Exploratory Data Analysis will show us what do with data

- Three important steps to keep in mind are
- Understand the data
- Clean the data
- Find a relationship between data

In [90]:

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

In [91]:

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

In [92]:

```
kashti.to_csv('kashti.csv')
```

In [93]:

```
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 [94]:

```
# How can we know we have to do EDA On data
```

In [95]:

```
# this is just to see how the data is
ks = kashti
```

In [96]:

```
ks.head()
```

Out[96]:

	survived	pclass	sex	age	sibsp	parch	fare	embarked	class	who	adult_male	deck
0	0	3	male	22.0	1	0	7.2500	S	Third	man	True	NaN
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
3	1	1	female	35.0	1	0	53.1000	S	First	woman	False	C
4	0	3	male	35.0	0	0	8.0500	S	Third	man	True	NaN

In [97]:

```
ks.shape # it shows number of rows and column
```

Out[97]:

```
(891, 15)
```

In [98]:

```
ks.describe()
```

Out[98]:

	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 [99]:

```
# unique values means number of values in one column
ks.nunique()
```

Out[99]:

```
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 [100…

```
# when I need column names
ks.columns
```

Out[100…

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

In [101…

```
# to find unique value of one column
ks['sex'].unique()
```

Out[101…

```
array(['male', 'female'], dtype=object)
```

In [102…

```
# Assignment
# ks['adult_male' , 'sex'].unique() , for multiple column following is workin

pd.unique(ks[['adult_male' , 'sex']].values.ravel('K'))
```

Out[102…

```
array([True, False, 'male', 'female'], dtype=object)
```

What if we need to clean the data

Cleaning and filtering the data :)

In [103…

```
# Find mussing values inside , sum se total missing values column ke show hon
# aba nechy zahir hy deck = 688 boht ziada missing values hain, ek solution t

ks.isnull().sum()
```

Out[103…

```
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 [104…

```
# how to drop deck , removing missing value, or cleaning data
ks_clean = ks.drop(['deck'] , axis =1 )
ks_clean.head()
```

Out[104…

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

In [105…

```
ks_clean.isnull().sum()
```

Out[105…

```
survived         0
pclass           0
sex              0
age            177
sibsp            0
parch            0
fare             0
embarked         2
class            0
who              0
adult_male       0
embark_town      2
alive            0
alone            0
dtype: int64
```

In [106…

```
ks_clean.shape
```

Out[106…

```
(891, 14)
```

In [107…

```
# removing all missing values
ks_clean = ks_clean.dropna()
```

In [108…

```
# clear hogaya sab data missing value ka
ks_clean.isnull().sum()
```

Out[108…

```
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
```

In [109…

```
ks_clean.shape
```

Out[109…

```
(712, 14)
```

In [110…

```
ks.shape
```

Out[110…

```
(891, 15)
```

In [111…

```
# Value count
# ek column ka name dena parega or phir oski value counts ajati hai
ks_clean['sex'].value_counts()
```

Out[111…

```
male      453
female    259
Name: sex, dtype: int64
```

In [112…

```
# its important to clean the data , ab dono ke describe dekty hain ks ka or k
ks.describe()
```

Out[112…

	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 [113…

```
ks_clean.describe()
# ab yaha dono ka mean dekhain,
# raw data me mean survival rate .38 hy or clean me .40 to iska matlab hy nul.
```

Out[113…

	survived	pclass	age	sibsp	parch	fare
count	712.000000	712.000000	712.000000	712.000000	712.000000	712.000000
mean	0.404494	2.240169	29.642093	0.514045	0.432584	34.567251
std	0.491139	0.836854	14.492933	0.930692	0.854181	52.938648
min	0.000000	1.000000	0.420000	0.000000	0.000000	0.000000
25%	0.000000	1.000000	20.000000	0.000000	0.000000	8.050000
50%	0.000000	2.000000	28.000000	0.000000	0.000000	15.645850
75%	1.000000	3.000000	38.000000	1.000000	1.000000	33.000000
max	1.000000	3.000000	80.000000	5.000000	6.000000	512.329200

Its important to clean outliers as follows :-

In [114…

