In this Titanic.csv dataset, what do i improved :- 1) Remove Cabin column , becuase it contain more than 70% of missing values in it. & it is not having strong correlation with label called "Survived" 2) Fill the missing values with average in 'Age' column. 3) Identify the outliers in Fare column using BoxPlot.

In [1]:
```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from scipy.stats import shapiro
```

In [2]:
```
# reading csv file
data_frame=pd.read_csv('datasets/titanic/train.csv')
```

In [3]:
```
# (rows,columns)
data_frame.shape
```

Out[3]:
```
(891, 12)
```

In [4]:
```
# rows*columns
data_frame.size
```

Out[4]:
```
10692
```

In [5]:
```
data_frame.head()
```

Out[5]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	S
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN	S

In [6]:
```
# randomly selected rows form dataframe
data_frame.sample(5)
```

Out[6]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
538	539	0	3	Risien, Mr. Samuel Beard	male	NaN	0	0	364498	14.5000	NaN	S

212	213	0	3	Perkin, Mr. John Henry	male	22.0	0	0	A/5 21174	7.2500	NaN	S
54	55	0	1	Ostby, Mr. Engelhart Cornelius	male	65.0	0	1	113509	61.9792	B30	C
188	189	0	3	Bourke, Mr. John	male	40.0	1	1	364849	15.5000	NaN	Q
781	782	1	1	Dick, Mrs. Albert Adrian (Vera Gillespie)	female	17.0	1	0	17474	57.0000	B20	S

In [7]:

```
# to find insights of data
# ex- data_tyoes,null values, columns etc
data_frame.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 891 entries, 0 to 890
Data columns (total 12 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   PassengerId  891 non-null    int64
 1   Survived     891 non-null    int64
 2   Pclass       891 non-null    int64
 3   Name         891 non-null    object
 4   Sex          891 non-null    object
 5   Age          714 non-null    float64
 6   SibSp        891 non-null    int64
 7   Parch        891 non-null    int64
 8   Ticket       891 non-null    object
 9   Fare         891 non-null    float64
 10  Cabin        204 non-null    object
 11  Embarked     889 non-null    object
dtypes: float64(2), int64(5), object(5)
memory usage: 83.7+ KB
```

In [8]:

```
# more insights from numerical columns
data_frame.describe()
```

Out[8]:

	PassengerId	Survived	Pclass	Age	SibSp	Parch	Fare
count	891.000000	891.000000	891.000000	714.000000	891.000000	891.000000	891.000000
mean	446.000000	0.383838	2.308642	29.699118	0.523008	0.381594	32.204208
std	257.353842	0.486592	0.836071	14.526497	1.102743	0.806057	49.693429
min	1.000000	0.000000	1.000000	0.420000	0.000000	0.000000	0.000000
25%	223.500000	0.000000	2.000000	20.125000	0.000000	0.000000	7.910400
50%	446.000000	0.000000	3.000000	28.000000	0.000000	0.000000	14.454200
75%	668.500000	1.000000	3.000000	38.000000	1.000000	0.000000	31.000000
max	891.000000	1.000000	3.000000	80.000000	8.000000	6.000000	512.329200

In [9]:

```
# for finding null-values
data_frame['Age'].isnull().sum()
```

Out[9]: 177

In [10]:
```
data_frame['Cabin'].isnull().sum()
```

Out[10]: 687

In [11]:
```
# missing values in %
(data_frame['Cabin'].isnull().sum()/891)*100
```

Out[11]: 77.10437710437711

In [12]:
```
# for finding pearson correlation between Numercial columns
data_frame.corr()
```

Out[12]:

	PassengerId	Survived	Pclass	Age	SibSp	Parch	Fare
PassengerId	1.000000	-0.005007	-0.035144	0.036847	-0.057527	-0.001652	0.012658
Survived	-0.005007	1.000000	-0.338481	-0.077221	-0.035322	0.081629	0.257307
Pclass	-0.035144	-0.338481	1.000000	-0.369226	0.083081	0.018443	-0.549500
Age	0.036847	-0.077221	-0.369226	1.000000	-0.308247	-0.189119	0.096067
SibSp	-0.057527	-0.035322	0.083081	-0.308247	1.000000	0.414838	0.159651
Parch	-0.001652	0.081629	0.018443	-0.189119	0.414838	1.000000	0.216225
Fare	0.012658	0.257307	-0.549500	0.096067	0.159651	0.216225	1.000000

Data Cleaning :-

In [13]:
```
# Removing Column 'Cabin' because it contain 77% missing values
print("missing values in Cabin in %",(data_frame['Cabin'].isnull().sum()/891)*100)
data_frame=data_frame.drop(['Cabin'],axis=1)
```

```
missing values in Cabin in % 77.10437710437711
```

In [14]:
```
data_frame.head(10)
```

Out[14]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Embarked
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	S
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	S
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	S
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	S
5	6	0	3	Moran, Mr. James	male	NaN	0	0	330877	8.4583	Q
6	7	0	1	McCarthy, Mr. Timothy J	male	54.0	0	0	17463	51.8625	S

