Assignment 8Data Visualization I

Use the inbuilt dataset 'titanic'. The dataset contains 891 rows and contains information about the passengers who boarded the unfortunate Titanic ship. Use the Seaborn library to see if we can find any patterns in the data. Write a code to check how the price of the ticket (column name: 'fare') for each passenger is distributed by plotting a histogram

Assignment 9Data Visualization II

1. Use the inbuilt dataset 'titanic' as used in the above problem. Plot a box plot for distribution of age with respect to each gender along with the information about whether they survived or not. (Column names : 'sex' and 'age') Write observations on the inference from the above statistics.

In [7]:
```
import pandas as pd
import numpy as np

import matplotlib.pyplot as plt
import seaborn as sns
dataset=pd.read_csv("C:\\Users\\Admin\\Desktop\\dataset\\Titanic-Dataset.csv")
```

In [8]:
```
dataset.head()
```

Out[8]:

	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

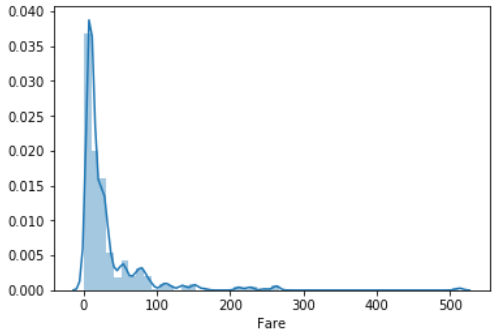
Distributional plots, as the name suggests are type of plots that show the statistical distribution of data. The distplot() shows the histogram distribution of data for a single column. The column name is passed as a parameter to the distplot() function. Let's see how the price of the ticket for each passenger is distributed

In [ ]:
```
Distribution Plots
a. Distplot
b. jointplot
c. Pairplot
d. Rugplot
These plots help us to visualize the distribution of data. We can use these plots to understand the mean, median, range, vari
ance, deviation, etc of the data.
a. Distplot
Dist plot gives us the histogram of the selected continuous variable.
It is an example of a univariate analysis.
We can change the number of bins i.e. number of vertical bars in a histogram
```

In [10]:
```
sns.distplot(dataset['Fare'])
```

```
C:\ProgramData\Anaconda3\lib\site-packages\scipy\stats\stats.py:1713: FutureWarning: Using a non-tuple sequence for multidime
nsional indexing is deprecated; use `arr[tuple(seq)]` instead of `arr[seq]`. In the future this will be interpreted as an arr
ay index, `arr[np.array(seq)]`, which will result either in an error or a different result.
  return np.add.reduce(sorted[indexer] * weights, axis=axis) / sumval
```

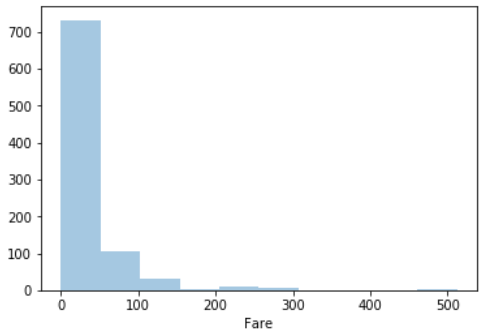
Out[10]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x18ab610>
```



OUTPUT:You can see that most of the tickets have been solved between 0-50 dollars. The line that you see represents the kernel density estimation.

In [12]:
```
#Kernal density estimation
#You can also pass the value for the bins parameter in order to see more or less details in the graph.
sns.distplot(dataset['Fare'], kde=False, bins=10)
```

Out[12]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x19ef270>
```

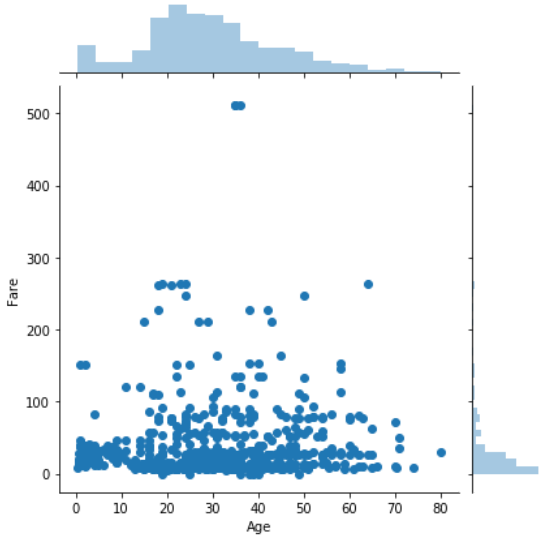


b. Joint Plot It is the combination of the distplot of two variables. It is an example of bivariate analysis. We additionally obtain a scatter plot between the variable to reflecting their linear relationship. We can customize the scatter plot into a hexagonal plot, where, more the color intensity, the more will be the number of observations.

