Missing value #1 (Fill some data)

1. Detect the missing value

In [1]:

```
import pandas as pd
df = pd.read_excel('dataset.xlsx', sheet_name='missing')
```

In [2]:

```
df.head()
```

Out[2]:

	Sex	Height
0	F	162.0
1	M	162.0
2	F	163.0
3	M	165.0
4	M	167.0

In [3]:

```
df
```

Out[3]:

	Sex	Height
0	F	162.0
1	M	162.0
2	F	163.0
3	M	165.0
4	M	167.0
5	M	165.0
6	M	169.0
7	F	155.0
8	M	163.0
9	M	166.0
10	M	162.0
11	M	166.0
12	F	164.0
13	F	164.0
14	F	161.0
15	M	171.0
16	F	160.0
17	F	151.0
18	F	162.0
19	M	170.0
20	M	165.0
21	M	NaN
22	F	158.0
23	M	161.0
24	F	159.0
25	F	161.0
26	F	156.0
27	M	166.0
28	F	NaN
29	F	156.0
30	F	152.0

In [4]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 31 entries, 0 to 30
Data columns (total 2 columns):
 #   Column  Non-Null Count  Dtype  
---  ------  --------------  -----  
 0   Sex     31 non-null     object 
 1   Height  29 non-null     float64
dtypes: float64(1), object(1)
memory usage: 624.0+ bytes
```

In [5]:

```
df.isna().sum()
```

Out[5]:

```
Sex       0
Height    2
dtype: int64
```

In [6]:

```
df[df.isna().any(axis=1)]
```

Out[6]:

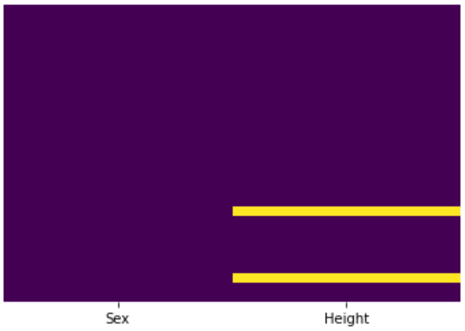
	Sex	Height
21	M	NaN
28	F	NaN

In [7]:

```
import seaborn as sns
sns.heatmap(df.isnull(),yticklabels=False,cbar=False,cmap='viridis')
```

Out[7]:

```
<AxesSubplot:>
```



2. Fill the N/A value with mean

In [8]:

```
df_fill_with_mean = df.copy()
```

In [9]:

```
df_fill_with_mean[df_fill_with_mean.isna().any(axis=1)]
```

Out[9]:

	Sex	Height
21	M	NaN
28	F	NaN

In [10]:

```
avg_height = df_fill_with_mean['Height'].mean()
```

In [11]:

```
avg_height
```

Out[11]:

```
162.13793103448276
```

In [12]:

```
df_fill_with_mean['Height'] = df_fill_with_mean['Height'].fillna(avg_height)
```

In [13]:

```
df_fill_with_mean.isna().sum()
```

Out[13]:

```
Sex       0
Height    0
dtype: int64
```

In [14]:

```
df_fill_with_mean.iloc[[21,28]]
```

Out[14]:

	Sex	Height
21	M	162.137931
28	F	162.137931

In [15]:

```
df_fill_with_mean[df.isna().any(axis=1)]
```

Out[15]:

	Sex	Height
21	M	162.137931
28	F	162.137931

3. Filling the N/A value with mean of each group

In [16]:

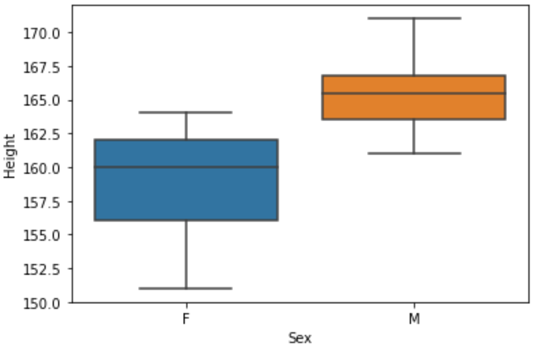
```
df_fill_with_mean_of_group = df.copy()
```

In [17]:

```
sns.boxplot(x='Sex',y='Height',data=df_fill_with_mean_of_group)
```

Out[17]:

```
<AxesSubplot:xlabel='Sex', ylabel='Height'>
```



In [18]:

```
M = df_fill_with_mean_of_group.loc[df_fill_with_mean_of_group['Sex']=='M']
F = df_fill_with_mean_of_group.loc[df_fill_with_mean_of_group['Sex']=='F']
```

In [19]:

```
M.mean()
```

```
/var/folders/50/yc3xx4j955ndlwshz8251btr0000gn/T/ipykernel_35580/3049135688.py:1: FutureWarning: Dropping of nuisance c
olumns in DataFrame reductions (with 'numeric_only=None') is deprecated; in a future version this will raise TypeError.
Select only valid columns before calling the reduction.
  M.mean()
```

Out[19]:

```
Height    165.571429
dtype: float64
```

In [20]:

```
F.mean()
```

```
/var/folders/50/yc3xx4j955ndlwshz8251btr0000gn/T/ipykernel_35580/1563806353.py:1: FutureWarning: Dropping of nuisance c
olumns in DataFrame reductions (with 'numeric_only=None') is deprecated; in a future version this will raise TypeError.
Select only valid columns before calling the reduction.
  F.mean()
```

Out[20]:

```
Height    158.933333
dtype: float64
```

In [21]:

```
import numpy as np
avg_M = np.average(M['Height'].dropna())
avg_F = np.average(F['Height'].dropna())
```

In [22]:

```
avg_F
```

Out[22]:

```
158.93333333333334
```

In [23]:

```
def replace_height(x):
    sex = x[0]
    h = x[1]
    if pd.isnull(h):
        if sex == 'M':
            return avg_M
        elif sex == 'F':
            return avg_F
    else:
        return h
```

In [24]:

```
df_fill_with_mean_of_group['Height'] = df_fill_with_mean_of_group[['Sex','Height']].apply(replace_height,axis=1)
```

In [25]:

```
df_fill_with_mean_of_group.isna().sum()
```

Out[25]:

```
Sex       0
Height    0
dtype: int64
```

In [26]:

```
df_fill_with_mean_of_group[df.isna().any(axis=1)]
```

Out[26]:

	Sex	Height
21	M	165.571429
28	F	158.933333

4. Filling the N/A value with scikid learn

In [27]:

```
df_sklearn = df.copy()
```

In [28]:

```
from sklearn.impute import SimpleImputer
my_fill_tech = SimpleImputer(strategy = 'median')
fill_data = my_fill_tech.fit_transform(df_sklearn.drop('Sex',axis=1))
```

In [29]:

```
df_sklearn['Height']=pd.DataFrame(fill_data)
```

In [30]:

```
df_sklearn.isna().sum()
```

Out[30]:

```
Sex       0
Height    0
dtype: int64
```

In [31]:
```
df_sklearn[df.isna().any(axis=1)]
```

Out[31]:

	Sex	Height
21	M	162.0
28	F	162.0

5. Filling the N/A value for caterical data

In [32]:
```
df = pd.read_excel('dataset.xlsx', sheet_name='missing2')
```

In [33]:
```
df_category = df.copy()
```

In [34]:
```
df_category.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 31 entries, 0 to 30
Data columns (total 2 columns):
 #   Column  Non-Null Count  Dtype 
---  ------  --------------  ----- 
 0   Sex     29 non-null     object
 1   Height  31 non-null     int64 
dtypes: int64(1), object(1)
memory usage: 624.0+ bytes
```

In [35]:
```
df_category.isna().sum()
```

Out[35]:
```
Sex       2
Height    0
dtype: int64
```

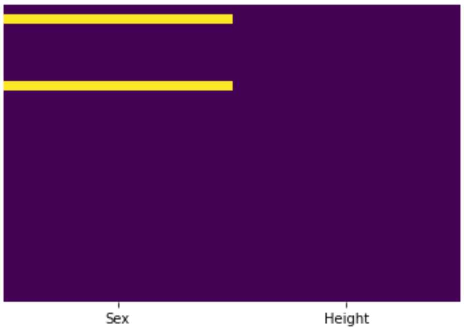
In [36]:
```
df_category[df_category.isna().any(axis=1)]
```

Out[36]:

	Sex	Height
1	NaN	162
8	NaN	163

In [37]:
```
sns.heatmap(df_category.isnull(),yticklabels=False,cbar=False,cmap='viridis')
```

Out[37]:
```
<AxesSubplot:>
```



In [38]:
```
len(df[df['Sex']=='F'])
```

Out[38]:
```
16
```

In [39]:
```
from sklearn.impute import SimpleImputer
my_fill_tech = SimpleImputer(strategy = 'most_frequent')
fill_data = my_fill_tech.fit_transform(df_category.drop('Height',axis=1))
```

In [40]:
```
df_category['Sex']=pd.DataFrame(fill_data)
```

In [41]:
```
df_category.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 31 entries, 0 to 30
Data columns (total 2 columns):
 #   Column  Non-Null Count  Dtype 
---  ------  --------------  ----- 
 0   Sex     31 non-null     object
 1   Height  31 non-null     int64 
dtypes: int64(1), object(1)
memory usage: 624.0+ bytes
```

In [42]:
```
df_category[df.isna().any(axis=1)]
```

Out[42]:

	Sex	Height
1	F	162
8	F	163