practical-08

April 26, 2024

[1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

[2]:
```
dataset = sns.load_dataset('titanic')
dataset
```

[2]:
```
     survived  pclass     sex   age  sibsp  parch     fare embarked   class  \
0           0       3    male  22.0      1      0   7.2500        S   Third
1           1       1  female  38.0      1      0  71.2833        C   First
2           1       3  female  26.0      0      0   7.9250        S   Third
3           1       1  female  35.0      1      0  53.1000        S   First
4           0       3    male  35.0      0      0   8.0500        S   Third
..        …     …     …   …    …    …      …      …     …
886         0       2    male  27.0      0      0  13.0000        S  Second
887         1       1  female  19.0      0      0  30.0000        S   First
888         0       3  female   NaN      1      2  23.4500        S   Third
889         1       1    male  26.0      0      0  30.0000        C   First
890         0       3    male  32.0      0      0   7.7500        Q   Third

       who  adult_male deck  embark_town alive  alone
0      man        True  NaN  Southampton    no  False
1    woman       False    C    Cherbourg   yes  False
2    woman       False  NaN  Southampton   yes   True
3    woman       False    C  Southampton   yes  False
4      man        True  NaN  Southampton    no   True
..     …         …  …          …   …    …
886    man        True  NaN  Southampton    no   True
887  woman       False    B  Southampton   yes   True
888  woman       False  NaN  Southampton    no  False
889    man        True    C    Cherbourg   yes   True
890    man        True  NaN   Queenstown    no   True

[891 rows x 15 columns]
```

[3]:
```
dataset.head()
```

```
[3]:    survived  pclass     sex   age  sibsp  parch     fare embarked  class  \
     0         0       3    male  22.0      1      0   7.2500        S  Third
     1         1       1  female  38.0      1      0  71.2833        C  First
     2         1       3  female  26.0      0      0   7.9250        S  Third
     3         1       1  female  35.0      1      0  53.1000        S  First
     4         0       3    male  35.0      0      0   8.0500        S  Third

          who  adult_male deck  embark_town alive  alone
     0    man        True  NaN  Southampton    no  False
     1  woman       False    C    Cherbourg   yes  False
     2  woman       False  NaN  Southampton   yes   True
     3  woman       False    C  Southampton   yes  False
     4    man        True  NaN  Southampton    no   True
```

```
[4]: dataset.isnull().sum()
```

```
[4]: 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
```

```
[5]: dataset.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 891 entries, 0 to 890
Data columns (total 15 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   survived     891 non-null    int64
 1   pclass       891 non-null    int64
 2   sex          891 non-null    object
 3   age          714 non-null    float64
 4   sibsp        891 non-null    int64
 5   parch        891 non-null    int64
 6   fare         891 non-null    float64
 7   embarked     889 non-null    object
 8   class        891 non-null    category
 9   who          891 non-null    object
 10  adult_male   891 non-null    bool
 11  deck         203 non-null    category
 12  embark_town  889 non-null    object
 13  alive        891 non-null    object
 14  alone        891 non-null    bool
dtypes: bool(2), category(2), float64(2), int64(4), object(5)
memory usage: 80.7+ KB
```

[6]:

```
# Finding patterns of data. --> Patterns of data can be find out with the help␣
 ↪of different types of plots
# A. Distribution Plots:
```

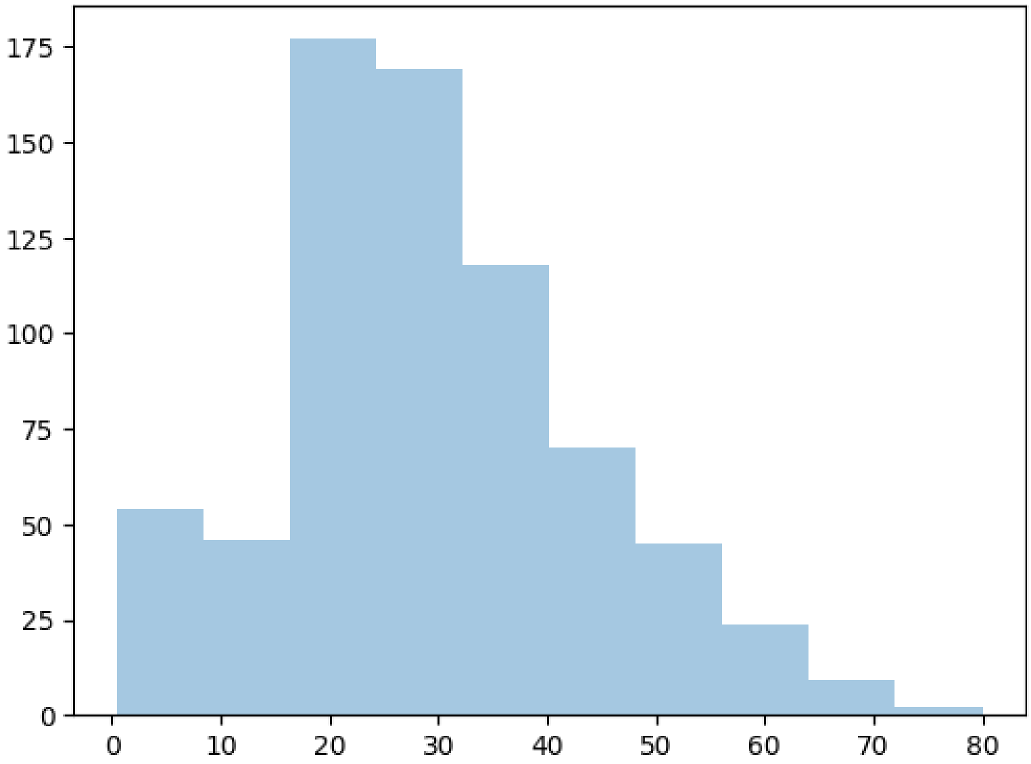
[7]:

```
# 1. Distplot
sns.distplot(x = dataset['age'], bins = 10,kde=False)
```

```
C:\Users\gugal\AppData\Local\Temp\ipykernel_19856\3431948374.py:2: UserWarning:

`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

  sns.distplot(x = dataset['age'], bins = 10,kde=False)
```