In [14]:
```
sns.jointplot(x='Age', y='Fare', data=dataset)
```

```
C:\ProgramData\Anaconda3\lib\site-packages\scipy\stats\stats.py:1713: FutureWarning: Using a non-tuple sequence for multidime
nsional indexing is deprecated; use `arr[tuple(seq)]` instead of `arr[seq]`. In the future this will be interpreted as an arr
ay index, `arr[np.array(seq)]`, which will result either in an error or a different result.
  return np.add.reduce(sorted[indexer] * weights, axis=axis) / sumval
```

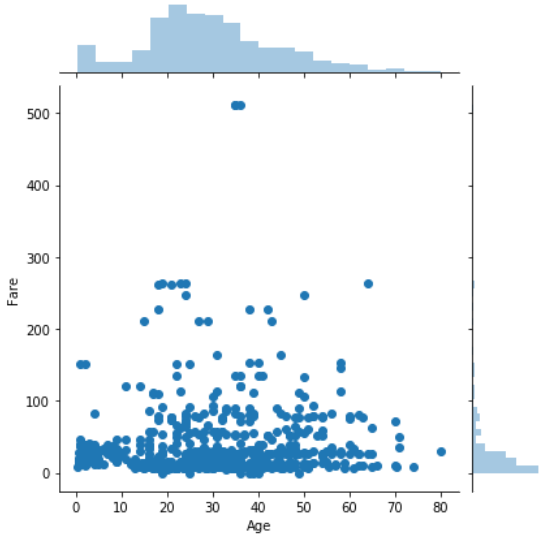
Out[14]:
```
<seaborn.axisgrid.JointGrid at 0x49f0670>
```



In [16]:
```
sns.jointplot(x = df['Age'], y = df['Fare'], kind = 'scatter')
```

```
C:\ProgramData\Anaconda3\lib\site-packages\scipy\stats\stats.py:1713: FutureWarning: Using a non-tuple sequence for multidime
nsional indexing is deprecated; use `arr[tuple(seq)]` instead of `arr[seq]`. In the future this will be interpreted as an arr
ay index, `arr[np.array(seq)]`, which will result either in an error or a different result.
  return np.add.reduce(sorted[indexer] * weights, axis=axis) / sumval
```

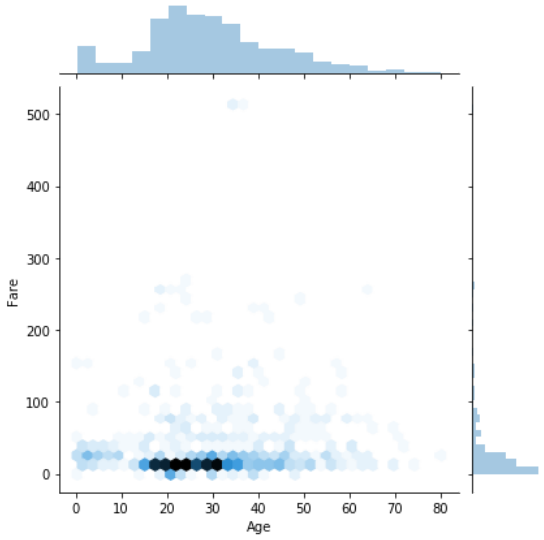
Out[16]:
```
<seaborn.axisgrid.JointGrid at 0x4ac27b0>
```



In [17]:
```
sns.jointplot(x = df['Age'], y = df['Fare'], kind = 'hex')
```

```
C:\ProgramData\Anaconda3\lib\site-packages\scipy\stats\stats.py:1713: FutureWarning: Using a non-tuple sequence for multidime
nsional indexing is deprecated; use `arr[tuple(seq)]` instead of `arr[seq]`. In the future this will be interpreted as an arr
ay index, `arr[np.array(seq)]`, which will result either in an error or a different result.
  return np.add.reduce(sorted[indexer] * weights, axis=axis) / sumval
```

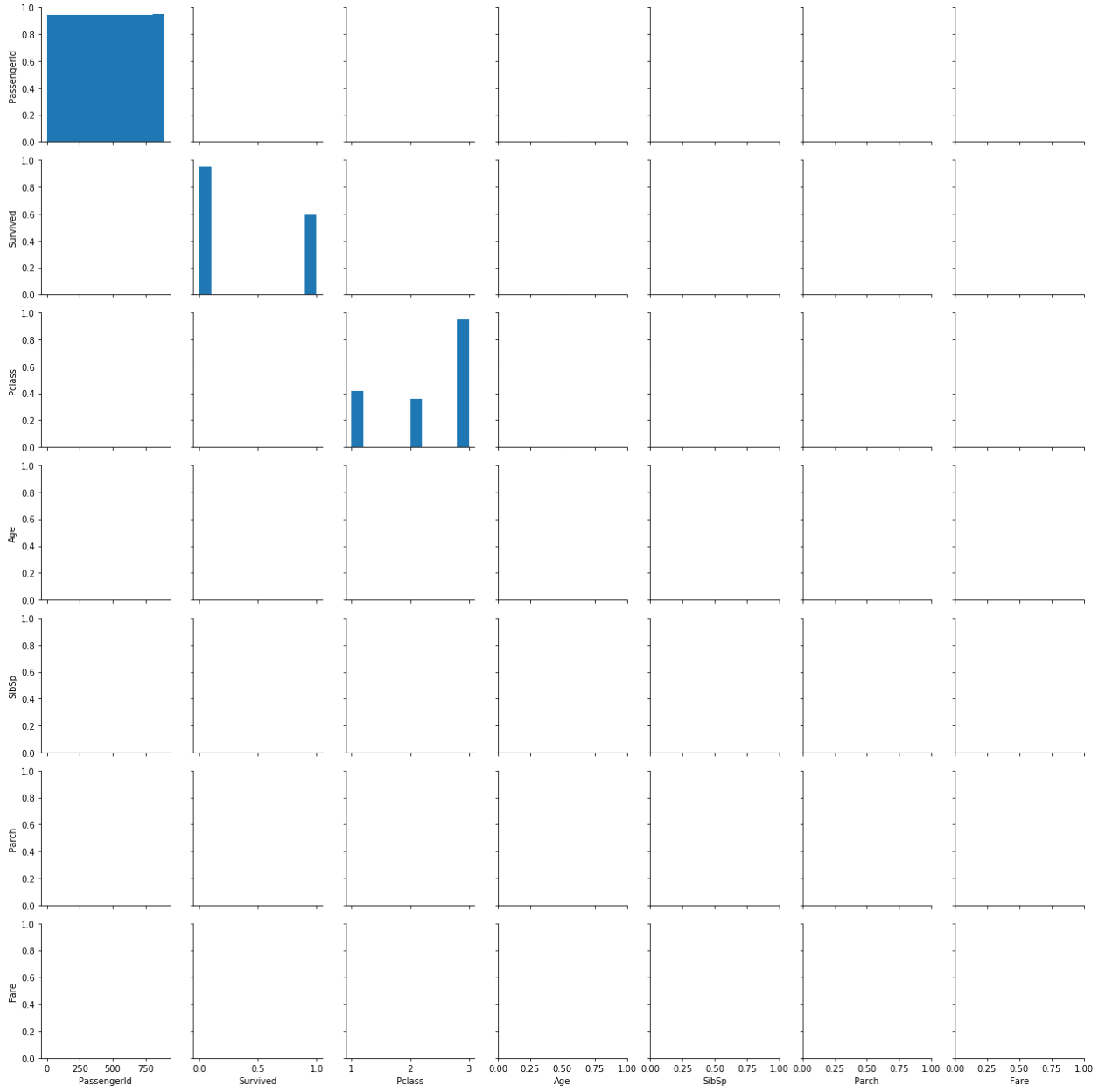
Out[17]:
```
<seaborn.axisgrid.JointGrid at 0xd15d6f0>
```



We can see that there no appropriate linear relation between age and fare. kind = ‘hex’ provides the hexagonal plot and kind = ‘reg’ provides a regression line on the graph.

