ADS Assignment 2

Titanic Ship Case Study

Problem Description: On April 15, 1912, during her maiden voyage, the Titanic sank after colliding with an iceberg, killing 1502 out of 2224 passengers and crew. Translated 32% survival rate.

- One of the reasons that the shipwreck led to such loss of life was that there were not enough lifeboats for the passengers and crew.
- Although there was some element of luck involved in surviving the sinking, some groups of people were more likely to survive than others, such as women, children, and the upper-class.

The problem associated with the Titanic dataset is to predict whether a passenger survived the disaster or not. The dataset contains various features such as passenger class, age, gender, cabin, fare, and whether the passenger had any siblings or spouses on board. These features can be used to build a predictive model to determine the likelihood of a passenger surviving the disaster. The dataset offers opportunities for feature engineering, data visualization, and model selection, making it a valuable resource for developing and testing data analysis and machine learning skills.

Perform Below Tasks to complete the assignment: -

1. Download the dataset: Dataset
2. Load the dataset

```
import pandas as pd
import numpy as np
```

```
import matplotlib.pyplot as plt
%matplotlib inline
import seaborn as sns
```

```
df=pd.read_csv('titanic.csv')
```

```
df
```

Name: Jyothsna Sriya Mahabhashyam **Reg No.:** 20BCR7067

	survived	pclass	sex	age	sibsp	parch	fare	embarked	class	who	adult_male	deck	embark_town	alive	alone
0	0	3	male	22.0	1	0	7.2500	S	Third	man	True	NaN	Southampton	no	False
1	1	1	female	38.0	1	0	71.2833	C	First	woman	False	C	Cherbourg	yes	False
2	1	3	female	26.0	0	0	7.9250	S	Third	woman	False	NaN	Southampton	yes	True
3	1	1	female	35.0	1	0	53.1000	S	First	woman	False	C	Southampton	yes	False
4	0	3	male	35.0	0	0	8.0500	S	Third	man	True	NaN	Southampton	no	True
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
886	0	2	male	27.0	0	0	13.0000	S	Second	man	True	NaN	Southampton	no	True
887	1	1	female	19.0	0	0	30.0000	S	First	woman	False	B	Southampton	yes	True
888	0	3	female	NaN	1	2	23.4500	S	Third	woman	False	NaN	Southampton	no	False
889	1	1	male	26.0	0	0	30.0000	C	First	man	True	C	Cherbourg	yes	True
890	0	3	male	32.0	0	0	7.7500	Q	Third	man	True	NaN	Queenstown	no	True

891 rows × 15 columns

```
df.dtypes
```

```
survived          int64
pclass            int64
sex              object
age             float64
sibsp             int64
parch             int64
fare            float64
embarked         object
class            object
who              object
adult_male         bool
deck             object
embark_town      object
alive            object
alone              bool
dtype: object
```

```
df.head()
```

	survived	pclass	sex	age	sibsp	parch	fare	embarked	class	who	adult_male	deck	embark_town	alive	alone
0	0	3	male	22.0	1	0	7.2500	S	Third	man	True	NaN	Southampton	no	False
1	1	1	female	38.0	1	0	71.2833	C	First	woman	False	C	Cherbourg	yes	False
2	1	3	female	26.0	0	0	7.9250	S	Third	woman	False	NaN	Southampton	yes	True
3	1	1	female	35.0	1	0	53.1000	S	First	woman	False	C	Southampton	yes	False
4	0	3	male	35.0	0	0	8.0500	S	Third	man	True	NaN	Southampton	no	True

```
df.tail()
```

	survived	pclass	sex	age	sibsp	parch	fare	embarked	class	who	adult_male	deck	embark_town	alive	alone
886	0	2	male	27.0	0	0	13.00	S	Second	man	True	NaN	Southampton	no	True
887	1	1	female	19.0	0	0	30.00	S	First	woman	False	B	Southampton	yes	True
888	0	3	female	NaN	1	2	23.45	S	Third	woman	False	NaN	Southampton	no	False
889	1	1	male	26.0	0	0	30.00	C	First	man	True	C	Cherbourg	yes	True
890	0	3	male	32.0	0	0	7.75	Q	Third	man	True	NaN	Queenstown	no	True

```
df.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    object
 9   who          891 non-null    object
 10  adult_male   891 non-null    bool
 11  deck         203 non-null    object
 12  embark_town  889 non-null    object
 13  alive        891 non-null    object
 14  alone        891 non-null    bool
dtypes: bool(2), float64(2), int64(4), object(7)
memory usage: 92.4+ KB
```

```
df['age']
```

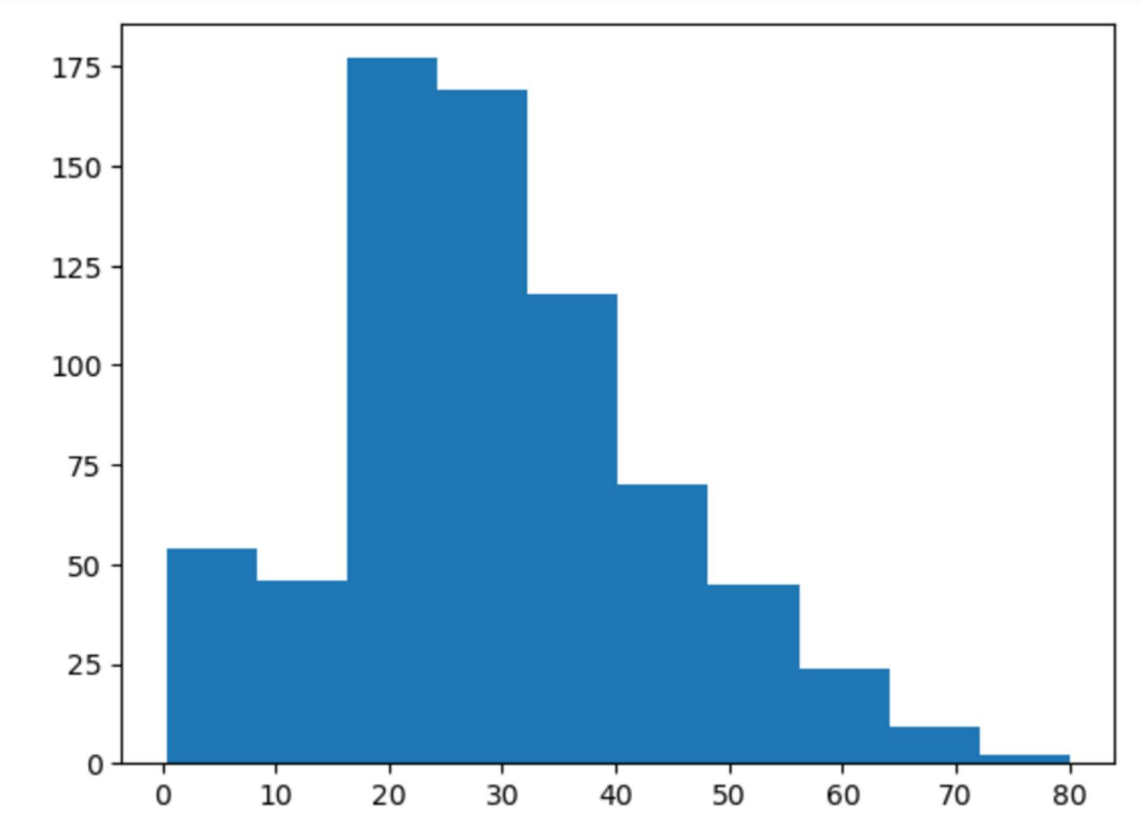
```
0      22.0
1      38.0
2      26.0
3      35.0
4      35.0
       ...
886    27.0
887    19.0
888     NaN
889    26.0
890    32.0
Name: age, Length: 891, dtype: float64
```

3. Perform Below Visualizations.

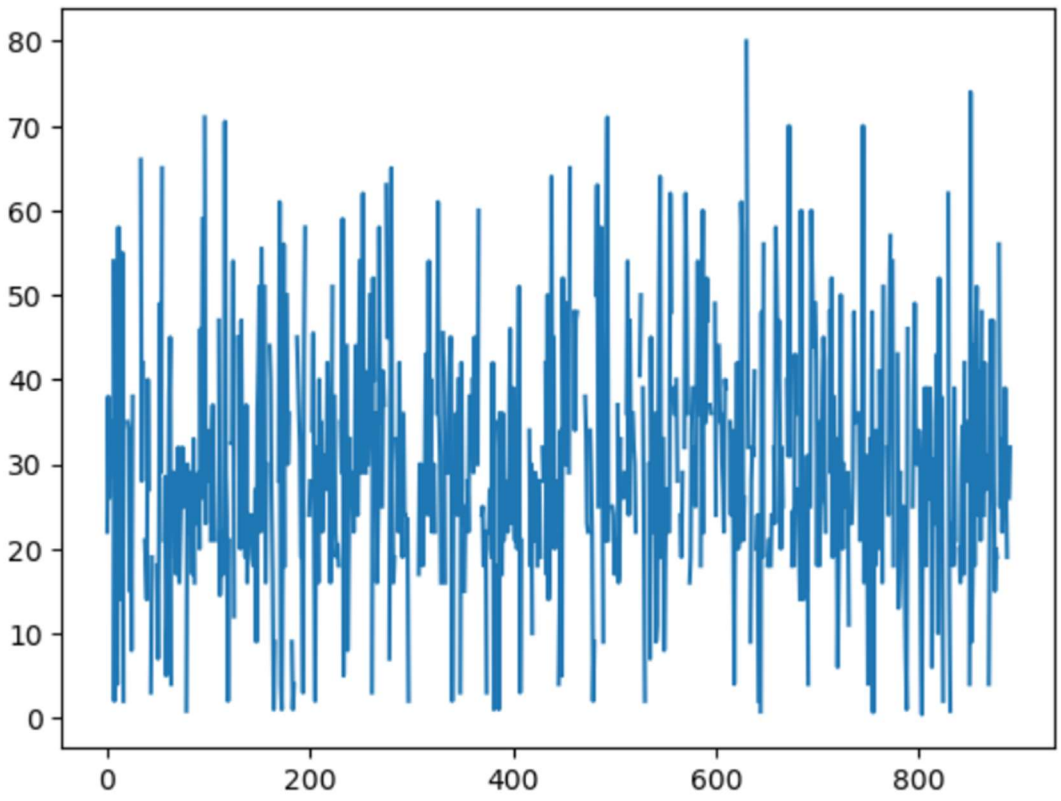
- Univariate Analysis

```
#histogram
plt.hist(df['age'])
```

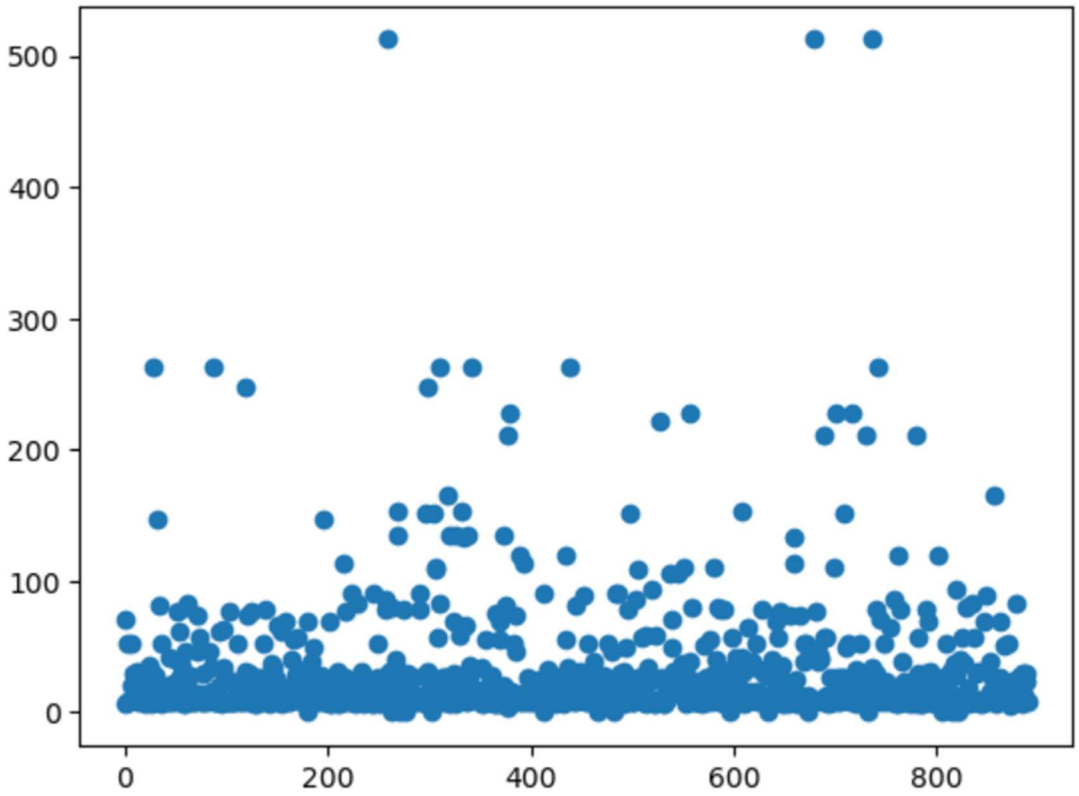
```
(array([ 54.,  46., 177., 169., 118.,  70.,  45.,  24.,   9.,   2.]),
 array([ 0.42 ,  8.378, 16.336, 24.294, 32.252, 40.21 , 48.168, 56.126,
        64.084, 72.042, 80.   ]),
 <BarContainer object of 10 artists>)
```



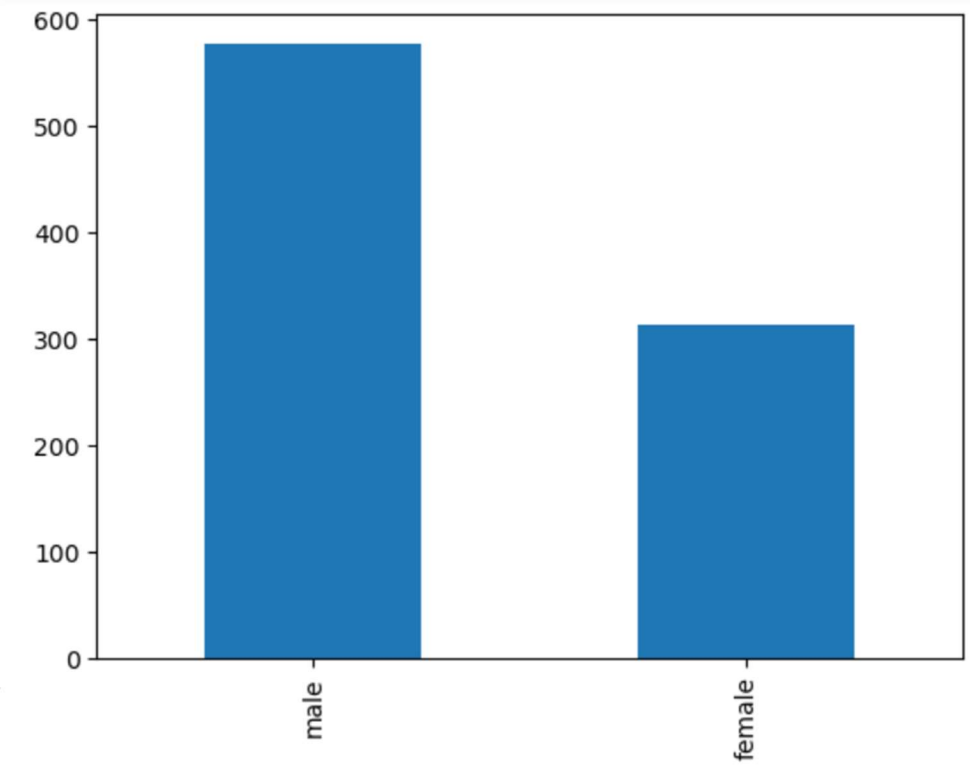

```
#Line plot
plt.plot(df.index,df['age'])
plt.show()
```



```
#Scatterplot
plt.scatter(df.index,df['fare'])
plt.show()
```

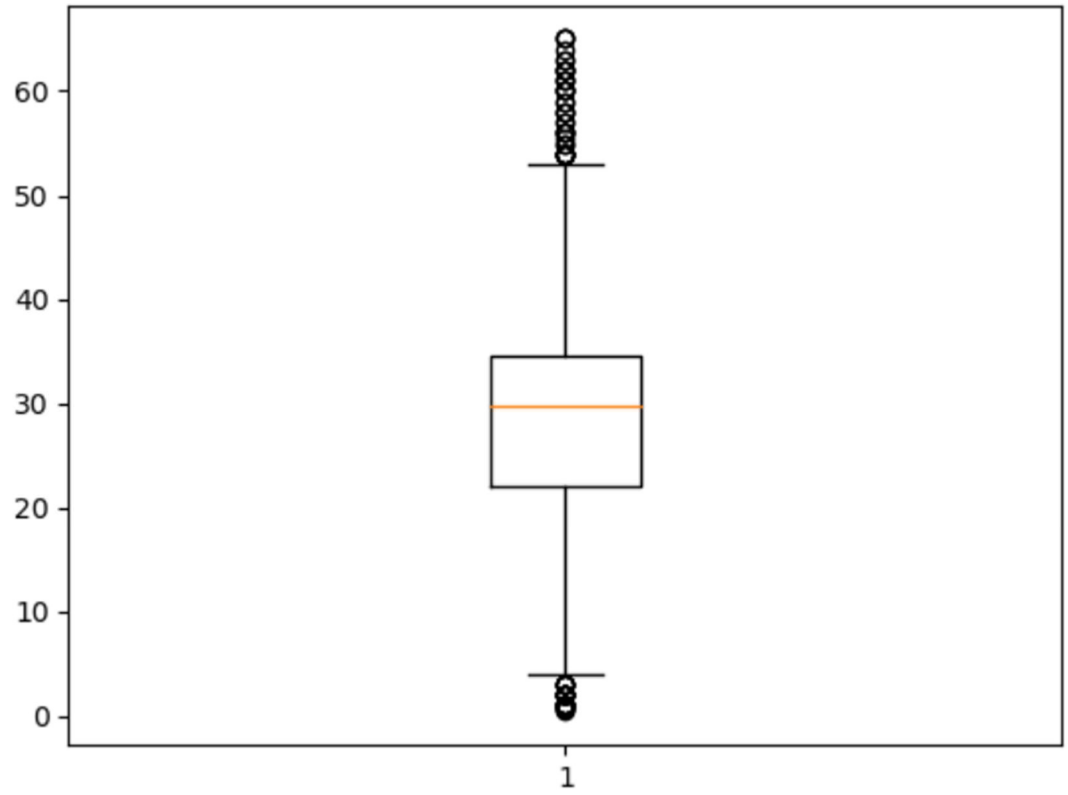



```
#Bar plot (Univariate)
df['sex'].value_counts().plot.bar()
```



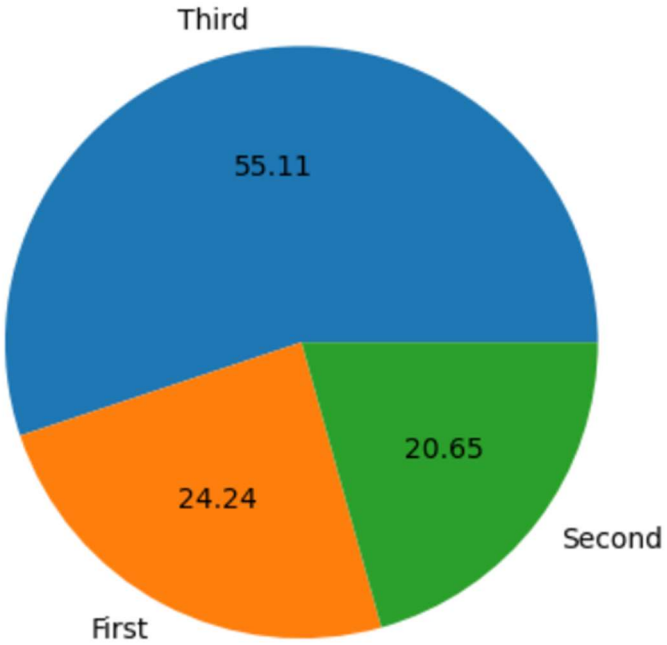
```
#box plot
plt.boxplot(df['age'])
```

```
{'whiskers': [<matplotlib.lines.Line2D at 0x1f7d35cee80>,
  <matplotlib.lines.Line2D at 0x1f7d359d190>],
 'caps': [<matplotlib.lines.Line2D at 0x1f7d359d460>,
  <matplotlib.lines.Line2D at 0x1f7d359d730>],
 'boxes': [<matplotlib.lines.Line2D at 0x1f7d35cebb0>],
 'medians': [<matplotlib.lines.Line2D at 0x1f7d359da00>],
 'fliers': [<matplotlib.lines.Line2D at 0x1f7d359dcd0>],
 'means': []}
```




```
# Pie chart
d=df['class'].value_counts()
list=df['class'].unique()
plt.pie(d,autopct='%.2f',labels=list)
```

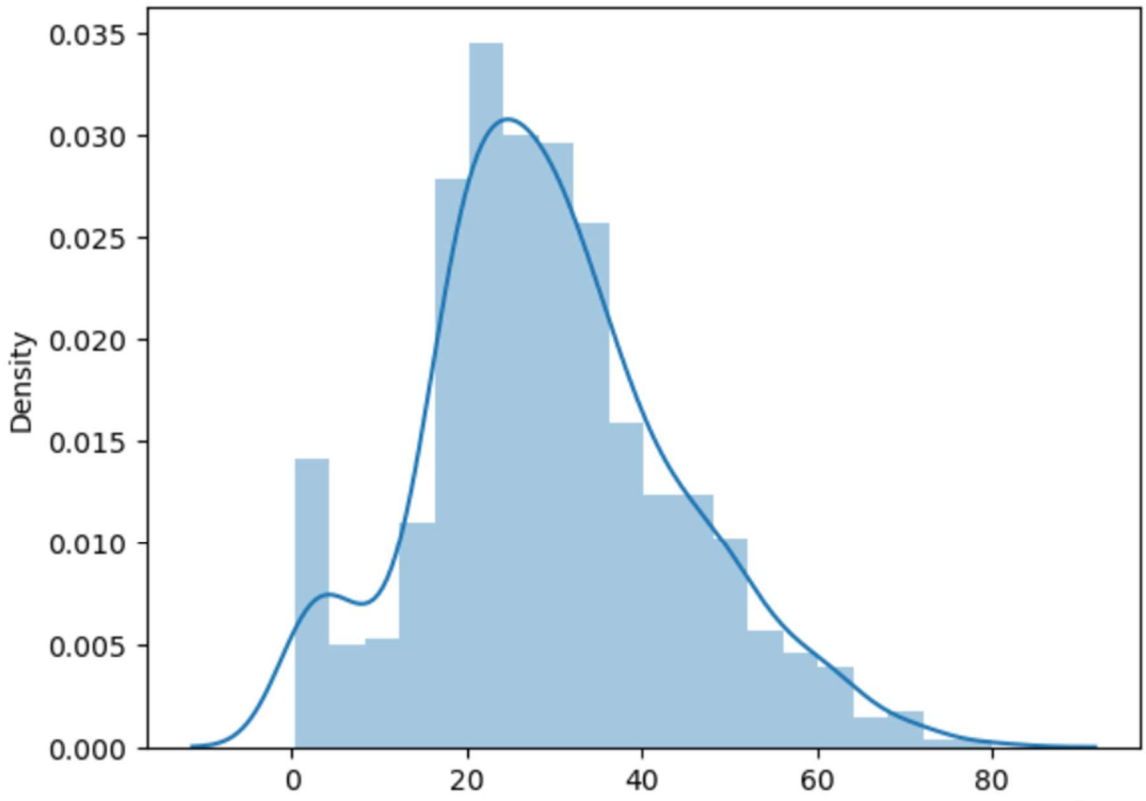
```
([<matplotlib.patches.Wedge at 0x1f7c595eb50>,
  <matplotlib.patches.Wedge at 0x1f7c596c2b0>,
  <matplotlib.patches.Wedge at 0x1f7c596c9d0>],
 [Text(-0.17571616709359641, 1.0858746836637898, 'Third'),
  Text(-0.5160762002061153, -0.9714243951954356, 'First'),
  Text(0.8765111456658484, -0.6646263698677192, 'Second')],
 [Text(-0.09584518205105258, 0.5922952819984307, '55.11'),
  Text(-0.2814961092033356, -0.5298678519247829, '24.24'),
  Text(0.47809698854500815, -0.3625234744733013, '20.65')])
```




```
sns.distplot(df.age)
```

```
C:\Users\Sriya\anaconda3\lib\site-packages\seaborn\distributions.py:2619: FutureWarning: `distplot` is a deprecated function an
d will be removed in a future version. Please adapt your code to use either `displot` (a figure-level function with similar fle
xibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```

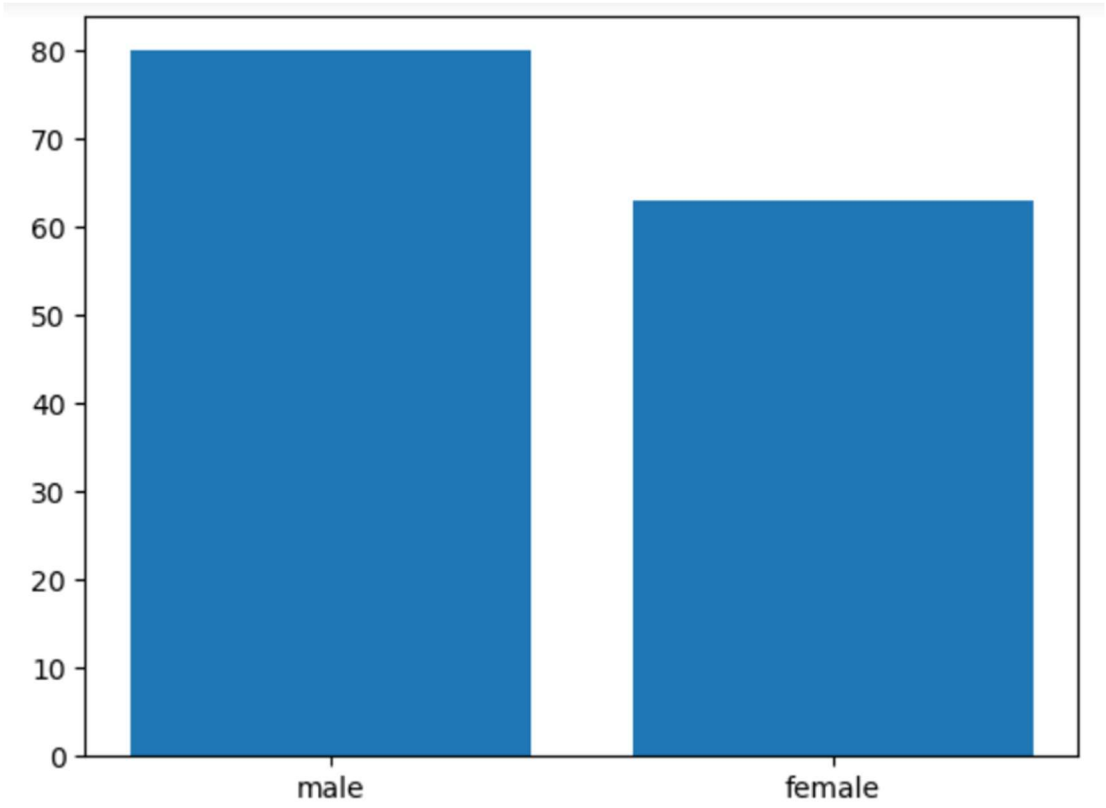
```
<AxesSubplot:xlabel='age', ylabel='Density'>
```



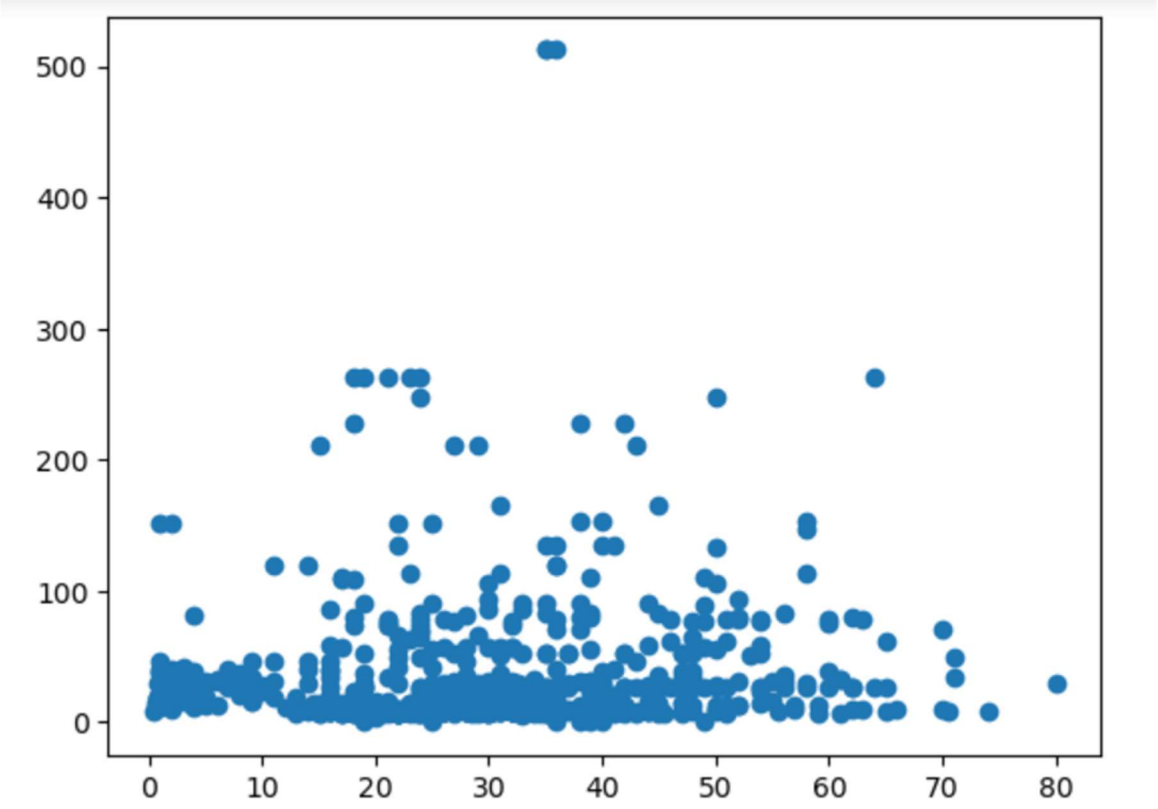
- Bi - Variate Analysis

```
#bar plot
plt.bar(df['sex'],df['age'])
```

```
<BarContainer object of 891 artists>
```



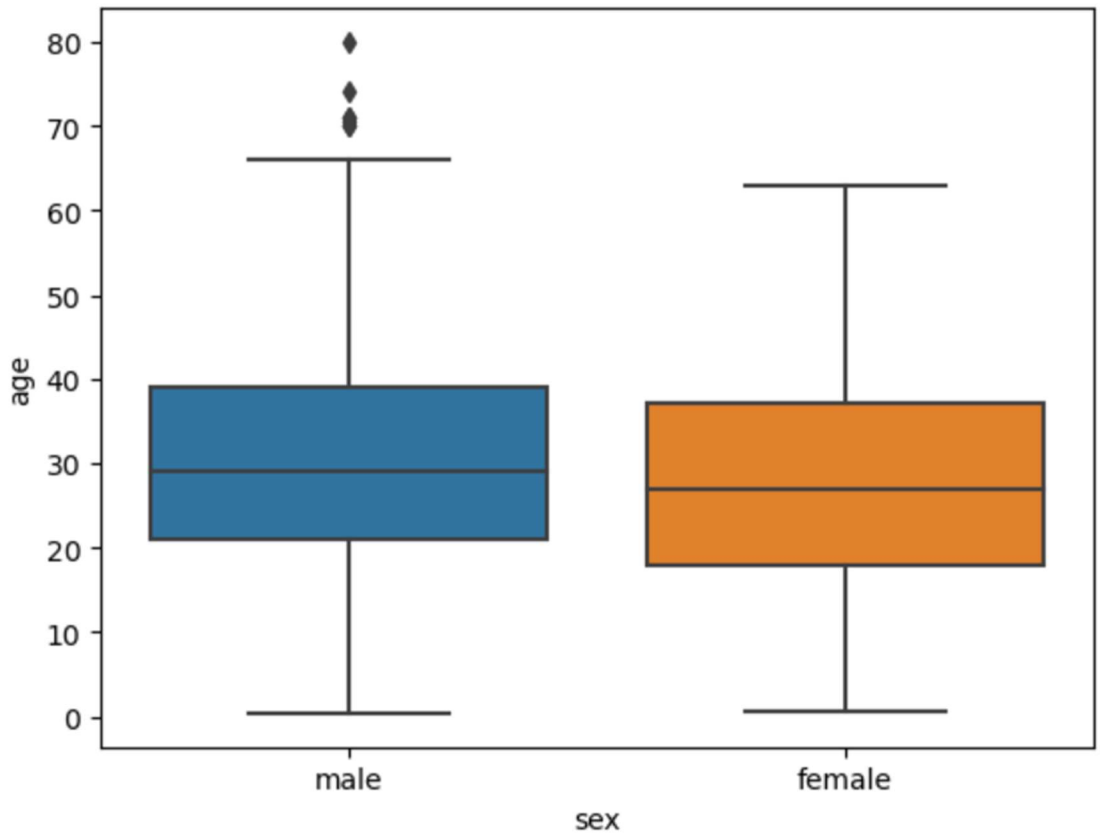

```
plt.scatter(df['age'],df['fare'])
plt.show()
```




```
sns.boxplot(df['sex'],df['age'])
```

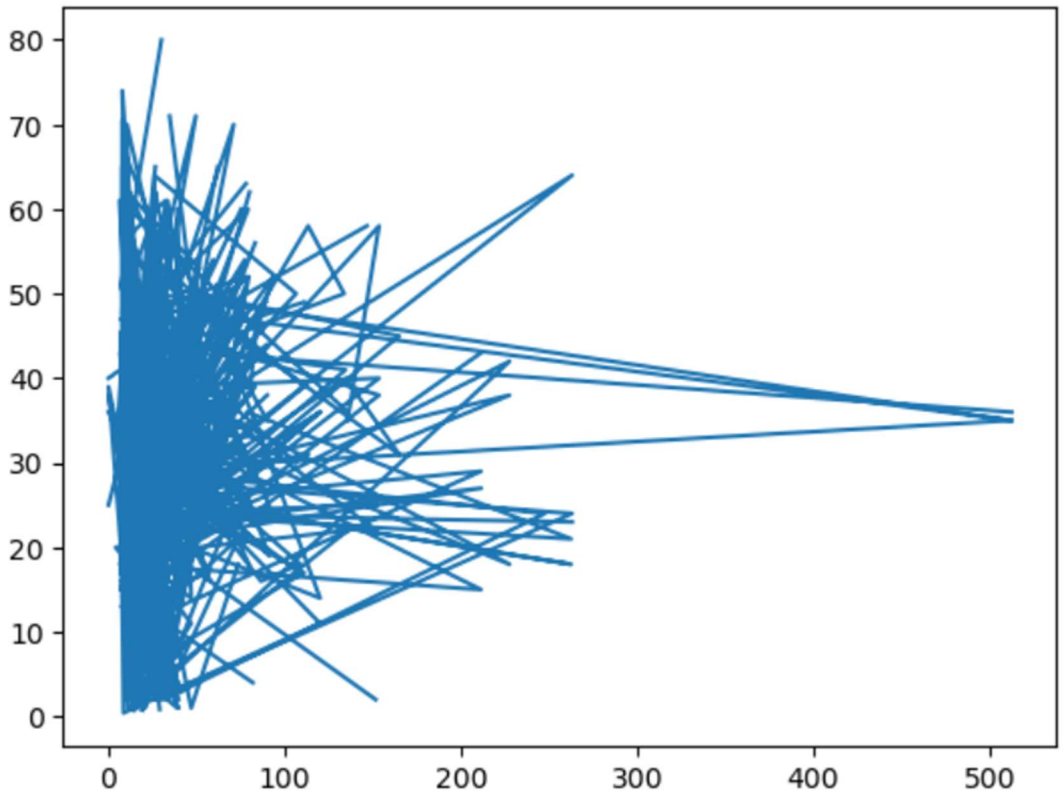
```
C:\Users\Sriya\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass the following variables as keyword ar
gs: x, y. From version 0.12, the only valid positional argument will be `data`, and passing other arguments without an explicit
keyword will result in an error or misinterpretation.
  warnings.warn(
```

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



```
plt.plot(df['fare'],df['age'])
```

```
[<matplotlib.lines.Line2D at 0x1f7c74089a0>]
```



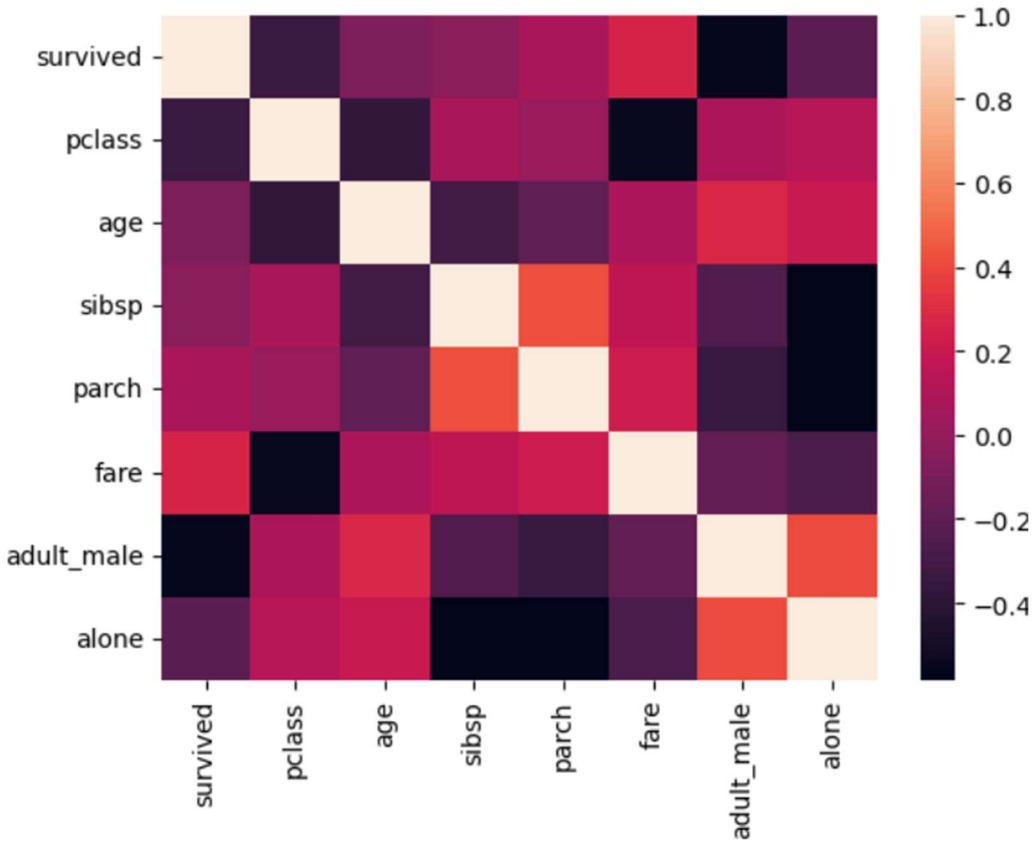
- Multi - Variate Analysis

```
df.corr()
```

	survived	pclass	age	sibsp	parch	fare	adult_male	alone
survived	1.000000	-0.338481	-0.077221	-0.035322	0.081629	0.257307	-0.557080	-0.203367
pclass	-0.338481	1.000000	-0.369226	0.083081	0.018443	-0.549500	0.094035	0.135207
age	-0.077221	-0.369226	1.000000	-0.308247	-0.189119	0.096067	0.280328	0.198270
sibsp	-0.035322	0.083081	-0.308247	1.000000	0.414838	0.159651	-0.253586	-0.584471
parch	0.081629	0.018443	-0.189119	0.414838	1.000000	0.216225	-0.349943	-0.583398
fare	0.257307	-0.549500	0.096067	0.159651	0.216225	1.000000	-0.182024	-0.271832
adult_male	-0.557080	0.094035	0.280328	-0.253586	-0.349943	-0.182024	1.000000	0.404744
alone	-0.203367	0.135207	0.198270	-0.584471	-0.583398	-0.271832	0.404744	1.000000

```
sns.heatmap(df.corr())
```

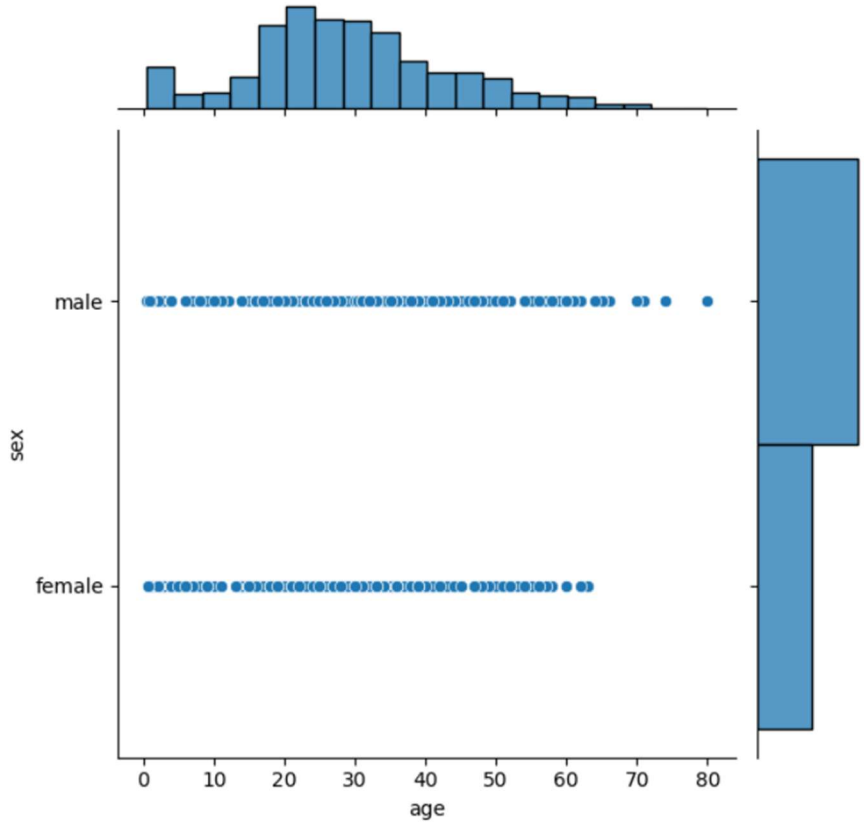
```
<AxesSubplot:>
```




```
sns.jointplot(df.age,df.sex)
```

```
C:\Users\Sriya\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass the following variables as keyword ar
gs: x, y. From version 0.12, the only valid positional argument will be `data`, and passing other arguments without an explicit
keyword will result in an error or misinterpretation.
  warnings.warn(
```

```
<seaborn.axisgrid.JointGrid at 0x1f7cc08bd00>
```



4. Perform descriptive statistics on the dataset.

```
#Mesure of Central Tendency
df.mean()
```

```
C:\Users\Sriya\AppData\Local\Temp\ipykernel_25880\1221912073.py:2: FutureWarning: Dropping of nuisance columns in DataFrame red
uctions (with 'numeric_only=None') is deprecated; in a future version this will raise TypeError.  Select only valid columns bef
ore calling the reduction.
  df.mean()
```

```
survived       0.383838
pclass         2.308642
age           29.699118
sibsp          0.523008
parch          0.381594
fare          32.204208
adult_male     0.602694
alone          0.602694
dtype: float64
```

```
df.mode()
```

	survived	pclass	sex	age	sibsp	parch	fare	embarked	class	who	adult_male	deck	embark_town	alive	alone
0	0	3	male	24.0	0	0	8.05	S	Third	man	True	C	Southampton	no	True

```
df.median()
```

```
C:\Users\Sriya\AppData\Local\Temp\ipykernel_25880\530051474.py:1: FutureWarning: Dropping of nuisance columns in DataFrame redu
ctions (with 'numeric_only=None') is deprecated; in a future version this will raise TypeError.  Select only valid columns befo
re calling the reduction.
  df.median()
```

```
survived       0.0000
pclass         3.0000
age           28.0000
sibsp          0.0000
parch          0.0000
fare          14.4542
adult_male     1.0000
alone          1.0000
dtype: float64
```

```
df.skew()
```

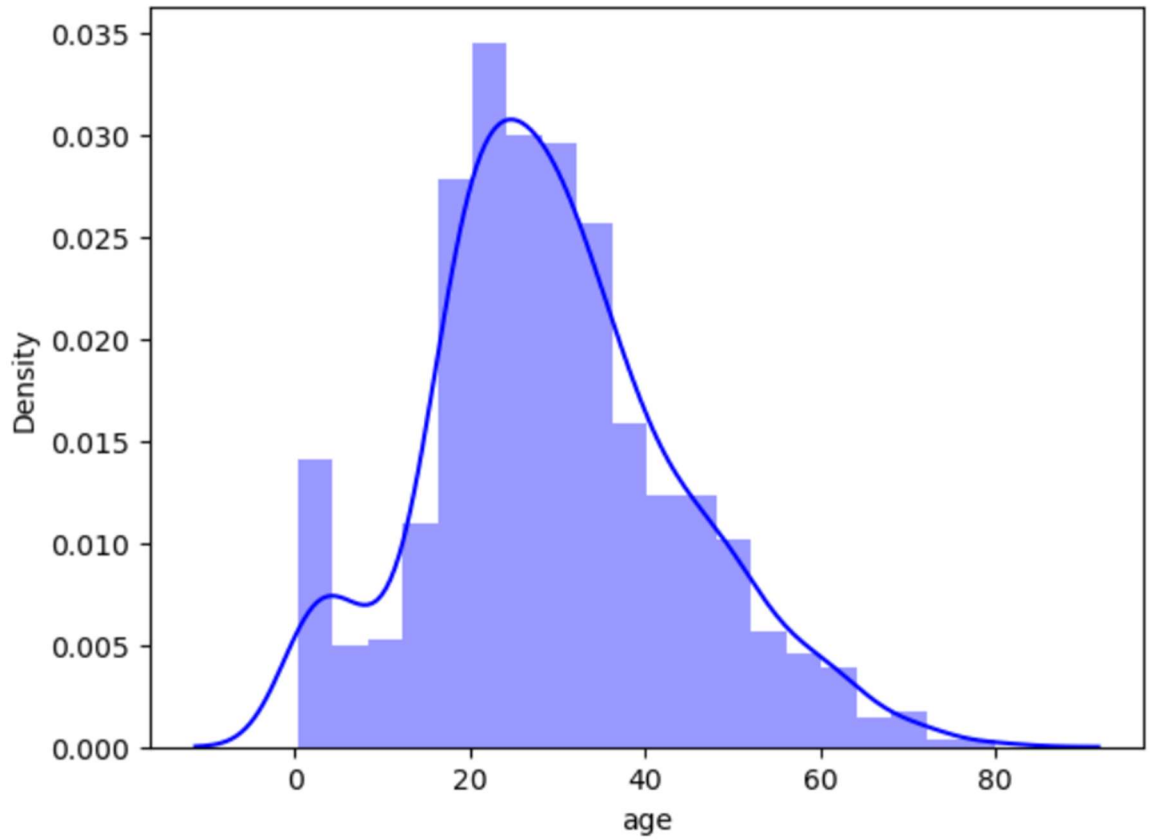
```
C:\Users\Sriya\AppData\Local\Temp\ipykernel_25880\1665899112.py:1: FutureWarning: Dropping of nuisance columns in DataFrame red
uctions (with 'numeric_only=None') is deprecated; in a future version this will raise TypeError.  Select only valid columns bef
ore calling the reduction.
  df.skew()
```

```
survived      0.478523
pclass       -0.630548
age           0.389108
sibsp         3.695352
parch         2.749117
fare          4.787317
adult_male   -0.420431
alone        -0.420431
dtype: float64
```

```
print(sns.distplot(df['age'],color='blue'))
```

```
C:\Users\Sriya\anaconda3\lib\site-packages\seaborn\distributions.py:2619: FutureWarning: `distplot` is a deprecated function an
d will be removed in a future version. Please adapt your code to use either `displot` (a figure-level function with similar fle
xibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```

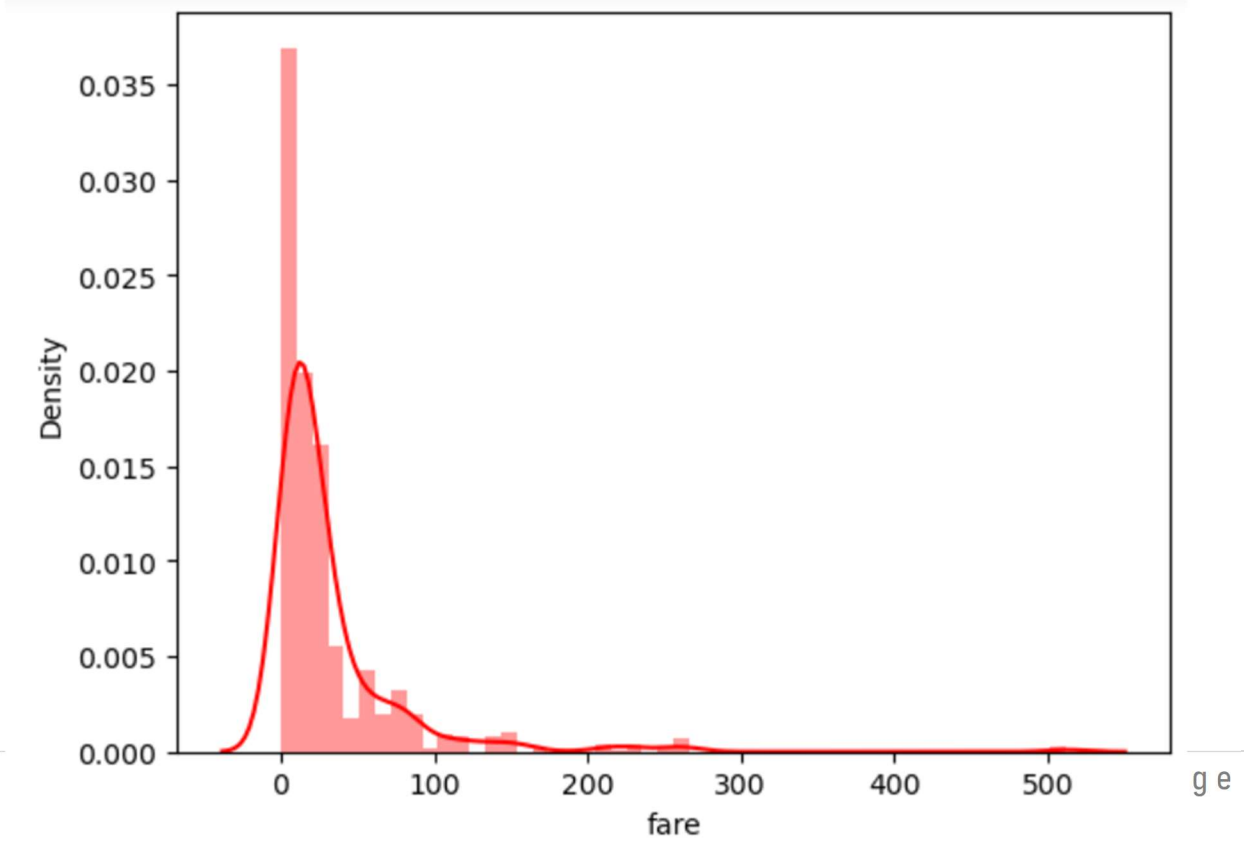
```
AxesSubplot(0.125,0.11;0.775x0.77)
```




```
print(sns.distplot(df['fare'],color='red'))
```

```
C:\Users\Sriya\anaconda3\lib\site-packages\seaborn\distributions.py:2619: FutureWarning: `distplot` is a deprecated function an
d will be removed in a future version. Please adapt your code to use either `displot` (a figure-level function with similar fle
xibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```

```
AxesSubplot(0.125,0.11;0.775x0.77)
```



```
df.kurt()
```

```
C:\Users\Sriya\AppData\Local\Temp\ipykernel_25880\1257127604.py:1: FutureWarning: Dropping of nuisance columns in DataFrame red
uctions (with 'numeric_only=None') is deprecated; in a future version this will raise TypeError.  Select only valid columns bef
ore calling the reduction.
  df.kurt()
```

```
survived      -1.775005
pclass        -1.280015
age            0.178274
sibsp         17.880420
parch          9.778125
fare          33.398141
adult_male    -1.827345
alone         -1.827345
dtype: float64
```

```
max(df.age)
```

```
80.0
```

```
min(df.age)
```

```
0.42
```

```
range=max(df.age)-min(df.age)
range
```

```
79.58
```

```
data=df.age
quan=data.quantile(q=[0.75,0.25])
quan
```

```
0.75    38.000
0.25    20.125
Name: age, dtype: float64
```

```
quan.iloc[0]
```

```
38.0
```

```
quan.iloc[1]
```

```
20.125
```

```
IQR=quan.iloc[0]-quan.iloc[1]
IQR
```

```
17.875
```

```
uex=quan.iloc[0]+(1.5*IQR)
uex
```

```
64.8125
```

```
lex=quan.iloc[1]-(1.5*IQR)
lex
```

```
-6.6875
```

```
df.var()
```

```
C:\Users\Sriya\AppData\Local\Temp\ipykernel_25880\1568254755.py:1: FutureWarning: Dropping of nuisance columns in DataFrame red
uctions (with 'numeric_only=None') is deprecated; in a future version this will raise TypeError.  Select only valid columns bef
ore calling the reduction.
  df.var()
```

```
survived         0.236772
pclass           0.699015
age            211.019125
sibsp            1.216043
parch            0.649728
fare          2469.436846
adult_male       0.239723
alone            0.239723
dtype: float64
```

```
df.std()
```

```
C:\Users\Sriya\AppData\Local\Temp\ipykernel_25880\3390915376.py:1: FutureWarning: Dropping of nuisance columns in DataFrame red
uctions (with 'numeric_only=None') is deprecated; in a future version this will raise TypeError.  Select only valid columns bef
ore calling the reduction.
  df.std()
```

```
survived       0.486592
pclass         0.836071
age           14.526497
sibsp          1.102743
parch          0.806057
fare          49.693429
adult_male     0.489615
alone          0.489615
dtype: float64
```

5. Handle the Missing values.

```
df.columns
```

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

```
df.dtypes
```

```
survived         int64
pclass           int64
sex             object
age            float64
sibsp            int64
parch            int64
fare           float64
embarked        object
class           object
who             object
adult_male        bool
deck            object
embark_town     object
alive           object
alone             bool
dtype: object
```

```
df.describe()
```

	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

```
df.isna()
```

	survived	pclass	sex	age	sibsp	parch	fare	embarked	class	who	adult_male	deck	embark_town	alive	alone
0	False	False	False	False	False	False	False	False	False	False	False	True	False	False	False
1	False	False	False	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	False	False	False
3	False	False	False	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	False	False	False
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
886	False	False	False	False	False	False	False	False	False	False	False	True	False	False	False
887	False	False	False	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	False	False	False
889	False	False	False	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	False	False	False

891 rows × 15 columns

```
df.isnull().any()
```

```
survived       False
pclass         False
sex            False
age             True
sibsp          False
parch          False
fare           False
embarked        True
class          False
who            False
adult_male     False
deck            True
embark_town     True
alive          False
alone          False
dtype: bool
```

```
df.isnull().sum()
```

```
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
```

```
df['age'].fillna(df['age'].mean(),inplace=True)
```

```
df['embarked']=df['embarked'].fillna(df['embarked'].mode()[0])
```

```
df['deck']=df['deck'].fillna(df['deck'].mode()[0])
```

```
df['embark_town']=df['embark_town'].fillna(df['embark_town'].mode()[0])
```

```
df.isnull().any()
```

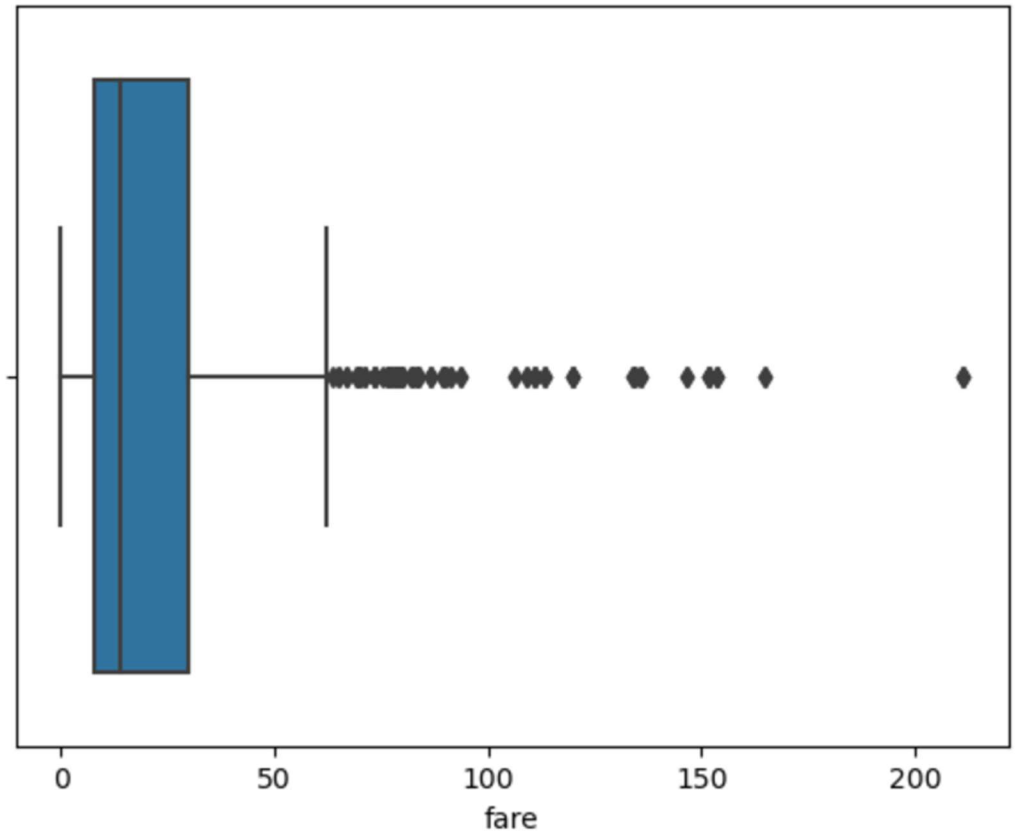
```
survived       False
pclass         False
sex            False
age            False
sibsp          False
parch          False
fare           False
embarked       False
class          False
who            False
adult_male     False
deck           False
embark_town    False
alive          False
alone          False
dtype: bool
```

6. Find the outliers and replace the outliers

```
sns.boxplot(df.fare)
```

```
C:\Users\Sriya\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass the following variable as a keyword a
rg: x. From version 0.12, the only valid positional argument will be `data`, and passing other arguments without an explicit ke
yword will result in an error or misinterpretation.
  warnings.warn(
```

```
<AxesSubplot:xlabel='fare'>
```

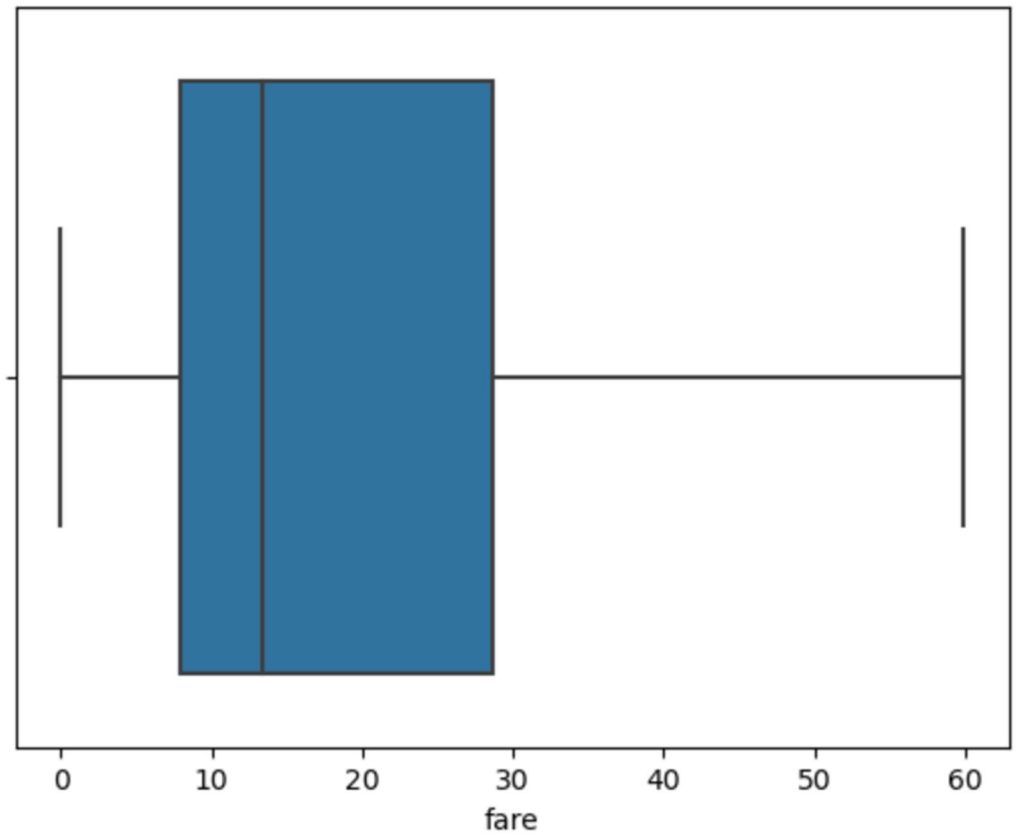



```
Q1 = df['fare'].quantile(0.25)
Q3 = df['fare'].quantile(0.75)
IQR = Q3 - Q1
whisker_width = 1.5
lower_whisker = Q1 -(whisker_width*IQR)
upper_whisker = Q3 + (whisker_width*IQR)
df['fare']=np.where(df['fare']>upper_whisker,upper_whisker,np.where(df['fare']<lower_whisker,lower_whisker,df['fare']))
```

```
sns.boxplot(df.fare)
```

```
C:\Users\Sriya\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass the following variable as a keyword a
rg: x. From version 0.12, the only valid positional argument will be `data`, and passing other arguments without an explicit ke
yword will result in an error or misinterpretation.
  warnings.warn(
```

```
<AxesSubplot:xlabel='fare'>
```



7. Check for Categorical columns and perform encoding.

```
df.head()
```

	survived	pclass	sex	age	sibsp	parch	fare	embarked	class	who	adult_male	deck	embark_town	alive	alone
0	0	3	male	22.0	1	0	7.25000	S	Third	man	True	C	Southampton	no	False
1	1	1	female	38.0	1	0	59.93755	C	First	woman	False	C	Cherbourg	yes	False
2	1	3	female	26.0	0	0	7.92500	S	Third	woman	False	C	Southampton	yes	True
3	1	1	female	35.0	1	0	53.10000	S	First	woman	False	C	Southampton	yes	False
4	0	3	male	35.0	0	0	8.05000	S	Third	man	True	C	Southampton	no	True

```
from sklearn.preprocessing import LabelEncoder
```

```
le=LabelEncoder()
```

```
df.sex=le.fit_transform(df.sex)
```

```
df.head()
```

	survived	pclass	sex	age	sibsp	parch	fare	embarked	class	who	adult_male	deck	embark_town	alive	alone
0	0	3	1	22.0	1	0	7.25000	S	Third	man	True	C	Southampton	no	False
1	1	1	0	38.0	1	0	59.93755	C	First	woman	False	C	Cherbourg	yes	False
2	1	3	0	26.0	0	0	7.92500	S	Third	woman	False	C	Southampton	yes	True
3	1	1	0	35.0	1	0	53.10000	S	First	woman	False	C	Southampton	yes	False
4	0	3	1	35.0	0	0	8.05000	S	Third	man	True	C	Southampton	no	True

```
df.embarked=le.fit_transform(df.embarked)
df.head()
```

	survived	pclass	sex	age	sibsp	parch	fare	embarked	class	who	adult_male	deck	embark_town	alive	alone
0	0	3	1	22.0	1	0	7.25000	2	Third	man	True	C	Southampton	no	False
1	1	1	0	38.0	1	0	59.93755	0	First	woman	False	C	Cherbourg	yes	False
2	1	3	0	26.0	0	0	7.92500	2	Third	woman	False	C	Southampton	yes	True
3	1	1	0	35.0	1	0	53.10000	2	First	woman	False	C	Southampton	yes	False
4	0	3	1	35.0	0	0	8.05000	2	Third	man	True	C	Southampton	no	True

```
df.who=le.fit_transform(df.who)
df.head()
```

	survived	pclass	sex	age	sibsp	parch	fare	embarked	class	who	adult_male	deck	embark_town	alive	alone
0	0	3	1	22.0	1	0	7.25000	2	Third	1	True	C	Southampton	no	False
1	1	1	0	38.0	1	0	59.93755	0	First	2	False	C	Cherbourg	yes	False
2	1	3	0	26.0	0	0	7.92500	2	Third	2	False	C	Southampton	yes	True
3	1	1	0	35.0	1	0	53.10000	2	First	2	False	C	Southampton	yes	False
4	0	3	1	35.0	0	0	8.05000	2	Third	1	True	C	Southampton	no	True

```
df['class']=le.fit_transform(df['class'])
df.head()
```

	survived	pclass	sex	age	sibsp	parch	fare	embarked	class	who	adult_male	deck	embark_town	alive	alone
0	0	3	1	22.0	1	0	7.25000	2	2	1	True	C	Southampton	no	False
1	1	1	0	38.0	1	0	59.93755	0	0	2	False	C	Cherbourg	yes	False
2	1	3	0	26.0	0	0	7.92500	2	2	2	False	C	Southampton	yes	True
3	1	1	0	35.0	1	0	53.10000	2	0	2	False	C	Southampton	yes	False
4	0	3	1	35.0	0	0	8.05000	2	2	1	True	C	Southampton	no	True

```
df.deck=le.fit_transform(df.deck)
df.head()
```

	survived	pclass	sex	age	sibsp	parch	fare	embarked	class	who	adult_male	deck	embark_town	alive	alone
0	0	3	1	22.0	1	0	7.25000	2	2	1	True	2	Southampton	no	False
1	1	1	0	38.0	1	0	59.93755	0	0	2	False	2	Cherbourg	yes	False
2	1	3	0	26.0	0	0	7.92500	2	2	2	False	2	Southampton	yes	True
3	1	1	0	35.0	1	0	53.10000	2	0	2	False	2	Southampton	yes	False
4	0	3	1	35.0	0	0	8.05000	2	2	1	True	2	Southampton	no	True

	survived	pclass	sex	age	sibsp	parch	fare	embarked	class	who	adult_male	deck	embark_town	alive	alone
0	0	3	1	22.0	1	0	7.25000	2	2	1	1	2	2	no	False
1	1	1	0	38.0	1	0	59.93755	0	0	2	0	2	0	yes	False
2	1	3	0	26.0	0	0	7.92500	2	2	2	0	2	2	yes	True
3	1	1	0	35.0	1	0	53.10000	2	0	2	0	2	2	yes	False
4	0	3	1	35.0	0	0	8.05000	2	2	1	1	2	2	no	True

```
df.alone=le.fit_transform(df.alone)
df.head()
```

	survived	pclass	sex	age	sibsp	parch	fare	embarked	class	who	adult_male	deck	embark_town	alive	alone
0	0	3	1	22.0	1	0	7.25000	2	2	1	1	2	2	no	0
1	1	1	0	38.0	1	0	59.93755	0	0	2	0	2	0	yes	0
2	1	3	0	26.0	0	0	7.92500	2	2	2	0	2	2	yes	1
3	1	1	0	35.0	1	0	53.10000	2	0	2	0	2	2	yes	0
4	0	3	1	35.0	0	0	8.05000	2	2	1	1	2	2	no	1

```
df.alive=le.fit_transform(df.alive)
df.head()
```

	survived	pclass	sex	age	sibsp	parch	fare	embarked	class	who	adult_male	deck	embark_town	alive	alone
0	0	3	1	22.0	1	0	7.2500	2	2	1	1	2	2	0	0
1	1	1	0	38.0	1	0	65.6344	0	0	2	0	2	0	1	0
2	1	3	0	26.0	0	0	7.9250	2	2	2	0	2	2	1	1
3	1	1	0	35.0	1	0	53.1000	2	0	2	0	2	2	1	0
4	0	3	1	35.0	0	0	8.0500	2	2	1	1	2	2	0	1

8. Split the data into dependent and independent variables.

```
y=df['alive']
y
```

```
0      0
1      1
2      1
3      1
4      0
      ..
886    0
887    1
888    0
889    1
890    0
Name: alive, Length: 891, dtype: int32
```

```
x=df.drop(columns=['alive'],axis=1)
x
```

	survived	pclass	sex	age	sibsp	parch	fare	embarked	class	who	adult_male	deck	embark_town	alone
0	0	3	1	22.000000	1	0	7.25000	2	2	1	1	2	2	0
1	1	1	0	38.000000	1	0	59.93755	0	0	2	0	2	0	0
2	1	3	0	26.000000	0	0	7.92500	2	2	2	0	2	2	1
3	1	1	0	35.000000	1	0	53.10000	2	0	2	0	2	2	0
4	0	3	1	35.000000	0	0	8.05000	2	2	1	1	2	2	1
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
886	0	2	1	27.000000	0	0	13.00000	2	1	1	1	2	2	1
887	1	1	0	19.000000	0	0	30.00000	2	0	2	0	1	2	1
888	0	3	0	29.699118	1	2	23.45000	2	2	2	0	2	2	0
889	1	1	1	26.000000	0	0	30.00000	0	0	1	1	2	0	1
890	0	3	1	32.000000	0	0	7.75000	1	2	1	1	2	1	1

849 rows × 14 columns

9. Scale the independent variables

```
name=x.columns
name
```

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

```
from sklearn.preprocessing import MinMaxScaler
scale=MinMaxScaler()
```

```
x_scaled=scale.fit_transform(x)
x_scaled
```

```
array([[0.        , 1.        , 1.        , ..., 0.33333333, 1.        ,
        0.        ],
       [1.        , 0.        , 0.        , ..., 0.33333333, 0.        ,
        0.        ],
       [1.        , 1.        , 0.        , ..., 0.33333333, 1.        ,
        1.        ],
       ...,
       [0.        , 1.        , 0.        , ..., 0.33333333, 1.        ,
        0.        ],
       [1.        , 0.        , 1.        , ..., 0.33333333, 0.        ,
        1.        ],
       [0.        , 1.        , 1.        , ..., 0.33333333, 0.5       ,
        1.        ]])
```

```
x=pd.DataFrame(x_scaled,columns=name)
x
```

	survived	pclass	sex	age	sibsp	parch	fare	embarked	class	who	adult_male	deck	embark_town	alone
0	0.0	1.0	1.0	0.334159	0.125	0.000000	0.120959	1.0	1.0	0.5	1.0	0.333333	1.0	0.0
1	1.0	0.0	0.0	0.581914	0.125	0.000000	1.000000	0.0	0.0	1.0	0.0	0.333333	0.0	0.0
2	1.0	1.0	0.0	0.396098	0.000	0.000000	0.132221	1.0	1.0	1.0	0.0	0.333333	1.0	1.0
3	1.0	0.0	0.0	0.535460	0.125	0.000000	0.885922	1.0	0.0	1.0	0.0	0.333333	1.0	0.0
4	0.0	1.0	1.0	0.535460	0.000	0.000000	0.134306	1.0	1.0	0.5	1.0	0.333333	1.0	1.0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
844	0.0	0.5	1.0	0.411583	0.000	0.000000	0.216892	1.0	0.5	0.5	1.0	0.333333	1.0	1.0
845	1.0	0.0	0.0	0.287705	0.000	0.000000	0.500521	1.0	0.0	1.0	0.0	0.166667	1.0	1.0
846	0.0	1.0	0.0	0.453377	0.125	0.333333	0.391241	1.0	1.0	1.0	0.0	0.333333	1.0	0.0
847	1.0	0.0	1.0	0.396098	0.000	0.000000	0.500521	0.0	0.0	0.5	1.0	0.333333	0.0	1.0
848	0.0	1.0	1.0	0.489006	0.000	0.000000	0.129301	0.5	1.0	0.5	1.0	0.333333	0.5	1.0

849 rows × 14 columns

10. Split the data into training and testing

```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y, test_size=0.2, random_state=0)
```

```
x_train.head()
```

	survived	pclass	sex	age	sibsp	parch	fare	embarked	class	who	adult_male	deck	embark_town	alone
140	0.0	1.0	0.0	0.367921	0.000	0.333333	0.232284	0.0	1.0	1.0	0.0	0.333333	0.0	0.0
439	0.0	0.5	1.0	0.384267	0.000	0.000000	0.159977	1.0	0.5	0.5	1.0	0.333333	1.0	1.0
817	0.0	0.5	1.0	0.384267	0.125	0.166667	0.563793	0.0	0.5	0.5	1.0	0.333333	0.0	0.0
378	0.0	1.0	1.0	0.246042	0.000	0.000000	0.061134	0.0	1.0	0.5	1.0	0.333333	0.0	1.0
491	0.0	1.0	1.0	0.258608	0.000	0.000000	0.110460	1.0	1.0	0.5	1.0	0.333333	1.0	1.0

```
x_test.head()
```

	survived	pclass	sex	age	sibsp	parch	fare	embarked	class	who	adult_male	deck	embark_town	alone
495	0.0	1.0	1.0	0.367921	0.000	0.000000	0.220285	0.0	1.0	0.5	1.0	0.333333	0.0	1.0
648	0.0	1.0	1.0	0.367921	0.000	0.000000	0.115031	1.0	1.0	0.5	1.0	0.333333	1.0	1.0
278	0.0	1.0	1.0	0.082684	0.500	0.166667	0.443746	0.5	1.0	0.0	0.0	0.333333	0.5	0.0
31	1.0	0.0	0.0	0.367921	0.125	0.000000	1.000000	0.0	0.0	1.0	0.0	0.166667	0.0	0.0
255	1.0	1.0	0.0	0.359135	0.000	0.333333	0.232284	0.0	1.0	1.0	0.0	0.333333	0.0	0.0

```
y_test.head()
```

```
495    0
648    0
278    0
31     1
255    1
Name: alive, dtype: int32
```

```
y_train.head()
```

```
140    0
439    0
817    0
378    0
491    0
Name: alive, dtype: int32
```