```
# How to fid outliers
ks_clean.columns
```

Out[114…

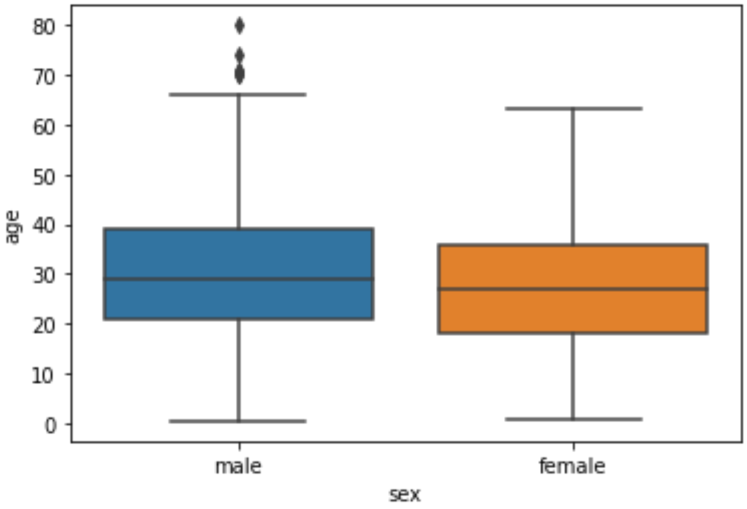
```
Index(['survived', 'pclass', 'sex', 'age', 'sibsp', 'parch', 'fare',
       'embarked', 'class', 'who', 'adult_male', 'embark_town', 'alive',
       'alone'],
      dtype='object')
```

In [115…

```
sns.boxplot ( x ='sex' , y = 'age' , data = ks_clean)
# following box plot se zahir hy age bahr ja raha hy jo dots hain, outliers ha
```

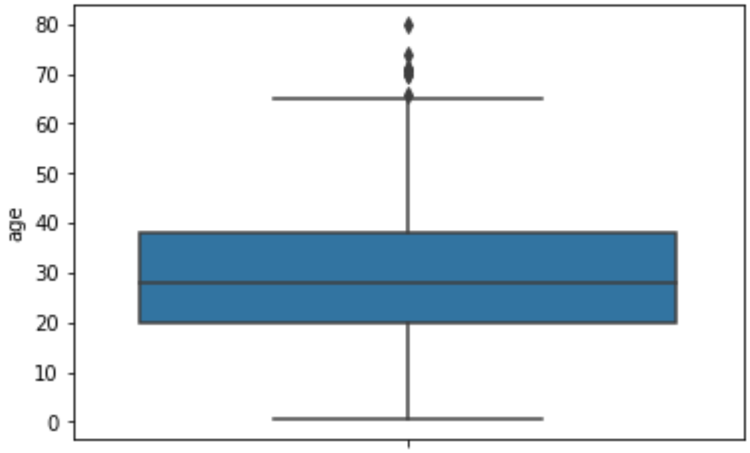
Out[115…

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



```
In [116…
sns.boxplot ( y = 'age' , data = ks_clean)
# sirf age ko dekty hain
# necy wali line box plot ke min value hy ,
# opar wali line max value hy
# darmyan wala box interquartile range hota hy
# or box k andar wali line mean hoti hy
# or jo en sab se bahr hy wo outlier hy
```

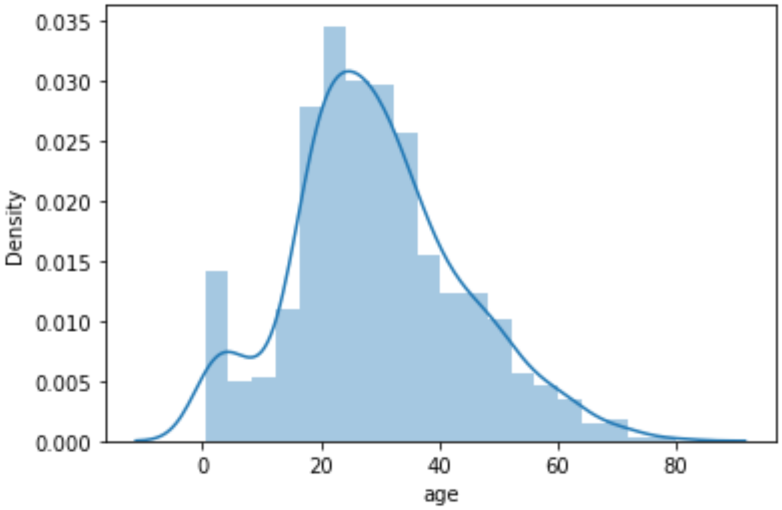
```
Out[116… <AxesSubplot:ylabel='age'>
```



```
In [117…
# esko further dekhny k liye hum dist or density plot dekty hai
# esko bell curve be bolty hain or normality graph be
# data ke dispression hy perfect bell curve nai hy left side pe khrab hy , ou
sns.distplot(ks_clean['age'])
```

```
C:\Users\Asad\anaconda3\lib\site-packages\seaborn\distributions.py:2557: Futur
eWarning: `distplot` is a deprecated function and will be removed in a future
version. Please adapt your code to use either `displot` (a figure-level functi
on with similar flexibility) or `histplot` (an axes-level function for histogr
ams).
  warnings.warn(msg, FutureWarning)
```

```
Out[117… <AxesSubplot:xlabel='age', ylabel='Density'>
```



In [118…

```
# Out liers removal

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

Out[118…

```
29.64209269662921
```

In [119…

```
ks_clean['age'] < 68
```

Out[119…

```
0      True
1      True
2      True
3      True
4      True
       ...
885    True
886    True
887    True
889    True
890    True
Name: age, Length: 712, dtype: bool
```

In [120…

```
ks_clean['age'].mean()
ks_clean.head()
```

Out[120…

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

In [121…

```
ks_clean=ks_clean[ks_clean['age'] <68]
ks_clean.head()
#['age']=ks_clean['age'] < 68
#ks_clean['age'].mean()
```

Out[121…

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

In [122…

```
ks_clean.shape
```

Out[122… (705, 14)

In [123…

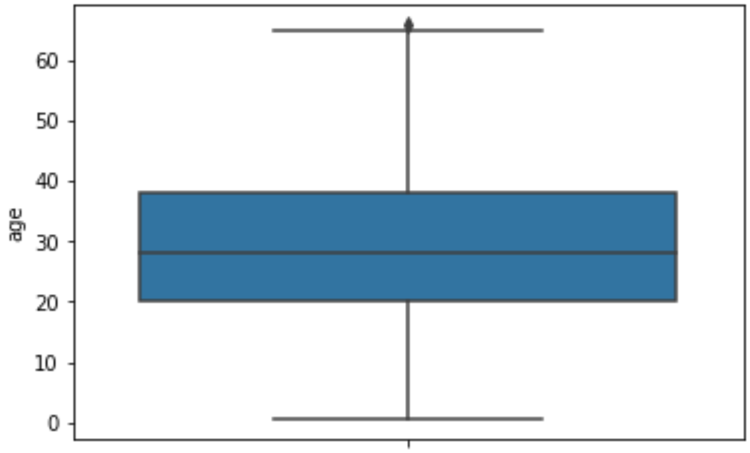
```
ks_clean['age'].mean()
```

Out[123… 29.21797163120567

In [124…

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

Out[124… <AxesSubplot:ylabel='age'>



In [125…

```
ks_clean.head()
```

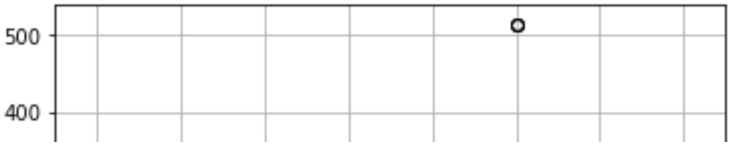
Out[125…

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

In [127…

```
ks_clean.boxplot()
# yaha pe zahir he fare wala column me out liers boht ziada hain
```

Out[127… <AxesSubplot:>

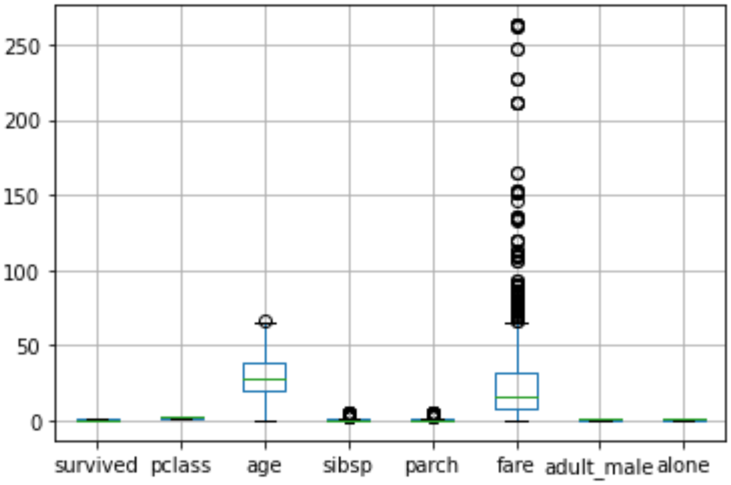


In [130…

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

Out[130…

```
<AxesSubplot:>
```



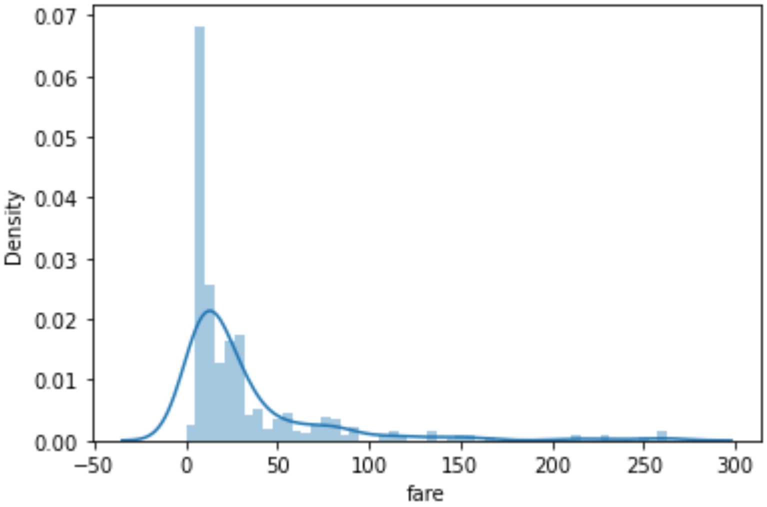
In [132…

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

```
C:\Users\Asad\anaconda3\lib\site-packages\seaborn\distributions.py:2557: Futur
eWarning: `distplot` is a deprecated function and will be removed in a future
version. Please adapt your code to use either `displot` (a figure-level functi
on with similar flexibility) or `histplot` (an axes-level function for histogr
ams).
  warnings.warn(msg, FutureWarning)
```

Out[132…

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

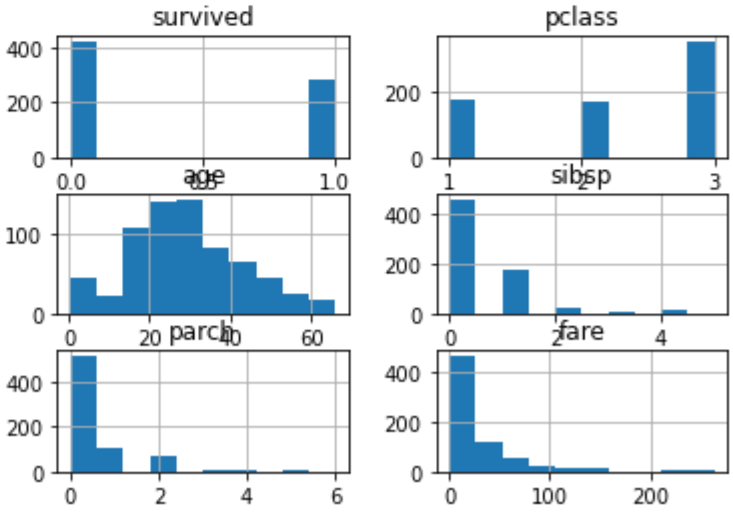


In [133…

```
ks_clean.hist()
```

Out[133…

```
array([[<AxesSubplot:title={'center':'survived'}>,
 <AxesSubplot:title={'center':'pclass'}>],
[<AxesSubplot:title={'center':'age'}>,
 <AxesSubplot:title={'center':'sibsp'}>],
[<AxesSubplot:title={'center':'parch'}>,
 <AxesSubplot:title={'center':'fare'}>]], dtype=object)
```