In [19]:
```
sns.pairplot(dataset)
```

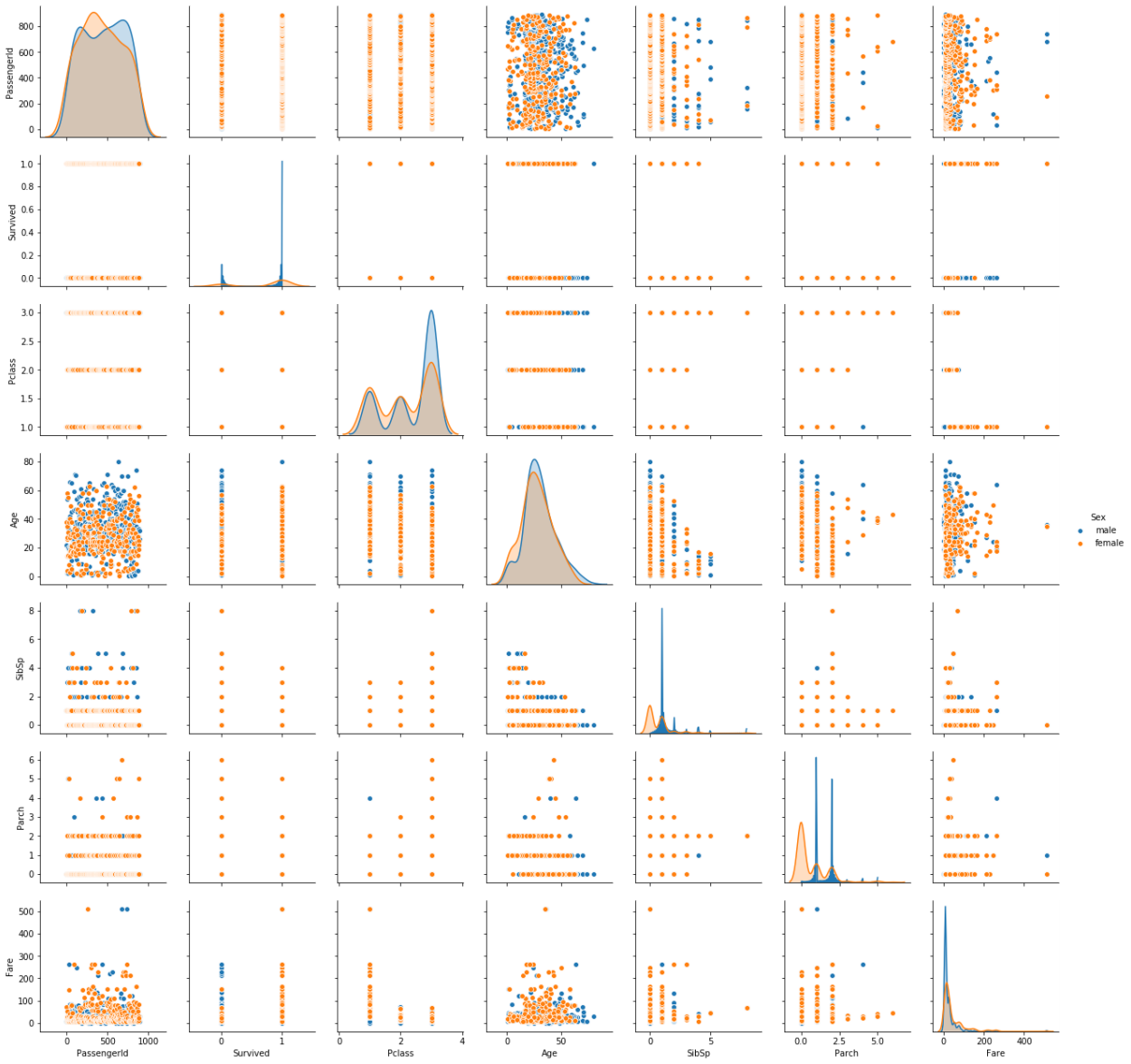
```
C:\ProgramData\Anaconda3\lib\site-packages\numpy\core\_methods.py:32: RuntimeWarning: invalid value encountered in reduce
  return umr_minimum(a, axis, None, out, keepdims, initial)
C:\ProgramData\Anaconda3\lib\site-packages\numpy\core\_methods.py:28: RuntimeWarning: invalid value encountered in reduce
  return umr_maximum(a, axis, None, out, keepdims, initial)

---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
<ipython-input-19-e506b35fe370> in <module>()
----> 1 sns.pairplot(dataset)

C:\ProgramData\Anaconda3\lib\site-packages\seaborn\axisgrid.py in pairplot(data, hue, hue_order, palette, vars, x_vars, y_var
s, kind, diag_kind, markers, height, aspect, dropna, plot_kws, diag_kws, grid_kws, size)
   2105     if grid.square_grid:
   2106         if diag_kind == "hist":
-> 2107             grid.map_diag(plt.hist, **diag_kws)
   2108         elif diag_kind == "kde":
   2109             diag_kws.setdefault("shade", True)

C:\ProgramData\Anaconda3\lib\site-packages\seaborn\axisgrid.py in map_diag(self, func, **kwargs)
   1397                     color = fixed_color
   1398
-> 1399                 func(data_k, label=label_k, color=color, **kwargs)
   1400
   1401             self._clean_axis(ax)

C:\ProgramData\Anaconda3\lib\site-packages\matplotlib\pyplot.py in hist(x, bins, range, density, weights, cumulative, bottom,
histtype, align, orientation, rwidth, log, color, label, stacked, normed, hold, data, **kwargs)
   3135                      histtype=histtype, align=align, orientation=orientation,
   3136                      rwidth=rwidth, log=log, color=color, label=label,
-> 3137                      stacked=stacked, normed=normed, data=data, **kwargs)
   3138     finally:
   3139         ax._hold = washold

C:\ProgramData\Anaconda3\lib\site-packages\matplotlib\__init__.py in inner(ax, *args, **kwargs)
   1865                         "the Matplotlib list!)" % (label_namer, func.__name__),
   1866                         RuntimeWarning, stacklevel=2)
-> 1867             return func(ax, *args, **kwargs)
   1868
   1869         inner.__doc__ = _add_data_doc(inner.__doc__,

C:\ProgramData\Anaconda3\lib\site-packages\matplotlib\axes\_axes.py in hist(***failed resolving arguments***)
   6637             # this will automatically overwrite bins,
   6638             # so that each histogram uses the same bins
-> 6639             m, bins = np.histogram(x[i], bins, weights=w[i], **hist_kwargs)
   6640             m = m.astype(float)  # causes problems later if it's an int
   6641             if mlast is None:

C:\ProgramData\Anaconda3\lib\site-packages\numpy\lib\histograms.py in histogram(a, bins, range, normed, weights, density)
    700     a, weights = _ravel_and_check_weights(a, weights)
    701
--> 702     bin_edges, uniform_bins = _get_bin_edges(a, bins, range, weights)
    703
    704     # Histogram is an integer or a float array depending on the weights.

C:\ProgramData\Anaconda3\lib\site-packages\numpy\lib\histograms.py in _get_bin_edges(a, bins, range, weights)
    353             raise ValueError('`bins` must be positive, when an integer')
    354
--> 355         first_edge, last_edge = _get_outer_edges(a, range)
    356
    357     elif np.ndim(bins) == 1:

C:\ProgramData\Anaconda3\lib\site-packages\numpy\lib\histograms.py in _get_outer_edges(a, range)
    240         if first_edge > last_edge:
    241             raise ValueError(
--> 242                 'max must be larger than min in range parameter.')
    243         if not (np.isfinite(first_edge) and np.isfinite(last_edge)):
    244             raise ValueError(

ValueError: max must be larger than min in range parameter.
```


PassengerId
Survived
Pclass
Age
SibSp
Parch
Fare
0 250 500 750
0.0 0.5 1.0
1 2 3
0.00 0.25 0.50 0.75 1.00

In [21]:
```
sns.pairplot(dataset, hue='Sex')
```

```
C:\ProgramData\Anaconda3\lib\site-packages\scipy\stats\stats.py:1713: FutureWarning: Using a non-tuple sequence for multidime
nsional indexing is deprecated; use `arr[tuple(seq)]` instead of `arr[seq]`. In the future this will be interpreted as an arr
ay index, `arr[np.array(seq)]`, which will result either in an error or a different result.
  return np.add.reduce(sorted[indexer] * weights, axis=axis) / sumval
C:\ProgramData\Anaconda3\lib\site-packages\statsmodels\nonparametric\kde.py:448: RuntimeWarning: invalid value encountered in
greater
  X = X[np.logical_and(X > clip[0], X < clip[1])] # won't work for two columns.
C:\ProgramData\Anaconda3\lib\site-packages\statsmodels\nonparametric\kde.py:448: RuntimeWarning: invalid value encountered in
less
  X = X[np.logical_and(X > clip[0], X < clip[1])] # won't work for two columns.
```

Out[21]: <seaborn.axisgrid.PairGrid at 0xefe5730>

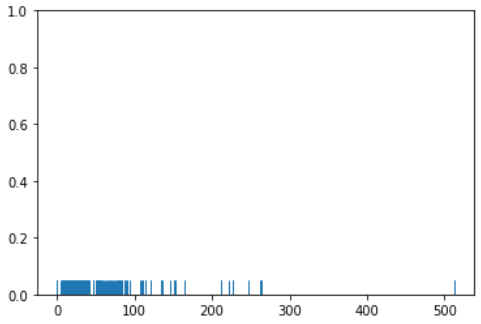


OUTPUT:In the output you can see the information about the males in orange and the information about the female in blue (as shown in the legend). From the joint plot on the top left, you can clearly see that among the surviving passengers, the majority were female.

d.Rug Plot It draws a dash mark instead of a uniform distribution as in distplot. It is an example of a univariate analysis.

In [22]:
```
sns.rugplot(dataset['Fare'])
```

Out[22]:
```
<matplotlib.axes._subplots.AxesSubplot at 0xd6670f0>
```



OUTPUT:From the output, you can see that as was the case with the distplot(), most of the instances for the fares have values between 0 and 100.

Categorical plots, as the name suggests are normally used to plot categorical data. The categorical plots plot the values in the categorical column against another categorical column or a numeric column. Let's see some of the most commonly used categorical data.

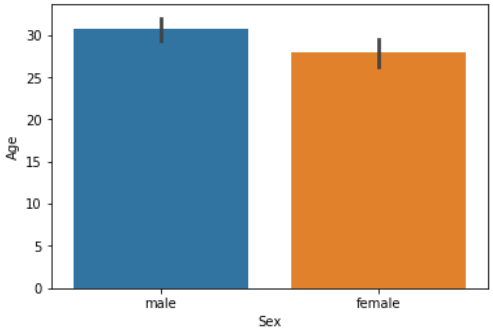
a.The Bar Plot

It is an example of bivariate analysis. On the x-axis, we have a categorical variable and on the y-axis, we have a continuous variable. The barplot() is used to display the mean value for each value in a categorical column, against a numeric column. The first parameter is the categorical column, the second parameter is the numeric column while the third parameter is the dataset. For instance, if you want to know the mean value of the age of the male and female passengers, you can use the bar plot as follows.

In [24]:
```
sns.barplot(x='Sex', y='Age', data=dataset)
```

```
C:\ProgramData\Anaconda3\lib\site-packages\scipy\stats\stats.py:1713: FutureWarning: Using a non-tuple sequence for multidime
nsional indexing is deprecated; use `arr[tuple(seq)]` instead of `arr[seq]`. In the future this will be interpreted as an arr
ay index, `arr[np.array(seq)]`, which will result either in an error or a different result.
  return np.add.reduce(sorted[indexer] * weights, axis=axis) / sumval
```

Out[24]:
```
<matplotlib.axes._subplots.AxesSubplot at 0xf6980b0>
```

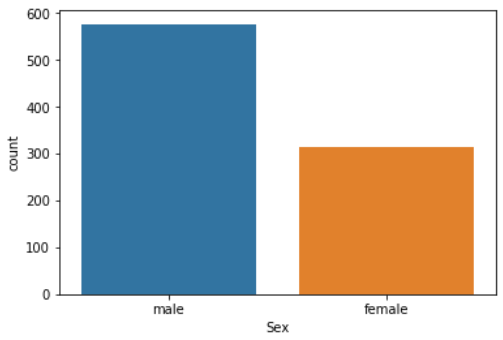


OUTPUT:From the output, you can clearly see that the average age of male passengers is just less than 40 while the average age of female passengers is around 33.