7	8	0	3	Palsson, Master. Gosta Leonard	male	2.0	3	1	349909	21.0750	S
8	9	1	3	Johnson, Mrs. Oscar W (Elisabeth Vilhelmina Berg)	female	27.0	0	2	347742	11.1333	S
9	10	1	2	Nasser, Mrs. Nicholas (Adele Achem)	female	14.0	1	0	237736	30.0708	C

In [15]:
```
print("Missing values in Age in %",(data_frame['Age'].isnull().sum()/891)*100)
```

```
Missing values in Age in % 19.865319865319865
```

In [16]:
```
# Age columns contain less missing values ,
# thus we are filling it with mean() values

data_frame.fillna(data_frame['Age'].mean(),inplace=True)
```

In [17]:
```
data_frame.sample(2)
```

Out[17]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Embarked
526	527	1	2	Ridsdale, Miss. Lucy	female	50.0	0	0	W./C. 14258	10.5	S
559	560	1	3	de Messemaeker, Mrs. Guillaume Joseph (Emma)	female	36.0	1	0	345572	17.4	S

In [18]:
```
# all missing values in age column removed
data_frame.isnull().sum()
```

Out[18]:
```
PassengerId    0
Survived       0
Pclass         0
Name           0
Sex            0
Age            0
SibSp          0
Parch          0
Ticket         0
Fare           0
Embarked       0
dtype: int64
```

In [19]:
```
data_frame.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 891 entries, 0 to 890
Data columns (total 11 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   PassengerId  891 non-null    int64
 1   Survived     891 non-null    int64
 2   Pclass       891 non-null    int64
 3   Name         891 non-null    object
 4   Sex          891 non-null    object
 5   Age          891 non-null    float64
 6   SibSp        891 non-null    int64
 7   Parch        891 non-null    int64
 8   Ticket       891 non-null    object
```

```
 9   Fare         891 non-null    float64
 10  Embarked     891 non-null    object
dtypes: float64(2), int64(5), object(4)
memory usage: 76.7+ KB
```

In [20]:
```
# converting Age column values from float64 to integer
print("Size of Age column :- ",data_frame['Age'].nbytes)
data_frame['Age']=data_frame['Age'].astype(int)
```

```
Size of Age column :-  7128
```

In [21]:
```
data_frame.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 891 entries, 0 to 890
Data columns (total 11 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   PassengerId  891 non-null    int64
 1   Survived     891 non-null    int64
 2   Pclass       891 non-null    int64
 3   Name         891 non-null    object
 4   Sex          891 non-null    object
 5   Age          891 non-null    int32
 6   SibSp        891 non-null    int64
 7   Parch        891 non-null    int64
 8   Ticket       891 non-null    object
 9   Fare         891 non-null    float64
 10  Embarked     891 non-null    object
dtypes: float64(1), int32(1), int64(5), object(4)
memory usage: 73.2+ KB
```

In [22]:
```
print("Size of Age columns after conversion :- ",data_frame['Age'].nbytes)
```

```
Size of Age columns after conversion :-  3564
```

In [23]:
```
data_frame['Embarked'].value_counts()
```

Out[23]:
```
S                     644
C                     168
Q                      77
29.69911764705882       2
Name: Embarked, dtype: int64
```

EDA Using Visualization plots

Exploratory Data Analysis (EDA) is an approach to analyze the data using visual techniques.

EDA Using Univariate Analysis

Univariate Analysis :- Statistical analysis using uni(single) variate(variable or column) It can be Inferential and Descriptive.

```
UseD in statistic to describe the type of data that contain only
one attribute or characteristic.
ex :- population of any village

it work on numeric data and categorical data(collection of
info. devided into groups ex:- 0,1)
```

```
uses ex :- mean ( or average ) of population
```

Inferential Statistic :- It allows us to make predictions from the data.

```
ex :- On the basis of health of population , predict
      the survival of population . etc
      uses hypothesis testing , etc..
```

Descriptive statistic :- Used to describe the data using chart,graph, etc

In [24]:
```
data_frame.head(5)
```

Out[24]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Embarked
0	1	0	3	Braund, Mr. Owen Harris	male	22	1	0	A/5 21171	7.2500	S
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38	1	0	PC 17599	71.2833	C
2	3	1	3	Heikkinen, Miss. Laina	female	26	0	0	STON/O2. 3101282	7.9250	S
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35	1	0	113803	53.1000	S
4	5	0	3	Allen, Mr. William Henry	male	35	0	0	373450	8.0500	S

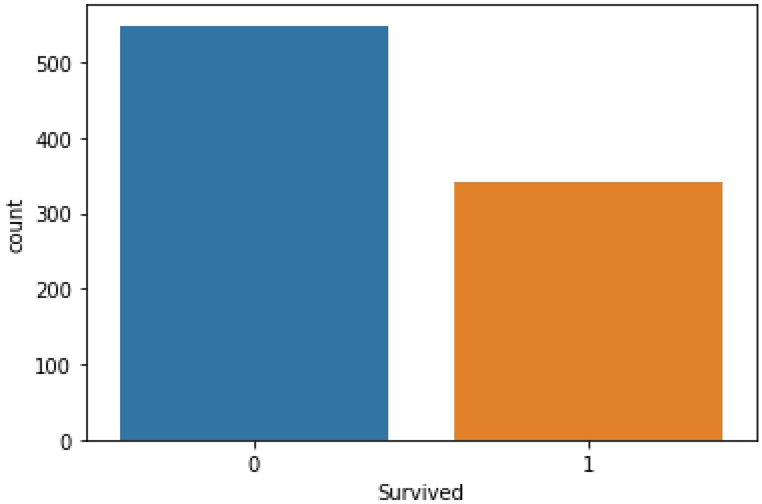
In [25]:
```
# Analysing columns from left to right

# NOTE :- we are not using Passengerld & Name column for analysis , because
# these feature will never help to predict the Survived people
```

sns.countplot() :- used for categorical data analysis

used to Show the counts of observations in each categorical bin using bars.

In [26]:
```
# this bar graph tell us that total number of Survival is less then Not-Survived people
sns.countplot(x='Survived',data=data_frame)
```

Out[26]:
```
<AxesSubplot:xlabel='Survived', ylabel='count'>
```

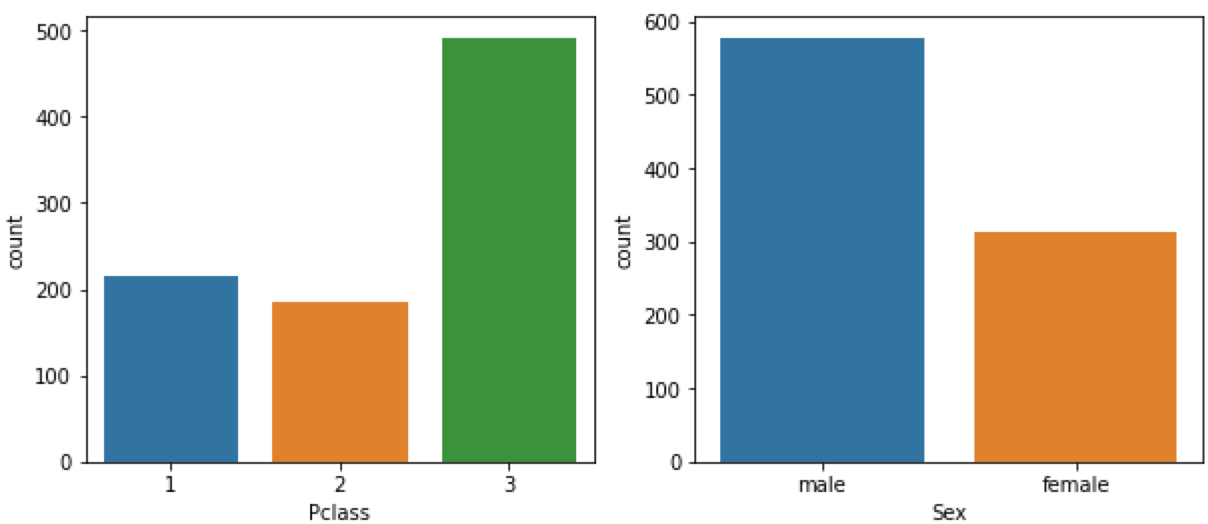


In [27]:

```
plt.figure(figsize=(10,9))

# More passengers were travelled in 3rd Class,
# may be , due to cheaper price.
plt.subplot(2,2,1)
sns.countplot(x='Pclass',data=data_frame)

# Male ratio in titanic was higher as compare to female
plt.subplot(2,2,2)
sns.countplot(x='Sex',data=data_frame)
```

Out[27]:

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



sns.piechart() :-

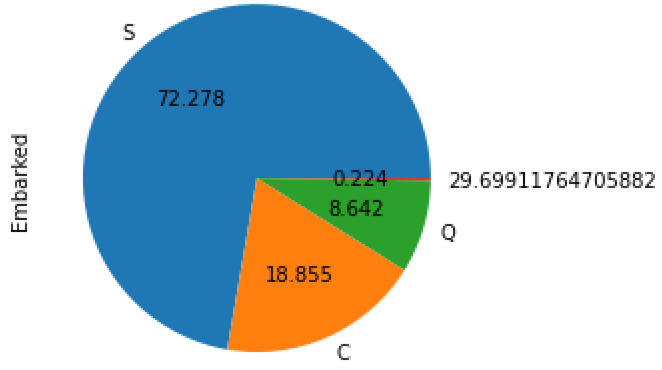
A Pie Chart is a circular statistical plot that can display only one series of data.

In [28]:

```
# This pie chart tell us that we have 2 unknown values in "Embarked" column

data_frame['Embarked'].value_counts().plot(kind='pie',autopct='%.3f')
```

Out[28]:

```
<AxesSubplot:ylabel='Embarked'>
```



In [29]:
```
# removing These two unknown values by droping these two rows
```

In [30]:
```
data_frame=data_frame.drop(data_frame[data_frame['Embarked']==29.69911764705882].index,a
```

In [31]:
```
(data_frame["Embarked"]==29.69911764705882).sum()
```

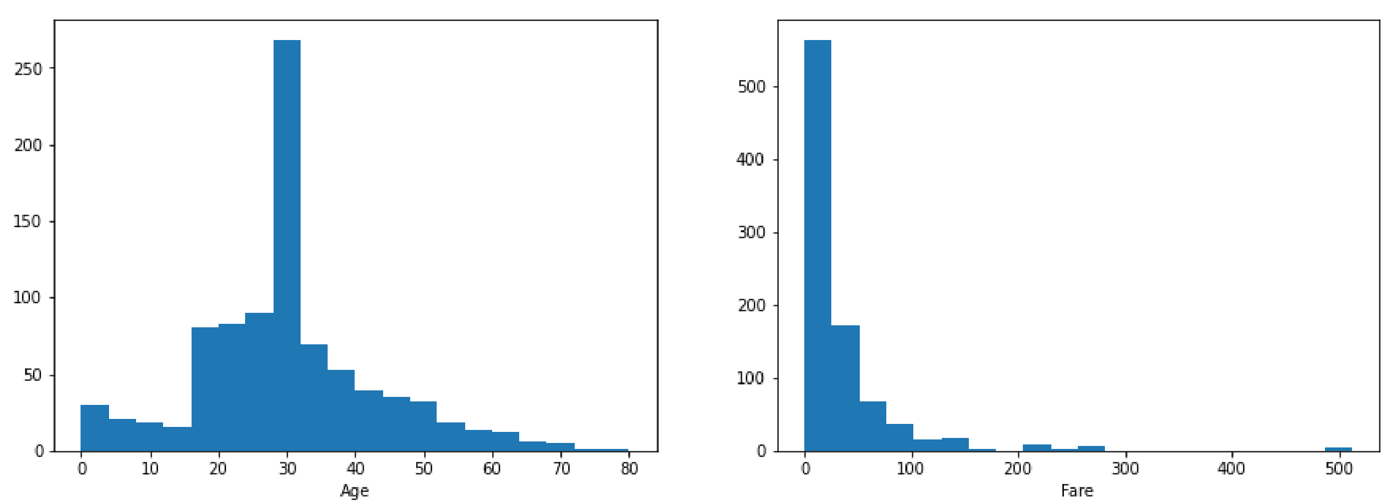
Out[31]:
```
0
```

Histogram :-A histogram is a representation of the distribution of data

Used for numerical columns analysis

In [32]:
```
plt.figure(figsize=(15,5))

# Histogram for Age,
# indicates that no. of travellers is large in the age b/w 20-35
plt.subplot(1,2,1)
plt.hist(data_frame['Age'],bins=20)
plt.xlabel("Age")

# Histogram for Fare,
# indicates that large amount of people by cheap Ticket
plt.subplot(1,2,2)
plt.hist(data_frame['Fare'],bins=20)
plt.xlabel("Fare")
```

Out[32]:
```
Text(0.5, 0, 'Fare')
```



2) Distplot :-

The Distplot depicts(to show) the data by a histogram and a line in combination to it.

kde :- used to find the skewness in data, ex:- -ve / +ve skewness or normal destribution

+ve means :- towards Right side

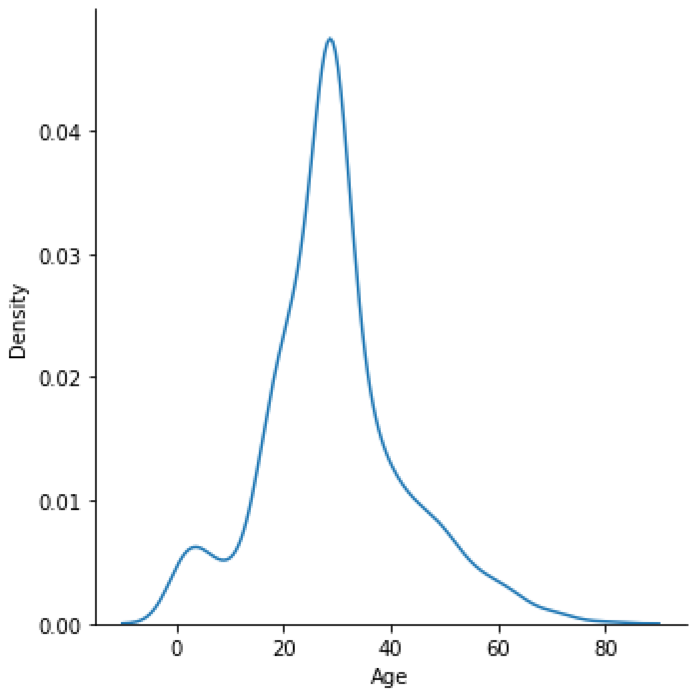
-ve means :- towards left side

0 means :- normal distribution

In [33]:
```
# skewness in Age column is towards Right side( +ve skewness)
print("skewness value in Age column :- ",data_frame['Age'].skew())
sns.displot(data_frame['Age'],kind="kde")
```

```
skewness value in Age column :-  0.4569465528010798
```

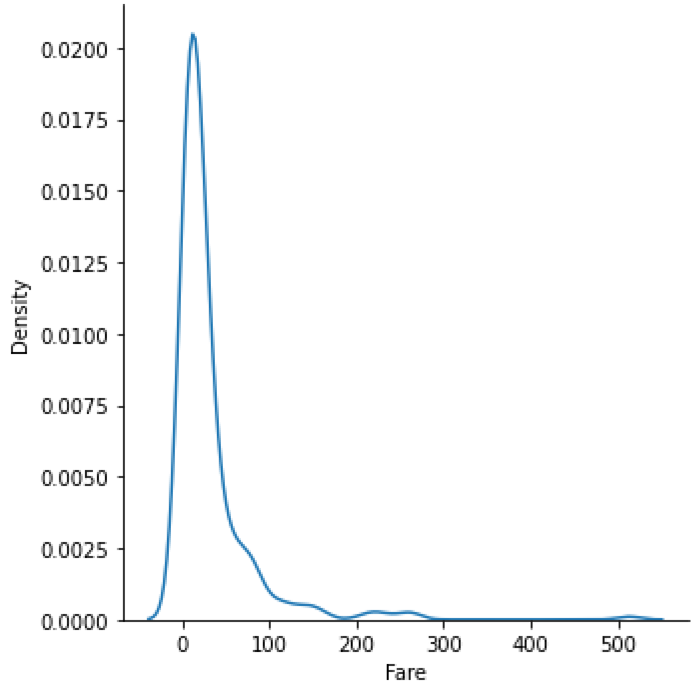
Out[33]:
```
<seaborn.axisgrid.FacetGrid at 0x289aac9780>
```



In [34]:
```
# Skewness in Fare column is towards Left side ( -ve skewness)
print("skewness value in Fare column :- ",data_frame['Fare'].skew())
sns.displot(data_frame['Fare'],kind="kde")
```

```
skewness value in Fare column :-  4.801440211044194
```

Out[34]:
```
<seaborn.axisgrid.FacetGrid at 0x289aac9810>
```

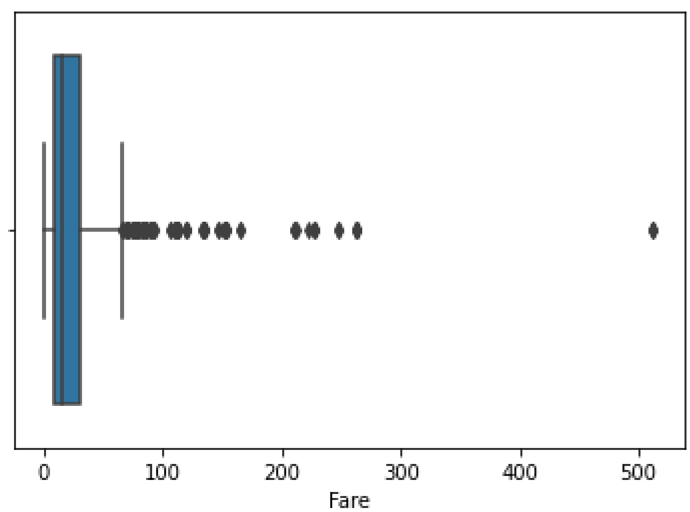


BoxPlot() :- Box Plot is the visual representation of the depicting(to show or describe) groups of numerical data through their quartiles.

Boxplot is also used for detect the outlier in data set.

In [35]:
```
# Some outliers present in Fare
sns.boxplot(x='Fare',data=data_frame)
```

Out[35]: <AxesSubplot:xlabel='Fare'>



In [36]:
```
data_frame[data_frame['Fare']>200].head()
```

Out[36]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Embarked
27	28	0	1	Fortune, Mr. Charles Alexander	male	19	3	2	19950	263.0000	S
88	89	1	1	Fortune, Miss. Mabel Helen	female	23	3	2	19950	263.0000	S
118	119	0	1	Baxter, Mr. Quigg Edmond	male	24	0	1	PC 17558	247.5208	C
258	259	1	1	Ward, Miss. Anna	female	35	0	0	PC 17755	512.3292	C

299	300	1	1	Baxter, Mrs. James (Helene DeLaudeniere Chaput)	female	50	0	1	PC 17558	247.5208	C

EDA using Bivariate and Multivariate Analysis

Bivariate analysis refers to the analysis of two variables to determine relationships between them

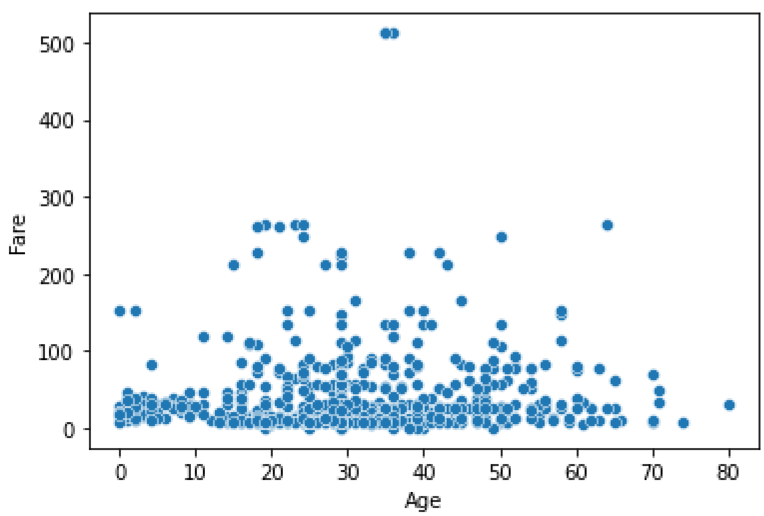
Multivariate analysis is based in observation and analysis of more than one statistical outcome variable at a time..

1) ScatterPlot (Numerical-Numerical):-

Scatter plot do good work to find relationship between numeric-numeric columns , but note we can use numeric-categorical or categorical-categorical analysis also

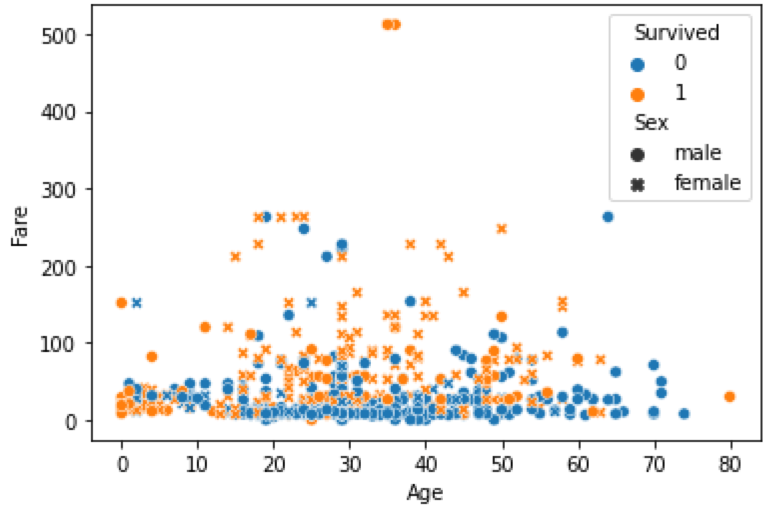
In [37]:
```
# Bi_Variate Analysis
sns.scatterplot(x='Age',y='Fare',data=data_frame)
```

Out[37]:
```
<AxesSubplot:xlabel='Age', ylabel='Fare'>
```



In [38]:
```
# Multi_Variate Analysis b/w Age and Fare
sns.scatterplot(x='Age',y='Fare',data=data_frame,hue='Survived',style='Sex')
```

Out[38]:
```
<AxesSubplot:xlabel='Age', ylabel='Fare'>
```

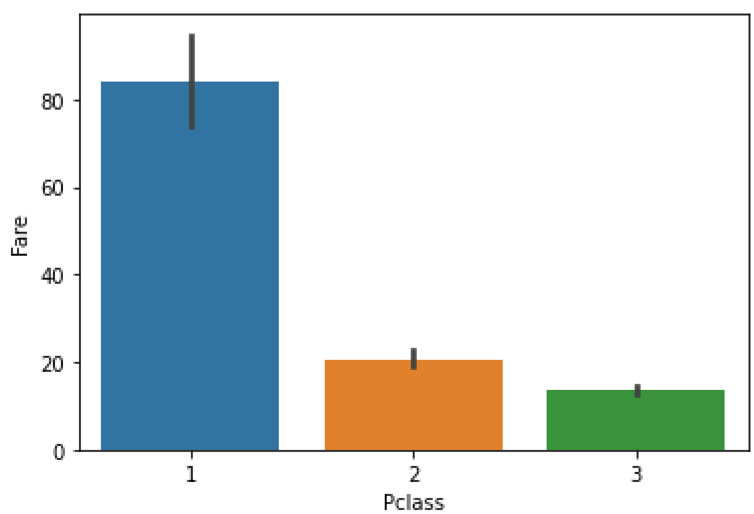


2) BarPlot(Numeric-Categorical)

A bar plot shows catergorical data as rectangular bars with heights proportional to the value they represent.

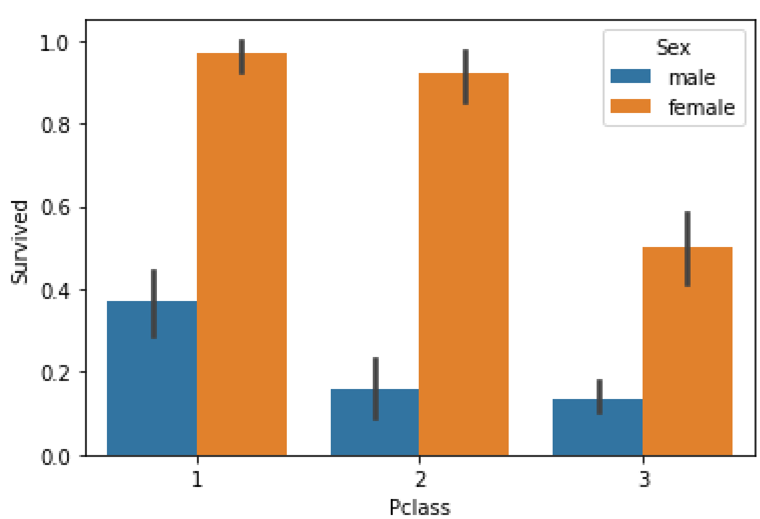
In [39]:
```
# bi-Variate Analysis
sns.barplot(x='Pclass',y='Fare',data=data_frame)
```

Out[39]:
```
<AxesSubplot:xlabel='Pclass', ylabel='Fare'>
```



In [40]:
```
# Multi-Variate Analysis
sns.barplot(x='Pclass',y='Survived',data=data_frame,hue='Sex')
```

Out[40]:
```
<AxesSubplot:xlabel='Pclass', ylabel='Survived'>
```

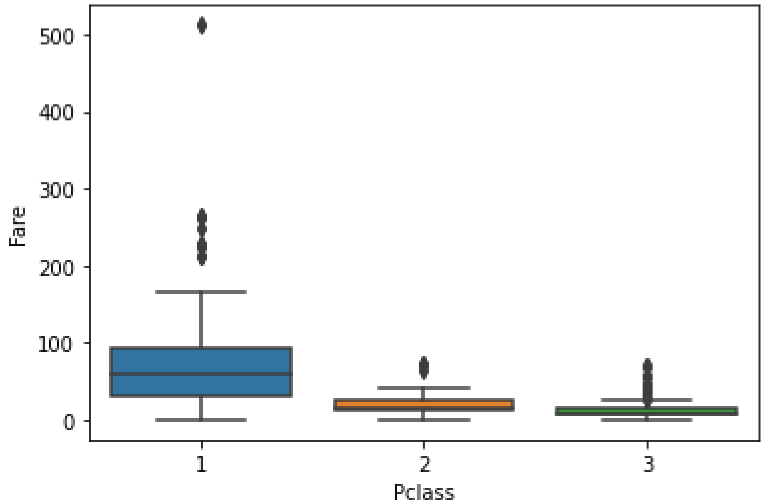


BoxPlot (Numeric-Categorical):-

Box Plot is the visual representation of the depicting groups of numerical data through their quartiles.

In [41]:
```
# for Bi-Variate Analysis
sns.boxplot(x='Pclass',y='Fare',data=data_frame)
```

Out[41]: <AxesSubplot:xlabel='Pclass', ylabel='Fare'>

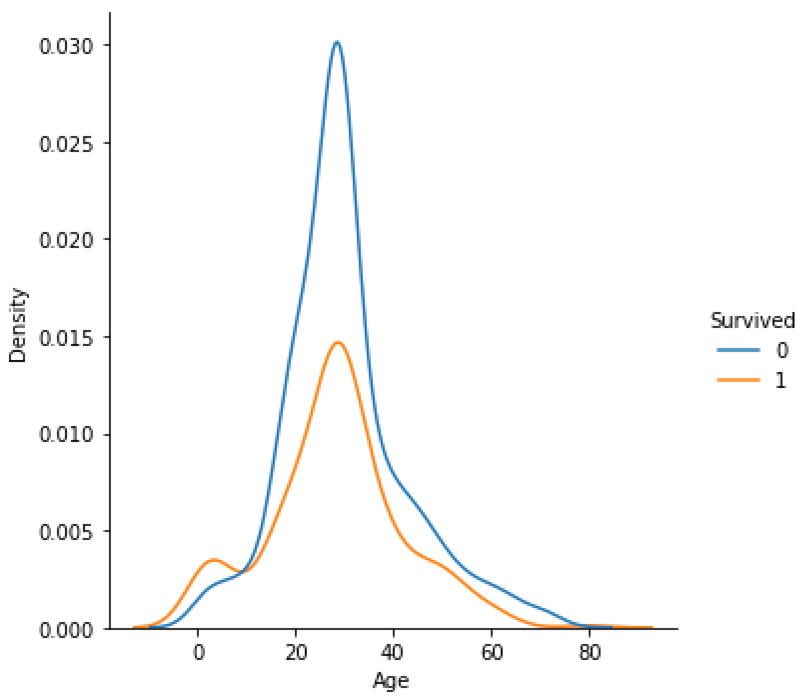


Displot (Numeric-Categorical):-

A Distplot or distribution plot, depicts the variation in the data distribution. Seaborn Distplot represents the overall distribution of continuous data variables.

In [42]:
```
# displot gives the information that,
# Survival chaances of child was much higher then Younger and older generation
sns.displot(x='Age',data=data_frame,hue='Survived',kind='kde')
```

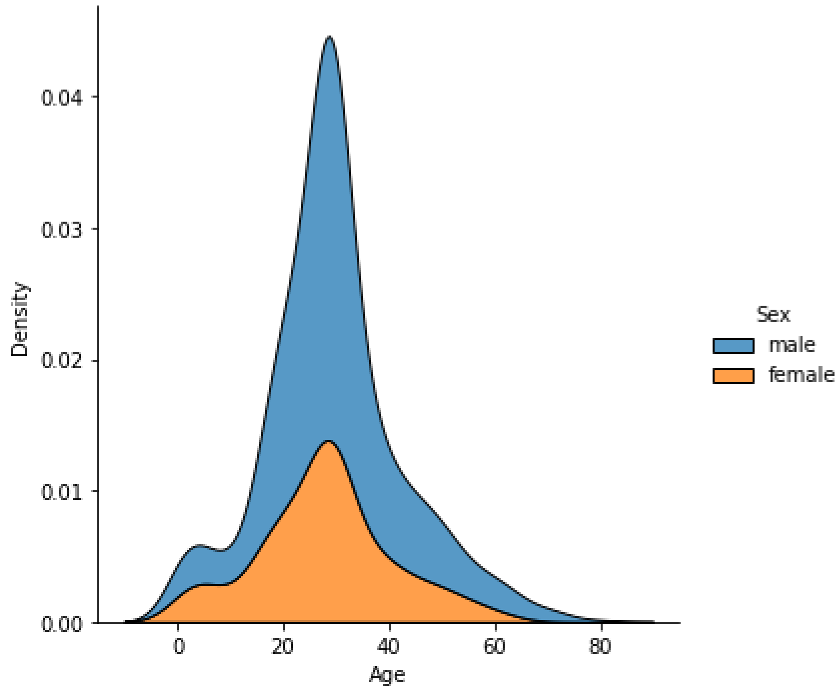
Out[42]: <seaborn.axisgrid.FacetGrid at 0x289e007a30>



In [43]:
```
sns.displot(x='Age',data=data_frame,hue='Sex',multiple='stack',kind='kde')
```

<seaborn.axisgrid.FacetGrid at 0x289e06f6d0>

Out[43]:



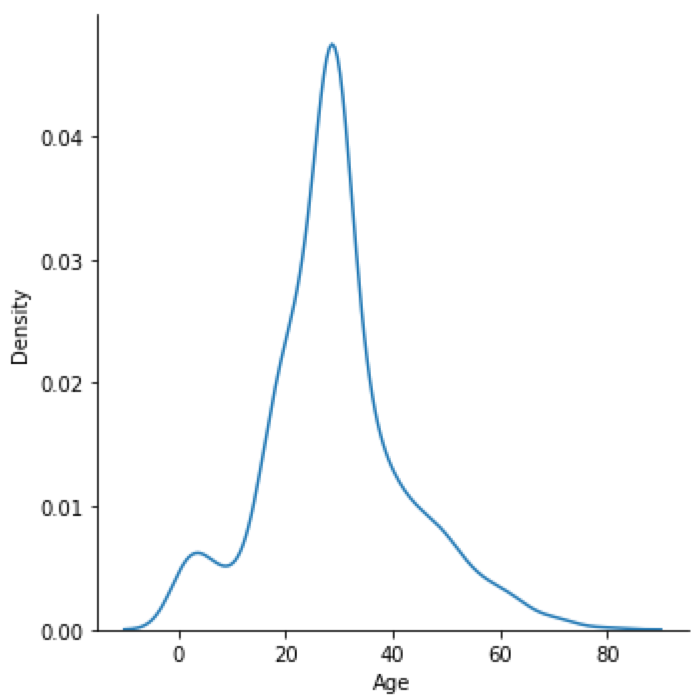
Hypothesis Testing :-

Applying hypothesis testing to find whether, Age column data is normal distributed or not.

In [52]:
```
# Graph shows that it is sligtly -ve curve (left skewness)
# thus is is not nomal distributed
print("Skewness in Age column is ",data_frame['Age'].skew())
sns.displot(x='Age',data=data_frame,kind='kde')
```

```
Skewness in Age column is  0.4569465528010798
```

Out[52]:
```
<seaborn.axisgrid.FacetGrid at 0x289ac02e60>
```



In [53]:
```
H0 = 'Data is normal'
Ha = 'Data is not normal'
alpha = 0.05
```

In [54]:
```
p_value = round(shapiro(data_frame['Age'])[1], 2)
```

In [55]:
```
# Shapiro-Wilk's test gives that our Age column data is not normal distributed
# hence proved
if p_value > alpha:
    print(f"{p_value} > {alpha}. We fail to reject Null Hypothesis. {H0}")
else:
    print(f"{p_value} <= {alpha}. We reject Null Hypothesis. {Ha}")
```

```
0.0 <= 0.05. We reject Null Hypothesis. Data is not normal
```

In [ ]: