Exploratory Data Analysis

This will show us how we can do EDA using python.

Three important steps to keep in mind are:

1- Undestand the data

2- Clean the Data

3- Find relationship between data

In [ ]:

```
# Import Libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

Loading Dataset of **Titanic**.

In [ ]:

```
kashti = sns.load_dataset("titanic")
```

Download or Save dataset in CSV file

In [ ]:

```
kashti.to_csv("kashti.csv")
```

In [ ]:

```
kashti.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
```

In [ ]:

```
ks = kashti
```

In [ ]:

```
#Check krain k Dataset kis trah ka hai.
ks.head()
```

Out[ ]:

	survived	pclass	sex	age	sibsp	parch	fare	embarked	class	who	adult_male	deck	em
0	0	3	male	22.0	1	0	7.2500	S	Third	man	True	NaN	Sc
1	1	1	female	38.0	1	0	71.2833	C	First	woman	False	C	
2	1	3	female	26.0	0	0	7.9250	S	Third	woman	False	NaN	Sc
3	1	1	female	35.0	1	0	53.1000	S	First	woman	False	C	Sc

	survived	pclass	sex	age	sibsp	parch	fare	embarked	class	who	adult_male	deck	em
4	0	3	male	35.0	0	0	8.0500	S	Third	man	True	NaN	Sc

In [ ]:

```
#Rows and column k number Pta chal jata hai
ks.shape
```

Out[ ]:

```
(891, 15)
```

In [ ]:

```
ks.describe()
```

Out[ ]:

	survived	pclass	age	sibsp	parch	fare
count	891.000000	891.000000	714.000000	891.000000	891.000000	891.000000
mean	0.383838	2.308642	29.699118	0.523008	0.381594	32.204208
std	0.486592	0.836071	14.526497	1.102743	0.806057	49.693429
min	0.000000	1.000000	0.420000	0.000000	0.000000	0.000000
25%	0.000000	2.000000	20.125000	0.000000	0.000000	7.910400
50%	0.000000	3.000000	28.000000	0.000000	0.000000	14.454200
75%	1.000000	3.000000	38.000000	1.000000	0.000000	31.000000
max	1.000000	3.000000	80.000000	8.000000	6.000000	512.329200

In [ ]:

```
# find  unique Value
ks.nunique()
```

Out[ ]:

```
survived         2
pclass           3
sex              2
age             88
sibsp            7
parch            7
fare           248
embarked         3
class            3
who              3
adult_male       2
deck             7
embark_town      3
alive            2
alone            2
dtype: int64
```

In [ ]:

```
# Check column name
ks.columns
```

Out[ ]:

```
Index(['survived', 'pclass', 'sex', 'age', 'sibsp', 'parch', 'fare',
       'embarked', 'class', 'who', 'adult_male', 'deck', 'embark_town',
       'alive', 'alone'],
      dtype='object')
```

In [ ]:

```
# Chack unique valus in a column
ks["who"].unique()
```

Out[ ]:

```
array(['man', 'woman', 'child'], dtype=object)
```

In [ ]:

```
# Check unique values in multiple columns..
pd.unique(ks[['sex', 'who', "survived", "class"]].values.ravel())
```

Out[ ]:

```
array(['male', 'man', 0, 'Third', 'female', 'woman', 1, 'First', 'child',
       'Second'], dtype=object)
```

Cleaning and Filtering the Data

In [ ]:

```
# Find the Missing Values
ks.isnull()
```

Out[ ]:

	survived	pclass	sex	age	sibsp	parch	fare	embarked	class	who	adult_male	deck	embar
0	False	False	False	False	False	False	False	False	False	False	False	True	
1	False	False	False	False	False	False	False	False	False	False	False	False	
2	False	False	False	False	False	False	False	False	False	False	False	True	
3	False	False	False	False	False	False	False	False	False	False	False	False	
4	False	False	False	False	False	False	False	False	False	False	False	True	
...	...	...	...	...	...	...	...	...	...	...	...	...	
886	False	False	False	False	False	False	False	False	False	False	False	True	
887	False	False	False	False	False	False	False	False	False	False	False	False	
888	False	False	False	True	False	False	False	False	False	False	False	True	
889	False	False	False	False	False	False	False	False	False	False	False	False	
890	False	False	False	False	False	False	False	False	False	False	False	True	

891 rows × 15 columns

In [ ]:

```
# False mean k null nhi jis jis jgha true likha hia wo null values hain.
# The better way to find total number of missing values.
ks.isnull().sum()
```

Out[ ]:

```
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
```

In [ ]:

```
# removing missing value column (Cleaning Data)
ks_clean = ks.drop(["deck"],axis=1)
ks_clean.head()
```

Out[ ]:

	survived	pclass	sex	age	sibsp	parch	fare	embarked	class	who	adult_male	embark_t
0	0	3	male	22.0	1	0	7.2500	S	Third	man	True	Southam
1	1	1	female	38.0	1	0	71.2833	C	First	woman	False	Cherb
2	1	3	female	26.0	0	0	7.9250	S	Third	woman	False	Southam
3	1	1	female	35.0	1	0	53.1000	S	First	woman	False	Southam
4	0	3	male	35.0	0	0	8.0500	S	Third	man	True	Southam

In [ ]:

```
ks_clean.shape
```

Out[ ]:

```
(891, 14)
```

In [ ]:

```
891-177
```

Out[ ]:
```
714
```

Note for Data Cleaning:

- sb se pehle hm dekhin gy data me missing values kitni hai.
- agr kisi column me boht ziyada missing values hain e.g. **"deck"** to hm us column ko drop kr dy gy.
- ab jaise hm ne dekha **"age"** wale column me "177" missing values hai lakin hm use drop nhi kr skte Q k total values "891" hai. or difference boht km he
- To ab hm sirf null values hi remove krain gy.

In [ ]:
```
# Drop Null valus
ks_clean.dropna().shape
```

Out[ ]:
```
(712, 14)
```

In [ ]:
```
# Update Data after removing missing values.
ks_clean = ks_clean.dropna()
```

In [ ]:
```
ks_clean.shape
```

Out[ ]:
```
(712, 14)
```

In [ ]:
```
# Now we can check agin if our datacontain some missing values..
ks_clean.isnull().sum()
```

Out[ ]:
```
survived       0
pclass         0
sex            0
age            0
sibsp          0
parch          0
fare           0
embarked       0
class          0
who            0
adult_male     0
embark_town    0
alive          0
alone          0
dtype: int64
```

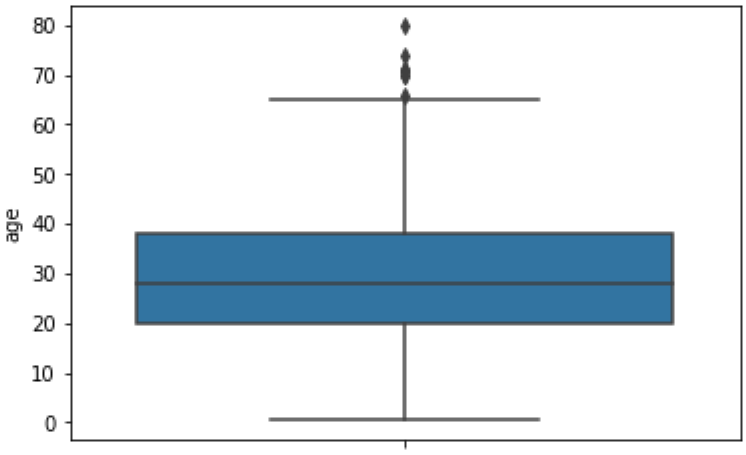
How to find the OutLier in data?

In [ ]:
```
ks_clean.columns
```

Out[ ]:
```
Index(['survived', 'pclass', 'sex', 'age', 'sibsp', 'parch', 'fare',
       'embarked', 'class', 'who', 'adult_male', 'embark_town', 'alive',
       'alone'],
      dtype='object')
```

In [ ]:
```
sns.boxplot(y="age",data=ks_clean)
```

Out[ ]:
```
<AxesSubplot:ylabel='age'>
```



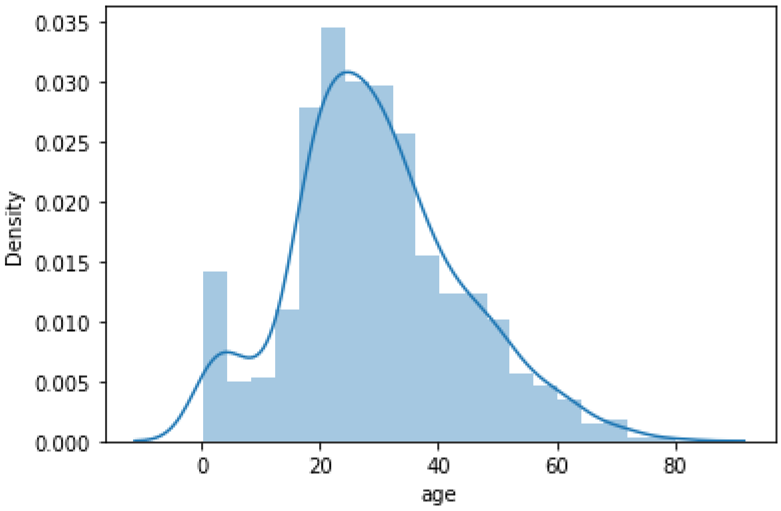
In [ ]:

```
# Normalty test, Histogram, Bell curve to check data is normal or not..
sns.distplot(ks_clean["age"])
```

```
c:\Users\Musharaf Ahsan\AppData\Local\Programs\Python\Python310\lib\site-packages\seabor
n\distributions.py:2619: FutureWarning: `distplot` is a deprecated function and will be
removed in a future version. Please adapt your code to use either `displot` (a figure-le
vel function with similar flexibility) or `histplot` (an axes-level function for histogr
ams).
  warnings.warn(msg, FutureWarning)
```

Out[ ]:

```
<AxesSubplot:xlabel='age', ylabel='Density'>
```



In [ ]:

```
ks_clean["age"].mean()
```

Out[ ]:

```
29.64209269662921
```

In [ ]:

```
ks_clean = ks_clean[ks_clean["age"] <68]
ks_clean.head()
```

Out[ ]:

	survived	pclass	sex	age	sibsp	parch	fare	embarked	class	who	adult_male	embark_t
0	0	3	male	22.0	1	0	7.2500	S	Third	man	True	Southam
1	1	1	female	38.0	1	0	71.2833	C	First	woman	False	Cherb
2	1	3	female	26.0	0	0	7.9250	S	Third	woman	False	Southam
3	1	1	female	35.0	1	0	53.1000	S	First	woman	False	Southam
4	0	3	male	35.0	0	0	8.0500	S	Third	man	True	Southam

In [ ]:

```
# Remaining valus after the removal of outliers..
ks_clean.shape
```

Out[ ]:

```
(705, 14)
```

In [ ]:

```
ks_clean["age"].mean()
```

Out[ ]:

```
29.21797163120567
```

Age mean diffrence after removing outliers:

- Mean with OutLiers: 29.64209269662921
- Mean without OutLiers: 29.21797163120567

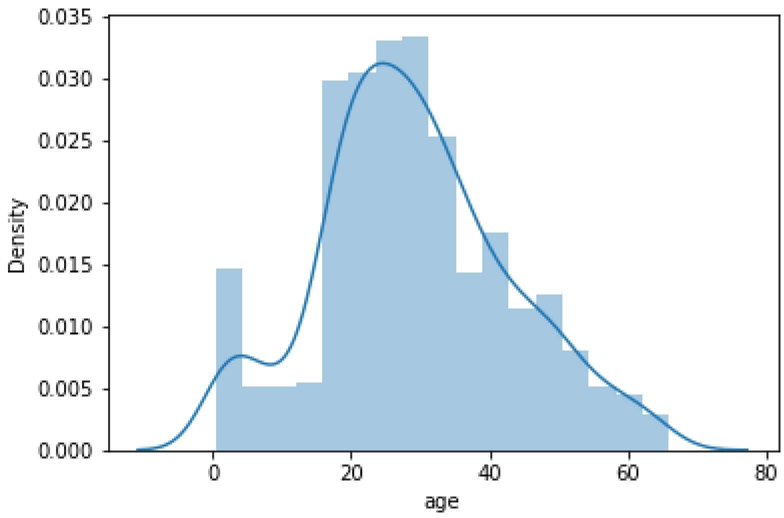
In [ ]:

```
sns.distplot(ks_clean["age"])
```

```
c:\Users\Musharaf Ahsan\AppData\Local\Programs\Python\Python310\lib\site-packages\seabor
n\distributions.py:2619: FutureWarning: `distplot` is a deprecated function and will be
removed in a future version. Please adapt your code to use either `displot` (a figure-le
vel function with similar flexibility) or `histplot` (an axes-level function for histogr
ams).
  warnings.warn(msg, FutureWarning)
```

Out[ ]:

```
<AxesSubplot:xlabel='age', ylabel='Density'>
```

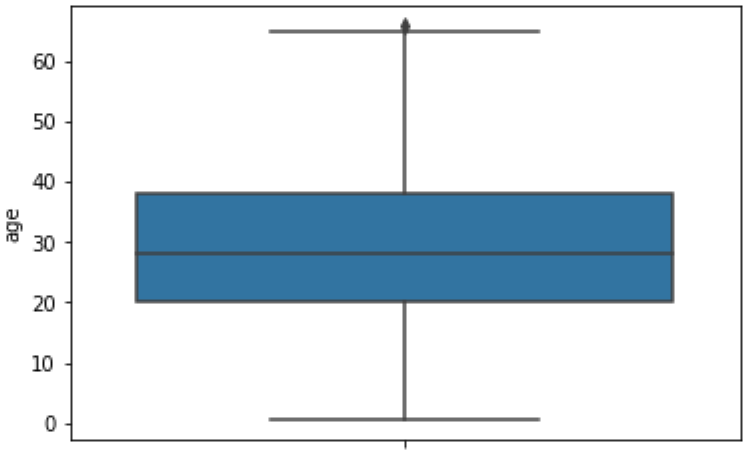


In [ ]:

```
sns.boxplot(y="age",data=ks_clean)
```

Out[ ]:

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

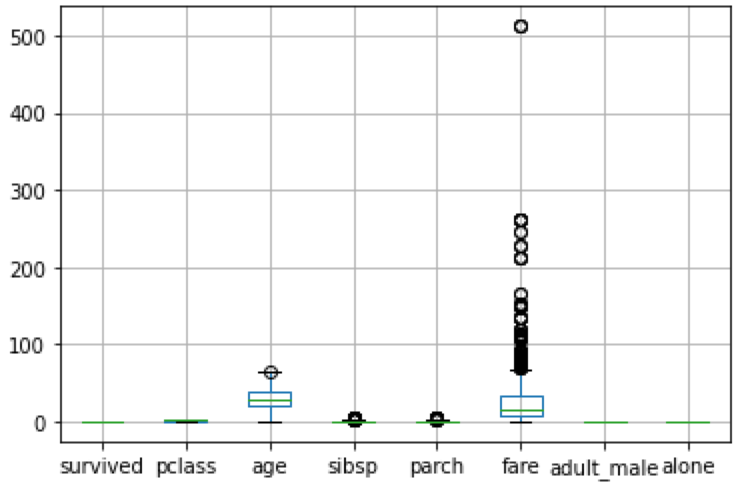


In [ ]:

```
ks_clean.boxplot()
```

Out[ ]:

```
<AxesSubplot:>
```

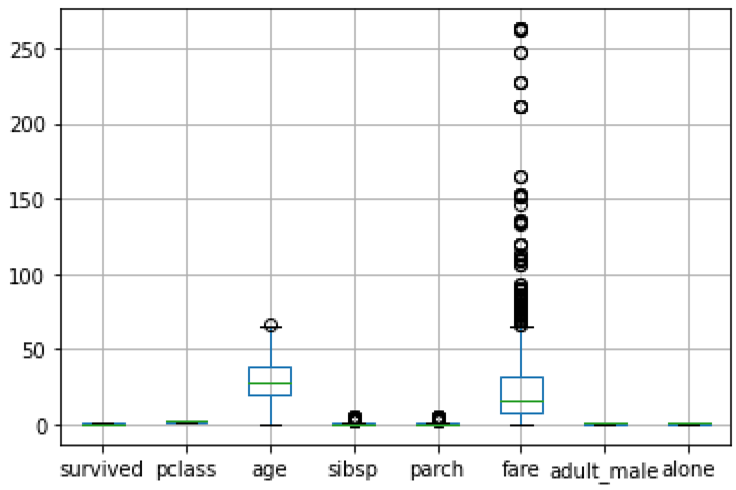


In [ ]:

```
ks_clean = ks_clean[ks_clean["fare"] <300]
ks_clean.boxplot()
```

Out[ ]:

```
<AxesSubplot:>
```



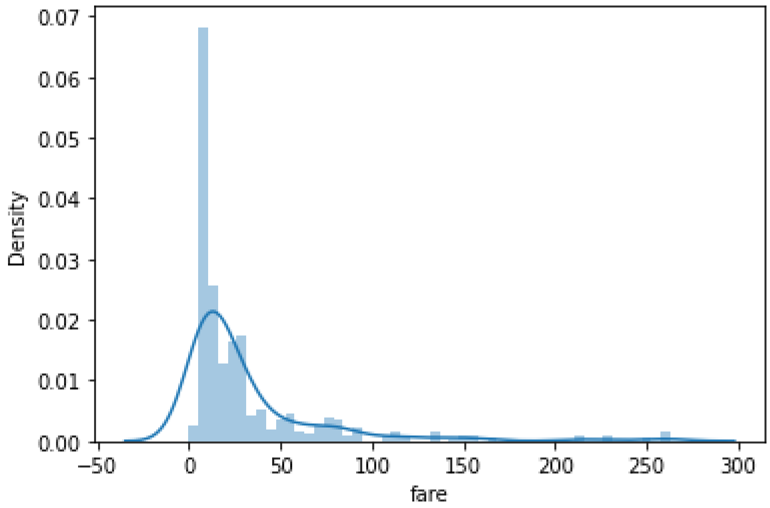
In [ ]:

```
sns.distplot(ks_clean["fare"])
```

```
c:\Users\Musharaf Ahsan\AppData\Local\Programs\Python\Python310\lib\site-packages\seaborn\distributions.py:2619: FutureWarning: `distplot` is a deprecated function and will be removed in a future version. Please adapt your code to use either `displot` (a figure-level function with similar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```

Out[ ]:

```
<AxesSubplot:xlabel='fare', ylabel='Density'>
```



Log Transformation

In [ ]:

```
ks_clean["fare_log"]=np.log(ks_clean["fare"])
ks_clean.head()
```

```
c:\Users\Musharaf Ahsan\AppData\Local\Programs\Python\Python310\lib\site-packages\pandas\core\arraylike.py:397: RuntimeWarning: divide by zero encountered in log
  result = getattr(ufunc, method)(*inputs, **kwargs)
```

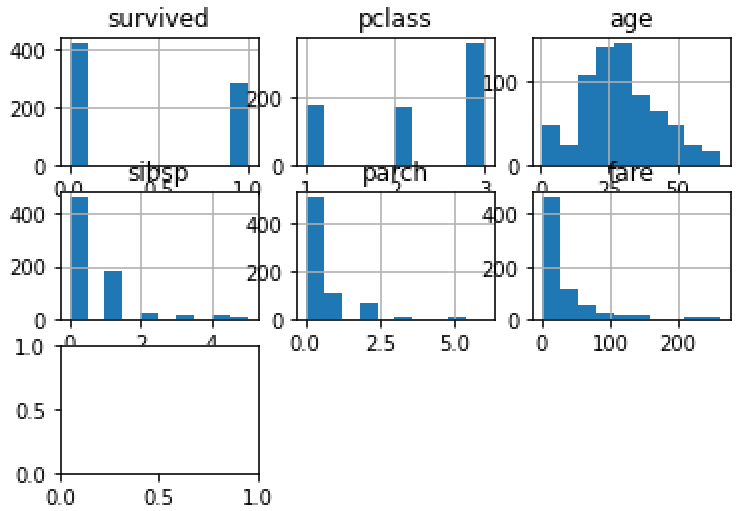
Out[ ]:

	survived	pclass	sex	age	sibsp	parch	fare	embarked	class	who	adult_male	embark_t
0	0	3	male	22.0	1	0	7.2500	S	Third	man	True	Southam
1	1	1	female	38.0	1	0	71.2833	C	First	woman	False	Cherb
2	1	3	female	26.0	0	0	7.9250	S	Third	woman	False	Southam
3	1	1	female	35.0	1	0	53.1000	S	First	woman	False	Southam
4	0	3	male	35.0	0	0	8.0500	S	Third	man	True	Southam

In [ ]:

```
ks_clean.hist()
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
c:\Users\Musharaf Ahsan\Desktop\Assignments\eda.ipynb Cell 43 in <cell line: 1>()
----> <a href='vscode-notebook-cell:/c%3A/Users/Musharaf%20Ahsan/Desktop/Assignments/ed
a.ipynb#ch0000057?line=0'>1</a> ks_clean.hist()

File c:\Users\Musharaf Ahsan\AppData\Local\Programs\Python\Python310\lib\site-packages\p
andas\plotting\_core.py:226, in hist_frame(data, column, by, grid, xlabelsize, xrot, yla
belsize, yrot, ax, sharex, sharey, figsize, layout, bins, backend, legend, **kwargs)
    135 """
    136 Make a histogram of the DataFrame's columns.
    137 
   (...)
    223     >>> hist = df.hist(bins=3)
    224 """
    225 plot_backend = _get_plot_backend(backend)
--> 226 return plot_backend.hist_frame(
    227     data,
    228     column=column,
    229     by=by,
    230     grid=grid,
    231     xlabelsize=xlabelsize,
    232     xrot=xrot,
    233     ylabelsize=ylabelsize,
    234     yrot=yrot,
    235     ax=ax,
    236     sharex=sharex,
    237     sharey=sharey,
    238     figsize=figsize,
    239     layout=layout,
    240     legend=legend,
    241     bins=bins,
    242     **kwargs,
    243 )

File c:\Users\Musharaf Ahsan\AppData\Local\Programs\Python\Python310\lib\site-packages\p
andas\plotting\_matplotlib\hist.py:501, in hist_frame(data, column, by, grid, xlabelsiz
e, xrot, ylabelsize, yrot, ax, sharex, sharey, figsize, layout, bins, legend, **kwds)
    499 if legend and can_set_label:
    500     kwds["label"] = col
--> 501 ax.hist(data[col].dropna().values, bins=bins, **kwds)
    502 ax.set_title(col)
    503 ax.grid(grid)

File c:\Users\Musharaf Ahsan\AppData\Local\Programs\Python\Python310\lib\site-packages\m
atplotlib\__init__.py:1412, in _preprocess_data.<locals>.inner(ax, data, *args, **kwarg
s)
   1409 @functools.wraps(func)
   1410 def inner(ax, *args, data=None, **kwargs):
   1411     if data is None:
-> 1412         return func(ax, *map(sanitize_sequence, args), **kwargs)
   1414     bound = new_sig.bind(ax, *args, **kwargs)
   1415     auto_label = (bound.arguments.get(label_namer)
   1416                   or bound.kwargs.get(label_namer))

File c:\Users\Musharaf Ahsan\AppData\Local\Programs\Python\Python310\lib\site-packages\m
atplotlib\axes\_axes.py:6635, in Axes.hist(self, x, bins, range, density, weights, cumul
ative, bottom, histtype, align, orientation, rwidth, log, color, label, stacked, **kwarg
s)
   6631 # Loop through datasets
   6632 for i in range(nx):
   6633     # this will automatically overwrite bins,
   6634     # so that each histogram uses the same bins
-> 6635     m, bins = np.histogram(x[i], bins, weights=w[i], **hist_kwargs)
   6636     tops.append(m)
   6637 tops = np.array(tops, float)  # causes problems later if it's an int

File <__array_function__ internals>:180, in histogram(*args, **kwargs)

File c:\Users\Musharaf Ahsan\AppData\Local\Programs\Python\Python310\lib\site-packages\n
umpy\lib\histograms.py:793, in histogram(a, bins, range, normed, weights, density)
    681 r"""
    682 Compute the histogram of a dataset.
    683 
   (...)
    789 
    790 """
    791 a, weights = _ravel_and_check_weights(a, weights)
--> 793 bin_edges, uniform_bins = _get_bin_edges(a, bins, range, weights)
    795 # Histogram is an integer or a float array depending on the weights.
    796 if weights is None:

File c:\Users\Musharaf Ahsan\AppData\Local\Programs\Python\Python310\lib\site-packages\n
umpy\lib\histograms.py:426, in _get_bin_edges(a, bins, range, weights)
    423     if n_equal_bins < 1:
    424         raise ValueError('`bins` must be positive, when an integer')
--> 426     first_edge, last_edge = _get_outer_edges(a, range)
    428 elif np.ndim(bins) == 1:
    429     bin_edges = np.asarray(bins)

File c:\Users\Musharaf Ahsan\AppData\Local\Programs\Python\Python310\lib\site-packages\n
umpy\lib\histograms.py:315, in _get_outer_edges(a, range)
    312         raise ValueError(
    313             'max must be larger than min in range parameter.')
    314     if not (np.isfinite(first_edge) and np.isfinite(last_edge)):
--> 315         raise ValueError(
    316             "supplied range of [{}, {}] is not finite".format(first_edge, last_e
dge))
    317 elif a.size == 0:
    318     # handle empty arrays. Can't determine range, so use 0-1.
    319     first_edge, last_edge = 0, 1

ValueError: supplied range of [-inf, 5.572154032177765] is not finite
```

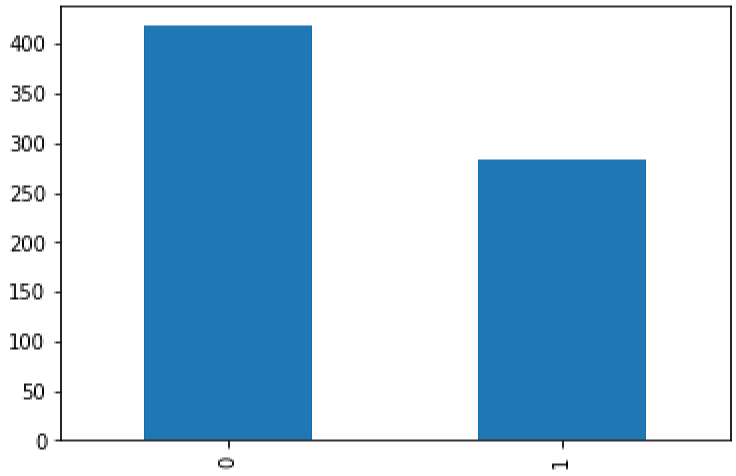


In [ ]:

```
pd.value_counts(ks_clean["survived"]).plot.bar()
```

Out[ ]:

```
<AxesSubplot:>
```

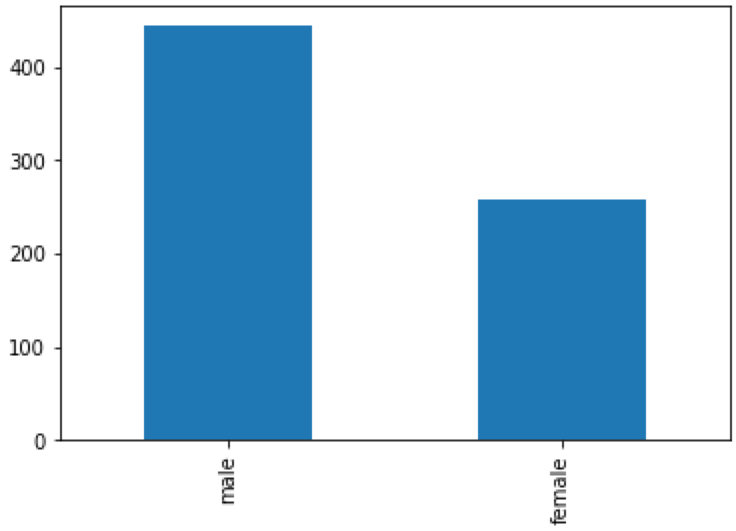


In [ ]:

```
pd.value_counts(ks_clean["sex"]).plot.bar()
```

Out[ ]:

```
<AxesSubplot:>
```

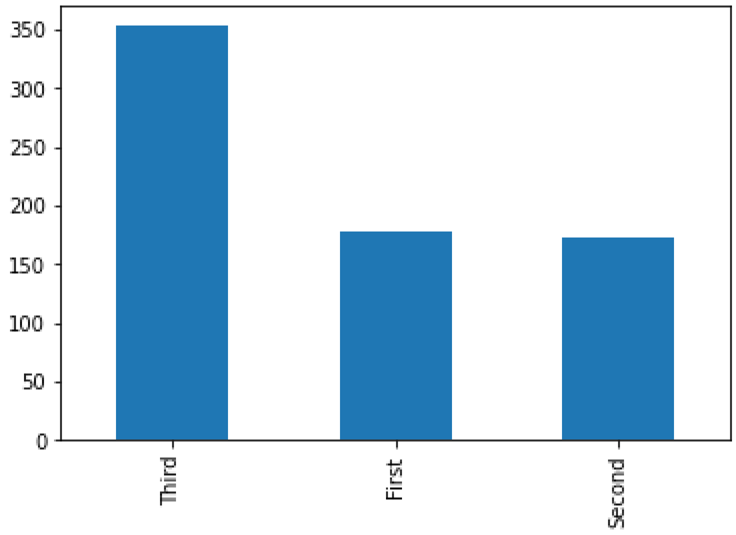


In [ ]:

```
pd.value_counts(ks_clean["class"]).plot.bar()
```

Out[ ]:

```
<AxesSubplot:>
```



In [ ]:

```
ks_clean.groupby(["sex","class"]).mean()
```

Out[ ]:

		survived	pclass	age	sibsp	parch	fare	adult_male	alone	fare_
sex	class									
female	First	0.963415	1.0	34.231707	0.560976	0.512195	103.696393	0.000000	0.353659	4.461
	Second	0.918919	2.0	28.722973	0.500000	0.621622	21.951070	0.000000	0.405405	2.985
	Third	0.460784	3.0	21.750000	0.823529	0.950980	15.875369	0.000000	0.372549	2.617
male	First	0.389474	1.0	40.067579	0.389474	0.336842	62.901096	0.968421	0.526316	N
	Second	0.153061	2.0	30.340102	0.377551	0.244898	21.221429	0.908163	0.632653	2.894
	Third	0.151394	3.0	26.143108	0.494024	0.258964	12.197757	0.888446	0.737052	N

In [ ]:

```
ks.groupby(["sex","class","who"]).mean()
```

Out[ ]:

			survived	pclass	age	sibsp	parch	fare	adult_male	alo
sex	class	who								
female	First	child	0.666667	1.0	10.333333	0.666667	1.666667	160.962500	0.0	0.0000
		man	NaN	NaN	NaN	NaN	NaN	NaN	NaN	N
		woman	0.978022	1.0	35.500000	0.549451	0.417582	104.317995	0.0	0.3736
	Second	child	1.000000	2.0	6.600000	0.700000	1.300000	29.240000	0.0	0.0000
		man	NaN	NaN	NaN	NaN	NaN	NaN	NaN	N
		woman	0.909091	2.0	32.179688	0.454545	0.500000	20.868624	0.0	0.4848
	Third	child	0.533333	3.0	7.100000	1.533333	1.100000	19.023753	0.0	0.1666

			survived	pclass	age	sibsp	parch	fare	adult_male	alo
sex	**class**	**who**								
		man	NaN	NaN	NaN	NaN	NaN	NaN	NaN	N
		woman	0.491228	3.0	27.854167	0.728070	0.719298	15.354351	0.0	0.4824
male	**First**	**child**	1.000000	1.0	5.306667	0.666667	2.000000	117.802767	0.0	0.0000
		man	0.352941	1.0	42.382653	0.302521	0.235294	65.951086	1.0	0.6302
		woman	NaN	NaN	NaN	NaN	NaN	NaN	NaN	N
	Second	**child**	1.000000	2.0	2.258889	0.888889	1.222222	27.306022	0.0	0.0000
		man	0.080808	2.0	33.588889	0.292929	0.131313	19.054124	1.0	0.7272
		woman	NaN	NaN	NaN	NaN	NaN	NaN	NaN	N
	Third	**child**	0.321429	3.0	6.515000	2.821429	1.321429	27.716371	0.0	0.0357
		man	0.119122	3.0	28.995556	0.294671	0.128527	11.340213	1.0	0.8244
		woman	NaN	NaN	NaN	NaN	NaN	NaN	NaN	N

Relationship

In [ ]:

```
ks_clean.corr()
```

Out[ ]:

	survived	pclass	age	sibsp	parch	fare	adult_male	alone	fare
survived	1.000000	-0.356549	-0.074335	-0.014483	0.095426	0.273531	-0.554567	-0.201175	0.334
pclass	-0.356549	1.000000	-0.365121	0.061354	0.022519	-0.617591	0.102930	0.156030	-0.76
age	-0.074335	-0.365121	1.000000	-0.308906	-0.186271	0.103100	0.275035	0.187284	0.13
sibsp	-0.014483	0.061354	-0.308906	1.000000	0.381803	0.197954	-0.311622	-0.629200	0.32
parch	0.095426	0.022519	-0.186271	0.381803	1.000000	0.259948	-0.366540	-0.574701	0.34
fare	0.273531	-0.617591	0.103100	0.197954	0.259948	1.000000	-0.228675	-0.333949	0.868
adult_male	-0.554567	0.102930	0.275035	-0.311622	-0.366540	-0.228675	1.000000	0.402214	-0.304
alone	-0.201175	0.156030	0.187284	-0.629200	-0.574701	-0.333949	0.402214	1.000000	-0.49
fare_log	0.334877	-0.766373	0.131457	0.321417	0.340691	0.868301	-0.304249	-0.497267	1.00

In [ ]:

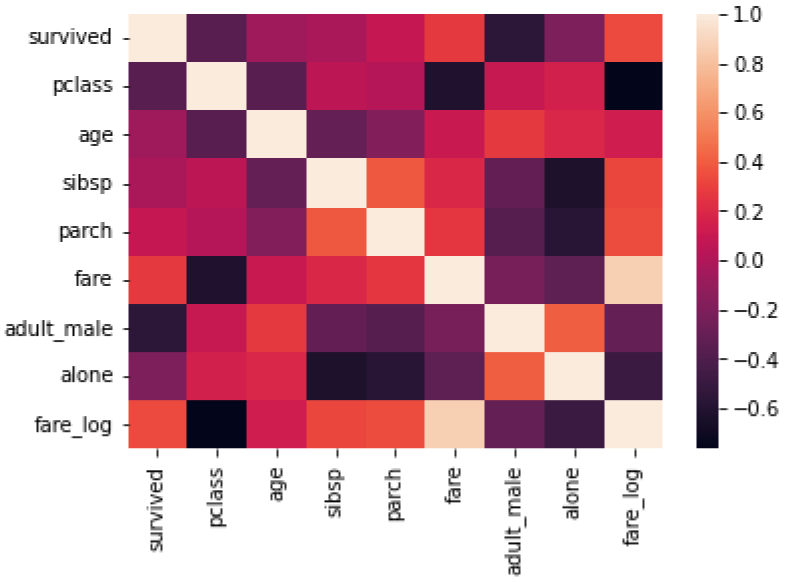
```
corr_ks_clean = ks_clean.corr()
```

In [ ]:

```
# Heatmap
sns.heatmap(corr_ks_clean)
```

Out[ ]:

```
<AxesSubplot:>
```

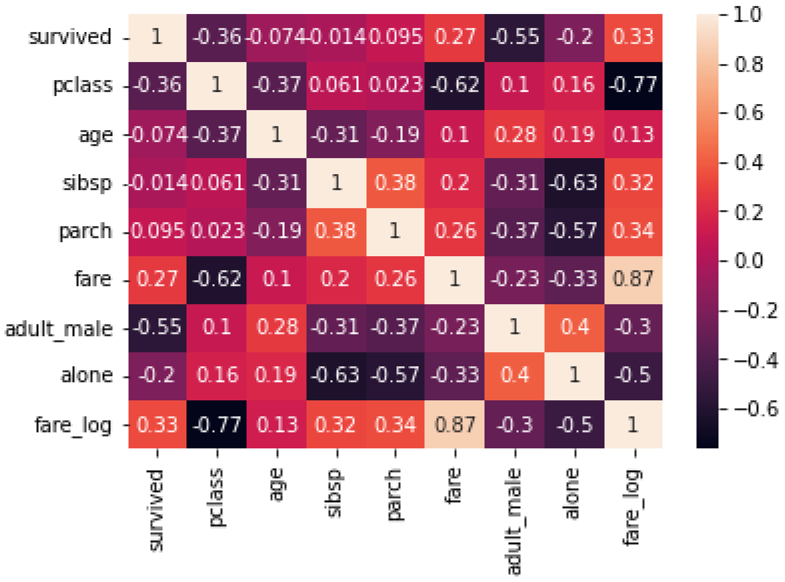


In [ ]:

```
sns.heatmap(corr_ks_clean,annot=True)
```

Out[ ]:

```
<AxesSubplot:>
```

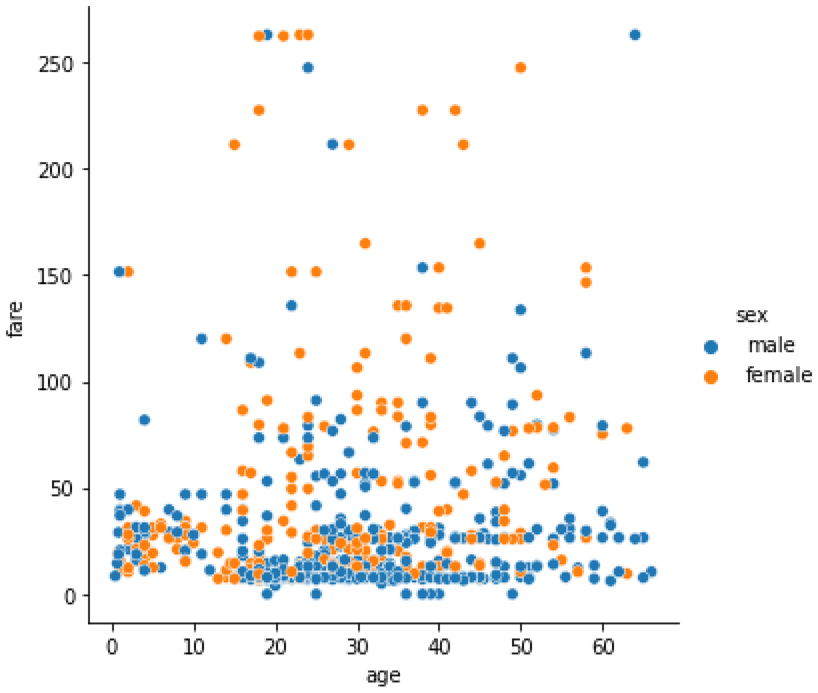


In [ ]:

```
sns.relplot(x="age",y="fare",hue="sex",data=ks_clean)
```

Out[ ]:

```
<seaborn.axisgrid.FacetGrid at 0x5fa7df4280>
```

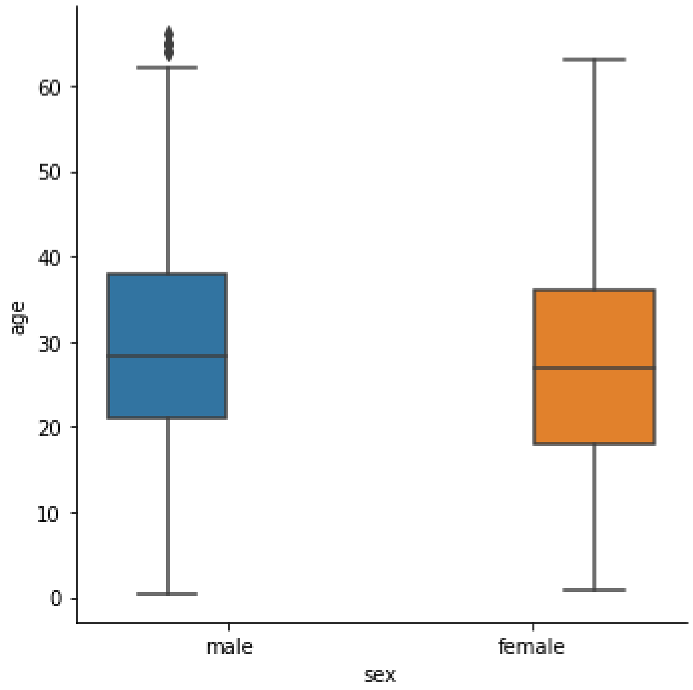


In [ ]:

```
sns.catplot(x="sex",y="age",hue="sex",data=ks_clean,kind="box")
```

Out[ ]:

```
<seaborn.axisgrid.FacetGrid at 0x5fa87f56c0>
```

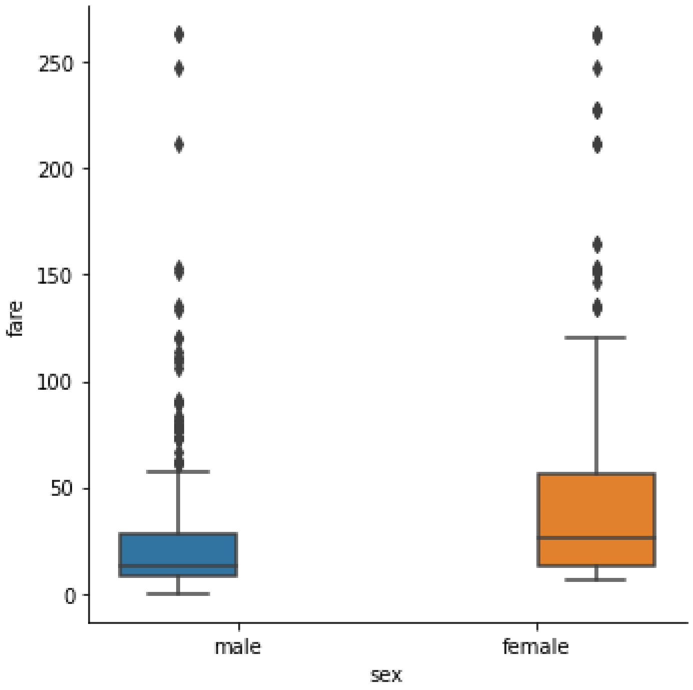


In [ ]:

```
sns.catplot(x="sex",y="fare",hue="sex",data=ks_clean,kind="box")
```

Out[ ]:

```
<seaborn.axisgrid.FacetGrid at 0x5fa87f7550>
```

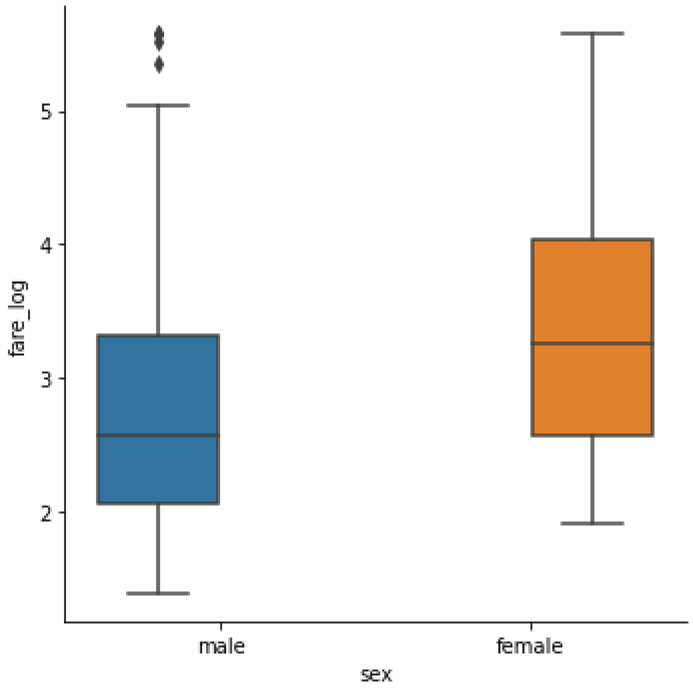


In [ ]:

```
sns.catplot(x="sex",y="fare_log",hue="sex",data=ks_clean,kind="box")
```

Out[ ]:

```
<seaborn.axisgrid.FacetGrid at 0x5fa89e6e90>
```


fare_log
male
female
sex