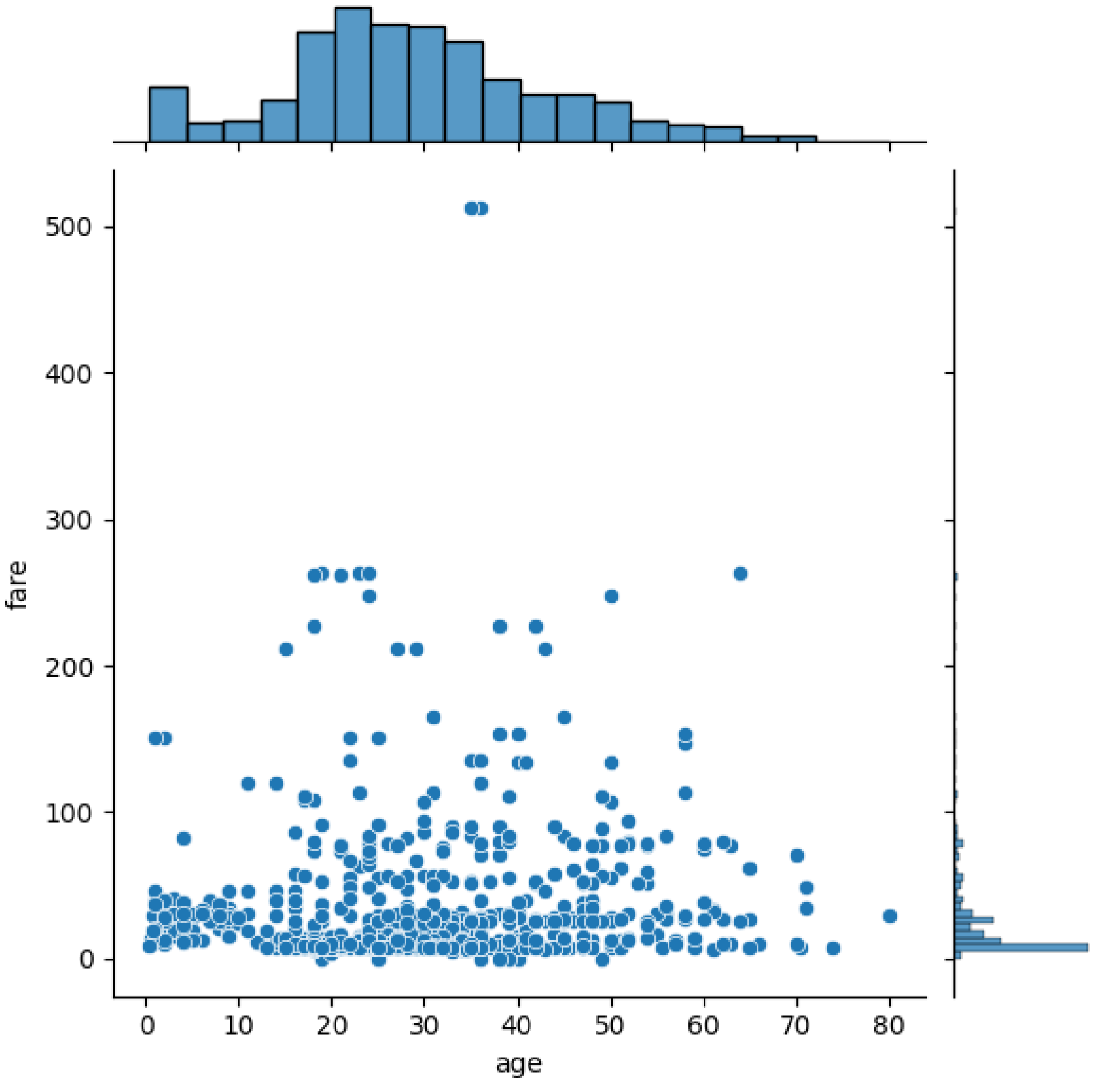
[7]:

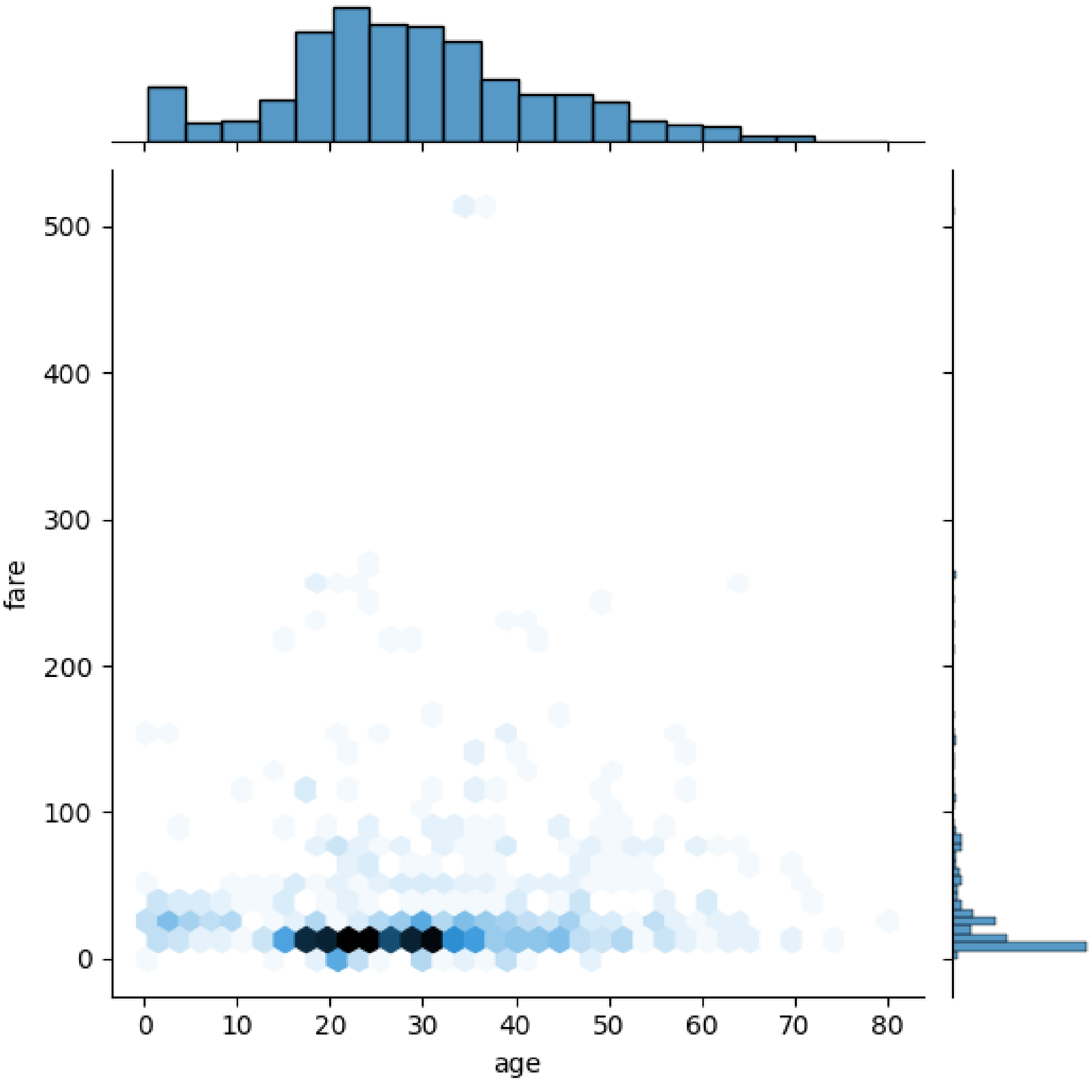
```
<Axes: >
```



```
[8]: # 2. Joint Plot
     # For Plot 1:
     sns.jointplot(x = dataset['age'], y = dataset['fare'], kind = 'scatter')
     # For Plot 2:
     sns.jointplot(x = dataset['age'], y = dataset['fare'], kind = 'hex')
```

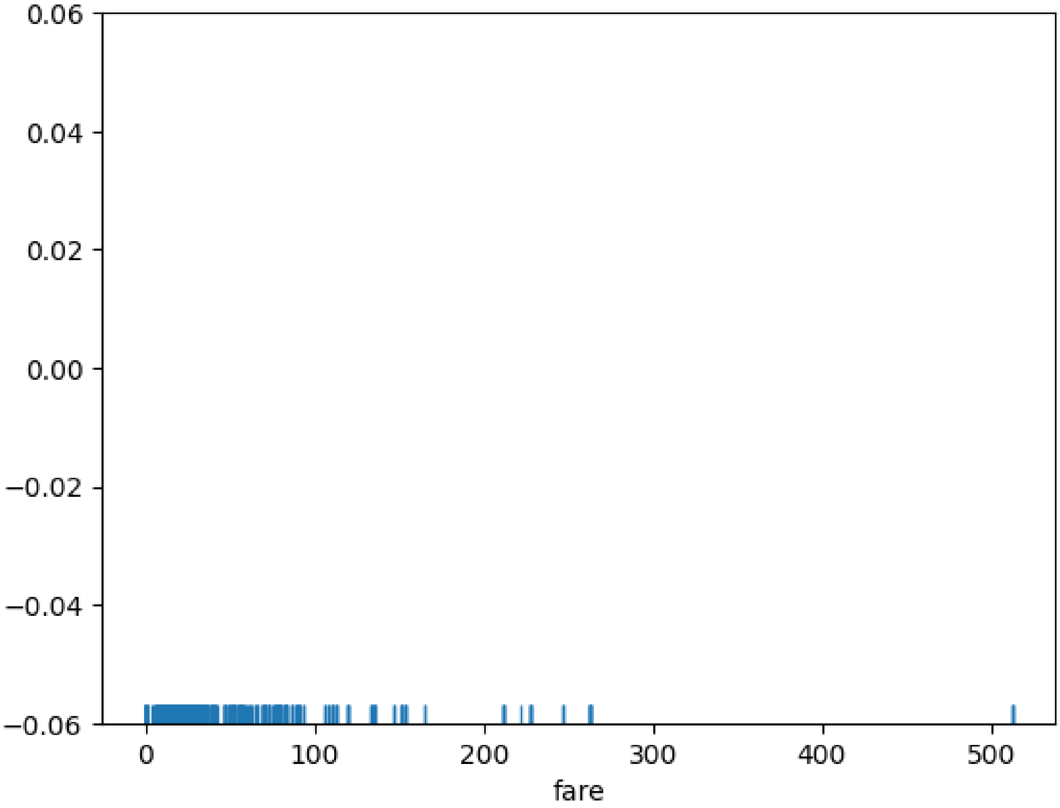
```
[8]: <seaborn.axisgrid.JointGrid at 0x21c05499f70>
```


500
400
300
200
100
0
fare
0
10
20
30
40
50
60
70
80
age



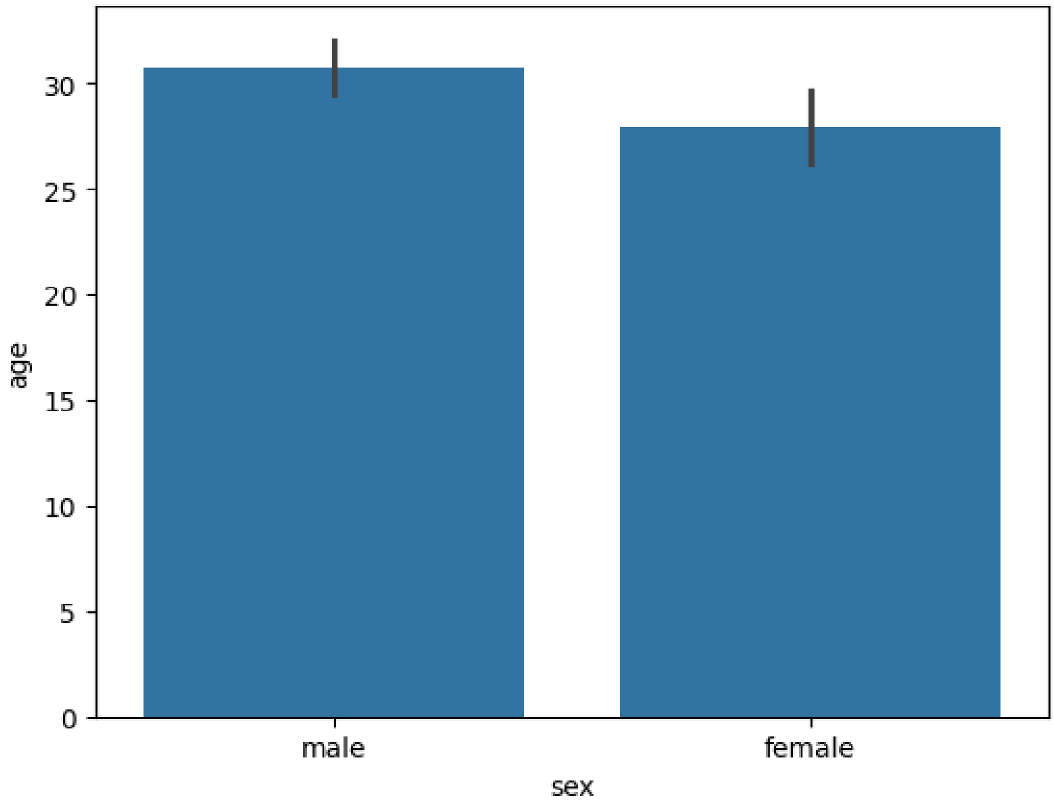
```
[9]: # 3. Rug Plot
     sns.rugplot(dataset['fare'])
```

```
[9]: <Axes: xlabel='fare'>
```



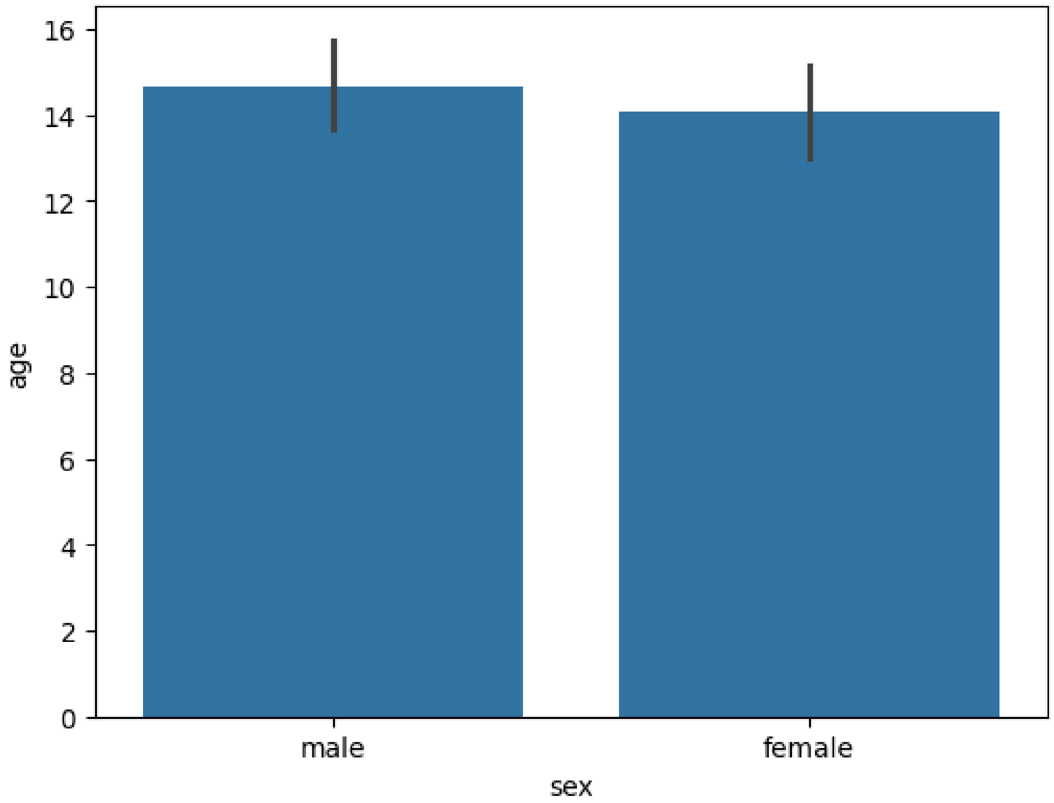
```
[10]: # B. Categorical Plots
      # 1. The Bar Plot
      sns.barplot(x='sex', y='age', data=dataset)
```

```
[10]: <Axes: xlabel='sex', ylabel='age'>
```



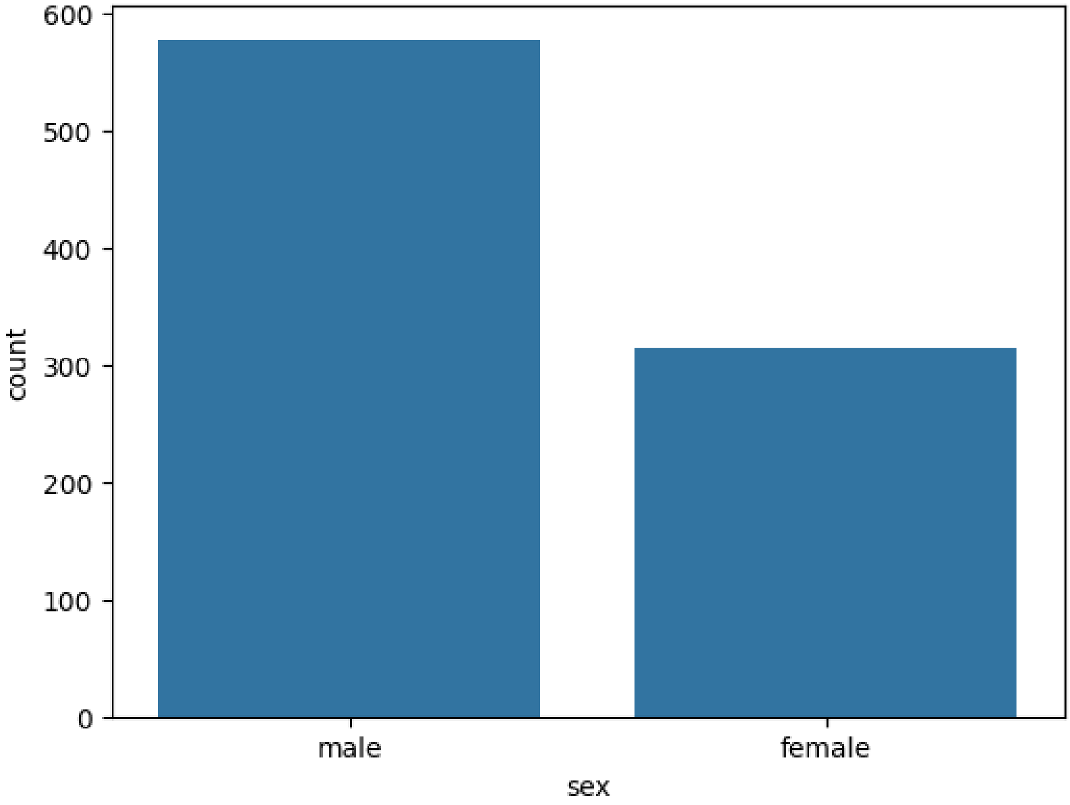
```
[11]: sns.barplot(x='sex', y='age', data=dataset, estimator=np.std) # calculated␣
       ↪other values rather than average
```

```
[11]: <Axes: xlabel='sex', ylabel='age'>
```



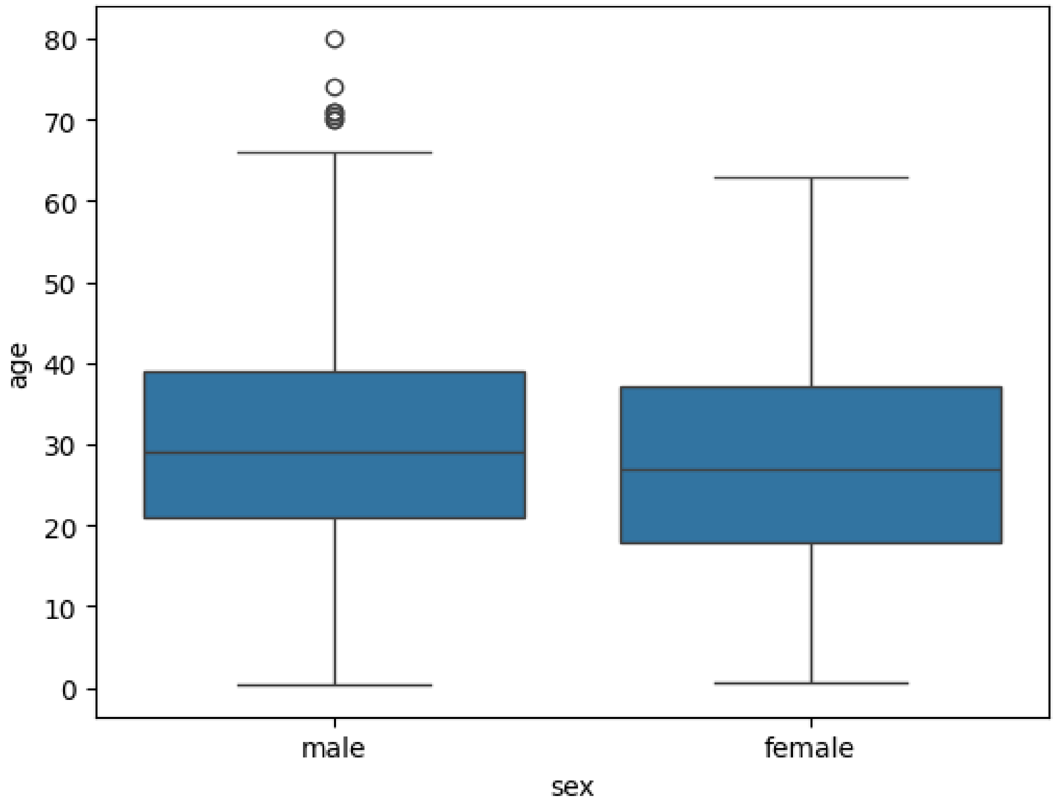
[12]:
```
# 2. The Count Plot
sns.countplot(x='sex', data=dataset)
```

[12]:
```
<Axes: xlabel='sex', ylabel='count'>
```



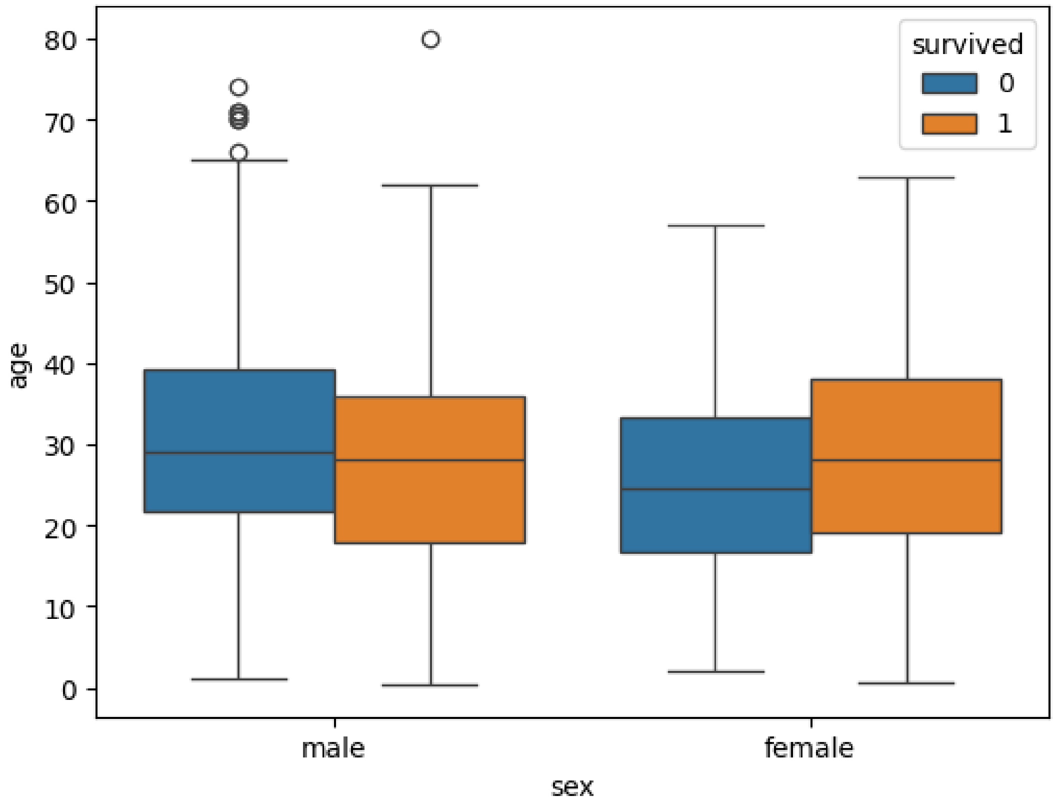
[13]:
```
# 3. The Box Plot
sns.boxplot(x='sex', y='age', data=dataset)
```

[13]:
```
<Axes: xlabel='sex', ylabel='age'>
```



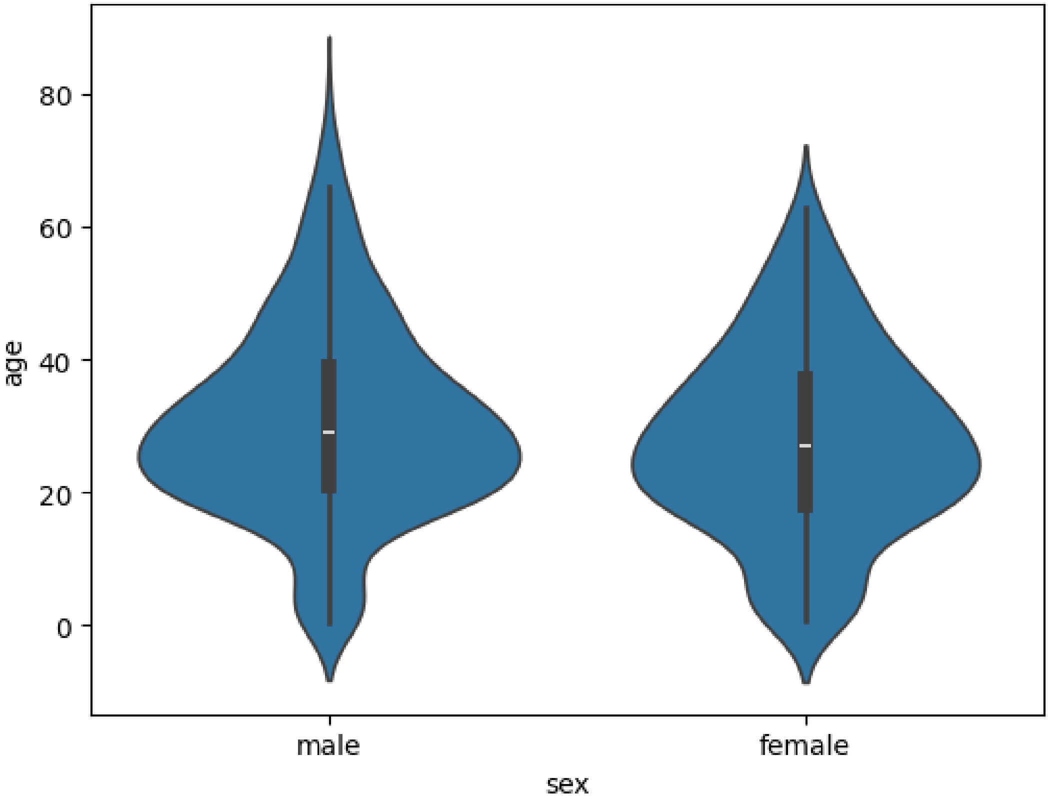
[14]:
```
sns.boxplot(x='sex', y='age', data=dataset, hue="survived")  #  along with the␣
  ↪information about whether or not they survived
```

