Missing Values

Missing values occurs in dataset when some of the informations is not stored for a variable There are 3 mechanisms

1 .Missing Completely at Random, MCAR:

Missing completely at random (MCAR) is a type of missing data mechanism in which the probability of a value being missing is unrelated to both the observed data and the missing data. In other words, if the data is MCAR, the missing values are randomly distributed throughout the dataset, and there is no systematic reason for why they are missing.

For example, in a survey about the prevalence of a certain disease, the missing data might be MCAR if the survey participants with missing values for certain questions were selected randomly and their missing responses are not related to their disease status or any other variables measured in the survey.

2. Missing at Random MAR:

Missing at Random (MAR) is a type of missing data mechanism in which the probability of a value being missing depends only on the observed data, but not on the missing data itself. In other words, if the data is MAR, the missing values are systematically related to the observed data, but not to the missing data. Here are a few examples of missing at random:

Income data: Suppose you are collecting income data from a group of people, but some participants choose not to report their income. If the decision to report or not report income is related to the participant's age or gender, but not to their income level, then the data is missing at random.

Medical data: Suppose you are collecting medical data on patients, including their blood pressure, but some patients do not report their blood pressure. If the patients who do not report their blood pressure are more likely to be younger or have healthier lifestyles, but the missingness is not related to their actual blood pressure values, then the data is missing at random.

3. Missing data not at random (MNAR)

It is a type of missing data mechanism where the probability of missing values depends on the value of the missing data itself. In other words, if the data is MNAR, the missingness is not random and is dependent on unobserved or unmeasured factors that are associated with the missing values.

For example, suppose you are collecting data on the income and job satisfaction of employees in a company. If employees who are less satisfied with their jobs are more likely to refuse to report their income, then the data is not missing at random. In this case, the missingness is dependent on job satisfaction, which is not directly observed or measured.

In [1]:

```
import seaborn as sns
```

In [3]:

```
df = sns.load_dataset("titanic")
```

In [4]:

```
df.head()
```

Out[4]:

	survived	pclass	sex	age	sibsp	parch	fare	embarked	class	who	adult_ma
0	0	3	male	22.0	1	0	7.2500	S	Third	man	Tr
1	1	1	female	38.0	1	0	71.2833	C	First	woman	Fal
2	1	3	female	26.0	0	0	7.9250	S	Third	woman	Fal
3	1	1	female	35.0	1	0	53.1000	S	First	woman	Fal
4	0	3	male	35.0	0	0	8.0500	S	Third	man	Tr

In [6]:

```
## For checking the values in dataset
## isnull means says that NAN value

df.isnull().sum()
```

Out[6]:

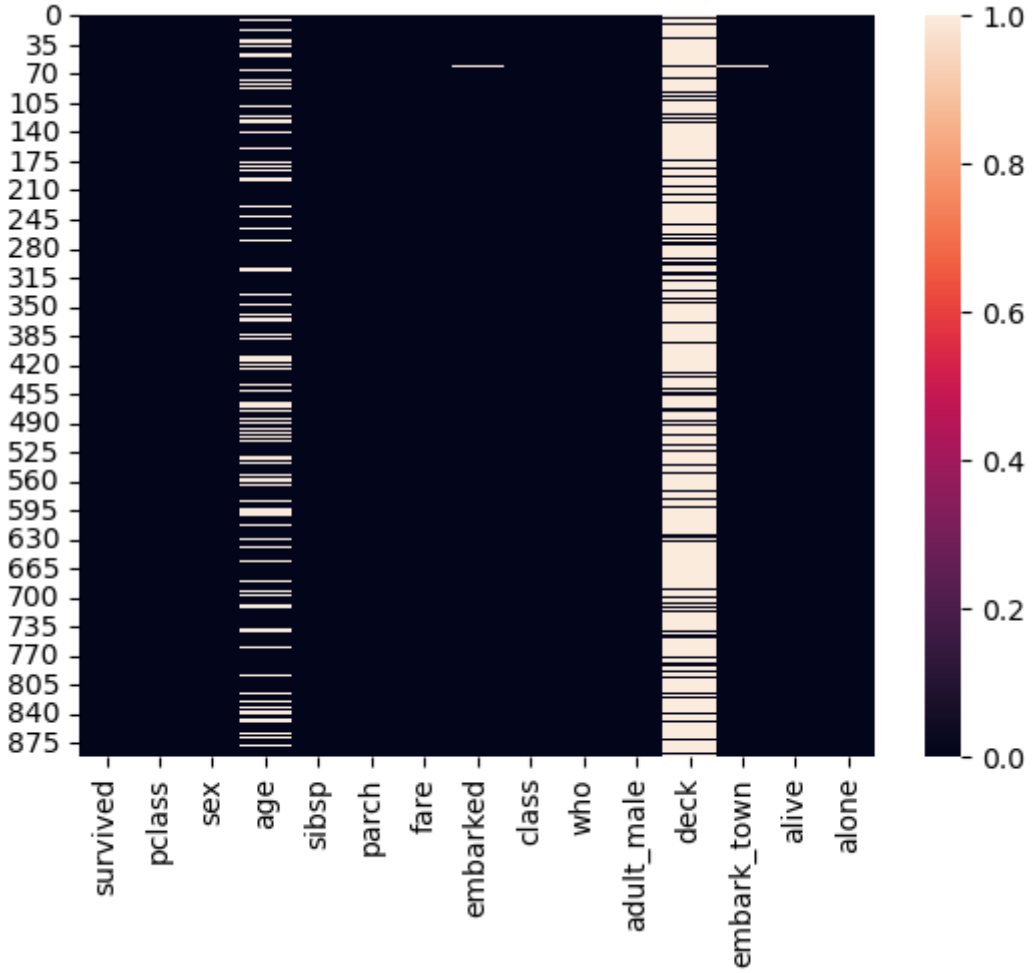
```
survived         0
pclass           0
sex              0
age            177
sibsp            0
parch            0
fare             0
embarked         2
class            0
who              0
adult_male       0
deck           688
embark_town      2
alive            0
alone            0
dtype: int64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

In [7]:

```
sns.heatmap(df.isnull())
```

Out[7]:

```
<Axes: >
```



In [11]:

```
## Handling missing by deleting rows
## Row wise deletion

df.dropna().shape
```

Out[11]:

```
(182, 15)
```

In [12]:

```
## Actual data in titanic dataset with

df.shape
```

Out[12]:

```
(891, 15)
```

In [14]:

```
## Handling missing by deleting columns
## Column wise deletion

df.dropna(axis=1)
```

Out[14]:

	survived	pclass	sex	sibsp	parch	fare	class	who	adult_male	alive	alone
0	0	3	male	1	0	7.2500	Third	man	True	no	False
1	1	1	female	1	0	71.2833	First	woman	False	yes	False
2	1	3	female	0	0	7.9250	Third	woman	False	yes	True
3	1	1	female	1	0	53.1000	First	woman	False	yes	False
4	0	3	male	0	0	8.0500	Third	man	True	no	True

Imputation Technique

1-Mean Value Imputation

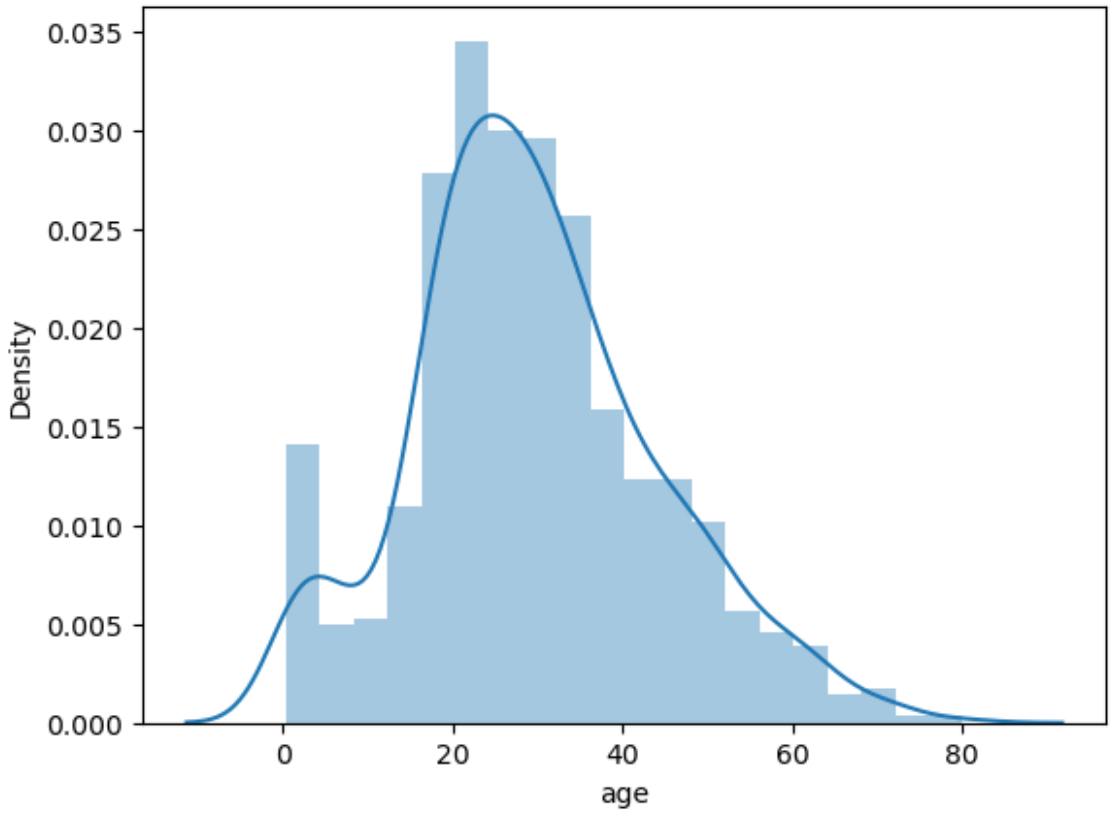
In [16]:

```
sns.distplot(df['age'])
```

```
C:\Users\Mr Abhi\AppData\Local\Temp\ipykernel_14204\3234920688.py:1: User
Warning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.
0.

Please adapt your code to use either `displot` (a figure-level function w
ith
similar flexibility) or `histplot` (an axes-level function for histogram
s).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751 (http
s://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751)

  sns.distplot(df['age'])
```

Out[16]:

```
<Axes: xlabel='age', ylabel='Density'>
```



In [17]:

```
df.age.isnull().sum()
```

Out[17]:

177

In [26]:

```
df['Mean_Age']=df['age'].fillna(df['age'].mean)
```

In [27]:

```
df[['Mean_Age','age']]

## This technique will work when your data is normally disrtibuted
```

Out[27]:

	Mean_Age	age
0	22.0	22.0
1	38.0	38.0
2	26.0	26.0
3	35.0	35.0
4	35.0	35.0
...	...	...
886	27.0	27.0
887	19.0	19.0
888	<bound method NDFrame._add_numeric_operations....	NaN
889	26.0	26.0
890	32.0	32.0

891 rows × 2 columns

2-Median value Imputation

If you have a outliers in dataset use this technique

In [28]:

```
df['Median_Age']=df['age'].fillna(df['age'].median)
```

In [30]:

```
df[['Median_Age','Mean_Age','age']]
```

Out[30]:

	Median_Age	Mean_Age	age
0	22.0	22.0	22.0
1	38.0	38.0	38.0
2	26.0	26.0	26.0
3	35.0	35.0	35.0
4	35.0	35.0	35.0
...	...	...	...
886	27.0	27.0	27.0
887	19.0	19.0	19.0
888	<bound method NDFrame._add_numeric_operations....	<bound method NDFrame._add_numeric_operations....	NaN
889	26.0	26.0	26.0
890	32.0	32.0	32.0

891 rows × 3 columns

Mode Value Imputation--Categorical

In [33]:

```
df[df['embarked'].isnull()]
```

Out[33]:

	survived	pclass	sex	age	sibsp	parch	fare	embarked	class	who	adult_mal
61	1	1	female	38.0	0	0	80.0	NaN	First	woman	Fals
829	1	1	female	62.0	0	0	80.0	NaN	First	woman	Fals

In [34]:

```
df['embarked'].unique()
```

Out[34]:

```
array(['S', 'C', 'Q', nan], dtype=object)
```

In [39]:

```
mode = df[df['age'].notna()]['embarked'].mode()[0]
```

In [40]:

```
mode
```

Out[40]:

```
'S'
```

In [41]:

```
df['embarked_mode']=df['embarked'].fillna(mode)
```

In [43]:

```
df[['embarked_mode','embarked']]
```

Out[43]:

	embarked_mode	embarked
0	S	S
1	C	C
2	S	S
3	S	S
4	S	S
...	...	...
886	S	S
887	S	S
888	S	S
889	C	C
890	Q	Q

891 rows × 2 columns

In [44]:

```
df['embarked_mode'].isnull().sum()
```

Out[44]:

```
0
```

In [48]:

```
df['embarked'].isnull().sum()
```

Out[48]:

```
2
```

In [ ]:

```

```