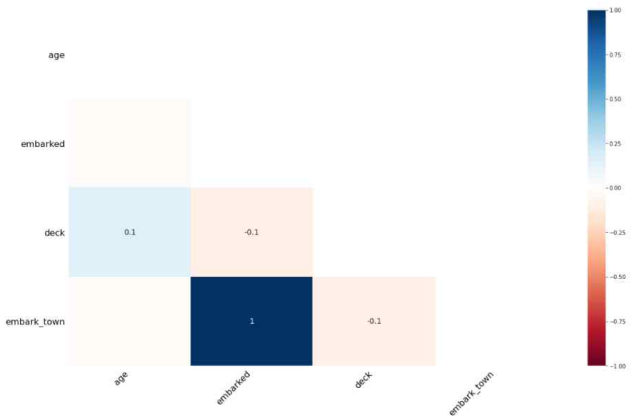
```
In [4]: #결측값 시각화3
        msno.bar(df)
Out[4]: <AxesSubplot:>
```



- 각 열의 결측치가 아닌 유효값의 수를 보여주는 막대 그래프입니다.
- 결측치가 많은 'deck'칼럼의 경우, 유의미한 데이터 개수가 203개임을 파악할 수 있습니다.
- matrix가 '결측치가 많은 컬럼이 어디인가?'를 직관적으로 보여줬다면, bar는 '유효값이 가장 적은 컬럼이 어디인가?'를 직관적으로 보여준다고 할 수 있습니다.

2.3 msno.heatmap(데이터셋)

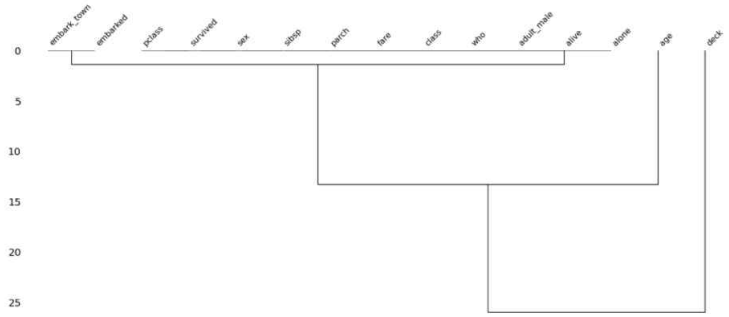
```
In [5]: #결측값 시각화4
        msno.heatmap(df)
Out[5]: <AxesSubplot:>
```



- 각 열에 결측치가 없거나 아예 비어있는 변수들(전부 NaN인 경우)은 유의미한 상관관계를 나타내기 힘들기 때문에, 자동으로 제외한 채 그려집니다. 위에서는 볼 수 없지만, 규모가 더 크고 결측치가 많은 데이터셋에서는 <1, >-1 등으로 표시되기도 합니다(상관관계가 0.243등 세밀하게 나타나는 경우 등이 있으며, 여기서 +/-의 표시가 긍정/부정의 의미는 아닙니다).
- msno.matrix에서 짐작했던 것처럼, 'embarked'와 'embark_town' 컬럼의 상관관계가 1로 상당히 높게 나타났음을 확인할 수 있습니다.
- 이처럼 msno.heatmap은 변수 간 데이터 상관관계를 파악해 특정 변수를 선택할 때는 효과적이지만, 이보다 더 큰 관계를 나타내기엔 설명력이 부족하고, 방대한 데이터셋의 경우 적용하기 어려운 단점이 있습니다.

2.4 msno.dendrogram(데이터셋)

```
In [6]: #결측값 시각화5
        msno.dendrogram(df)
Out[6]: <AxesSubplot:>
```



- 계층적 클러스터링 알고리즘을 활용해 두 그룹을 중심으로 표현한 결측값 시각화 방법입니다.
- 트리의 각 단계에서 나머지 클러스터의 거리를 최소화하는(근접한) 조합에 따라 변수들을 분할합니다. 여기에선 ('embark_town', 'embarked'), ('pclass', 'survived', 'sex', 'sibsp', 'parch', 'fare', 'class', 'who', 'adult_male', 'alive', 'alone'), ('age'), ('deck') 조합으로 묶여있습니다.
- 위에서부터 아래로 읽어내며, 변수 집합이 단조로울수록 전체 거리가 0에 가깝고, 평균 거리(y축)가 0에 더 가깝습니다.
- 0의 위치에 함께 묶인 조합은 서로의 존재를 완전히 예측하는, 즉 관련성이 높은 조합임을 의미합니다. 아래로 내려갈수록 서로를 잘 예측하긴 하지만, 결측값 등의 존재로 인해 불완전한 관계임을 의미합니다. 컬럼 별 결측값을 확인했을 때 아래와 같은 결과를 확인할 수 있었습니다.

```
survived       0
pclass         0
sex            0
age          177
sibsp          0
parch          0
fare           0
embarked       2
class          0
who            0
adult_male     0
deck         688
embark_town    2
alive          0
alone          0
dtype: int64
```

- 상대적으로 결측치가 많았던 'age', 'deck'가 그래프에서 아래에 위치함을 확인할 수 있습니다.
- 이처럼 dendrogram 그래프는 변수들 간 관계도를 보다 직접적으로 명시한다는 점에서 장점을 가진 방법입니다.

원하는 방식에 따라 결측값을 다양하게 확인해보세요!