[14]:
```
<Axes: xlabel='sex', ylabel='age'>
```



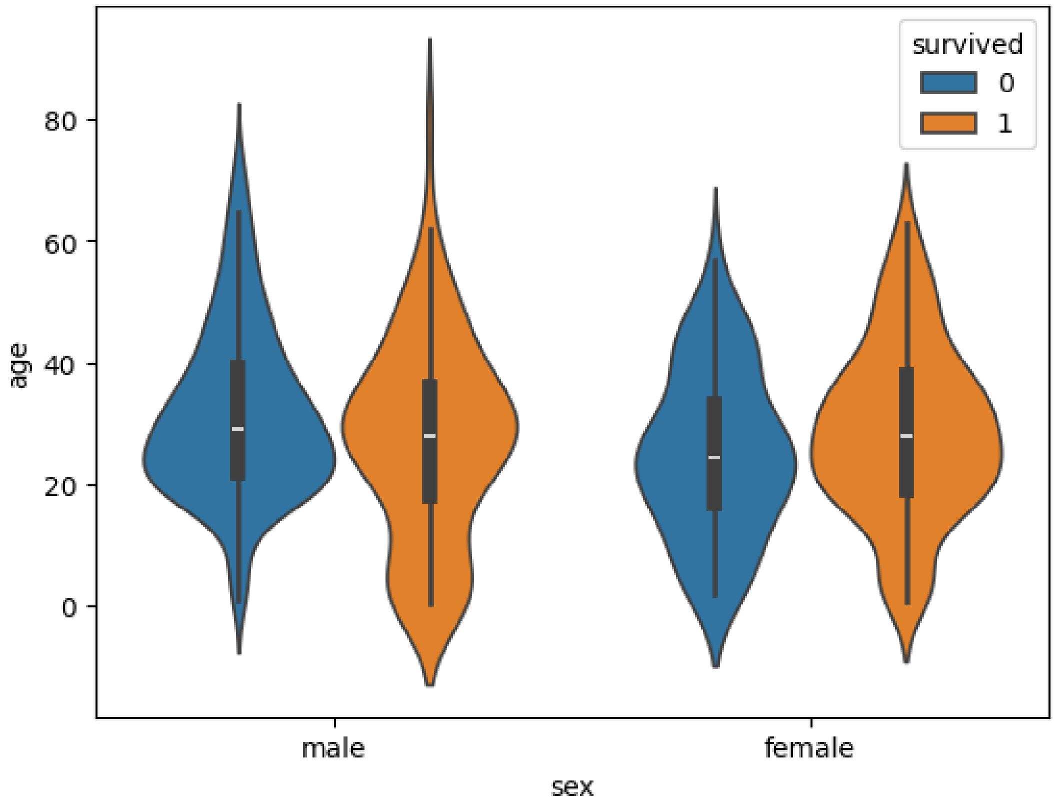
[15]:
```
# 4. The Violin Plot --> the violin plot allows us to display all the␣
↪components that actually correspond to the data point.
sns.violinplot(x='sex', y='age', data=dataset)
```

[15]:
```
<Axes: xlabel='sex', ylabel='age'>
```



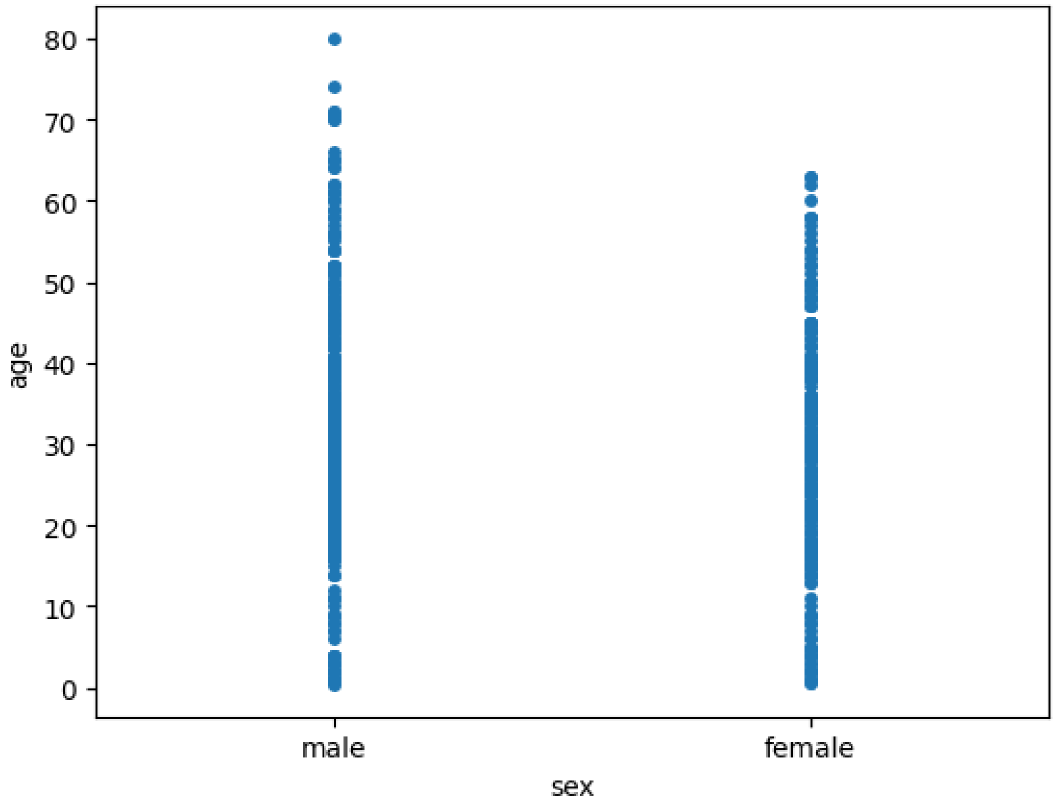
```
[16]: sns.violinplot(x='sex',y='age',data=dataset,hue='survived')  # along with the␣
       ↪information about whether or not they survived
```

```
[16]: <Axes: xlabel='sex', ylabel='age'>
```



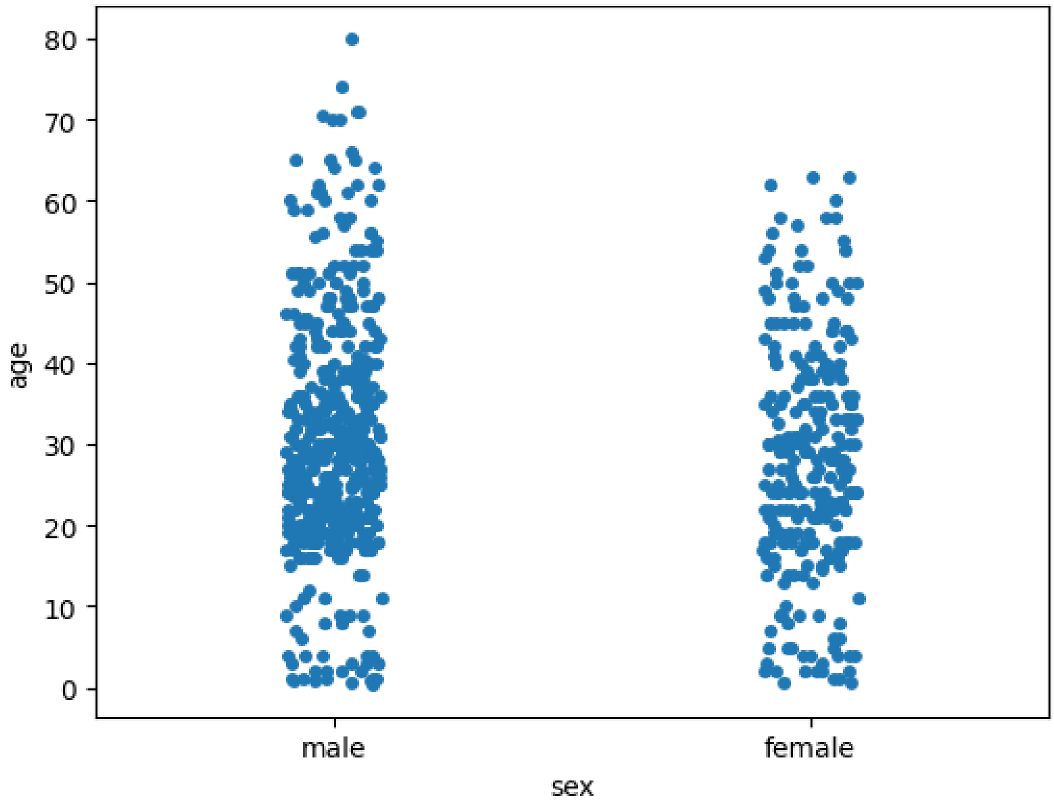
```
[17]: # C. Advanced Plots:
      # 1. The Strip Plot
      sns.stripplot(x='sex', y='age', data=dataset, jitter=False)
```

```
[17]: <Axes: xlabel='sex', ylabel='age'>
```



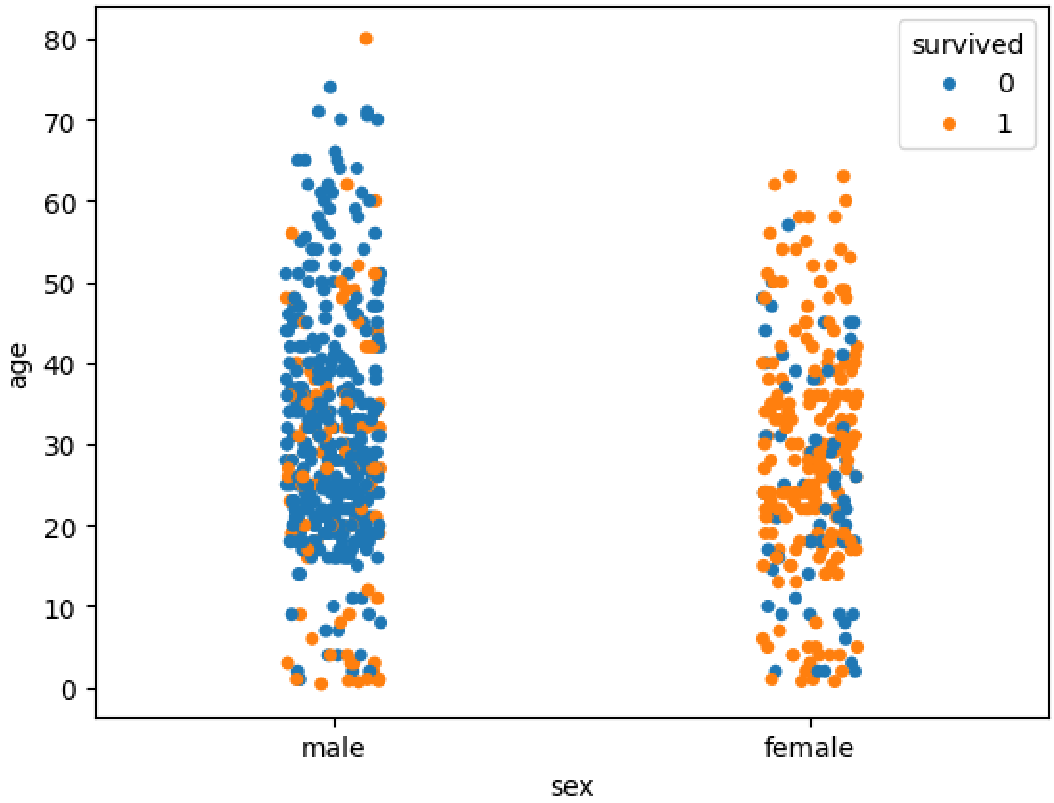
```
[18]: sns.stripplot(x='sex', y='age', data=dataset, jitter=True)
```

```
[18]: <Axes: xlabel='sex', ylabel='age'>
```



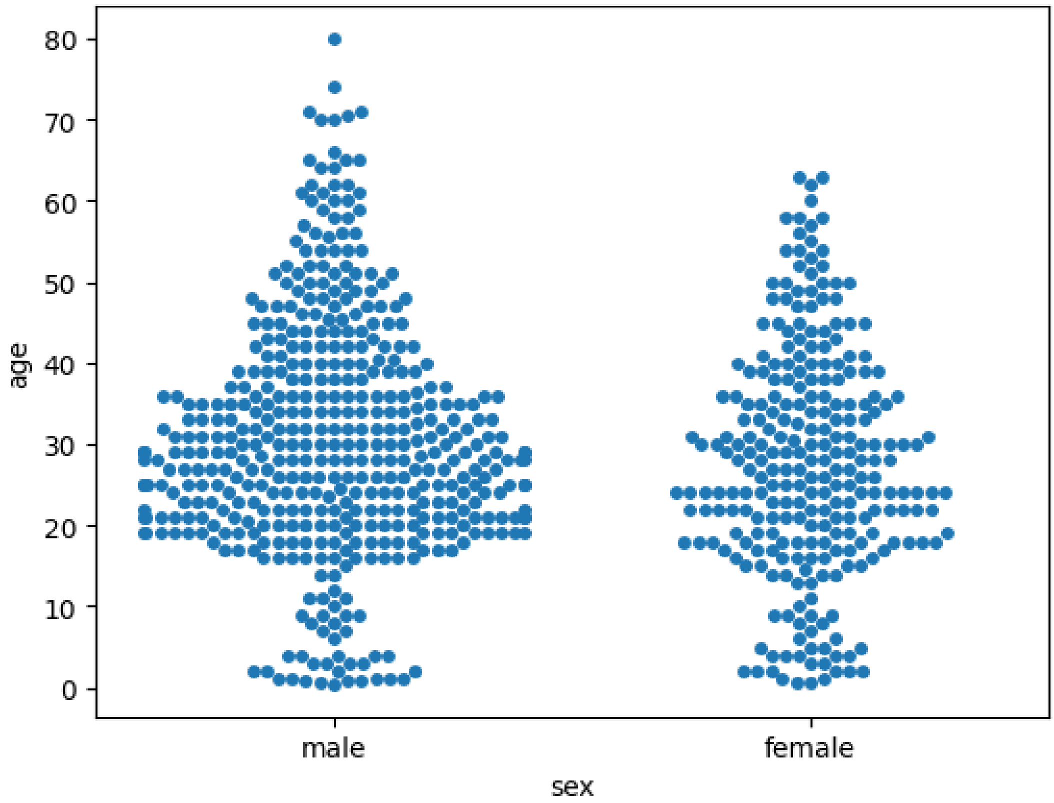
```
[19]: sns.stripplot(x='sex', y='age', data=dataset, jitter=True, hue='survived')
```

```
[19]: <Axes: xlabel='sex', ylabel='age'>
```



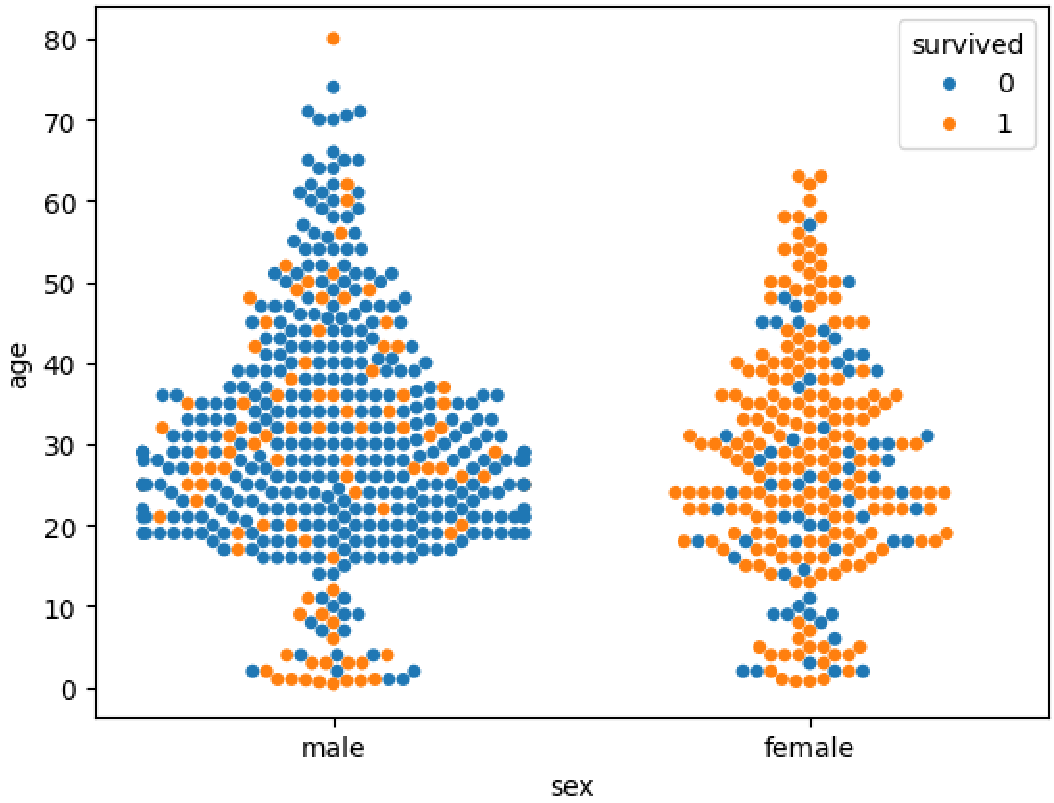
```
[20]: # 2. The Swarm Plot
      sns.swarmplot(x='sex', y='age', data=dataset)
```

```
[20]: <Axes: xlabel='sex', ylabel='age'>
```



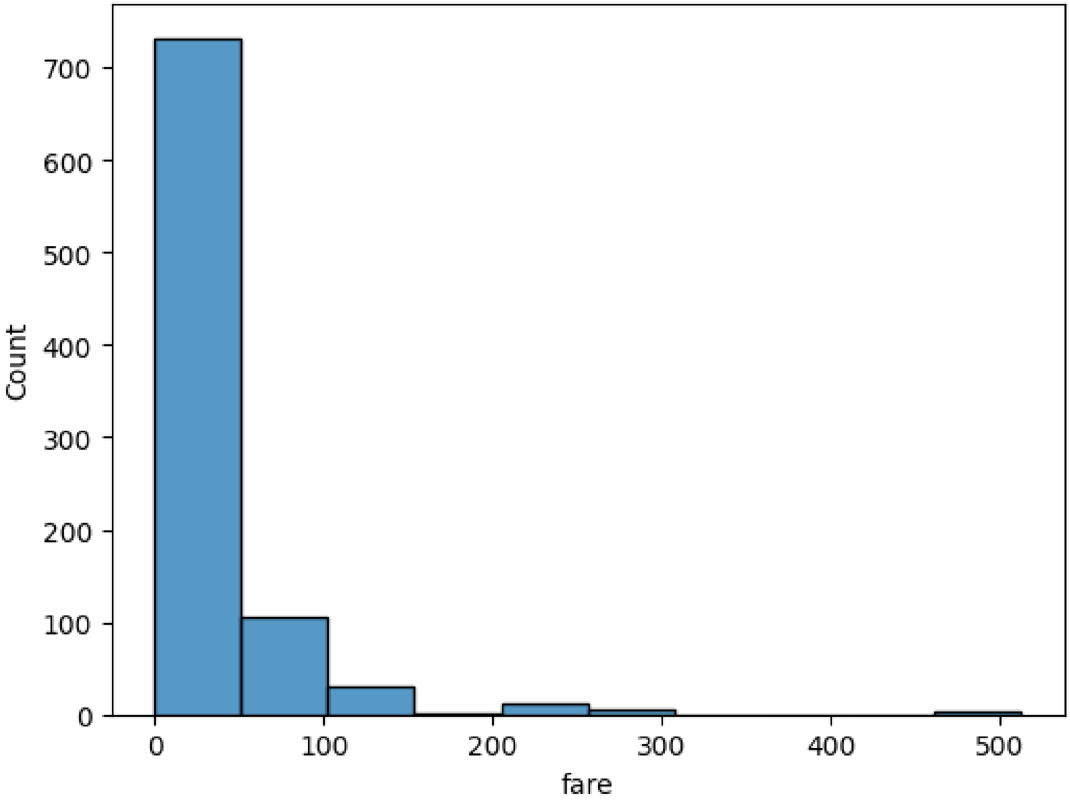
```
[21]: sns.swarmplot(x='sex', y='age', data=dataset, hue='survived')
```

```
[21]: <Axes: xlabel='sex', ylabel='age'>
```



[26]:
```
#  Checking how the price of the ticket (column name: 'fare') for each␣
↪passenger is distributed by plotting a histogram.
sns.histplot(dataset['fare'], kde=False, bins=10)
```

[26]:
```
<Axes: xlabel='fare', ylabel='Count'>
```


Count
0
100
200
300
400
500
600
700
fare
0
100
200
300
400
500