b.Count Plot It counts the number of occurrences of categorical variables. It is an example of a univariate analysis. The count plot is similar to the bar plot, however it displays the count of the categories in a specific column. For instance, if we want to count the number of males and women passenger we can do so using count plot as follows:

In [25]:
```
sns.countplot(x='Sex', data=dataset)
```

Out[25]:
```
<matplotlib.axes._subplots.AxesSubplot at 0xf91a810>
```

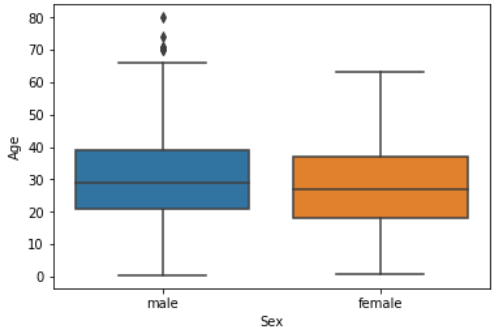


c. Box Plot It is a 5 point summary plot. It gives the information about the maximum, minimum, mean, first quartile, and third quartile of a continuous variable. Also, it equips us with knowledge of outliers. We can plot this for a single continuous variable or can analyze different categorical variables based on a continuous variable.

Now let's plot a box plot that displays the distribution for the age with respect to each gender. You need to pass the categorical column as the first parameter (which is sex in our case) and the numeric column (age in our case) as the second parameter. Finally, the dataset is passed as the third parameter, take a look at the following script:

```
In [26]: sns.boxplot(x='Sex', y='Age', data=dataset)
```

```
Out[26]: <matplotlib.axes._subplots.AxesSubplot at 0xf66c730>
```



OUTPUT:Let's try to understand the box plot for female. The first quartile starts at around 5 and ends at 22 which means that 25% of the passengers are aged between 5 and 25. The second quartile starts at around 23 and ends at around 32 which means that 25% of the passengers are aged between 23 and 32. Similarly, the third quartile starts and ends between 34 and 42, hence 25% passengers are aged within this range and finally the fourth or last quartile starts at 43 and ends around 65.

1. Using hue parameter:

While the points are plotted in two dimensions, another dimension can be added to the plot by coloring the points according to a third variable.

Syntax:

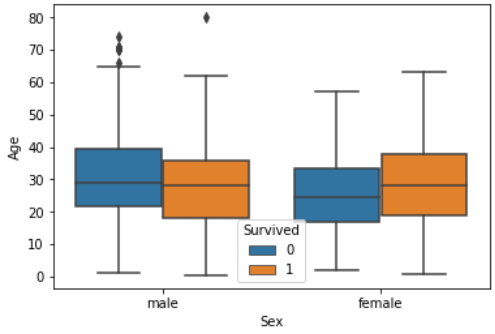
seaborn.boxplot(x, y, hue, data);

if you want to see the box plots of forage of passengers of both genders, along with the information about whether or not they survived, you can pass the survived as value to the hue parameter as shown below:

OUTPUT:Now in addition to the information about the age of each gender, you can also see the distribution of the passengers who survived. For instance, you can see that among the male passengers, on average more younger people survived as compared to the older ones. Similarly, you can see that the variation among the age of female passengers who did not survive is much greater than the age of the surviving female passengers.

```
In [27]: sns.boxplot(x='Sex', y='Age', data=dataset, hue="Survived")
```

```
Out[27]: <matplotlib.axes._subplots.AxesSubplot at 0xf8bb4f0>
```

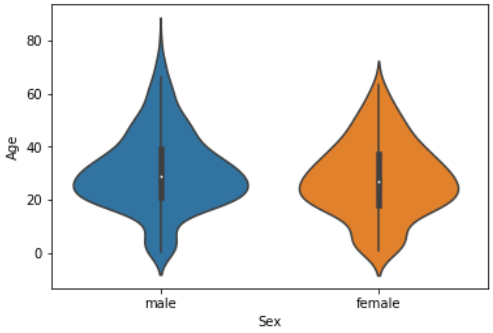


d.VIOLIN PLOT The violin plot is similar to the box plot, however, the violin plot allows us to display all the components that actually correspond to the data point. The violinplot() function is used to plot the violin plot. Like the box plot, the first parameter is the categorical column, the second parameter is the numeric column while the third parameter is the dataset.

In [29]:
```
sns.violinplot(x='Sex', y='Age', data=dataset)
```

```
C:\ProgramData\Anaconda3\lib\site-packages\scipy\stats\stats.py:1713: FutureWarning: Using a non-tuple sequence for multidime
nsional indexing is deprecated; use `arr[tuple(seq)]` instead of `arr[seq]`. In the future this will be interpreted as an arr
ay index, `arr[np.array(seq)]`, which will result either in an error or a different result.
  return np.add.reduce(sorted[indexer] * weights, axis=axis) / sumval
```

Out[29]:
```
<matplotlib.axes._subplots.AxesSubplot at 0xf986090>
```



OUtPUT:You can see from the figure above that violin plots provide much more information about the data as compared to the box plot. Instead of plotting the quartile, the violin plot allows us to see all the components that actually correspond to the data. The area where the violin plot is thicker has a higher number of instances for the age. For instance, from the violin plot for males, it is clearly evident that the number of passengers with age between 20 and 40 is higher than all the rest of the age brackets.

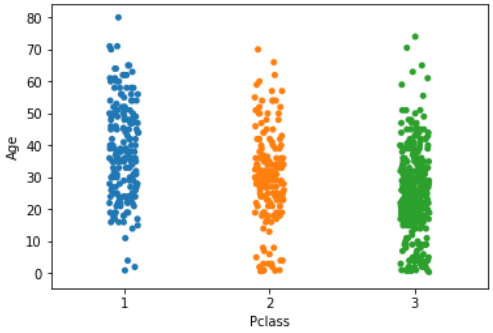
Advanced Plots As the name suggests, they are advanced because they ought to fuse the distribution and categorical encodings.

a.Strip Plot

It's a plot between a continuous variable and a categorical variable. It plots as a scatter plot but supplementarily uses categorical encodings of the categorical variable. The stripplot() function is used to plot the violin plot. Like the box plot, the first parameter is the categorical column, the second parameter is the numeric column while the third parameter is the dataset. Look at the following script:

In [31]:
```
sns.stripplot(y = dataset['Age'], x = dataset['Pclass'])
```

Out[31]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x49ad1b0>
```

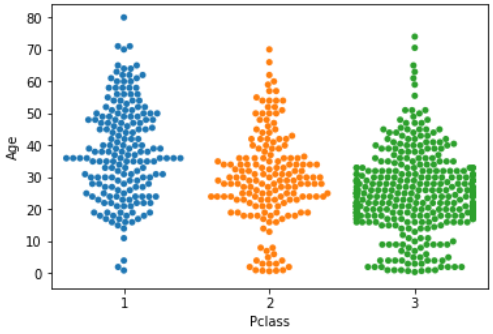


OUTPUT:We can observe that in class 1 and class 2, children around 10 years are not present and the people having age above 60 are mostly accommodated in class 1.

In [ ]:
```
b. Swarm Plot
It is the combination of a strip plot and a violin plot.
Along with the number of data points, it also provides their respective distribution.
```

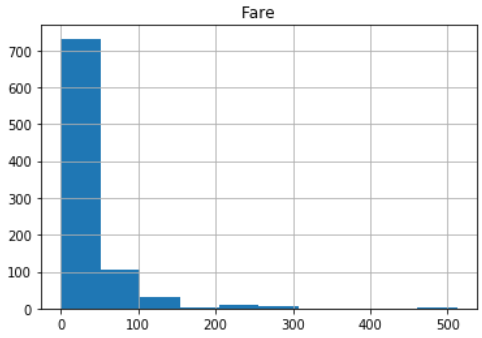
In [34]:
```
sns.swarmplot(y = dataset['Age'], x = dataset['Pclass'])
```

Out[34]:
```
<matplotlib.axes._subplots.AxesSubplot at 0xe35c870>
```



In [35]:
```
dataset.hist('Fare')
```

Out[35]:
```
array([[<matplotlib.axes._subplots.AxesSubplot object at 0x0F892A50>]],
      dtype=object)
```



In [ ]: