titanic-traindata

June 15, 2024

[1]:

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

[133]:

```
train_df=pd.read_csv("C:\\Users\\pandeysunny2315\\Downloads\\titanic\\train.
 ↪csv")
```

[134]:

```
train_df.head()
```

[134]:

```
   PassengerId  Survived  Pclass  \
0            1         0       3
1            2         1       1
2            3         1       3
3            4         1       1
4            5         0       3

                                                Name     Sex   Age  SibSp  \
0                            Braund, Mr. Owen Harris    male  22.0      1
1  Cumings, Mrs. John Bradley (Florence Briggs Th…  female  38.0      1
2                             Heikkinen, Miss. Laina  female  26.0      0
3       Futrelle, Mrs. Jacques Heath (Lily May Peel)  female  35.0      1
4                           Allen, Mr. William Henry    male  35.0      0

   Parch            Ticket     Fare Cabin Embarked
0      0         A/5 21171   7.2500   NaN        S
1      0          PC 17599  71.2833   C85        C
2      0  STON/O2. 3101282   7.9250   NaN        S
3      0            113803  53.1000  C123        S
4      0            373450   8.0500   NaN        S
```

[135]:

```
train_df.drop(columns=("Cabin"), inplace=True)
```

[136]:

```
train_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 891 entries, 0 to 890
Data columns (total 11 columns):
 #   Column       Non-Null Count  Dtype
```

```
---  ------       --------------  -----
 0   PassengerId  891 non-null    int64
 1   Survived     891 non-null    int64
 2   Pclass       891 non-null    int64
 3   Name         891 non-null    object
 4   Sex          891 non-null    object
 5   Age          714 non-null    float64
 6   SibSp        891 non-null    int64
 7   Parch        891 non-null    int64
 8   Ticket       891 non-null    object
 9   Fare         891 non-null    float64
 10  Embarked     889 non-null    object
dtypes: float64(2), int64(5), object(4)
memory usage: 76.7+ KB
```

[137]:

```python
train_df["Age"].fillna(train_df["Age"].mean(), inplace=True)
```

```
C:\Users\pandeysunny2315\AppData\Local\Temp\ipykernel_7880\1036321305.py:1:
FutureWarning: A value is trying to be set on a copy of a DataFrame or Series
through chained assignment using an inplace method.
The behavior will change in pandas 3.0. This inplace method will never work
because the intermediate object on which we are setting values always behaves as
a copy.

For example, when doing 'df[col].method(value, inplace=True)', try using
'df.method({col: value}, inplace=True)' or df[col] = df[col].method(value)
instead, to perform the operation inplace on the original object.


  train_df["Age"].fillna(train_df["Age"].mean(), inplace=True)
```

[138]:

```python
train_df["Embarked"].fillna('S', inplace=True)
```

```
C:\Users\pandeysunny2315\AppData\Local\Temp\ipykernel_7880\4256062730.py:1:
FutureWarning: A value is trying to be set on a copy of a DataFrame or Series
through chained assignment using an inplace method.
The behavior will change in pandas 3.0. This inplace method will never work
because the intermediate object on which we are setting values always behaves as
a copy.

For example, when doing 'df[col].method(value, inplace=True)', try using
'df.method({col: value}, inplace=True)' or df[col] = df[col].method(value)
instead, to perform the operation inplace on the original object.


  train_df["Embarked"].fillna('S', inplace=True)
```

[139]:

```python
train_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 891 entries, 0 to 890
Data columns (total 11 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   PassengerId  891 non-null    int64
 1   Survived     891 non-null    int64
 2   Pclass       891 non-null    int64
 3   Name         891 non-null    object
 4   Sex          891 non-null    object
 5   Age          891 non-null    float64
 6   SibSp        891 non-null    int64
 7   Parch        891 non-null    int64
 8   Ticket       891 non-null    object
 9   Fare         891 non-null    float64
 10  Embarked     891 non-null    object
dtypes: float64(2), int64(5), object(4)
memory usage: 76.7+ KB
```

[140]:

```
train_df['Survived']=train_df['Survived'].astype('category')
train_df['Age']=train_df['Age'].astype('int')
train_df['Embarked']=train_df['Embarked'].astype('category')
train_df['Sex']=train_df['Sex'].astype('category')
train_df['Pclass']=train_df['Pclass'].astype('category')
```

[141]:

```
train_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 891 entries, 0 to 890
Data columns (total 11 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   PassengerId  891 non-null    int64
 1   Survived     891 non-null    category
 2   Pclass       891 non-null    category
 3   Name         891 non-null    object
 4   Sex          891 non-null    category
 5   Age          891 non-null    int32
 6   SibSp        891 non-null    int64
 7   Parch        891 non-null    int64
 8   Ticket       891 non-null    object
 9   Fare         891 non-null    float64
 10  Embarked     891 non-null    category
dtypes: category(4), float64(1), int32(1), int64(3), object(2)
memory usage: 49.4+ KB
```

[142]:

```
death_percent=round((train_df['Survived'].value_counts().values[0]/891)*100)
death_percent
```

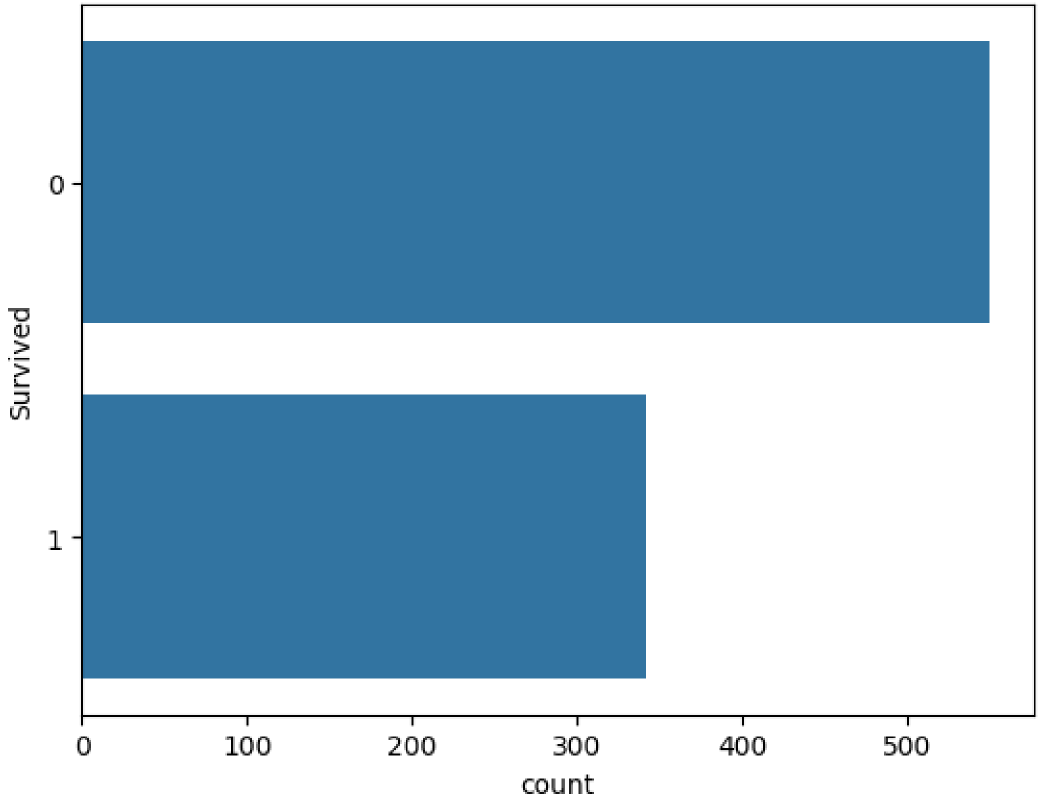
[142]:

```
62
```

[143]:

```
sns.countplot(train_df['Survived'])
death_percent=round((train_df['Survived'].value_counts().values[0]/891)*100)
print('Out of 891 {} people deid in the accident'.format(death_percent))
```

```
Out of 891 62 people deid in the accident
```

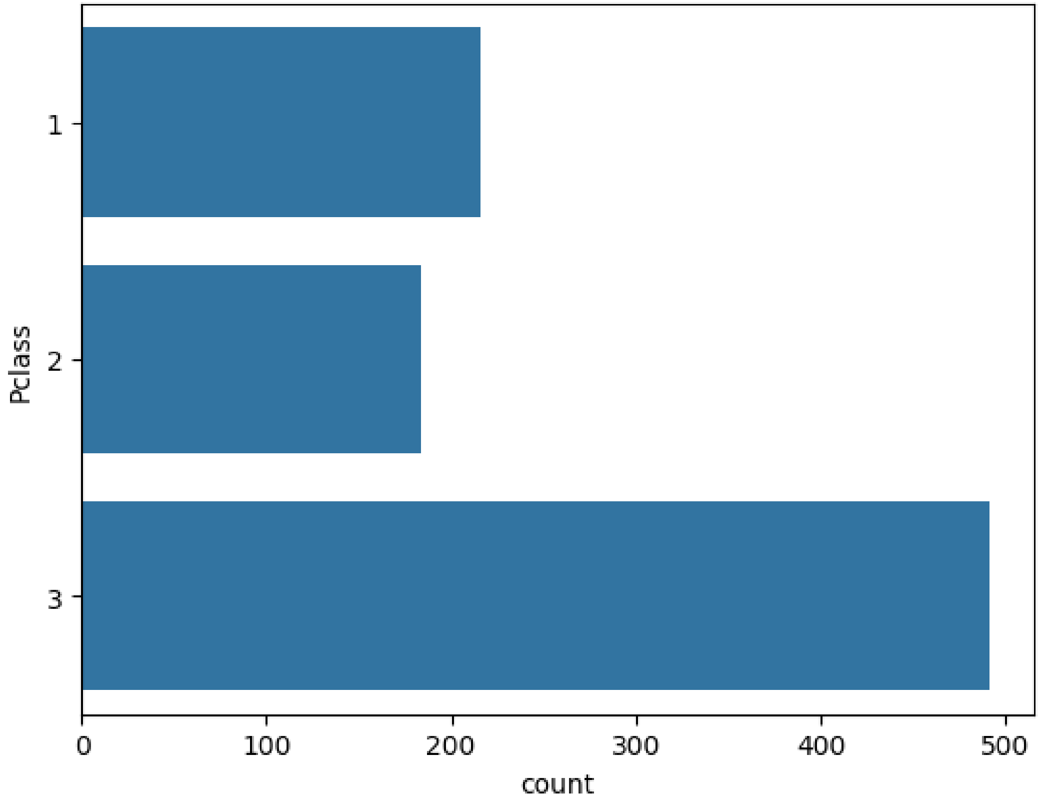


[144]:

```
sns.countplot(train_df['Pclass'])
print((train_df['Pclass'].value_counts()))
print(((train_df['Pclass'].value_counts()/891)*100))
```

```
Pclass
3    491
1    216
2    184
Name: count, dtype: int64
Pclass
3    55.106622
```

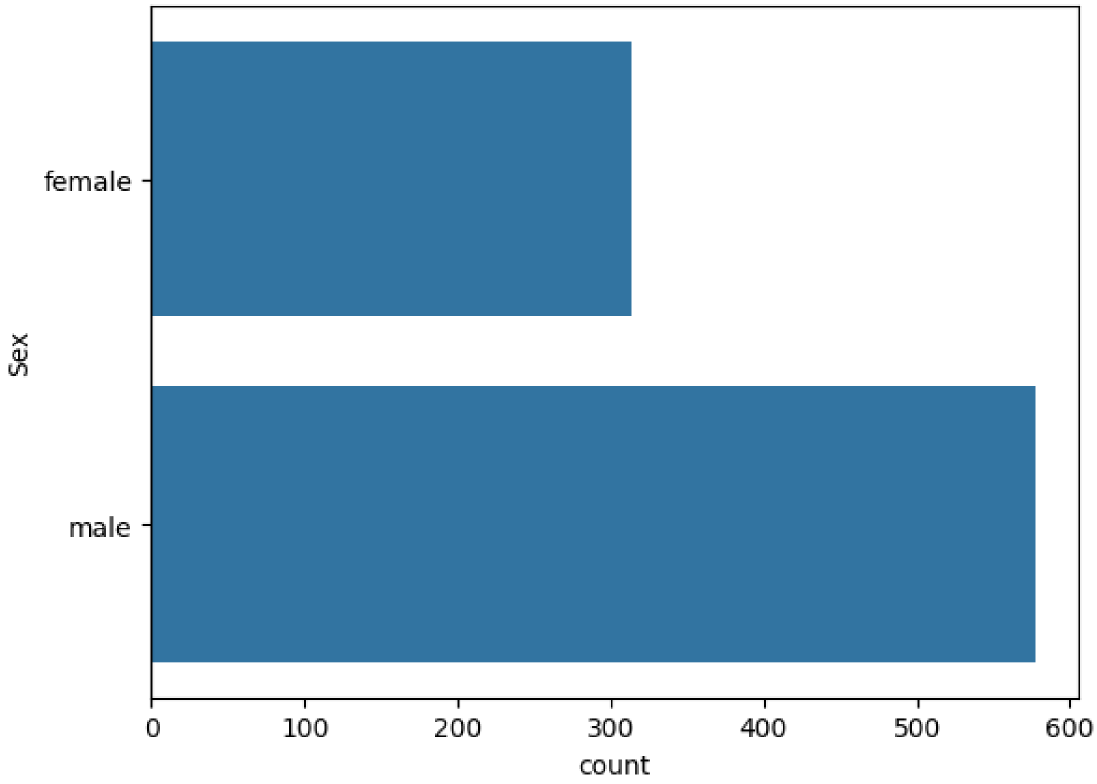
```
1    24.242424
2    20.650954
Name: count, dtype: float64
```



[58]:

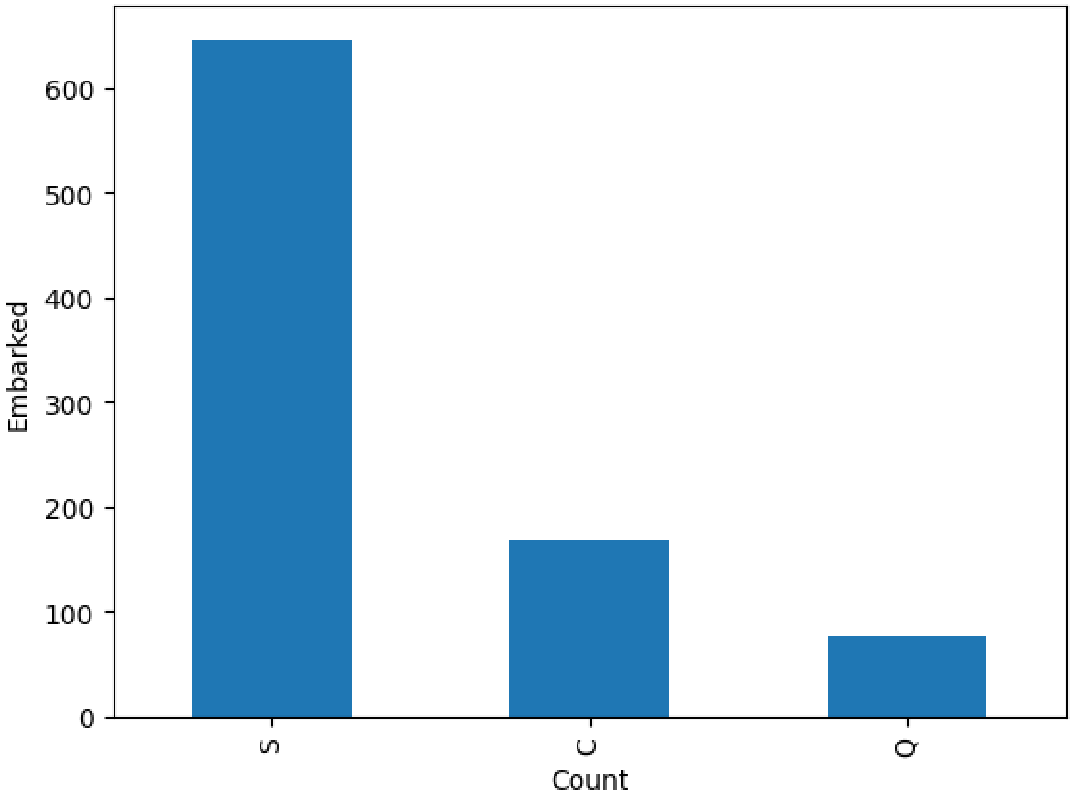
```
sns.countplot(train_df['Sex'])
print((train_df['Sex'].value_counts()))
print((train_df['Sex'].value_counts()/891)*100)
```

```
Sex
male      577
female    314
Name: count, dtype: int64
Sex
male      64.758698
female    35.241302
Name: count, dtype: float64
```



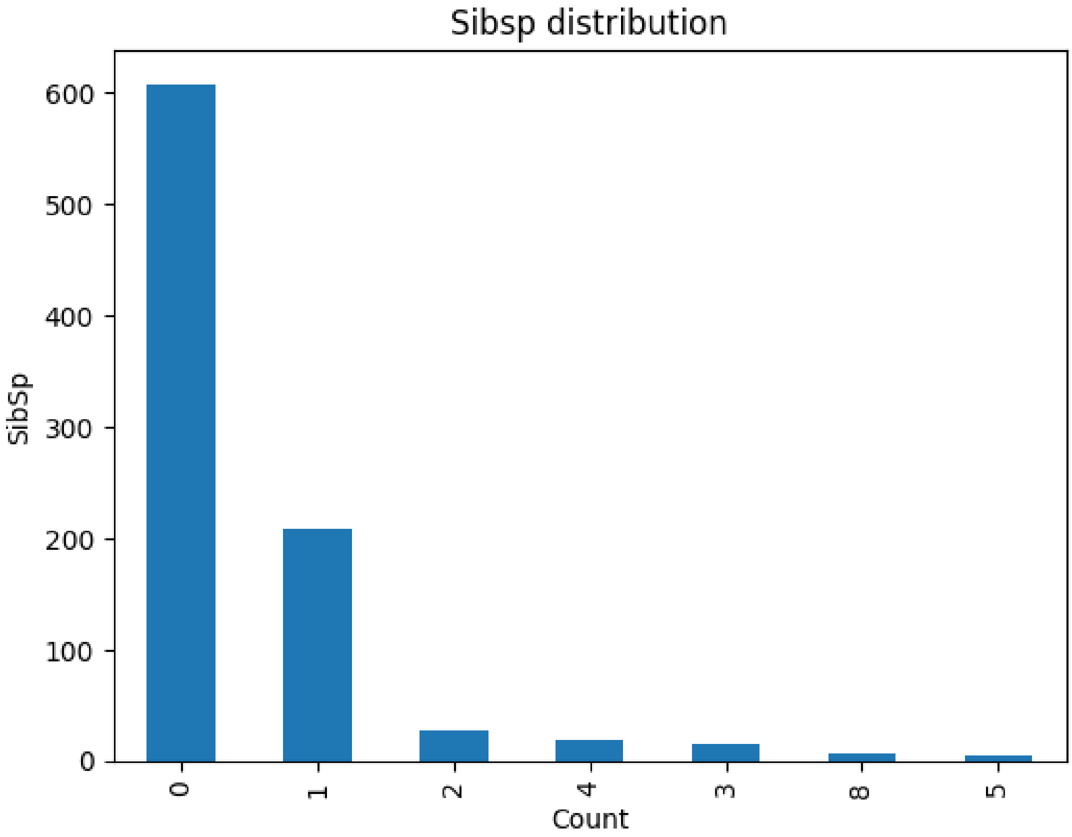
```
[79]: print((train_df['Embarked'].value_counts()))
      print((train_df['Embarked'].value_counts()/891)*100)
      Embarked_counts=train_df['Embarked'].value_counts()
      Embarked_counts.plot(kind='bar')
      plt.xlabel('Count')
      plt.ylabel('Embarked')
      plt.show()
```

```
Embarked
S    646
C    168
Q     77
Name: count, dtype: int64
Embarked
S    72.502806
C    18.855219
Q     8.641975
Name: count, dtype: float64
```



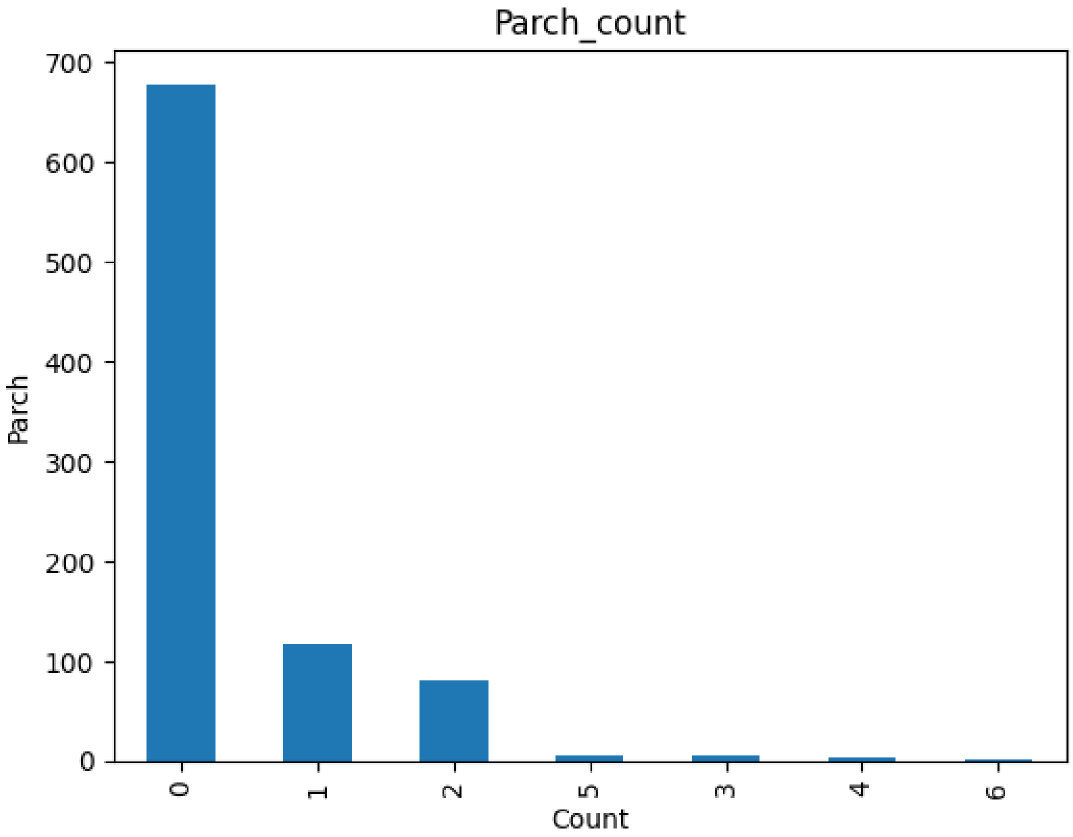
[93]:
```
print((train_df['SibSp'].value_counts()/891)*100)
SibSp_counts=train_df['SibSp'].value_counts() #.plot(kind='barh')
SibSp_counts.plot(kind='bar')
plt.xlabel('Count')
plt.ylabel('SibSp')
plt.title('Sibsp distribution')
plt.show()
```

```
SibSp
0    68.237935
1    23.456790
2     3.142536
4     2.020202
3     1.795735
8     0.785634
5     0.561167
Name: count, dtype: float64
```



[94]:
```
print((train_df['Parch'].value_counts()/891)*100)
Parch_count=(train_df['Parch'].value_counts())
Parch_count.plot(kind='bar')
plt.xlabel('Count')
plt.ylabel('Parch')
plt.title('Parch_count')
plt.show()
```

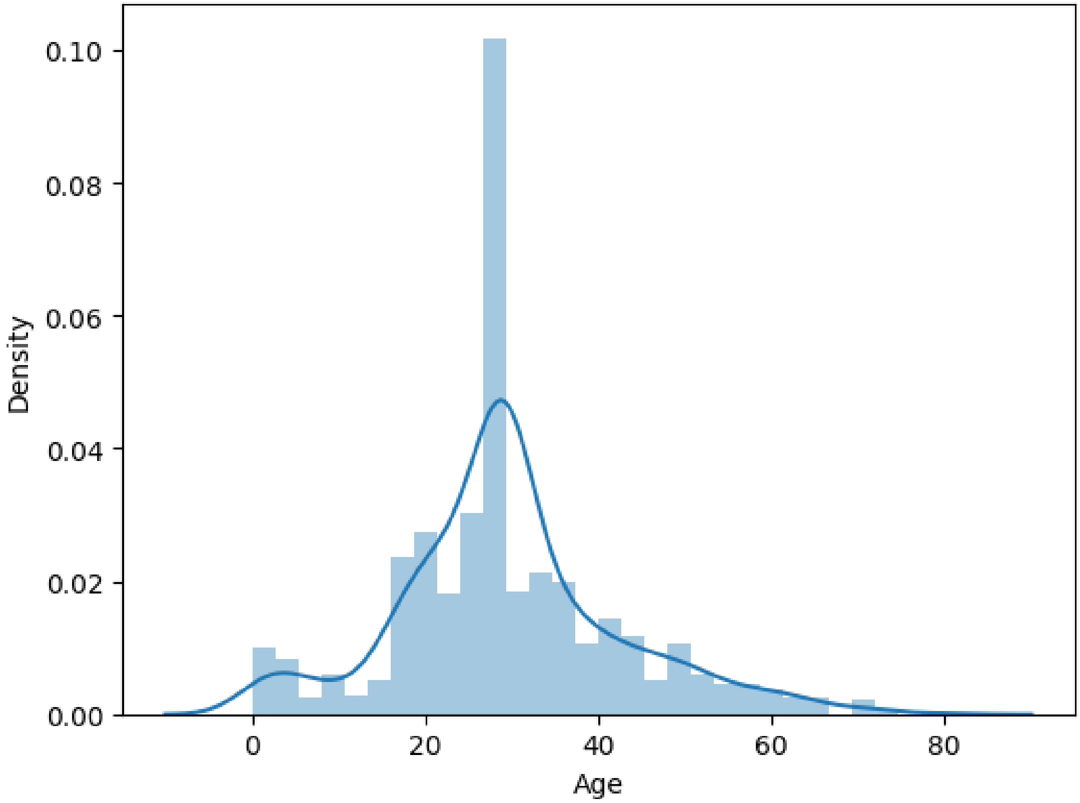
```
Parch
0    76.094276
1    13.243547
2     8.978676
5     0.561167
3     0.561167
4     0.448934
6     0.112233
Name: count, dtype: float64
```



[96]:
```
sns.distplot(train_df['Age'])
print(train_df['Age'].skew())
print(train_df['Age'].kurt())
```

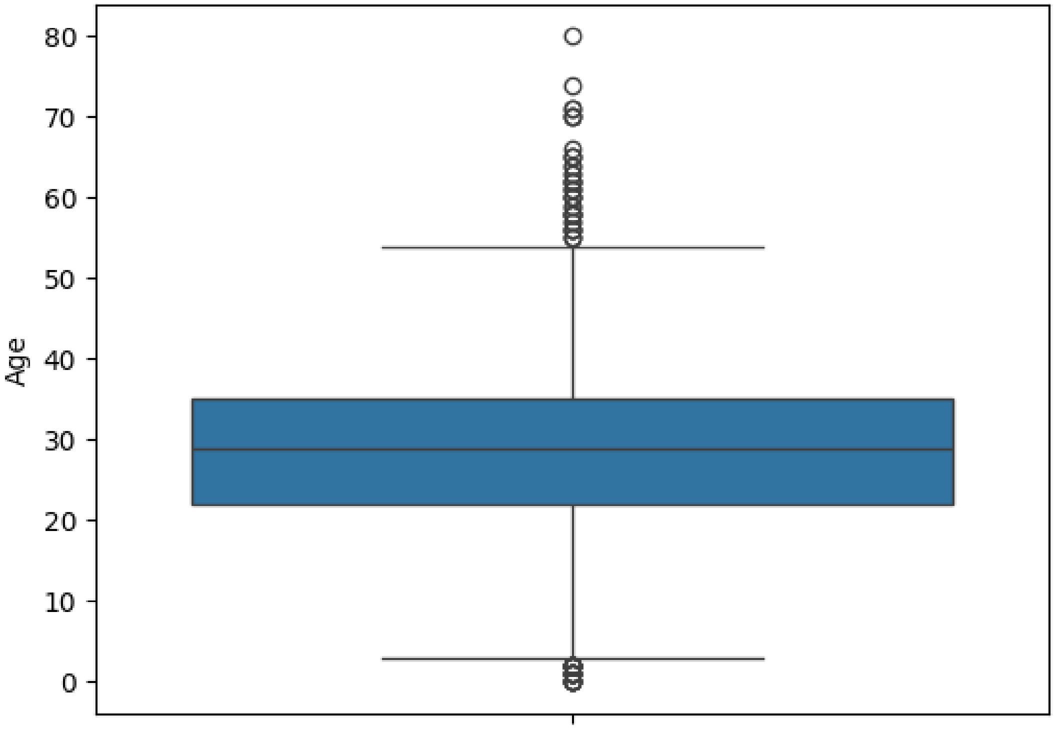
```
C:\Users\pandeysunny2315\AppData\Local\Temp\ipykernel_7880\101640707.py:1:
UserWarning:

`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(train_df['Age'])
```

```
0.45956263424701577
0.9865867453652877
```



```
[97]: sns.boxplot(train_df['Age'])
```

```
[97]: <Axes: ylabel='Age'>
```



```
[174]: def Family_type(number):
           if number==0:
               return 'Alone'
           elif number > 0 and number <= 4:
               return 'Medium'
           else:
               return 'Large'
```

```
[182]: train_df['Family_size'].sample(10)
```

```
[182]: 11     0
       96     0
       98     1
       634    5
       94     0
       260    0
       736    4
       221    0
       768    1
       675    0
       Name: Family_size, dtype: int64
```

[178]:
```
#train_df['Family_size']= train_df['Parch'] + train_df['SibSp']
(train_df['Family_size'] > 4).sum()
```

[178]:
```
47
```

[186]:
```
train_df['Family_type']=train_df['Family_size'].apply(Family_type)
```

[187]:
```
train_df['Family_size'].value_counts()
```

[187]:
```
Family_size
0     537
1     161
2     102
3      29
5      22
4      15
6      12
10      7
7       6
Name: count, dtype: int64
```

[188]:
```
train_df['Family_type'].value_counts()
```

[188]:
```
Family_type
Alone     537
Medium    307
Large      47
Name: count, dtype: int64
```

[199]:
```
train_df.drop(columns={'SibSp','Parch','Family_size'}, inplace=True)
```

[200]:
```
train_df.sample(5)
```

[200]:
```
     PassengerId Survived Pclass                       Name     Sex  Age  \
450          451        0      2      West, Mr. Edwy Arthur    male   36
656          657        0      3      Radeff, Mr. Alexander    male   29
56            57        1      2          Rugg, Miss. Emily  female   21
443          444        1      2  Reynaldo, Ms. Encarnacion  female   28
810          811        0      3     Alexander, Mr. William    male   26

         Ticket     Fare Embarked Family_type
450  C.A. 34651  27.7500        S      Medium
656      349223   7.8958        S       Alone
56   C.A. 31026  10.5000        S       Alone
443      230434  13.0000        S       Alone
810        3474   7.8875        S       Alone
```

```
[192]: pd.crosstab(train_df['Family_type'],train_df['Survived']).apply(lambda r:␣
         ↪round((r/r.sum())*100, 1), axis=1)
```

```
[192]: Survived        0     1
       Family_type
       Alone        69.6  30.4
       Large        85.1  14.9
       Medium       44.0  56.0
```

```
[197]: pd.crosstab(train_df['Sex'],train_df['Survived']).apply(lambda r: round((r/r.
         ↪sum())*100, 1), axis=1)
```

```
[197]: Survived     0     1
       Sex
       female    25.8  74.2
       male      81.1  18.9
```

```
[198]: pd.crosstab(train_df['Pclass'], train_df['Survived']).apply(lambda r: round((r/
         ↪r.sum())*100, 1), axis=1)
```

```
[198]: Survived     0     1
       Pclass
       1         37.0  63.0
       2         52.7  47.3
       3         75.8  24.2
```

```
[203]: pd.crosstab(train_df['Embarked'],train_df['Survived']).apply(lambda r: round((r/
         ↪r.sum())*100,1), axis=1)
```

```
[203]: Survived     0     1
       Embarked
       C         44.6  55.4
       Q         61.0  39.0
       S         66.1  33.9
```

```
[204]: train_df['Embarked'].value_counts()
```

```
[204]: Embarked
       S    646
       C    168
       Q     77
       Name: count, dtype: int64
```

```
[209]: # Chances of female survived is higher than male survived as you can see 74.2%␣
         ↪are females and only 18.9% of mens were survived.
       # People travelling in pclass 1 are having more likely to survive than pclass 3␣
         ↪and  pclass 2.
```

```
# somehow people travelling "Medium group" have more chances of surviving than people travelling "Alone" are with "Large group".
# people going to C are more likely to survived.
# people in the range of 20-40 had a higher chance of not surviving.
```

[ ]:

```
```