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

df=sns.load_dataset("titanic")
df.head()
```

```
   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
```

```
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.shape
```

```
(891, 15)
```

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

```
survived       False
pclass         False
sex            False
age            False
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.deck.value_counts()
```

```
deck
C    59
B    47
D    33
E    32
A    15
F    13
```

```
G     4
Name: count, dtype: int64
```

```
df["deck"].fillna(df["deck"].mode()[0],inplace=True)
df.deck.value_counts()
```

```
deck
C    747
B     47
D     33
E     32
A     15
F     13
G      4
Name: count, dtype: int64
```

```
df["embarked"].fillna(df["embarked"].mode()[0],inplace=True)
df.embarked.value_counts()
```

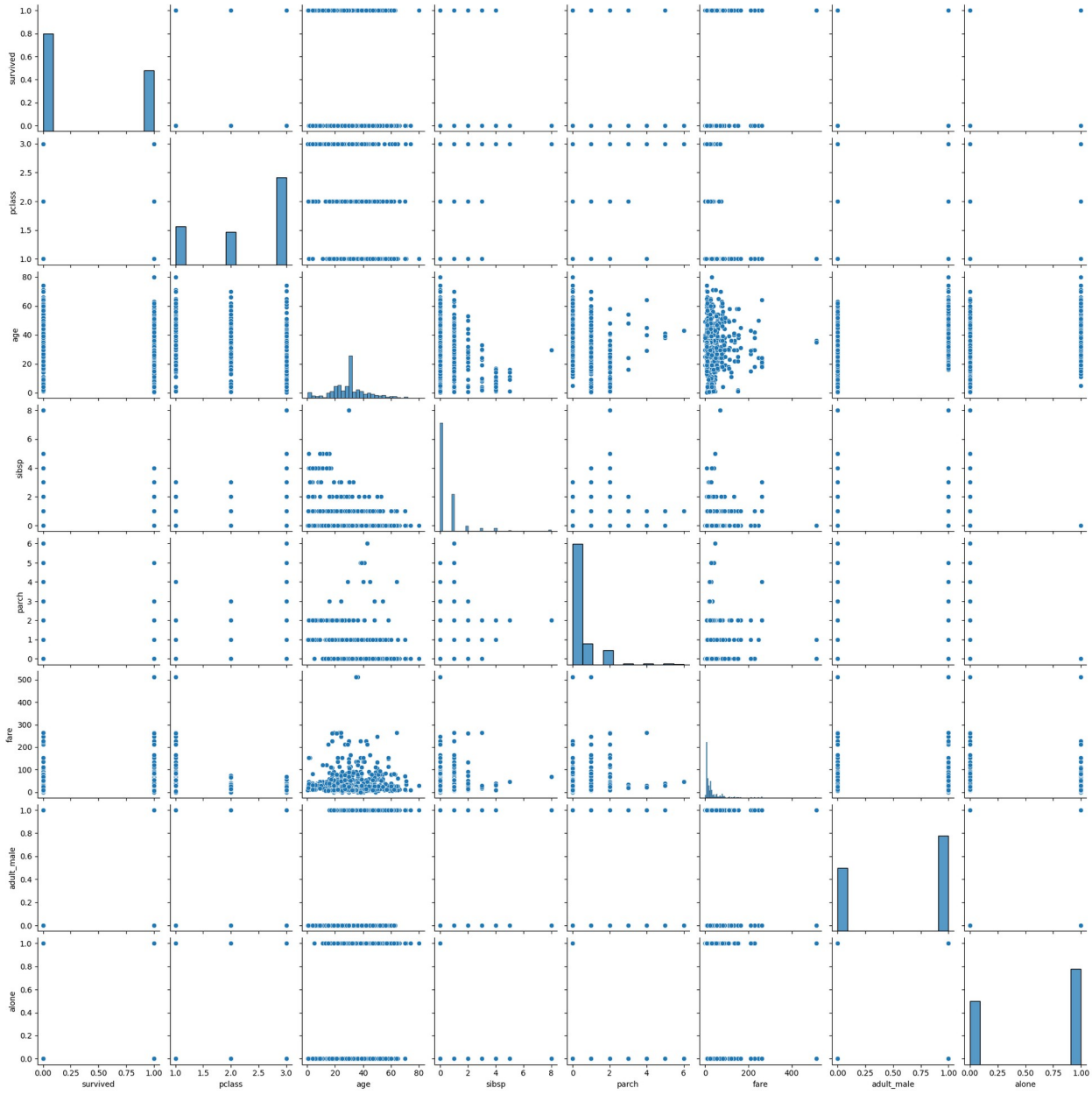
```
embarked
S    646
C    168
Q     77
Name: count, dtype: int64
```

```
df["embark_town"].fillna(df["embark_town"].mode()[0],inplace=True)
df["embark_town"].value_counts()
```

```
embark_town
Southampton    646
Cherbourg      168
Queenstown      77
Name: count, dtype: int64
```

```
sns.pairplot(data=df)
```

```
<__array_function__ internals>:200: RuntimeWarning: Converting input
from bool to <class 'numpy.uint8'> for compatibility.
<__array_function__ internals>:200: RuntimeWarning: Converting input
from bool to <class 'numpy.uint8'> for compatibility.
C:\Users\HP\AppData\Roaming\Python\Python311\site-packages\seaborn\
axisgrid.py:118: UserWarning: The figure layout has changed to tight
  self._figure.tight_layout(*args, **kwargs)

<seaborn.axisgrid.PairGrid at 0x24e1d0cdc10>
```




```
df.head()
```

	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    C  Southampton    no  False
1  woman       False    C    Cherbourg   yes  False
2  woman       False    C  Southampton   yes   True
3  woman       False    C  Southampton   yes  False
4    man        True    C  Southampton    no   True
```

```
df.drop(['alive','class','who','embark_town'],axis=1)
```

```
     survived  pclass     sex        age  sibsp  parch     fare
embarked  \
0           0       3    male  22.000000      1      0   7.2500
S
1           1       1  female  38.000000      1      0  71.2833
C
2           1       3  female  26.000000      0      0   7.9250
S
3           1       1  female  35.000000      1      0  53.1000
S
4           0       3    male  35.000000      0      0   8.0500
S
..        ...     ...     ...        ...    ...    ...      ...       .
..
886         0       2    male  27.000000      0      0  13.0000
S
887         1       1  female  19.000000      0      0  30.0000
S
888         0       3  female  29.699118      1      2  23.4500
S
889         1       1    male  26.000000      0      0  30.0000
C
890         0       3    male  32.000000      0      0   7.7500
Q

     adult_male deck  alone
0          True    C  False
1         False    C  False
2         False    C   True
3         False    C  False
4          True    C   True
..          ...  ...    ...
886        True    C   True
887       False    B   True
888       False    C  False
889        True    C   True
890        True    C   True
```

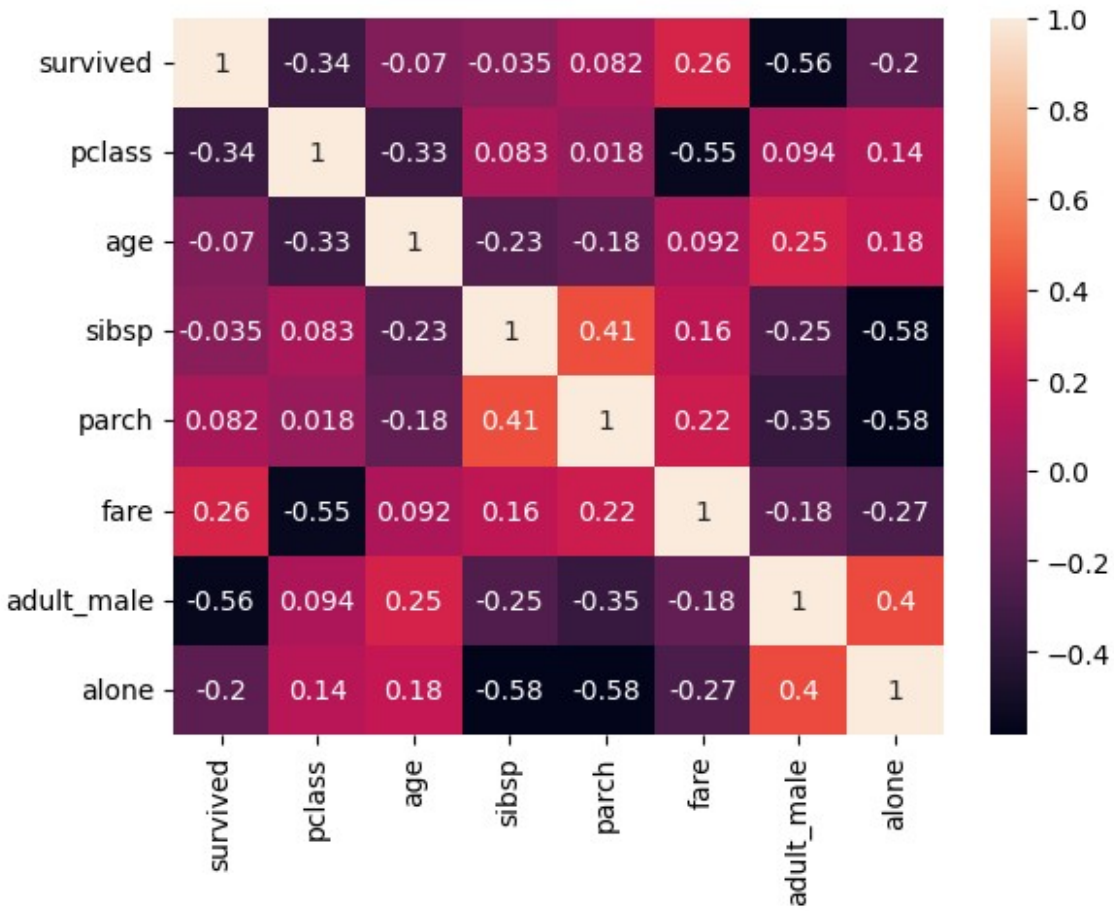
```
[891 rows x 11 columns]
```

```
corr=df.corr(numeric_only=True)
corr
```

```
             survived    pclass       age     sibsp     parch      fare
\
survived     1.000000 -0.338481 -0.069809 -0.035322  0.081629  0.257307

pclass      -0.338481  1.000000 -0.331339  0.083081  0.018443 -0.549500

age         -0.069809 -0.331339  1.000000 -0.232625 -0.179191  0.091566

sibsp       -0.035322  0.083081 -0.232625  1.000000  0.414838  0.159651

parch        0.081629  0.018443 -0.179191  0.414838  1.000000  0.216225

fare         0.257307 -0.549500  0.091566  0.159651  0.216225  1.000000

adult_male  -0.557080  0.094035  0.253236 -0.253586 -0.349943 -0.182024

alone       -0.203367  0.135207  0.179775 -0.584471 -0.583398 -0.271832


             adult_male     alone
survived      -0.557080 -0.203367
pclass         0.094035  0.135207
age            0.253236  0.179775
sibsp         -0.253586 -0.584471
parch         -0.349943 -0.583398
fare          -0.182024 -0.271832
adult_male     1.000000  0.404744
alone          0.404744  1.000000
```

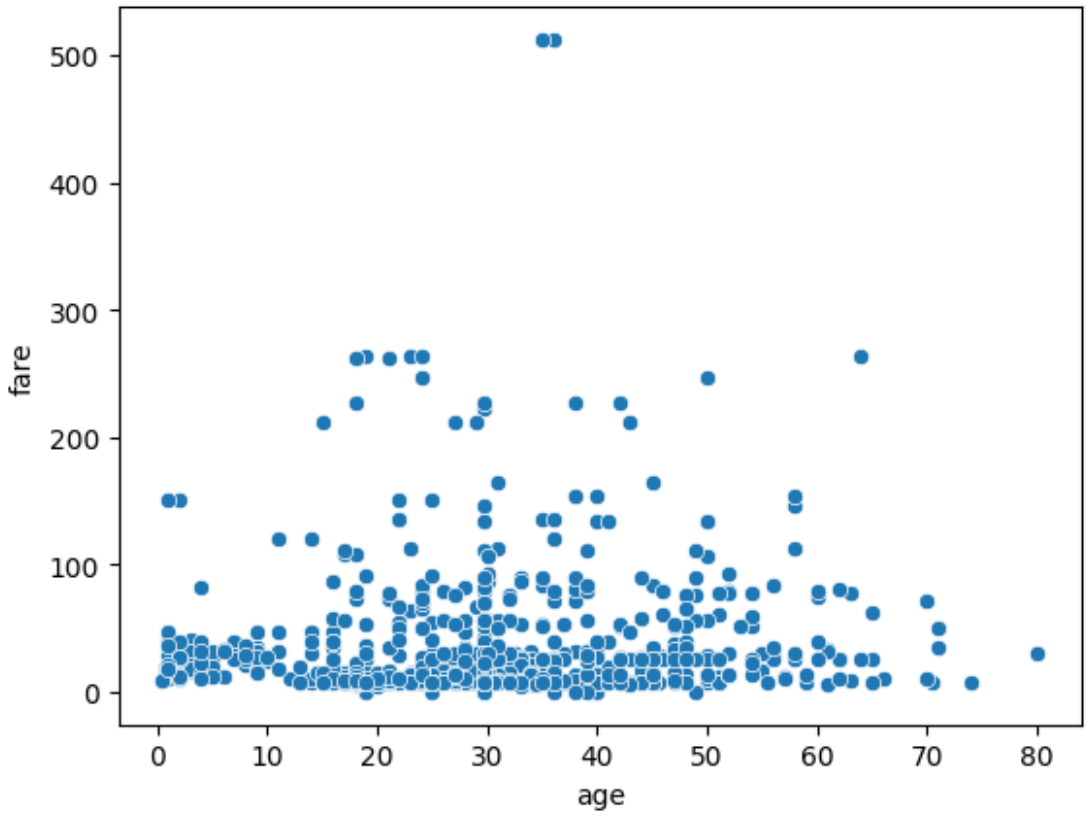
```
sns.heatmap(data=corr,annot=True,)
```

```
<Axes: >
```



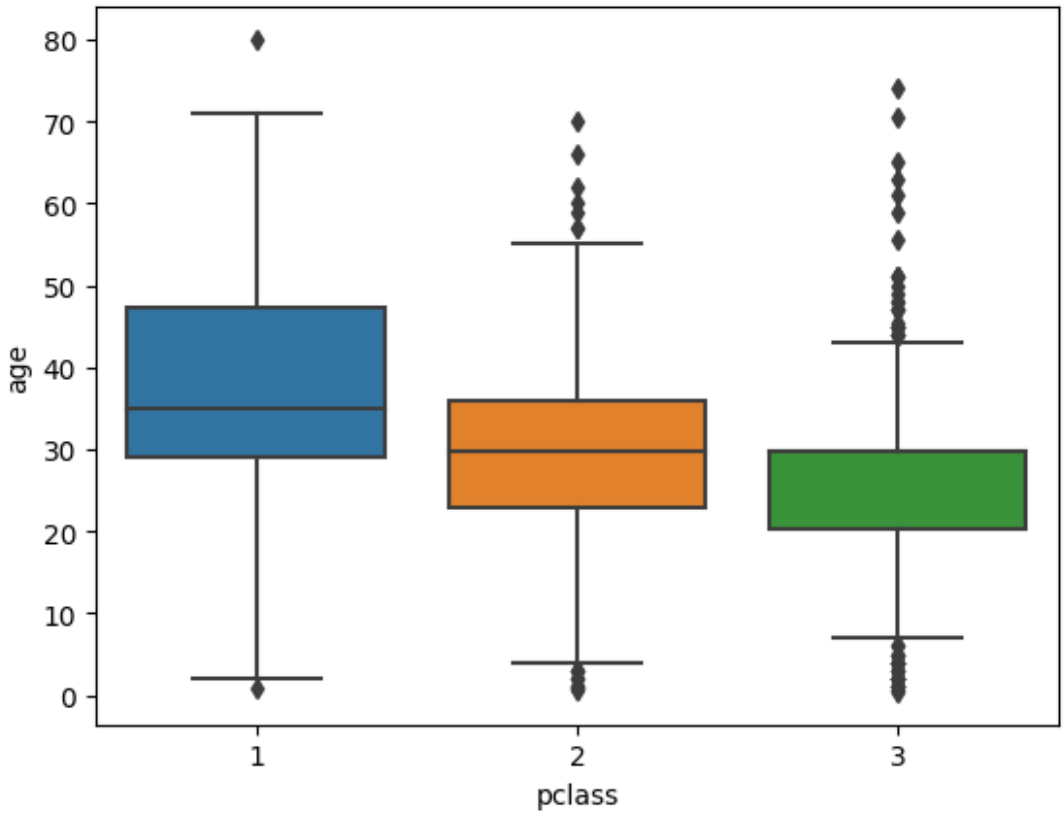

```
sns.scatterplot(x="age", y="fare",data=df)
```

```
<Axes: xlabel='age', ylabel='fare'>
```

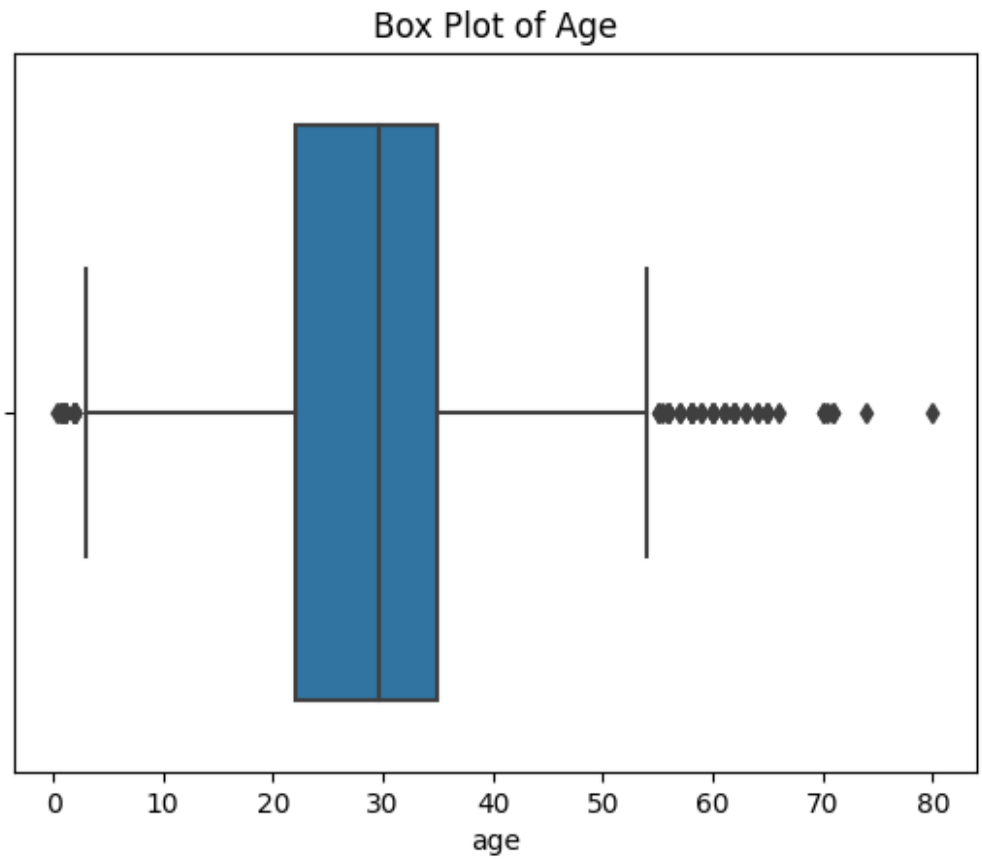


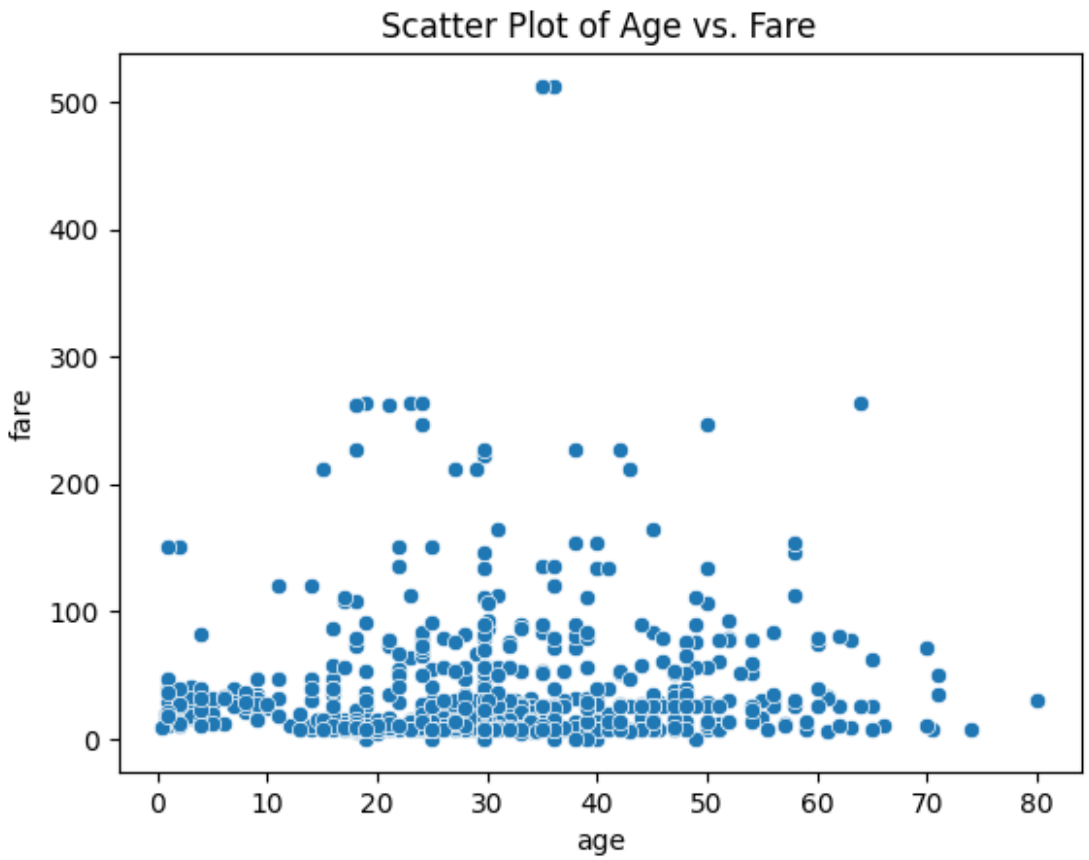

```
sns.boxplot(data=df, x='pclass', y='age')
```

```
<Axes: xlabel='pclass', ylabel='age'>
```




```
# Box plot for detecting outliers in a variable (e.g., 'Age')
sns.boxplot(x=df['age'])
plt.title('Box Plot of Age')
plt.show()
# Scatter plot for detecting outliers between 'Fare' and 'Age'
sns.scatterplot(data=df, x='age', y='fare')
plt.title('Scatter Plot of Age vs. Fare')
plt.show()
```


Box Plot of Age
0
10
20
30
40
50
60
70
80
age




```
q1 = df.age.quantile(0.25)
q3 = df.age.quantile(0.75)

IQR=q3-q1
IQR
```

```
13.0
```

```
 upper_limit = q3+1.5*IQR
 upper_limit
```

```
54.5
```

```
lower_limit = q1-1.5*IQR
lower_limit
```

```
2.5
```

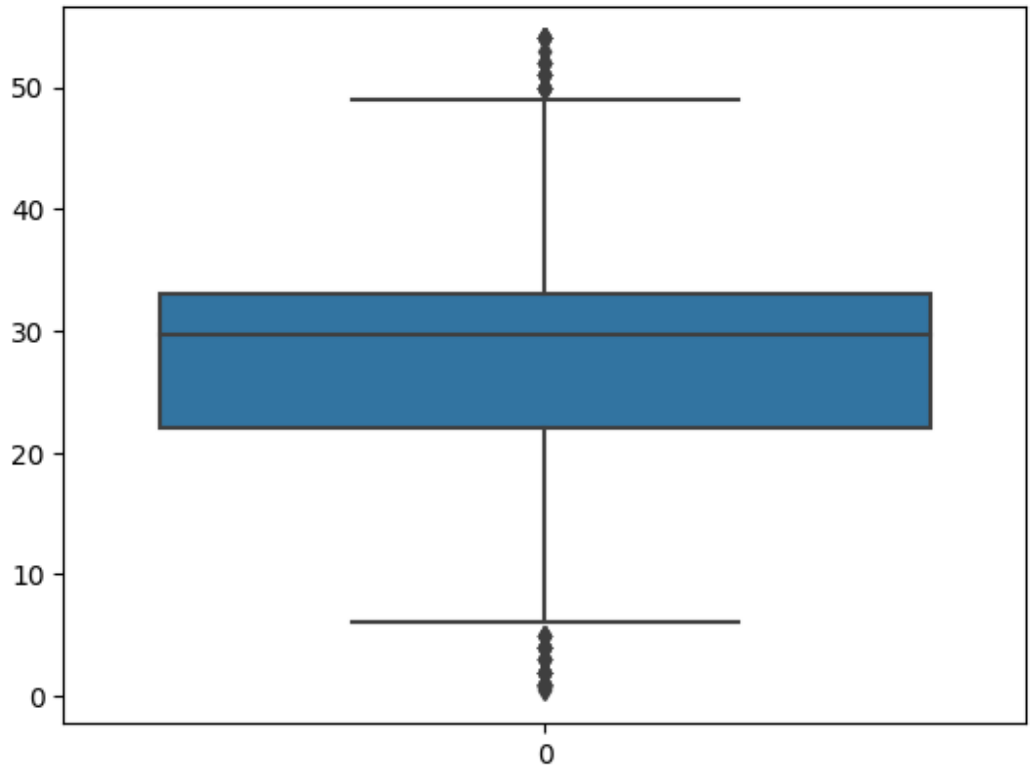
```
df.median(numeric_only=True)
```

```
survived       0.000000
pclass         3.000000
age           29.699118
sibsp          0.000000
parch          0.000000
```

```
fare          14.454200
adult_male     1.000000
alone          1.000000
dtype: float64
```

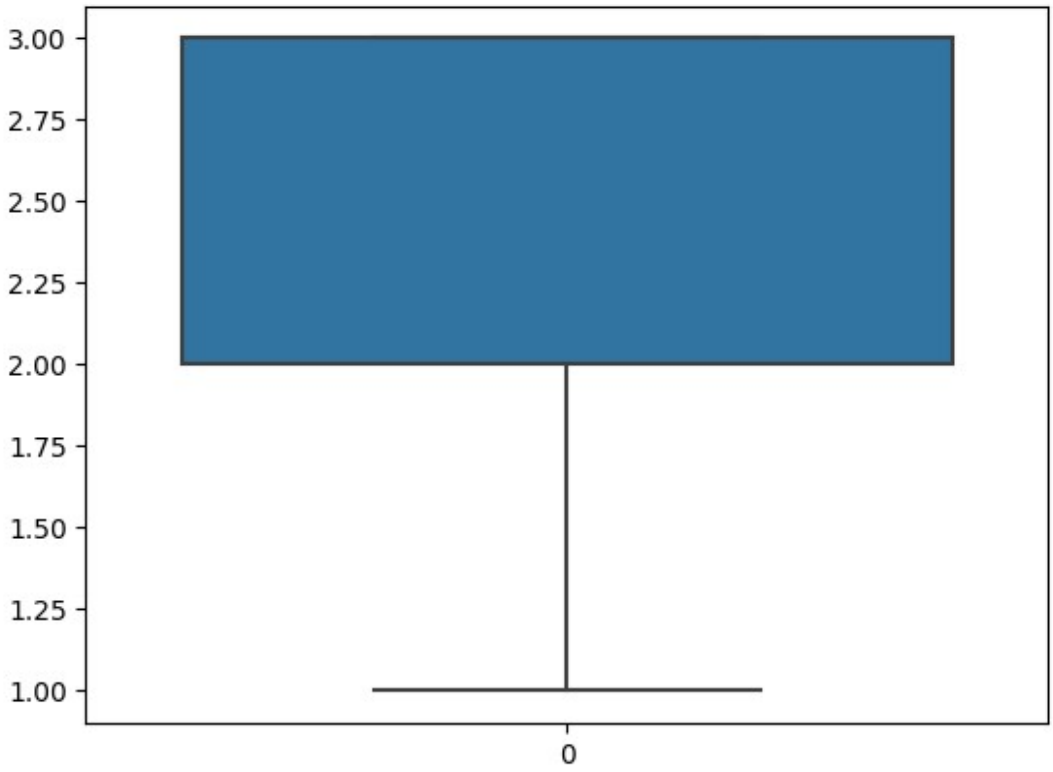
```
df['age']=np.where(df['age']>upper_limit,30,df['age'])

sns.boxplot(df.age)
```

```
<Axes: >
```




```
sns.boxplot(df.pclass)
```

```
<Axes: >
```




```
X=df.drop(columns=["embark_town","alive","alone","class","who","deck"]
,axis=1)

X.head()
```

	survived	pclass	sex	age	sibsp	parch	fare	embarked	adult_male
0	0	3	male	22.0	1	0	7.2500	S	True
1	1	1	female	38.0	1	0	71.2833	C	False
2	1	3	female	26.0	0	0	7.9250	S	False
3	1	1	female	35.0	1	0	53.1000	S	False
4	0	3	male	35.0	0	0	8.0500	S	True

```
X=df.drop(columns=["survived"],axis=1)

y=df["survived"]

from sklearn.preprocessing import LabelEncoder
le=LabelEncoder()
```

```
X["sex"]=le.fit_transform(X["sex"])
X["embarked"]=le.fit_transform(X["embarked"])
X["adult_male"]=le.fit_transform(X["adult_male"])

X.head()
```

```
   survived  pclass  sex   age  sibsp  parch     fare  embarked
adult_male
0         0       3    1  22.0      1      0   7.2500         2
1
1         1       1    0  38.0      1      0  71.2833         0
0
2         1       3    0  26.0      0      0   7.9250         2
0
3         1       1    0  35.0      1      0  53.1000         2
0
4         0       3    1  35.0      0      0   8.0500         2
1
```

```
mapping=dict(zip(le.classes_,range(len(le.classes_))))
mapping
```

```
{False: 0, True: 1}
```

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

X_Scaled=ms.fit_transform(X)
X_Scaled=pd.DataFrame(ms.fit_transform(X),columns=X.columns)
X_Scaled.head()
```

```
   survived  pclass  sex       age  sibsp  parch      fare
embarked  \
0       0.0     1.0  1.0  0.402762  0.125    0.0  0.014151        1.0

1       1.0     0.0  0.0  0.701381  0.125    0.0  0.139136        0.0

2       1.0     1.0  0.0  0.477417  0.000    0.0  0.015469        1.0

3       1.0     0.0  0.0  0.645390  0.125    0.0  0.103644        1.0

4       0.0     1.0  1.0  0.645390  0.000    0.0  0.015713        1.0


   adult_male
0         1.0
1         0.0
2         0.0
3         0.0
4         1.0
```

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

print(x_train.shape,x_test.shape,y_train.shape,y_test.shape)
```

```
(712, 9) (179, 9) (712,) (179,)
```

```
from sklearn.linear_model import LinearRegression
lr=LinearRegression()

lr.fit(x_train,y_train)
```

```
LinearRegression()
```

```
y_pred=lr.predict(x_test)
y_pred
```

```
array([1.08619750e-15, 9.62392253e-16, 5.77631802e-16, 1.00000000e+00,
       1.00000000e+00, 1.00000000e+00, 1.00000000e+00, 1.00000000e+00,
       1.00000000e+00, 1.00000000e+00, 8.77963058e-16, 1.00000000e+00,
       1.02621963e-15, 1.00000000e+00, 1.00000000e+00, 6.63062415e-16,
       9.13481986e-16, 8.61050706e-16, 1.00001432e-15, 1.00000000e+00,
       8.04696537e-16, 1.00000000e+00, 1.02622702e-15, 8.05173570e-16,
       7.19114240e-16, 1.00000000e+00, 9.78646191e-16, 1.00000000e+00,
       1.00000000e+00, 6.35851065e-16, 1.11323716e-15, 1.00000000e+00,
       9.49084893e-16, 1.00000000e+00, 8.99900546e-16, 1.00000000e+00,
       1.05818559e-15, 8.76819140e-16, 8.53881110e-16, 1.06072990e-15,
       1.00000000e+00, 1.05686251e-15, 9.58281490e-16, 5.57510445e-16,
       1.00000000e+00, 9.62187720e-16, 9.62187720e-16, 1.00000000e+00,
       9.54024687e-16, 1.00005480e-15, 1.00000000e+00, 1.00000000e+00,
       1.00000000e+00, 8.85655726e-16, 1.00000000e+00, 6.95365532e-16,
       8.09957887e-16, 6.40443783e-16, 7.15583995e-16, 1.00000000e+00,
       1.03540069e-15, 1.25023893e-15, 1.00000000e+00, 9.27978215e-16,
       1.00000000e+00, 1.04267810e-15, 1.00000000e+00, 9.85435820e-16,
       1.00000000e+00, 1.00000000e+00, 1.00000000e+00, 1.00000000e+00,
       8.32855536e-16, 9.62096514e-16, 9.24906640e-16, 1.00000000e+00,
       6.61901606e-16, 1.00868786e-15, 9.93780483e-16, 1.01417325e-15,
       9.58730350e-16, 1.00000000e+00, 7.30628515e-16, 8.64267839e-16,
       9.96407278e-16, 1.00000000e+00, 1.00000000e+00, 1.00000000e+00,
       1.00000000e+00, 7.45514711e-16, 8.23509348e-16, 8.85655726e-16,
       1.00000000e+00, 1.00000000e+00, 6.85922853e-16, 1.09047334e-15,
       1.00000000e+00, 7.36160996e-16, 8.09957887e-16, 1.00000000e+00,
       6.17380264e-16, 9.95843117e-16, 8.85801111e-16, 9.42663063e-16,
       8.32952200e-16, 1.00000000e+00, 1.00000000e+00, 1.44699233e-15,
       7.88402851e-16, 1.00000000e+00, 1.09312060e-15, 1.00000000e+00,
       1.00000000e+00, 7.75244161e-16, 1.00000000e+00, 1.00000000e+00,
       1.00000000e+00, 1.00000000e+00, 1.03587109e-15, 1.00000000e+00,
       1.00000000e+00, 1.02580069e-15, 8.92355354e-16, 8.82886495e-16,
       9.85350728e-16, 9.35986080e-16, 1.09047583e-15, 1.00000000e+00,
       8.95183625e-16, 1.04014299e-15, 7.77141507e-16, 8.99870746e-16,
```

```
       9.29074673e-16, 6.47026346e-16, 8.10283199e-16, 8.99870746e-16,
       9.68430568e-16, 7.90295616e-16, 9.24311991e-16, 8.39574885e-16,
       8.75340448e-16, 1.00000000e+00, 9.62096514e-16, 7.56273646e-16,
       1.00000000e+00, 7.34004549e-16, 8.89646673e-16, 1.00000000e+00,
       1.00000000e+00, 9.61734235e-16, 8.25795703e-16, 1.00000000e+00,
       8.26558799e-16, 9.57258303e-16, 1.00000000e+00, 8.62512914e-16,
       1.13072141e-15, 1.00000000e+00, 8.39367276e-16, 1.00000000e+00,
       9.62187720e-16, 1.00000000e+00, 1.00000000e+00, 1.00000000e+00,
       8.85120418e-16, 9.28719787e-16, 1.11505312e-15, 9.07174212e-16,
       9.21421290e-16, 8.70896016e-16, 1.11446413e-15, 1.00000000e+00,
       9.62569696e-16, 9.58281490e-16, 1.00000000e+00, 9.62096514e-16,
       1.00000000e+00, 9.14292782e-16, 8.85564521e-16])
```

```
y_test
```

```
495    0
648    0
278    0
31     1
255    1
      ..
780    1
837    0
215    1
833    0
372    0
Name: survived, Length: 179, dtype: int64
```

```
survived=pd.DataFrame({"actual_survived":y_test,"Predicted
_survived":y_pred})
survived
```

```
     actual_survived  Predicted _survived
495                0         1.086197e-15
648                0         9.623923e-16
278                0         5.776318e-16
31                 1         1.000000e+00
255                1         1.000000e+00
..               ...                  ...
780                1         1.000000e+00
837                0         9.620965e-16
215                1         1.000000e+00
833                0         9.142928e-16
372                0         8.855645e-16

[179 rows x 2 columns]
```

```
from sklearn import metrics

print(metrics.r2_score(y_test,y_pred))
```

```
1.0
print(metrics.mean_squared_error(y_test,y_pred))
1.4836608224492802e-30
print(np.sqrt(metrics.mean_squared_error(y_test,y_pred)))
1.218056165556121e-15
```