EDA ---> Exploratory Data Analysis

Parts of EDA

1. Univariate Analysis --> Analysis of single independent column
2. Bivariate Analysis --> Analysis of two columns
3. Multivariate Analysis --> Analysis of more than one column

Data types 1. Numerical Data --> continous data --> age(year,date,month),height, weight 2. Categorical Data --> Descrete data --> total no. of employees

In [1]:
```
import numpy as np
import pandas as pd
```

In [2]:
```
import matplotlib.pyplot as plt   # Visulization liberary
import seaborn as sns             # matplotlib updated version
```

In [3]:
```
df=pd.read_csv("D:\\Summer Training Video\\ML\\titanic.csv")
```

In [4]:
```
df.head(10)
```

Out[4]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabi
0	892	0	3	Kelly, Mr. James	male	34.5	0	0	330911	7.8292	NaN
1	893	1	3	Wilkes, Mrs. James (Ellen Needs)	female	47.0	1	0	363272	7.0000	NaN
2	894	0	2	Myles, Mr. Thomas Francis	male	62.0	0	0	240276	9.6875	NaN
3	895	0	3	Wirz, Mr. Albert	male	27.0	0	0	315154	8.6625	NaN
4	896	1	3	Hirvonen, Mrs. Alexander (Helga E Lindqvist)	female	22.0	1	1	3101298	12.2875	NaN
5	897	0	3	Svensson, Mr. Johan Cervin	male	14.0	0	0	7538	9.2250	NaN
6	898	1	3	Connolly, Miss. Kate	female	30.0	0	0	330972	7.6292	NaN
7	899	0	2	Caldwell, Mr. Albert Francis	male	26.0	1	1	248738	29.0000	NaN
8	900	1	3	Abrahim, Mrs. Joseph (Sophie Halaut Easu)	female	18.0	0	0	2657	7.2292	NaN
9	901	0	3	Davies, Mr. John Samuel	male	21.0	2	0	A/4 48871	24.1500	NaN

In [5]:
```
df.isnull().sum()
```

Out[5]:
```
PassengerId      0
Survived         0
Pclass           0
Name             0
Sex              0
Age             86
SibSp            0
Parch            0
Ticket           0
Fare             1
Cabin          327
Embarked         0
dtype: int64
```

In [7]:
```
df.count()
```

Out[7]:
```
PassengerId    418
Survived       418
Pclass         418
Name           418
Sex            418
Age            332
SibSp          418
Parch          418
Ticket         418
Fare           417
Cabin           91
Embarked       418
dtype: int64
```

In [8]:
```
df.shape
```

Out[8]:
```
(418, 12)
```

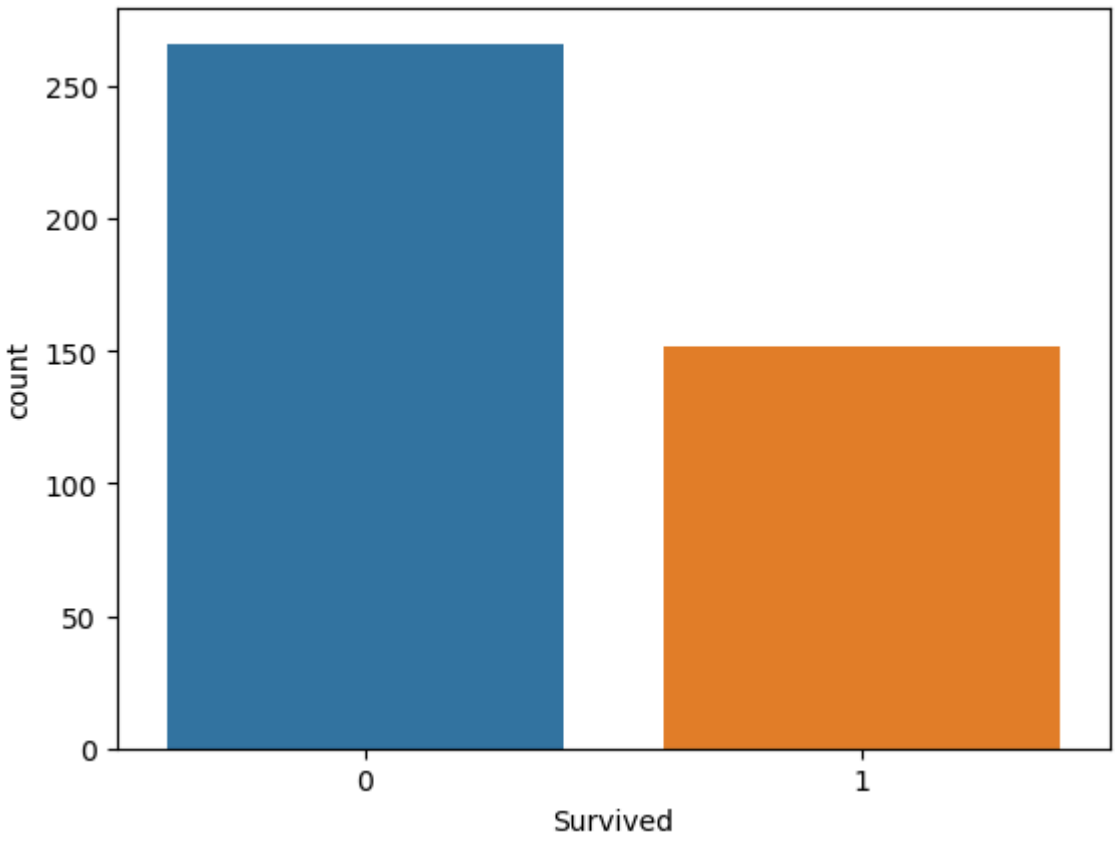
1) Univeriate Analysis

In [9]:
```
df.columns
```

Out[9]:
```
Index(['PassengerId', 'Survived', 'Pclass', 'Name', 'Sex', 'Age', 'SibSp',
       'Parch', 'Ticket', 'Fare', 'Cabin', 'Embarked'],
      dtype='object')
```

In [10]:
```
sns.countplot(x=df["Survived"])
```

Out[10]:
```
<Axes: xlabel='Survived', ylabel='count'>
```

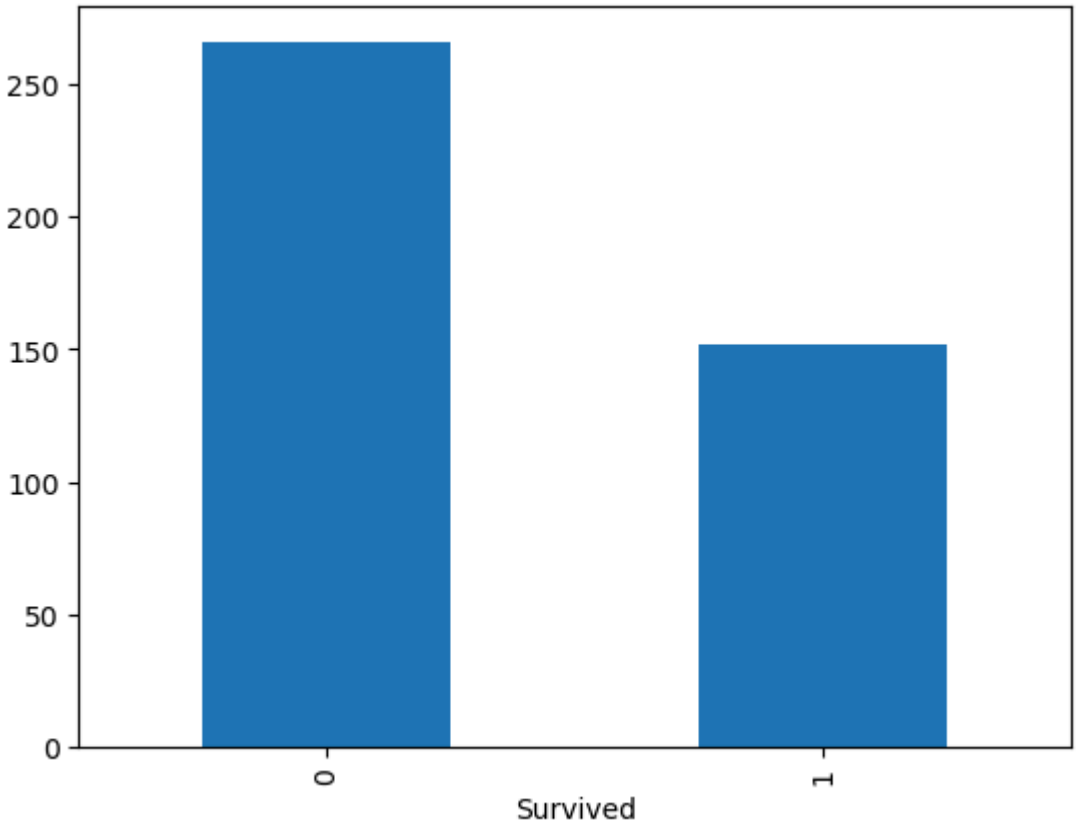


In [12]:
```
df['Survived'].value_counts()    # value count is only applicable on categorica
```

Out[12]:
```
Survived
0    266
1    152
Name: count, dtype: int64
```

In [13]:
```
df['Survived'].value_counts().plot(kind='bar')
```

Out[13]:
```
<Axes: xlabel='Survived'>
```

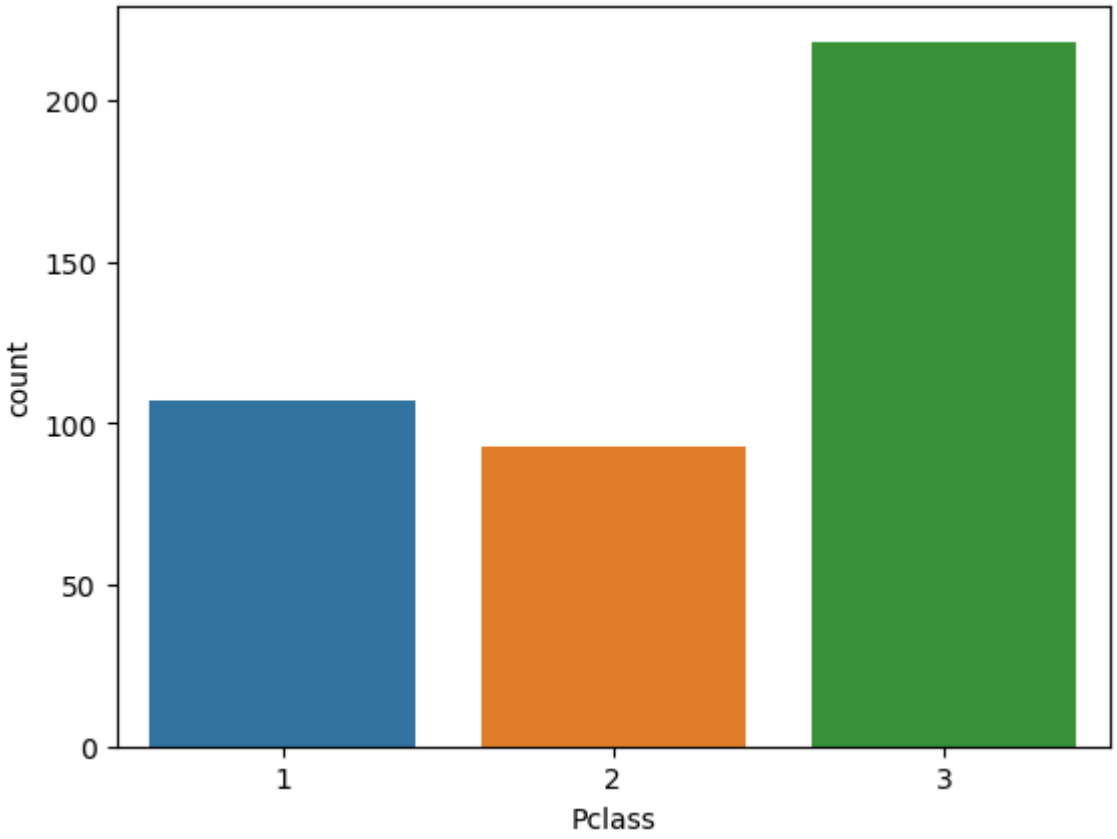


In [14]:
```
df['Pclass'].value_counts()
```

Out[14]:
```
Pclass
3    218
1    107
2     93
Name: count, dtype: int64
```

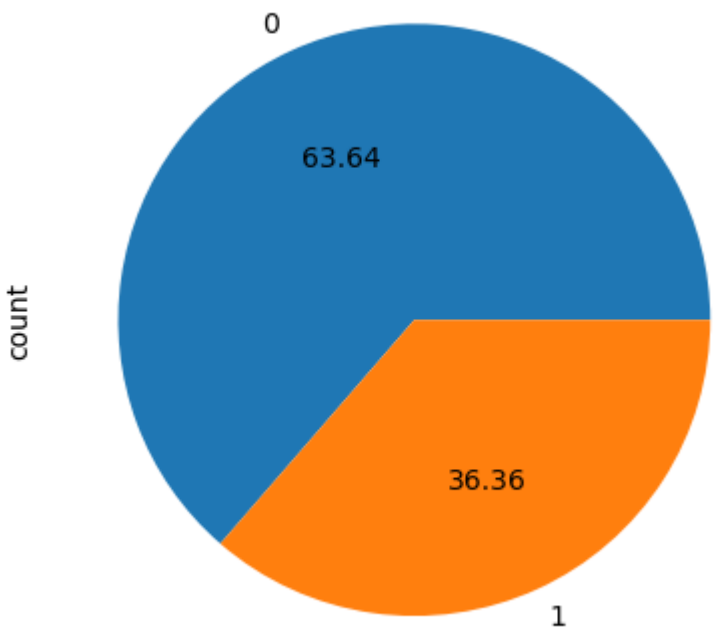
In [15]:
```
sns.countplot(x=df['Pclass'])
```

Out[15]:
```
<Axes: xlabel='Pclass', ylabel='count'>
```



In [16]:
```
df['Survived'].value_counts().plot(kind='pie',autopct='%.2f')
```

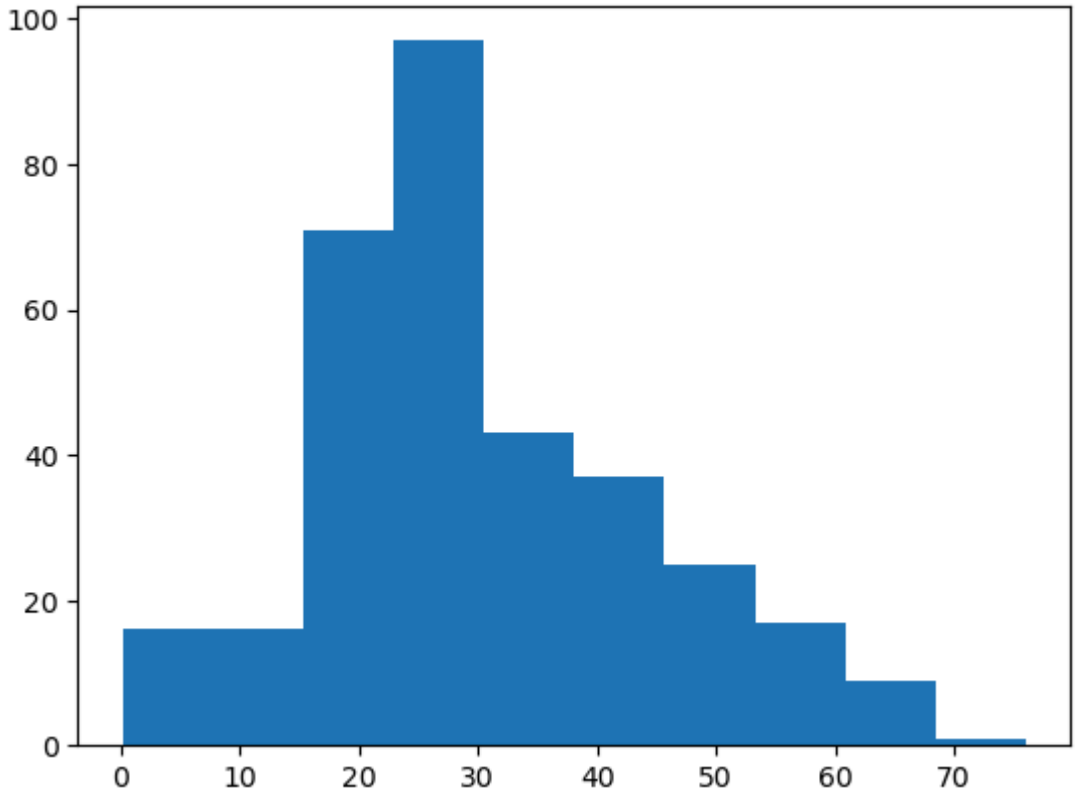
Out[16]: <Axes: ylabel='count'>



If we have numerical data then we use histogrambecause it finds the destribution

In [17]:
```
plt.hist(df['Age'])
```

Out[17]:
```
(array([16., 16., 71., 97., 43., 37., 25., 17.,  9.,  1.]),
 array([ 0.17 ,  7.753, 15.336, 22.919, 30.502, 38.085, 45.668, 53.251,
        60.834, 68.417, 76.   ]),
 <BarContainer object of 10 artists>)
```



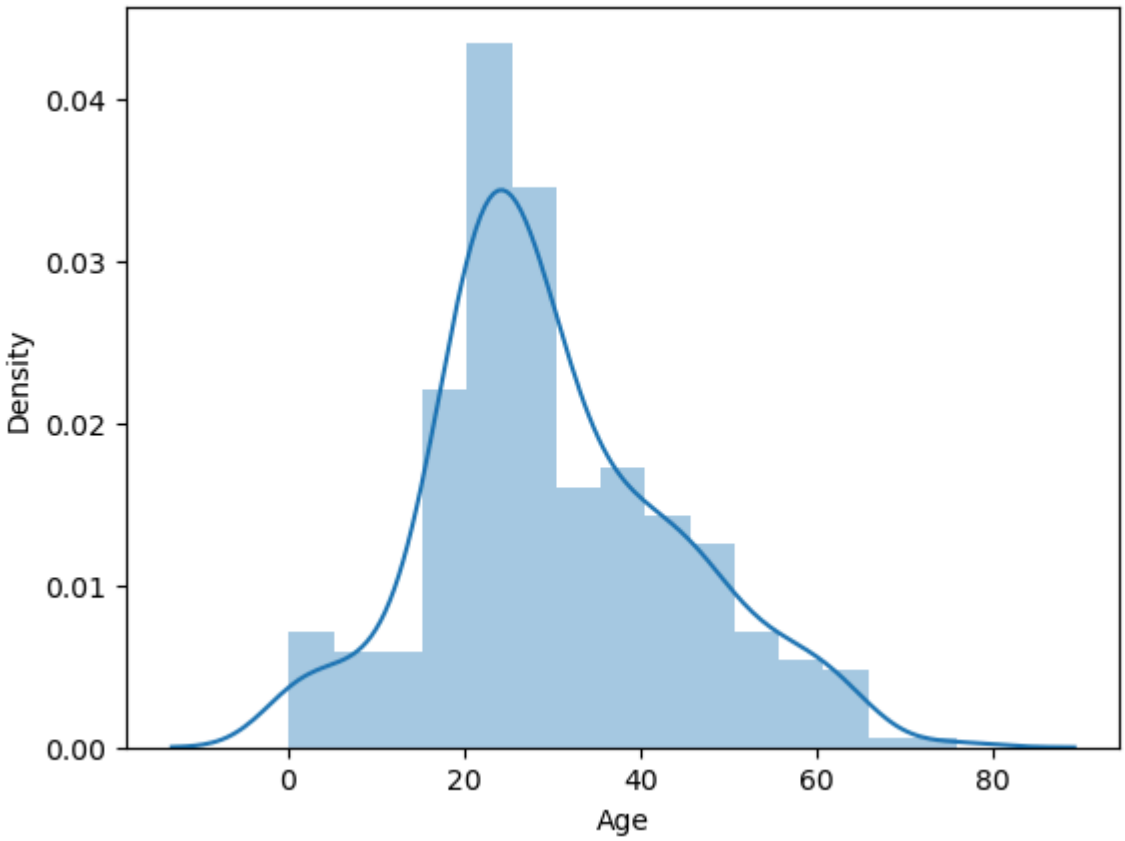
Distplot

```
curve --> KDE(Kurnel Density Extraction) used to find probability
```

In [19]:
```
sns.distplot(df['Age'])     # to find the peak value
```

```
C:\Users\yashs\AppData\Local\Temp\ipykernel_14348\1050488154.py:1: UserWarnin
g:

`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 with
similar flexibility) or `histplot` (an axes-level function for histograms).

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

  sns.distplot(df['Age'])     # to find the peak value
```

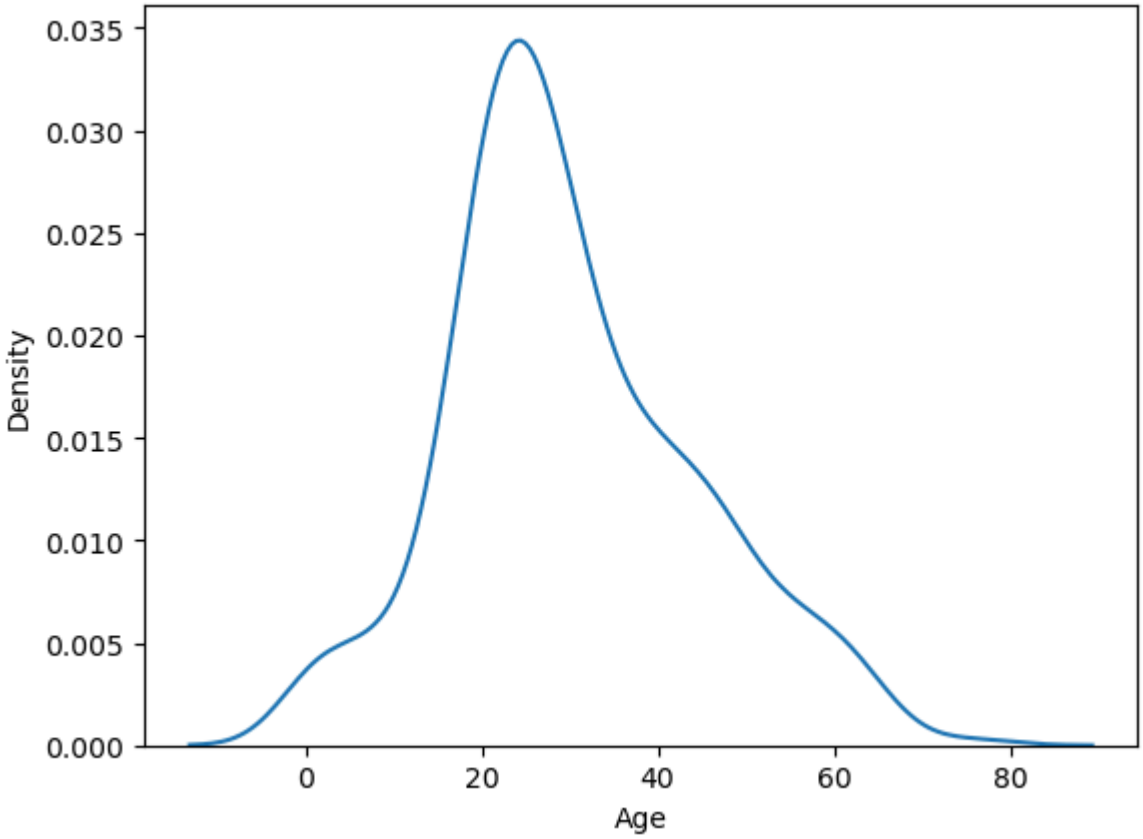
Out[19]:
```
<Axes: xlabel='Age', ylabel='Density'>
```



In [20]:
```
sns.distplot(df['Age'], hist = False)
```

```
C:\Users\yashs\AppData\Local\Temp\ipykernel_14348\4035848256.py:1: UserWarnin
g:

`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 with
similar flexibility) or `kdeplot` (an axes-level function for kernel density
plots).

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

  sns.distplot(df['Age'], hist = False)
```

Out[20]:
```
<Axes: xlabel='Age', ylabel='Density'>
```



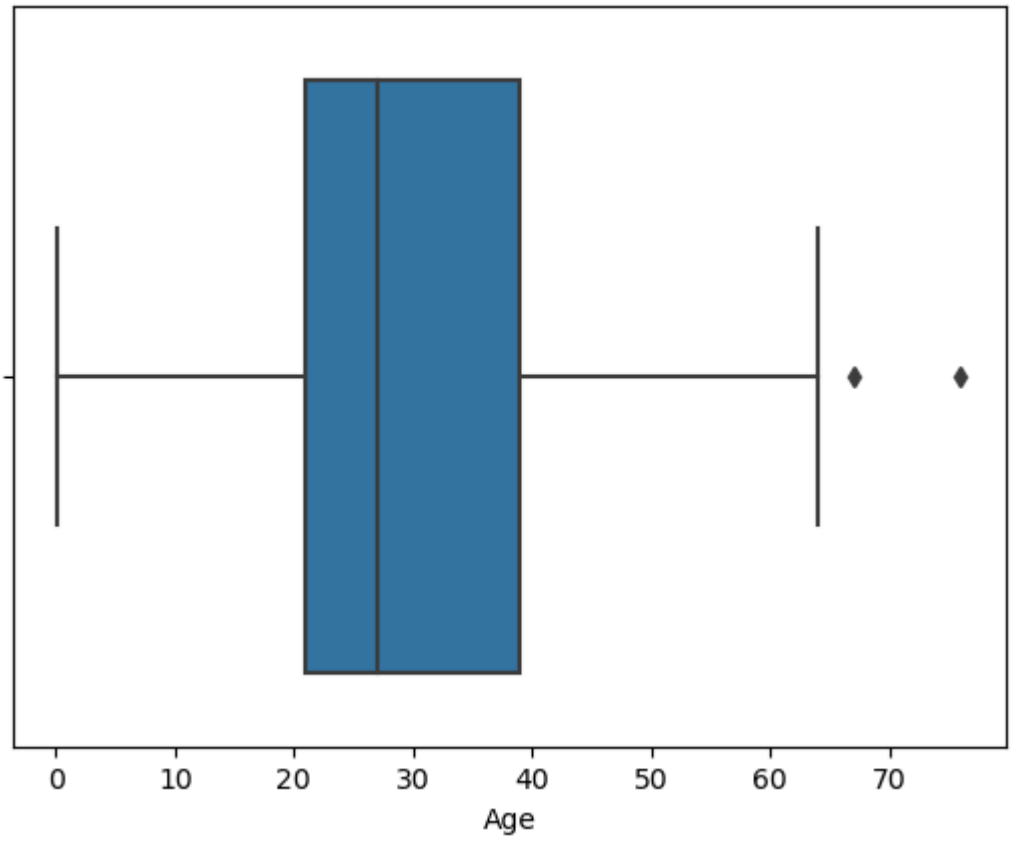
BoxPlot

```
Outliers will be present below the lower fence and upper fence

BoxPlot --> It is used to find the outliers

    1. lower fence
    2. 25% data
    3. IOR(Inter Quarantil range)(75% - 25%)
```

```
    4. 75% data
    5. upper fence
```

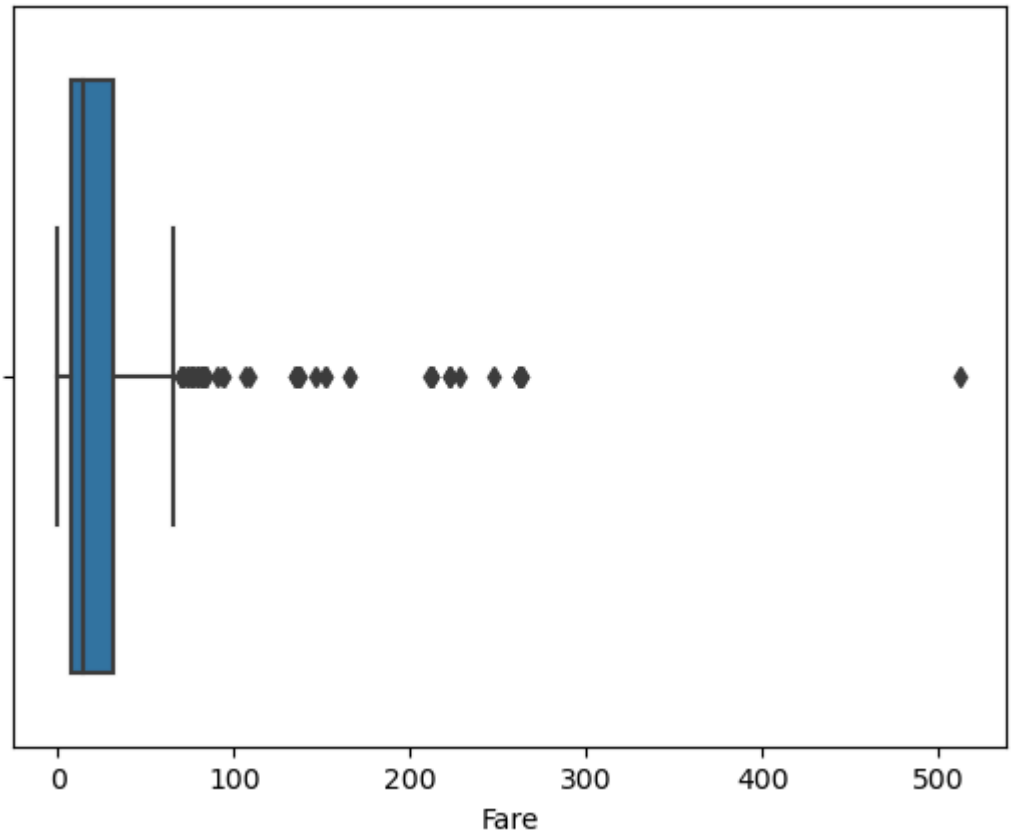
In [22]:
```
sns.boxplot(x=df['Age'])
```

Out[22]: <Axes: xlabel='Age'>



In [23]:
```
sns.boxplot(x=df['Fare'])
```

Out[23]: <Axes: xlabel='Fare'>



In [24]:
```
tips=pd.read_csv("D:\\Summer Training Video\\ML\\tips.csv")
```

In [25]:
```
tips
```

Out[25]:

	total_bill	tip	sex	smoker	day	time	size
0	16.99	1.01	Female	No	Sun	Dinner	2
1	10.34	1.66	Male	No	Sun	Dinner	3
2	21.01	3.50	Male	No	Sun	Dinner	3
3	23.68	3.31	Male	No	Sun	Dinner	2
4	24.59	3.61	Female	No	Sun	Dinner	4
...	...	...	...	...	...	...	...
239	29.03	5.92	Male	No	Sat	Dinner	3
240	27.18	2.00	Female	Yes	Sat	Dinner	2
241	22.67	2.00	Male	Yes	Sat	Dinner	2
242	17.82	1.75	Male	No	Sat	Dinner	2
243	18.78	3.00	Female	No	Thur	Dinner	2

244 rows × 7 columns

In [26]:
```
tips.head(10)
```

Out[26]:

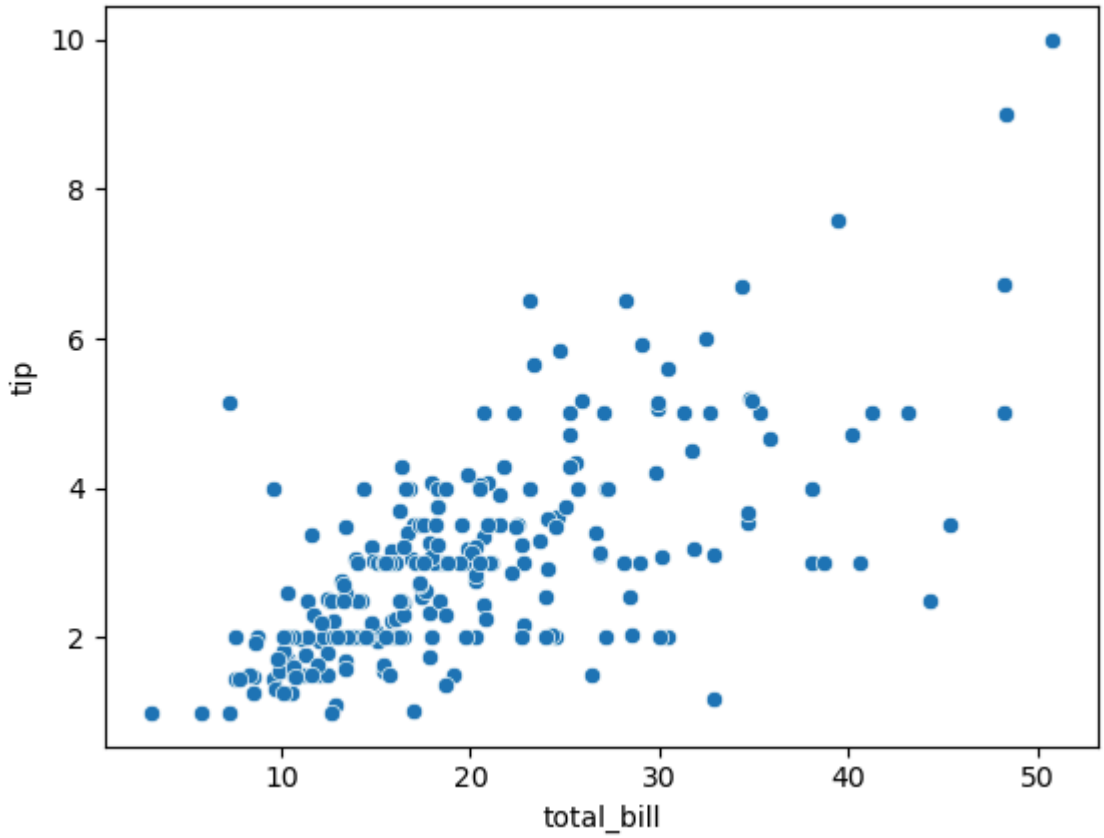
	total_bill	tip	sex	smoker	day	time	size
0	16.99	1.01	Female	No	Sun	Dinner	2
1	10.34	1.66	Male	No	Sun	Dinner	3
2	21.01	3.50	Male	No	Sun	Dinner	3
3	23.68	3.31	Male	No	Sun	Dinner	2
4	24.59	3.61	Female	No	Sun	Dinner	4
5	25.29	4.71	Male	No	Sun	Dinner	4
6	8.77	2.00	Male	No	Sun	Dinner	2
7	26.88	3.12	Male	No	Sun	Dinner	4
8	15.04	1.96	Male	No	Sun	Dinner	2
9	14.78	3.23	Male	No	Sun	Dinner	2

Bivariate Analysis

1. Scatterplot(numerical column - Numerical Column)

In [27]:
```
sns.scatterplot(x=tips['total_bill'],y=tips['tip'])
```

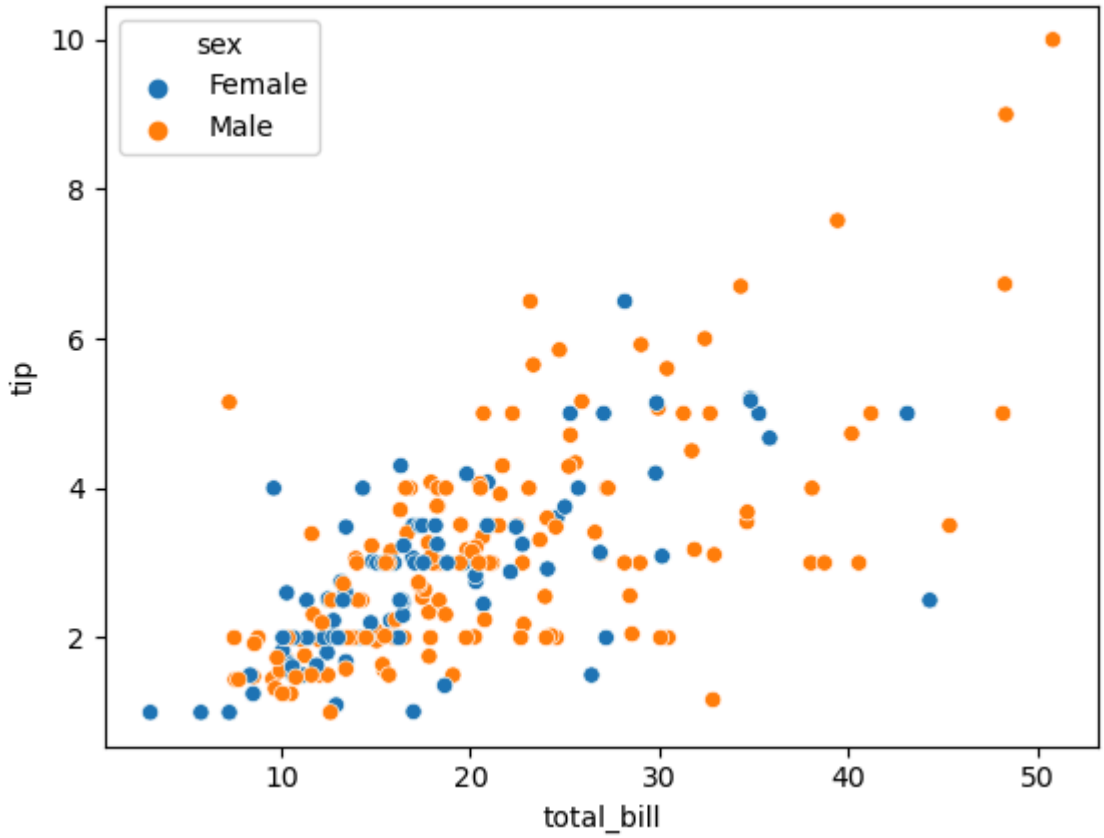
Out[27]:
```
<Axes: xlabel='total_bill', ylabel='tip'>
```




```
Hue , style , color , legend are hyper parameter from which we check the
relation between columns
```

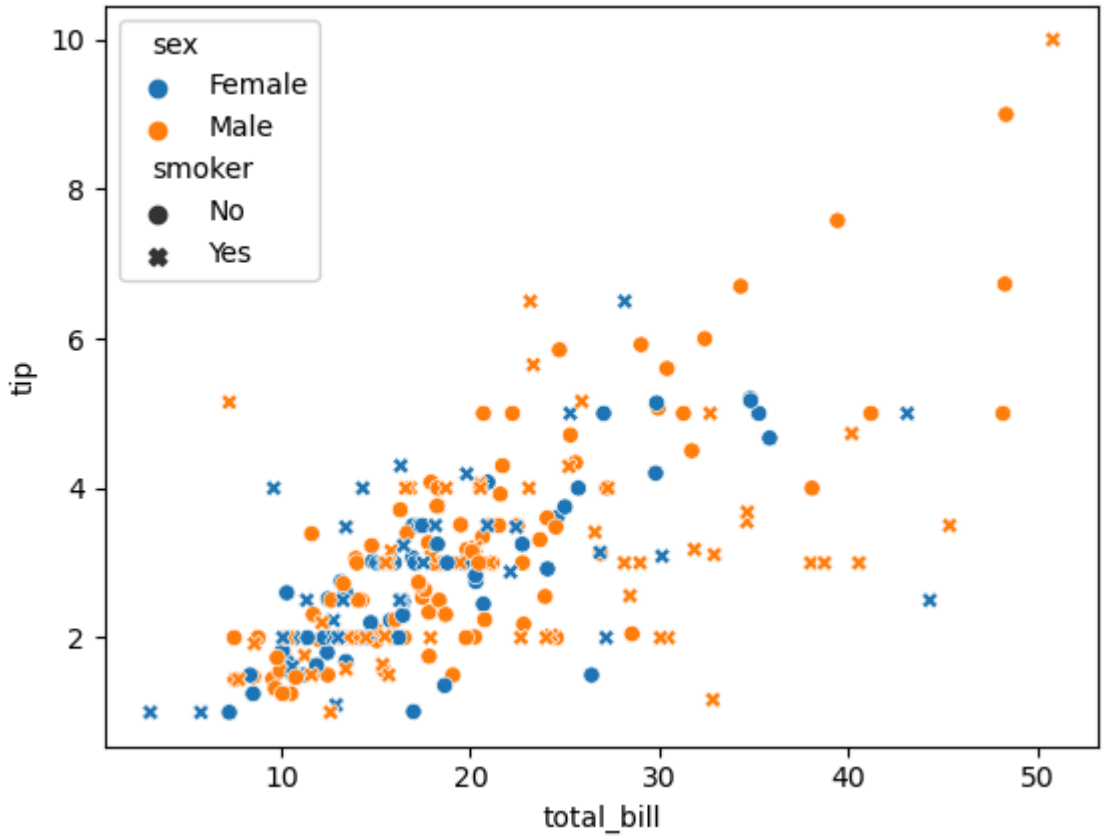
In [28]:
```
sns.scatterplot(x=tips['total_bill'],y=tips['tip'],data=tips,hue=tips['sex'])
```

Out[28]:
```
<Axes: xlabel='total_bill', ylabel='tip'>
```



In [31]:
```
sns.scatterplot(x=tips['total_bill'],y=tips['tip'],data=tips,hue=tips['sex'], s
```

Out[31]:
```
<Axes: xlabel='total_bill', ylabel='tip'>
```



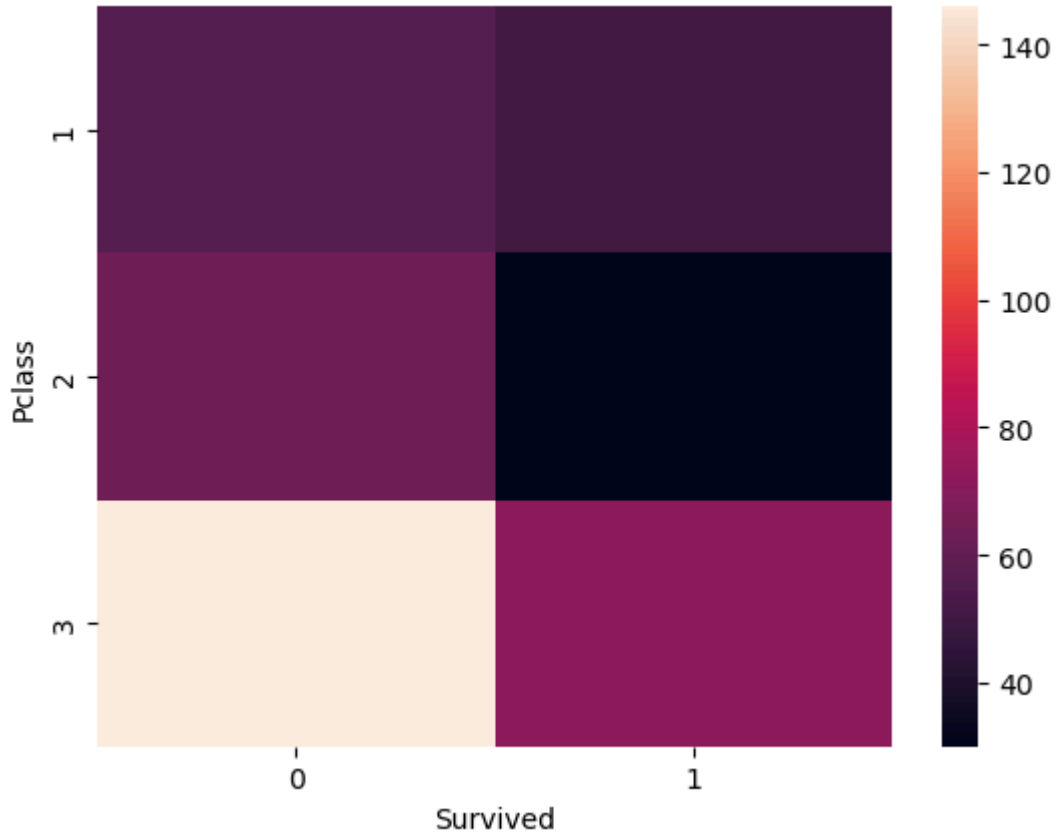
In [32]:
```
a=pd.crosstab(df['Pclass'],df['Survived'])
a
```

Out[32]:

Survived	0	1
Pclass		
1	57	50
2	63	30
3	146	72

In [33]:
```
sns.heatmap(a)   # heat map is used to classify dark color denotes less data
# heatmap is used on catogeicaldata
```

Out[33]: <Axes: xlabel='Survived', ylabel='Pclass'>



Aggeregation Function

In [42]:
```
df.empty
```

Out[42]: False

In [44]:
```
print(type(df))
```

```
<class 'pandas.core.frame.DataFrame'>
```

In [45]:
```
print(df.dtypes)
```

```
PassengerId      int64
Survived         int64
Pclass           int64
Name            object
Sex             object
Age            float64
SibSp            int64
Parch            int64
Ticket          object
Fare           float64
Cabin           object
Embarked        object
dtype: object
```

In [51]:
```
df.groupby('Pclass').mean
```

Out[51]:
```
<bound method GroupBy.mean of <pandas.core.groupby.generic.DataFrameGroupBy o
bject at 0x0000024FE3F18A10>>
```

In [52]:
```
for column in df.columns:
    if df[column].dtype == 'object':
        df[column] = pd.to_numeric(df[column], errors='coerce')
```

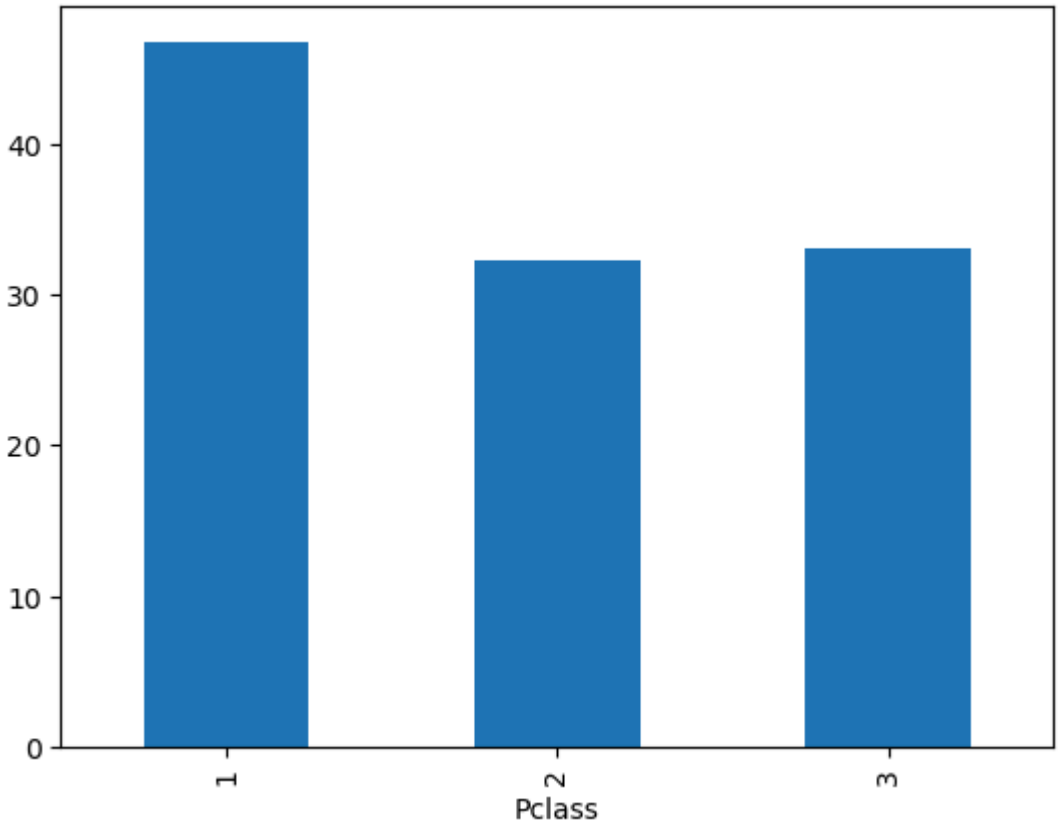
In [53]:
```
df.groupby('Pclass').mean()
```

Out[53]:

Pclass	PassengerId	Survived	Name	Sex	Age	SibSp	Parch	Ticket	F
1	1098.224299	0.467290	NaN	NaN	40.918367	0.476636	0.383178	55262.902778	94.280
2	1117.935484	0.322581	NaN	NaN	28.777500	0.376344	0.344086	162985.754386	22.202
3	1094.178899	0.330275	NaN	NaN	24.027945	0.463303	0.417431	317310.035928	12.459

In [54]:
```
((df.groupby('Pclass').mean()['Survived'])*100).plot(kind='bar')
```

Out[54]: <Axes: xlabel='Pclass'>



In [ ]:
```

```

In [ ]:
```

```