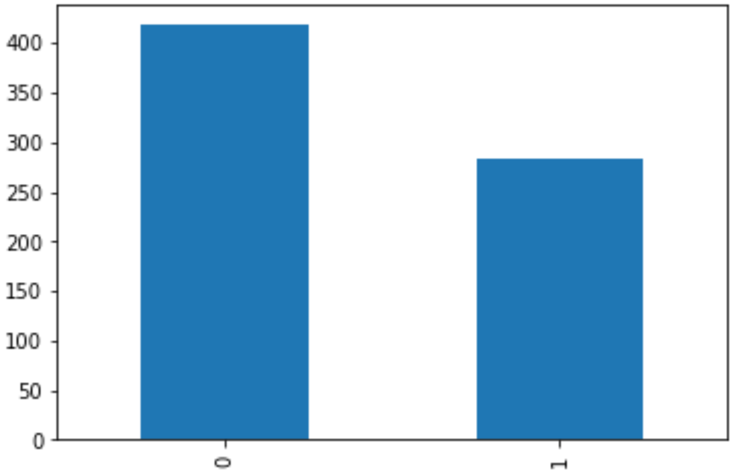
In [135…

```
# value count ka be bar graph bana skty hain ,

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

Out[135…

```
<AxesSubplot:>
```



In [136…

```
# group by karna chaye to

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

Out[136…

		survived	pclass	age	sibsp	parch	fare	adult_male	alone
sex	**class**								
female	**First**	0.963415	1.0	34.231707	0.560976	0.512195	103.696393	0.000000	0.353659
	Second	0.918919	2.0	28.722973	0.500000	0.621622	21.951070	0.000000	0.405405
	Third	0.460784	3.0	21.750000	0.823529	0.950980	15.875369	0.000000	0.372549

		survived	pclass	age	sibsp	parch	fare	adult_male	alone
sex	**class**								
male	**First**	0.389474	1.0	40.867579	0.389474	0.236842	63.901096	0.968421	0.526316

In [137…

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

Out[137…

		survived	pclass	age	sibsp	parch	fare	adult_male	alone
sex	**class**								
female	**First**	0.968085	1.0	34.611765	0.553191	0.457447	106.125798	0.000000	0.361702
	Second	0.921053	2.0	28.722973	0.486842	0.605263	21.970121	0.000000	0.421053
	Third	0.500000	3.0	21.750000	0.895833	0.798611	16.118810	0.000000	0.416667
male	**First**	0.368852	1.0	41.281386	0.311475	0.278689	67.226127	0.975410	0.614754
	Second	0.157407	2.0	30.740707	0.342593	0.222222	19.741782	0.916667	0.666667
	Third	0.135447	3.0	26.507589	0.498559	0.224784	12.661633	0.919308	0.760807

In [138…

```
# clean karny k bad data ke sari accuracy result change ho jaty hain
```

Relationship

In [141…

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

In [143…

```
sns.heatmap(corr_ks_clean)
# Heat map
# yaha pe heat map hamy co-relation dekha raha hy, right side pe bar me zero
# agar 0 se opar positive to positive relation or direct relation
# agar 0 se neechy ho to negative or in-direct relation
```

Out[143…

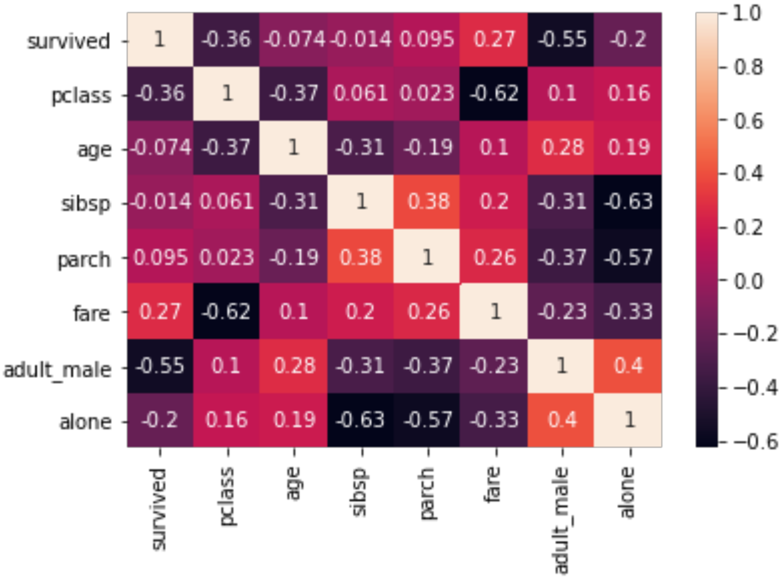
```
<AxesSubplot:>
```

In [144…

```
sns.heatmap(corr_ks_clean , annot=True ) # yaha pe values show hojani hain
```

Out[144…

```
<AxesSubplot:>
```

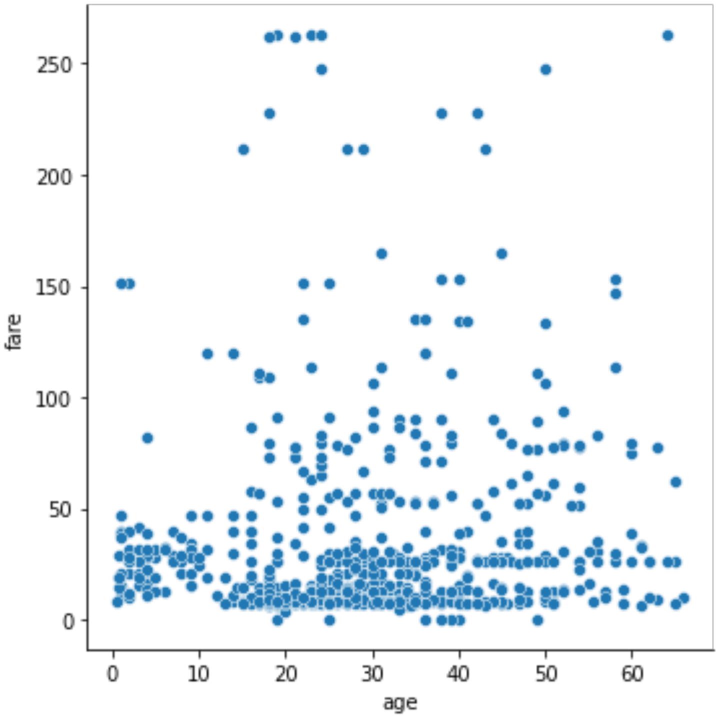


In [145…

```
sns.relplot(x='age' , y='fare' , data=ks_clean)
```

Out[145…

```
<seaborn.axisgrid.FacetGrid at 0x2277b600e20>
```

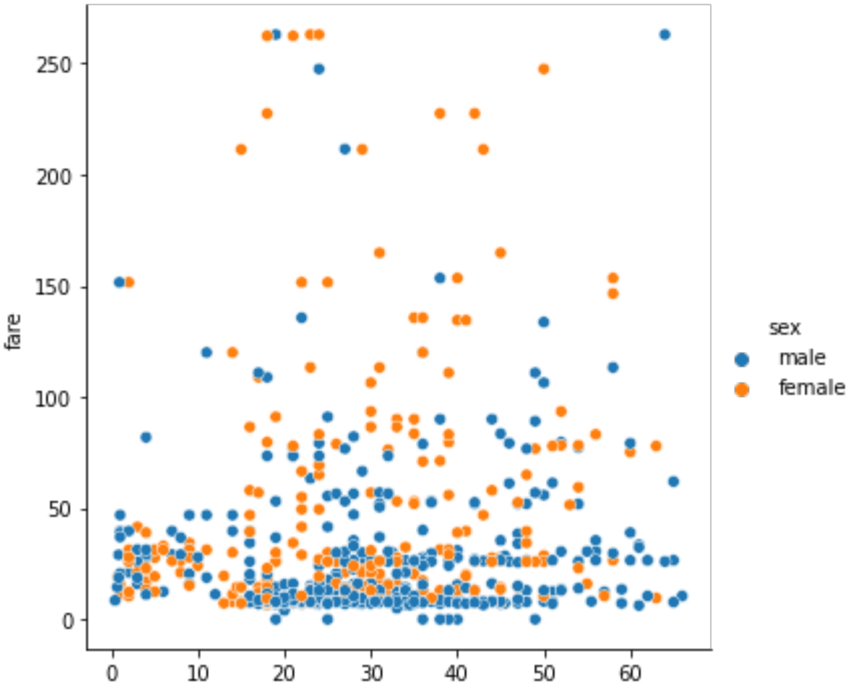


In [146…

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

Out[146…

```
<seaborn.axisgrid.FacetGrid at 0x2277b9f5b80>
```

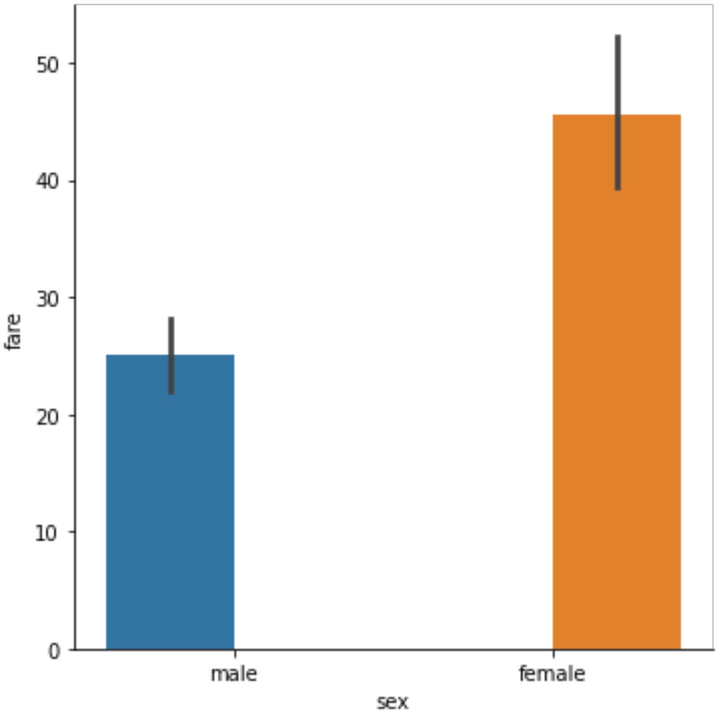


In [149…

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

Out[149…

```
<seaborn.axisgrid.FacetGrid at 0x2277bb474f0>
```

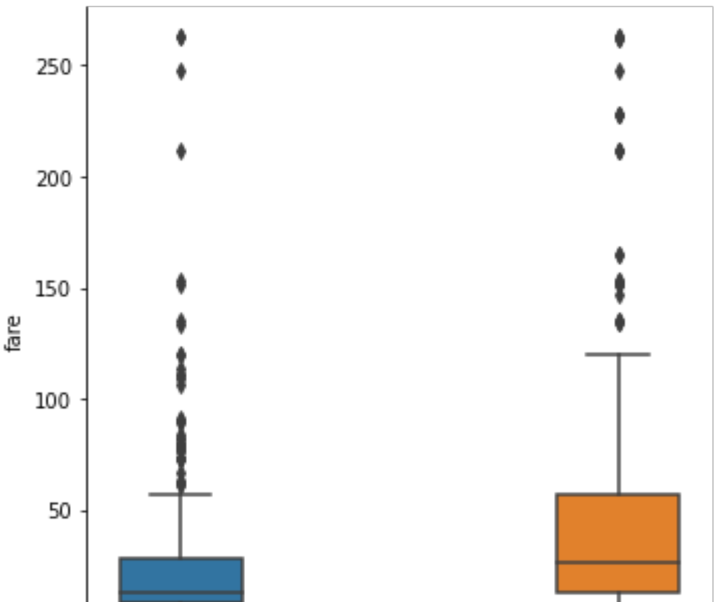


In [150…

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

Out[150…

```
<seaborn.axisgrid.FacetGrid at 0x2277bc10c70>
```

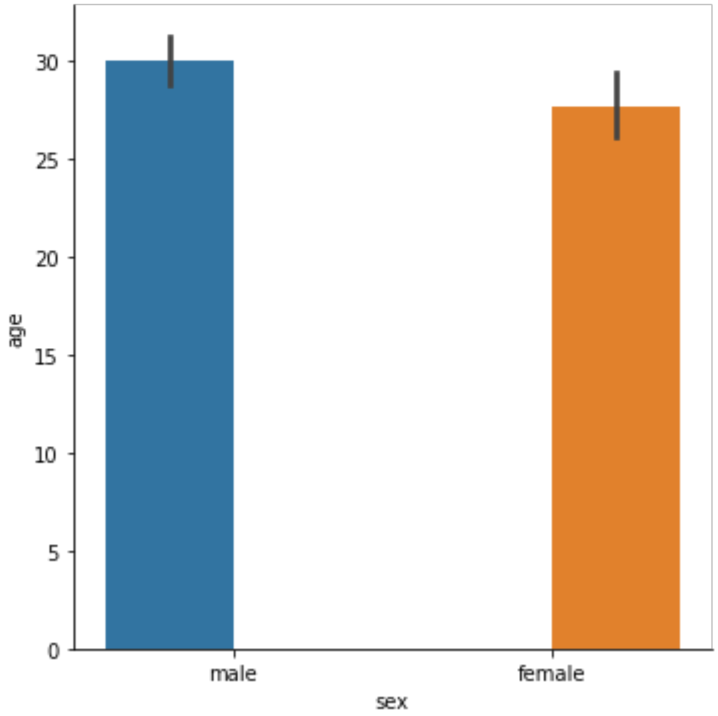


In [151…

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

Out[151…

```
<seaborn.axisgrid.FacetGrid at 0x2277bcbd1f0>
```



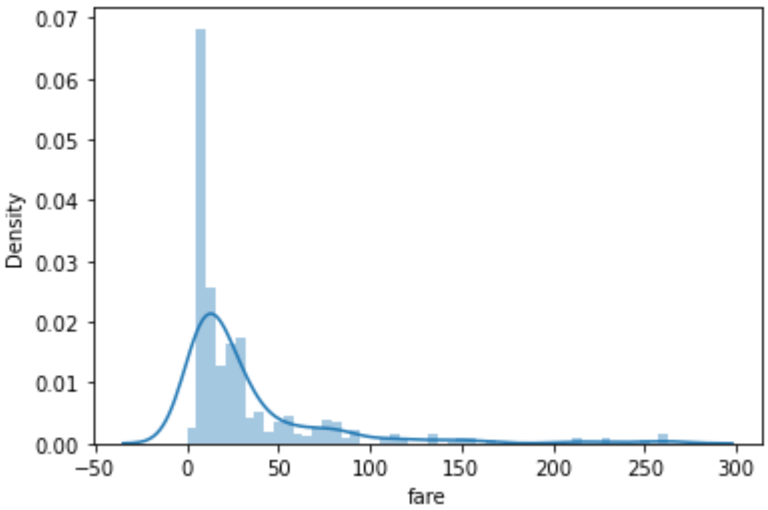
In [153…

```
# Agr log lain or new column banaye

sns.distplot(ks_clean['fare'])
ks_clean['fare_log'] = np.log(ks_clean['fare'])
```

```
C:\Users\Asad\anaconda3\lib\site-packages\seaborn\distributions.py:2557: Futur
eWarning: `distplot` is a deprecated function and will be removed in a future
version. Please adapt your code to use either `displot` (a figure-level functi
on with similar flexibility) or `histplot` (an axes-level function for histogr
ams).
  warnings.warn(msg, FutureWarning)
```

```
C:\Users\Asad\anaconda3\lib\site-packages\pandas\core\arraylike.py:358: Runtim
eWarning: divide by zero encountered in log
  result = getattr(ufunc, method)(*inputs, **kwargs)
```



In [155…

```
ks_clean.head()
```

Out[155…

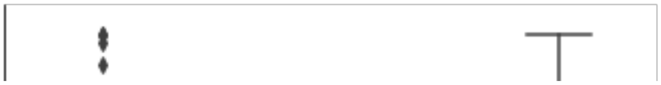
	survived	pclass	sex	age	sibsp	parch	fare	embarked	class	who	adult_male	emba
0	0	3	male	22.0	1	0	7.2500	S	Third	man	True	South
1	1	1	female	38.0	1	0	71.2833	C	First	woman	False	Ch
2	1	3	female	26.0	0	0	7.9250	S	Third	woman	False	South
3	1	1	female	35.0	1	0	53.1000	S	First	woman	False	South
4	0	3	male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

In [156…

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

Out[156…

```
<seaborn.axisgrid.FacetGrid at 0x2277bbbeaf0>
```



In [ ]: