In [1]:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings("ignore")
```

In [2]:

```
df=pd.read_csv("train.csv")
df
```

Out[2]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	F
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0
...	...	...	...	...	...	...	...	...	...	
886	887	0	2	Montvila, Rev. Juozas	male	27.0	0	0	211536	13.0
887	888	1	1	Graham, Miss. Margaret Edith	female	19.0	0	0	112053	30.0
888	889	0	3	Johnston, Miss. Catherine Helen "Carrie"	female	NaN	1	2	W./C. 6607	23.4
889	890	1	1	Behr, Mr. Karl Howell	male	26.0	0	0	111369	30.0
890	891	0	3	Dooley, Mr. Patrick	male	32.0	0	0	370376	7.7

891 rows × 12 columns

In [3]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 891 entries, 0 to 890
Data columns (total 12 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  Cabin        204 non-null    object 
 11  Embarked     889 non-null    object 
dtypes: float64(2), int64(5), object(5)
memory usage: 83.7+ KB
```

In [4]:

```
df.head()
```

Out[4]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Far
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.250
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.283
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.925
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.100
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.050

In [5]:

```
df.tail(7)
```

Out[5]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	F
884	885	0	3	Sutehall, Mr. Henry Jr	male	25.0	0	0	SOTON/OQ 392076	7.(
885	886	0	3	Rice, Mrs. William (Margaret Norton)	female	39.0	0	5	382652	29.'
886	887	0	2	Montvila, Rev. Juozas	male	27.0	0	0	211536	13.(
887	888	1	1	Graham, Miss. Margaret Edith	female	19.0	0	0	112053	30.(
888	889	0	3	Johnston, Miss. Catherine Helen "Carrie"	female	NaN	1	2	W./C. 6607	23.‹
889	890	1	1	Behr, Mr. Karl Howell	male	26.0	0	0	111369	30.(
890	891	0	3	Dooley, Mr. Patrick	male	32.0	0	0	370376	7.'

In [6]:
```
df.isna()
```

Out[6]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabi
0	False	False	False	False	False	False	False	False	False	False	Tru
1	False	False	False	False	False	False	False	False	False	False	Fals
2	False	False	False	False	False	False	False	False	False	False	Tru
3	False	False	False	False	False	False	False	False	False	False	Fals
4	False	False	False	False	False	False	False	False	False	False	Tru
...	...	...	...	...	...	...	...	...	...	...	.
886	False	False	False	False	False	False	False	False	False	False	Tru
887	False	False	False	False	False	False	False	False	False	False	Fals
888	False	False	False	False	False	True	False	False	False	False	Tru
889	False	False	False	False	False	False	False	False	False	False	Fals
890	False	False	False	False	False	False	False	False	False	False	Tru

891 rows × 12 columns

In [7]:
```
df.isna().sum()
```

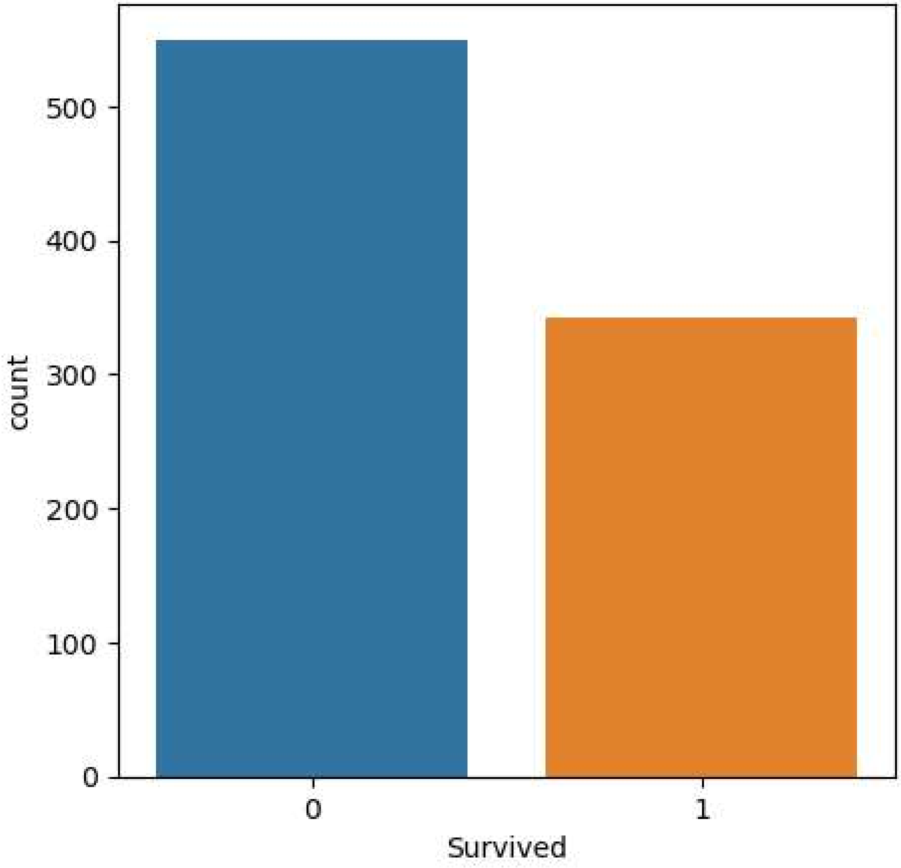
Out[7]:
```
PassengerId      0
Survived         0
Pclass           0
Name             0
Sex              0
Age            177
SibSp            0
Parch            0
Ticket           0
Fare             0
Cabin          687
Embarked         2
dtype: int64
```

Missing values in Age , Cabin , and Embarked.

70% of the data are missing in cabin.

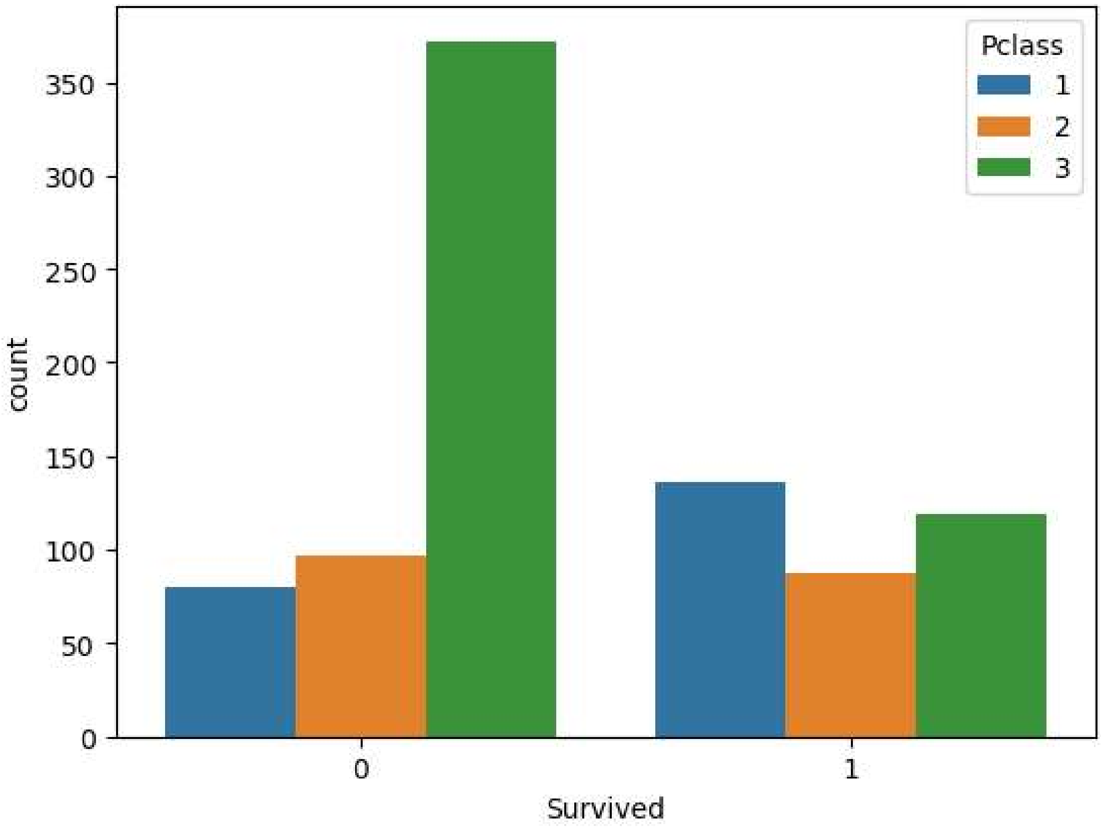
In [8]:

```
plt.figure(figsize=(5,5))
sns.countplot(x="Survived",data=df);
```



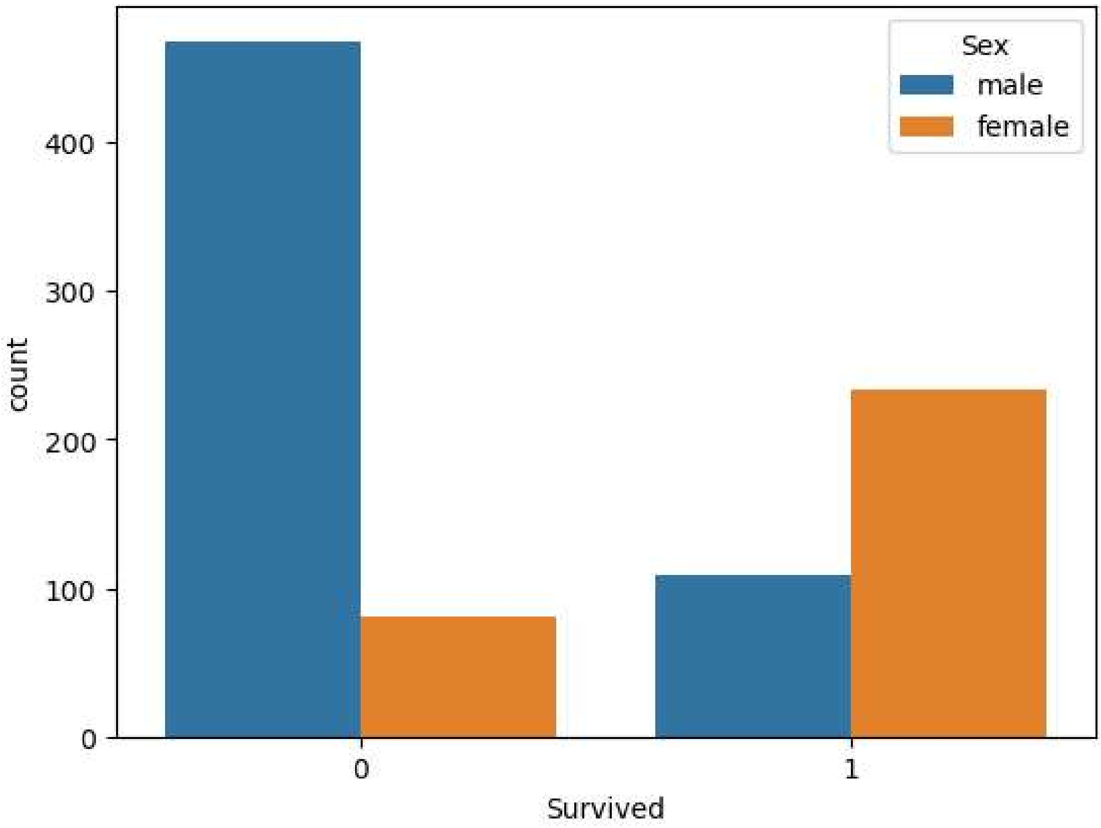
In [9]:

```
sns.countplot(x="Survived",hue='Pclass',data=df);
```



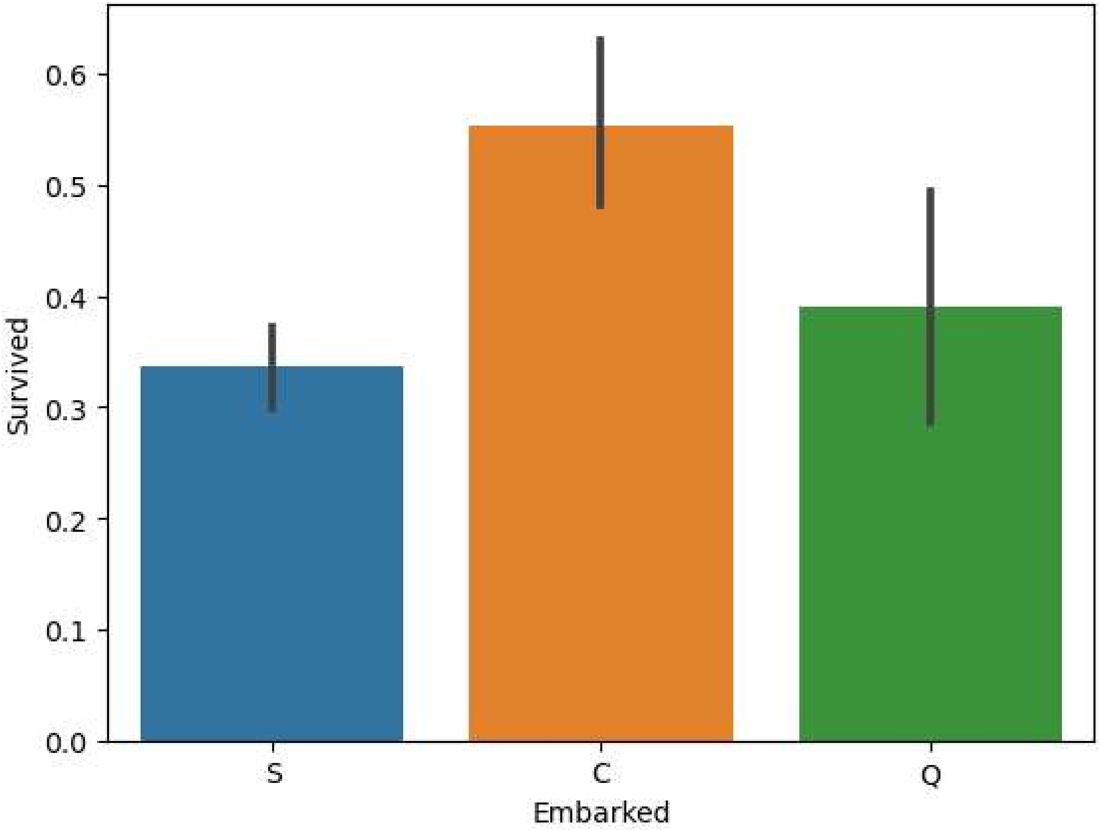
In [10]:

```
sns.countplot(x="Survived",hue="Sex",data=df);
```



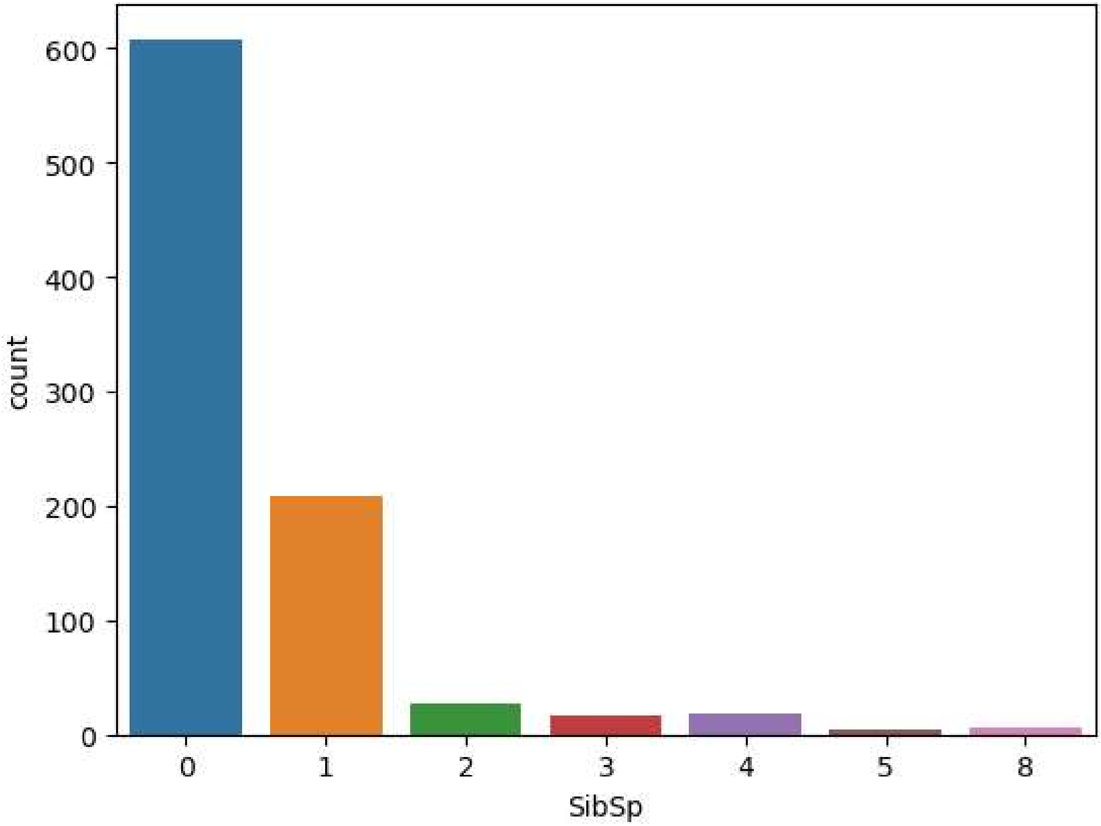
In [11]:

```
sns.barplot(data=df,x='Embarked',y='Survived');
```



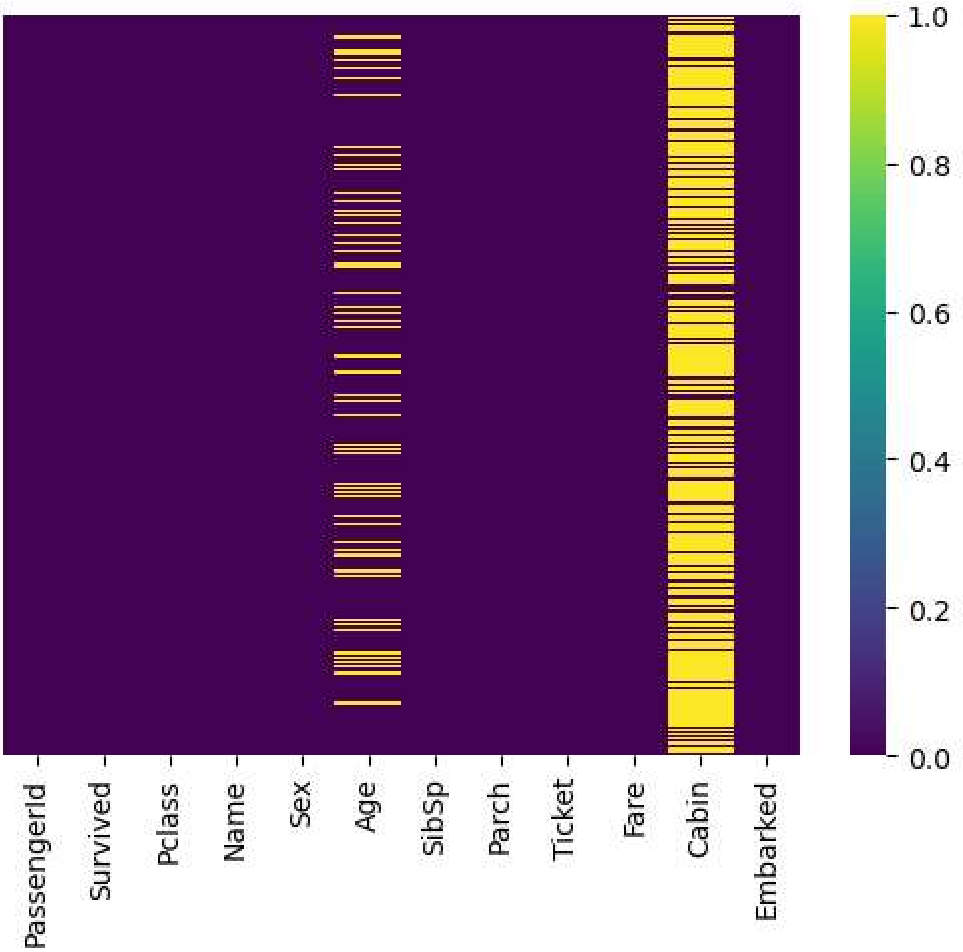
In [12]:

```
sns.countplot(x='SibSp',data=df);
```



In [13]:

```
sns.heatmap(df.isnull(),yticklabels=False,cbar=True,cmap="viridis");
```



In [14]:

```
df.corr().style.background_gradient(cmap='coolwarm').set_precision(3)
```

Out[14]:

	PassengerId	Survived	Pclass	Age	SibSp	Parch	Fare
PassengerId	1.000	-0.005	-0.035	0.037	-0.058	-0.002	0.013
Survived	-0.005	1.000	-0.338	-0.077	-0.035	0.082	0.257
Pclass	-0.035	-0.338	1.000	-0.369	0.083	0.018	-0.549
Age	0.037	-0.077	-0.369	1.000	-0.308	-0.189	0.096
SibSp	-0.058	-0.035	0.083	-0.308	1.000	0.415	0.160
Parch	-0.002	0.082	0.018	-0.189	0.415	1.000	0.216
Fare	0.013	0.257	-0.549	0.096	0.160	0.216	1.000

In [15]:

```
df.describe()
```

Out[15]:

	PassengerId	Survived	Pclass	Age	SibSp	Parch	Far
count	891.000000	891.000000	891.000000	714.000000	891.000000	891.000000	891.00000
mean	446.000000	0.383838	2.308642	29.699118	0.523008	0.381594	32.20420
std	257.353842	0.486592	0.836071	14.526497	1.102743	0.806057	49.69342
min	1.000000	0.000000	1.000000	0.420000	0.000000	0.000000	0.00000
25%	223.500000	0.000000	2.000000	20.125000	0.000000	0.000000	7.91040
50%	446.000000	0.000000	3.000000	28.000000	0.000000	0.000000	14.45420
75%	668.500000	1.000000	3.000000	38.000000	1.000000	0.000000	31.00000
max	891.000000	1.000000	3.000000	80.000000	8.000000	6.000000	512.32920

In [16]:

```
df.count()#  Age,Cabin and Embarked have missing values
```

Out[16]:

```
PassengerId    891
Survived       891
Pclass         891
Name           891
Sex            891
Age            714
SibSp          891
Parch          891
Ticket         891
Fare           891
Cabin          204
Embarked       889
dtype: int64
```

In [17]:

```
df['Embarked'].value_counts()
```

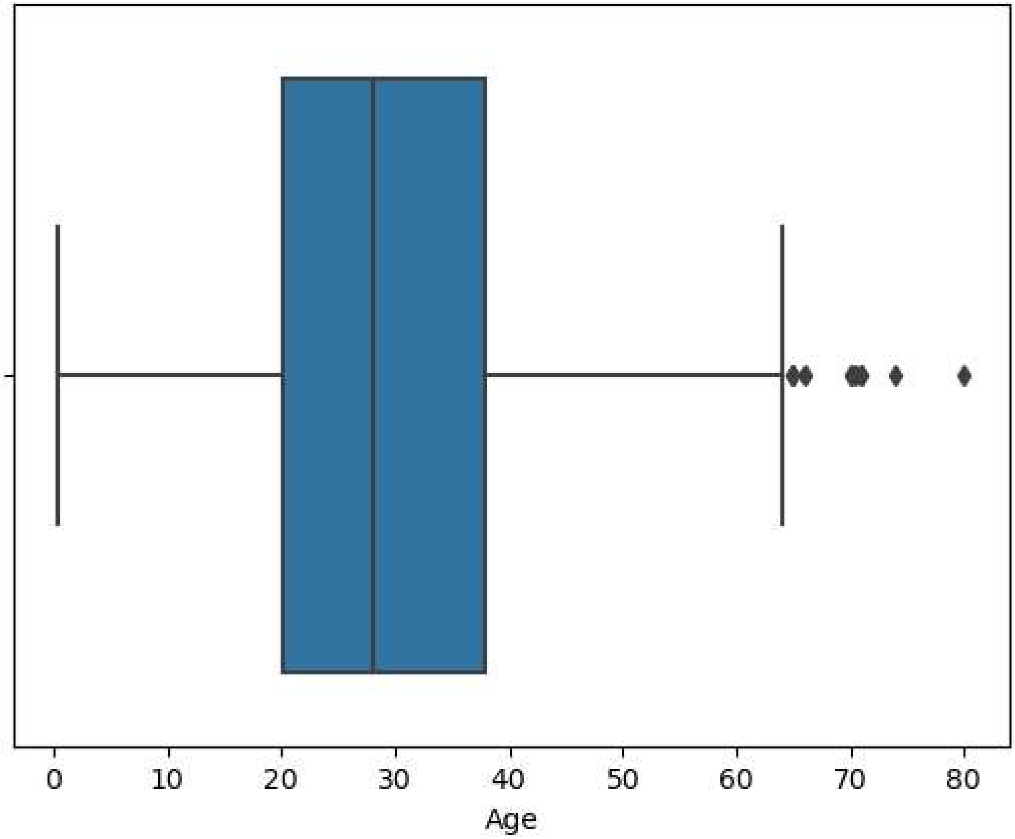
Out[17]:

```
S    644
C    168
Q     77
Name: Embarked, dtype: int64
```

Handling outlier

In [18]:

```
sns.boxplot(x="Age",data=df);
```



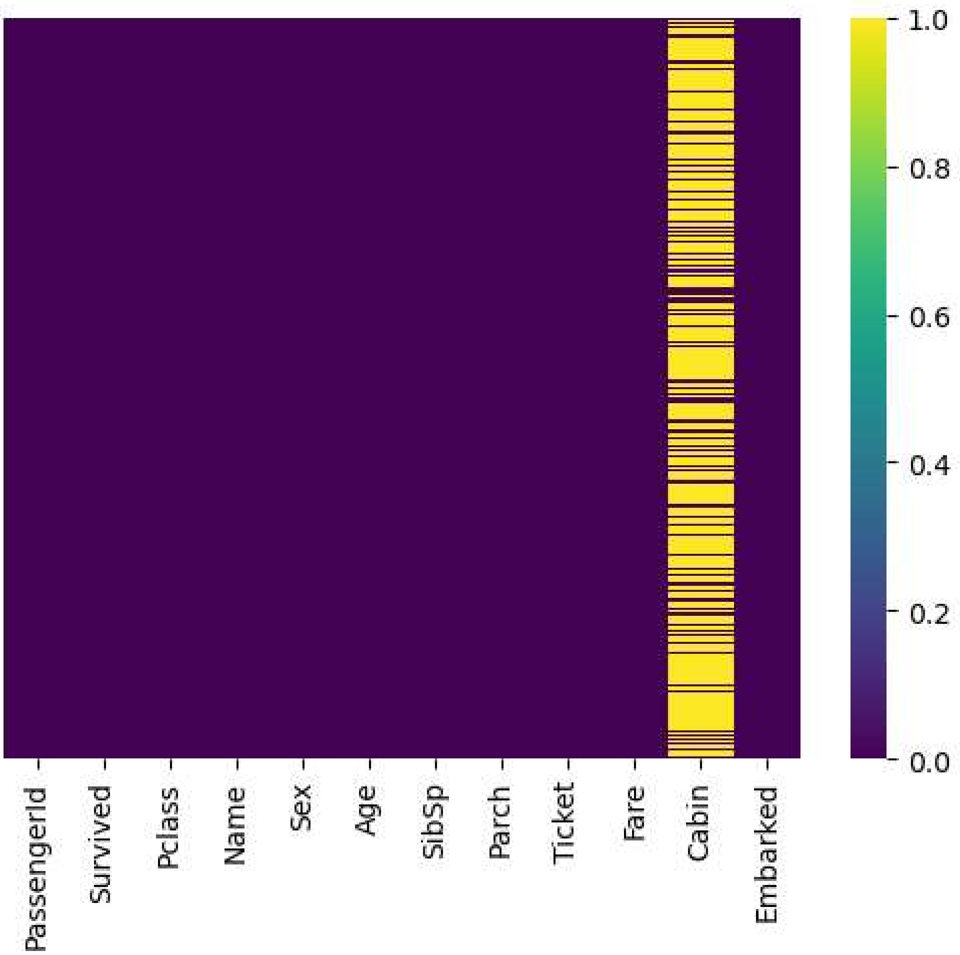
In [19]:

```
def fillage(cols):
    Age=cols[0]
    Pclass=cols[1]
    if(pd.isnull(Age)):
        if(Pclass)==1:
            return 38
        elif(Pclass)==2:
            return 29
        else:
            return 24
    else:
        return Age
```

In [20]:

```
df["Age"]=df[["Age","Pclass"]].apply(fillage,axis=1)
```

```
In [21]: sns.heatmap(df.isnull(),yticklabels=False,cbar=True,cmap="viridis");
```



```
In [22]: df.count()
```

```
Out[22]: PassengerId    891
         Survived       891
         Pclass         891
         Name           891
         Sex            891
         Age            891
         SibSp          891
         Parch          891
         Ticket         891
         Fare           891
         Cabin          204
         Embarked       889
         dtype: int64
```

```
In [23]: #df.drop('Cabin',axis=1,inplace=True) #  There are too many missing values
```

In [24]:

```
df["Embarked"].replace("",np.nan,inplace=True)
df["Embarked"].fillna("S",inplace=True)
```

In [25]:

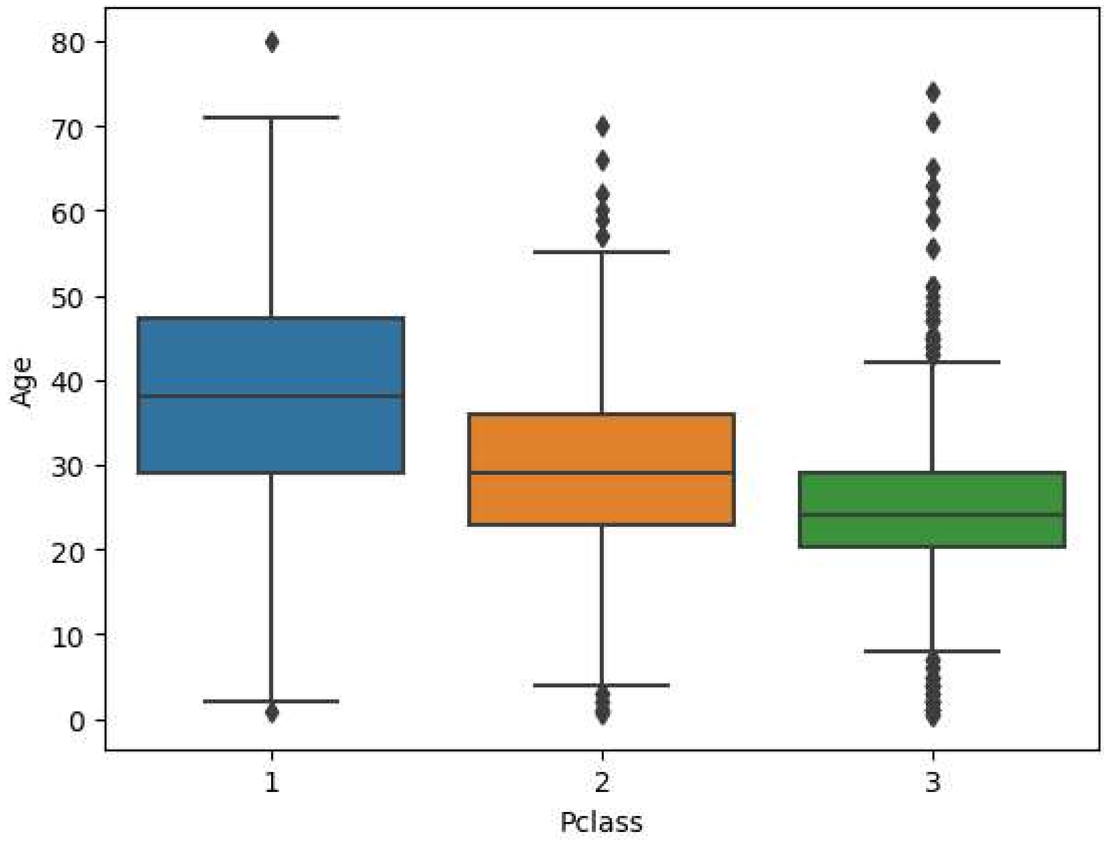
```
df.count()
```

Out[25]:

```
PassengerId    891
Survived       891
Pclass         891
Name           891
Sex            891
Age            891
SibSp          891
Parch          891
Ticket         891
Fare           891
Cabin          204
Embarked       891
dtype: int64
```

In [26]:

```
sns.boxplot(data=df,x="Pclass",y="Age");
```



In [27]:

```
df[(df["Pclass"]==1)&(df["Age"]>70)]
```

Out[27]:

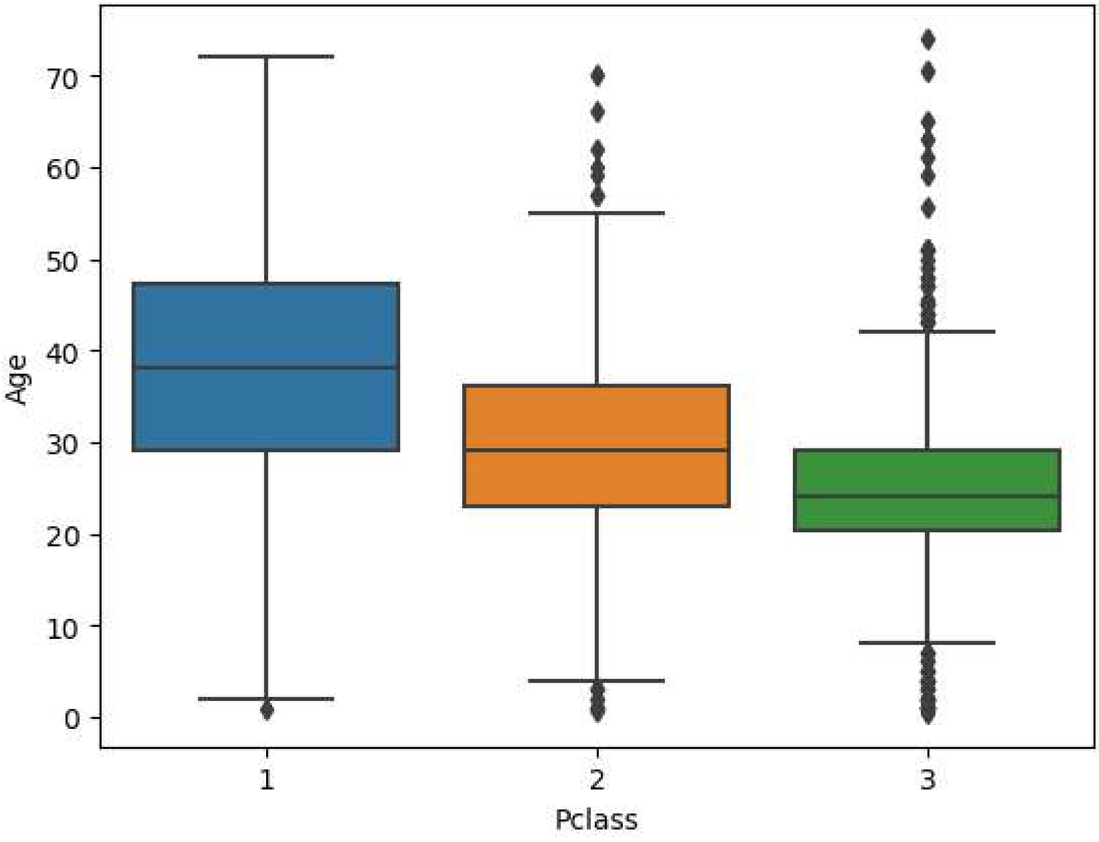
	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare
96	97	0	1	Goldschmidt, Mr. George B	male	71.0	0	0	PC 17754	34.6542
493	494	0	1	Artagaveytia, Mr. Ramon	male	71.0	0	0	PC 17609	49.5042
630	631	1	1	Barkworth, Mr. Algernon Henry Wilson	male	80.0	0	0	27042	30.0000

In [28]:

```
#replacing with upper whisker values
df.loc[[96,493,630],"Age"]=72
```

In [29]:

```
sns.boxplot(data=df,x="Pclass",y="Age");
```



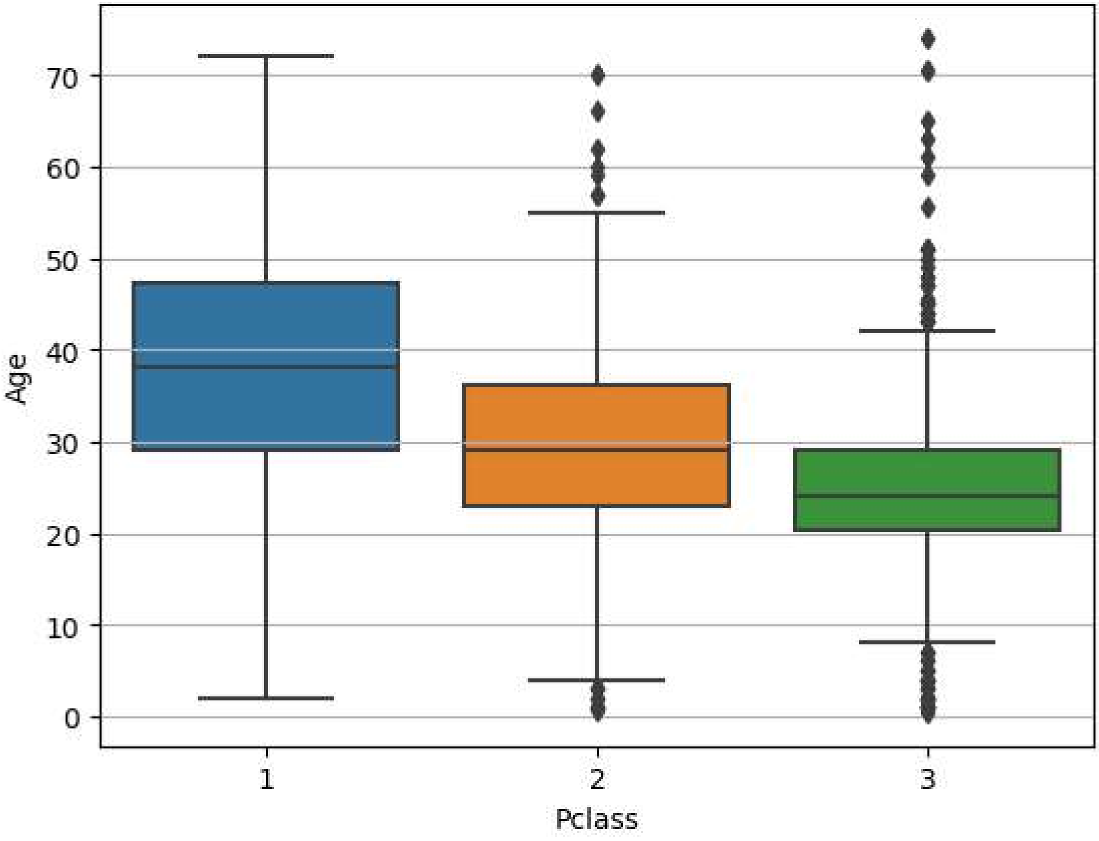
In [30]:
```
df[(df["Pclass"]==1)&(df["Age"]<1)]
```

Out[30]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cal
305	306	1	1	Allison, Master. Hudson Trevor	male	0.92	1	2	113781	151.55	C C

In [31]:
```
df.loc[305,"Age"]=2
```

In [32]:
```
plt.grid()
sns.boxplot(data=df,x="Pclass",y="Age");
```



In [33]:

```
df[(df["Pclass"]==2)&(df["Age"]<4)]
```

Out[33]:

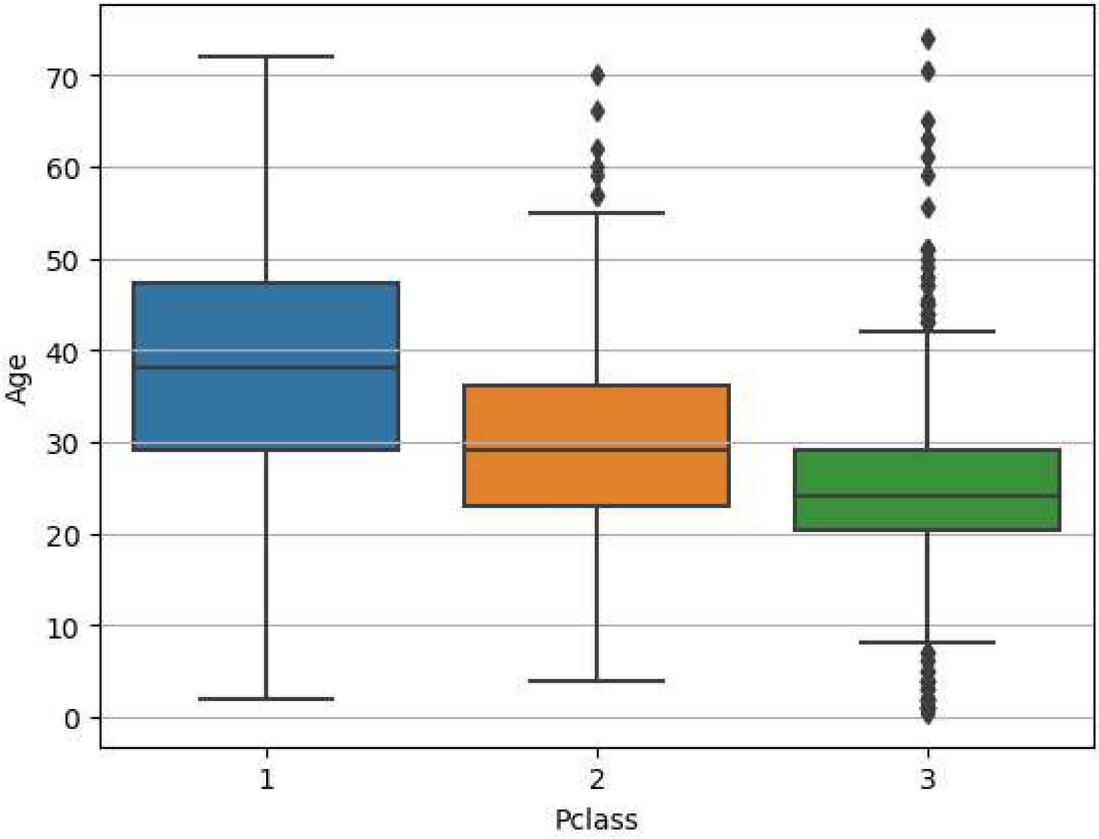
	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	
43	44	1	2	Laroche, Miss. Simonne Marie Anne Andree	female	3.00	1	2	SC/Paris 2123	4
78	79	1	2	Caldwell, Master. Alden Gates	male	0.83	0	2	248738	2
183	184	1	2	Becker, Master. Richard F	male	1.00	2	1	230136	3
193	194	1	2	Navratil, Master. Michel M	male	3.00	1	1	230080	2
340	341	1	2	Navratil, Master. Edmond Roger	male	2.00	1	1	230080	2
407	408	1	2	Richards, Master. William Rowe	male	3.00	1	1	29106	1
530	531	1	2	Quick, Miss. Phyllis May	female	2.00	1	1	26360	2
755	756	1	2	Hamalainen, Master. Viljo	male	0.67	1	1	250649	1
827	828	1	2	Mallet, Master. Andre	male	1.00	0	2	S.C./PARIS 2079	3
831	832	1	2	Richards, Master. George Sibley	male	0.83	1	1	29106	1

In [34]:

```
df.loc[[43,78,183,193,340,407,530,755,827,831],"Age"]=4
```

In [35]:

```
plt.grid()
sns.boxplot(data=df,x="Pclass",y="Age");
```



In [36]:

```
df[(df["Pclass"]==2)&(df["Age"]>54)]
```

Out[36]:

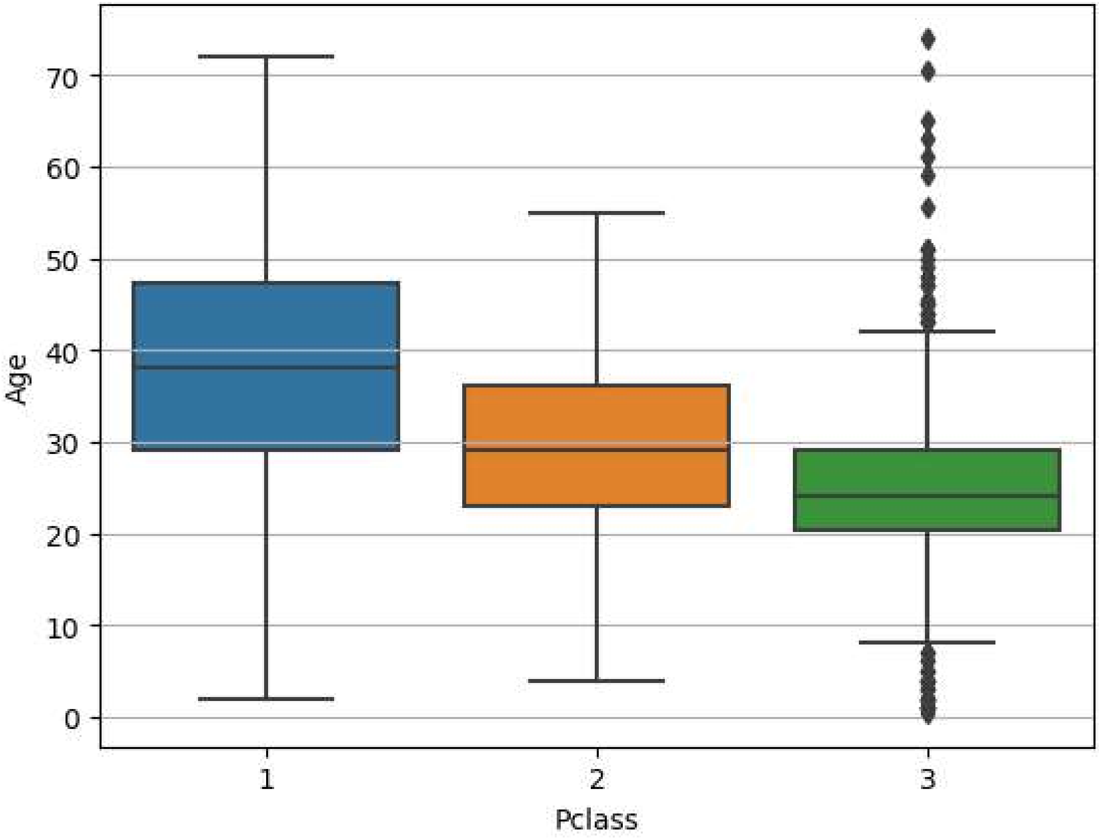
	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare
15	16	1	2	Hewlett, Mrs. (Mary D Kingcome)	female	55.0	0	0	248706	16.00
33	34	0	2	Wheadon, Mr. Edward H	male	66.0	0	0	C.A. 24579	10.50
232	233	0	2	Sjostedt, Mr. Ernst Adolf	male	59.0	0	0	237442	13.50
570	571	1	2	Harris, Mr. George	male	62.0	0	0	S.W./PP 752	10.50
626	627	0	2	Kirkland, Rev. Charles Leonard	male	57.0	0	0	219533	12.35
672	673	0	2	Mitchell, Mr. Henry Michael	male	70.0	0	0	C.A. 24580	10.50
684	685	0	2	Brown, Mr. Thomas William Solomon	male	60.0	1	1	29750	39.00
772	773	0	2	Mack, Mrs. (Mary)	female	57.0	0	0	S.O./P.P. 3	10.50

In [37]:

```
df.loc[[15,33,232,570,626,672,684,772],"Age"]=55
```

In [38]:

```
plt.grid()
sns.boxplot(data=df,x="Pclass",y="Age");
```



In [39]:

```
df[(df["Pclass"]==3)&(df["Age"]<9)]
```

Out[39]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fa
7	8	0	3	Palsson, Master. Gosta Leonard	male	2.00	3	1	349909	21.07
10	11	1	3	Sandstrom, Miss. Marguerite Rut	female	4.00	1	1	PP 9549	16.70
16	17	0	3	Rice, Master. Eugene	male	2.00	4	1	382652	29.12
24	25	0	3	Palsson, Miss. Torborg Danira	female	8.00	3	1	349909	21.07
50	51	0	3	Panula, Master. Juha Niilo	male	7.00	4	1	3101295	39.68
63	64	0	3	Skoog, Master. Harald	male	4.00	3	2	347088	27.90
119	120	0	3	Andersson, Miss. Ellis Anna Maria	female	2.00	4	2	347082	31.27
164	165	0	3	Panula, Master. Eino Viljami	male	1.00	4	1	3101295	39.68
171	172	0	3	Rice, Master. Arthur	male	4.00	4	1	382652	29.12
172	173	1	3	Johnson, Miss. Eleanor Ileen	female	1.00	1	1	347742	11.13
184	185	1	3	Kink-Heilmann, Miss. Luise Gretchen	female	4.00	0	2	315153	22.02
205	206	0	3	Strom, Miss. Telma Matilda	female	2.00	0	1	347054	10.46
233	234	1	3	Asplund, Miss. Lillian Gertrud	female	5.00	4	2	347077	31.38
261	262	1	3	Asplund, Master. Edvin Rojj Felix	male	3.00	4	2	347077	31.38
278	279	0	3	Rice, Master. Eric	male	7.00	4	1	382652	29.12

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fa
348	349	1	3	Coutts, Master. William Loch "William"	male	3.00	1	1	C.A. 37671	15.90
374	375	0	3	Palsson, Miss. Stina Viola	female	3.00	3	1	349909	21.07
381	382	1	3	Nakid, Miss. Maria ("Mary")	female	1.00	0	2	2653	15.74
386	387	0	3	Goodwin, Master. Sidney Leonard	male	1.00	5	2	CA 2144	46.90
448	449	1	3	Baclini, Miss. Marie Catherine	female	5.00	2	1	2666	19.25
469	470	1	3	Baclini, Miss. Helene Barbara	female	0.75	2	1	2666	19.25
479	480	1	3	Hirvonen, Miss. Hildur E	female	2.00	0	1	3101298	12.28
642	643	0	3	Skoog, Miss. Margit Elizabeth	female	2.00	3	2	347088	27.90
644	645	1	3	Baclini, Miss. Eugenie	female	0.75	2	1	2666	19.25
691	692	1	3	Karun, Miss. Manca	female	4.00	0	1	349256	13.41
751	752	1	3	Moor, Master. Meier	male	6.00	0	1	392096	12.47
777	778	1	3	Emanuel, Miss. Virginia Ethel	female	5.00	0	0	364516	12.47
787	788	0	3	Rice, Master. George Hugh	male	8.00	4	1	382652	29.12
788	789	1	3	Dean, Master. Bertram Vere	male	1.00	1	2	C.A. 2315	20.57
803	804	1	3	Thomas, Master. Assad Alexander	male	0.42	0	1	2625	8.51

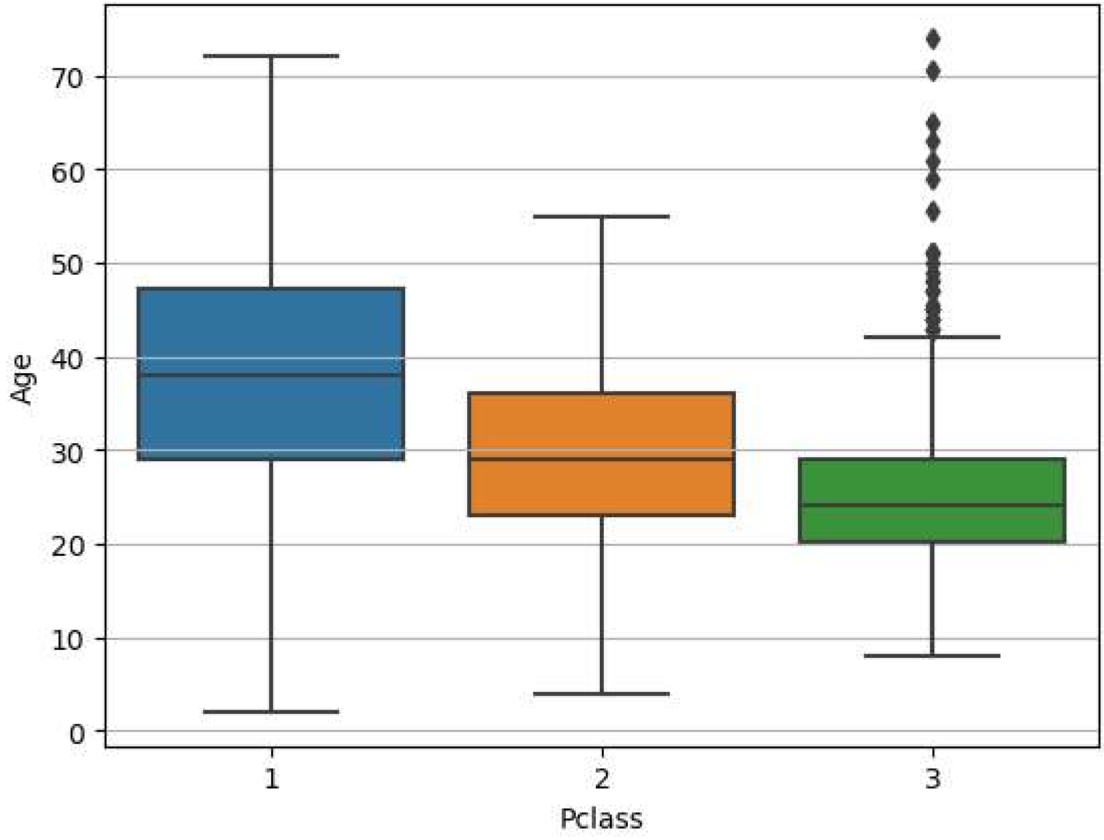
	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fa
813	814	0	3	Andersson, Miss. Ebba Iris Alfrida	female	6.00	4	2	347082	31.27
824	825	0	3	Panula, Master. Urho Abraham	male	2.00	4	1	3101295	39.68
850	851	0	3	Andersson, Master. Sigvard Harald Elias	male	4.00	4	2	347082	31.27
869	870	1	3	Johnson, Master. Harold Theodor	male	4.00	1	1	347742	11.13

In [40]:

```
df.loc[[7,10,16,24,50,63,119,164,171,172,184,205,233,261,278,348,374,381,3
```

In [41]:

```
plt.grid()
sns.boxplot(data=df,x="Pclass",y="Age");
```



In [42]:

```
df[(df["Pclass"]==3)&(df["Age"]>42)]
```

Out[42]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	F
94	95	0	3	Coxon, Mr. Daniel	male	59.0	0	0	364500	7.2
116	117	0	3	Connors, Mr. Patrick	male	70.5	0	0	370369	7.7
129	130	0	3	Ekstrom, Mr. Johan	male	45.0	0	0	347061	6.9
132	133	0	3	Robins, Mrs. Alexander A (Grace Charity Laury)	female	47.0	1	0	A/5. 3337	14.5
152	153	0	3	Meo, Mr. Alfonzo	male	55.5	0	0	A.5. 11206	8.0
160	161	0	3	Cribb, Mr. John Hatfield	male	44.0	0	1	371362	16.1
167	168	0	3	Skoog, Mrs. William (Anna Bernhardina Karlsson)	female	45.0	1	4	347088	27.9
203	204	0	3	Youseff, Mr. Gerious	male	45.5	0	0	2628	7.2
222	223	0	3	Green, Mr. George Henry	male	51.0	0	0	21440	8.0
276	277	0	3	Lindblom, Miss. Augusta Charlotta	female	45.0	0	0	347073	7.7
280	281	0	3	Duane, Mr. Frank	male	65.0	0	0	336439	7.7
326	327	0	3	Nysveen, Mr. Johan Hansen	male	61.0	0	0	345364	6.2
338	339	1	3	Dahl, Mr. Karl Edwart	male	45.0	0	0	7598	8.0
362	363	0	3	Barbara, Mrs. (Catherine David)	female	45.0	0	1	2691	14.4
406	407	0	3	Widegren, Mr. Carl/Charles Peter	male	51.0	0	0	347064	7.7
414	415	1	3	Sundman, Mr. Johan Julian	male	44.0	0	0	STON/O 2. 3101269	7.9
482	483	0	3	Rouse, Mr. Richard Henry	male	50.0	0	0	A/5 3594	8.0

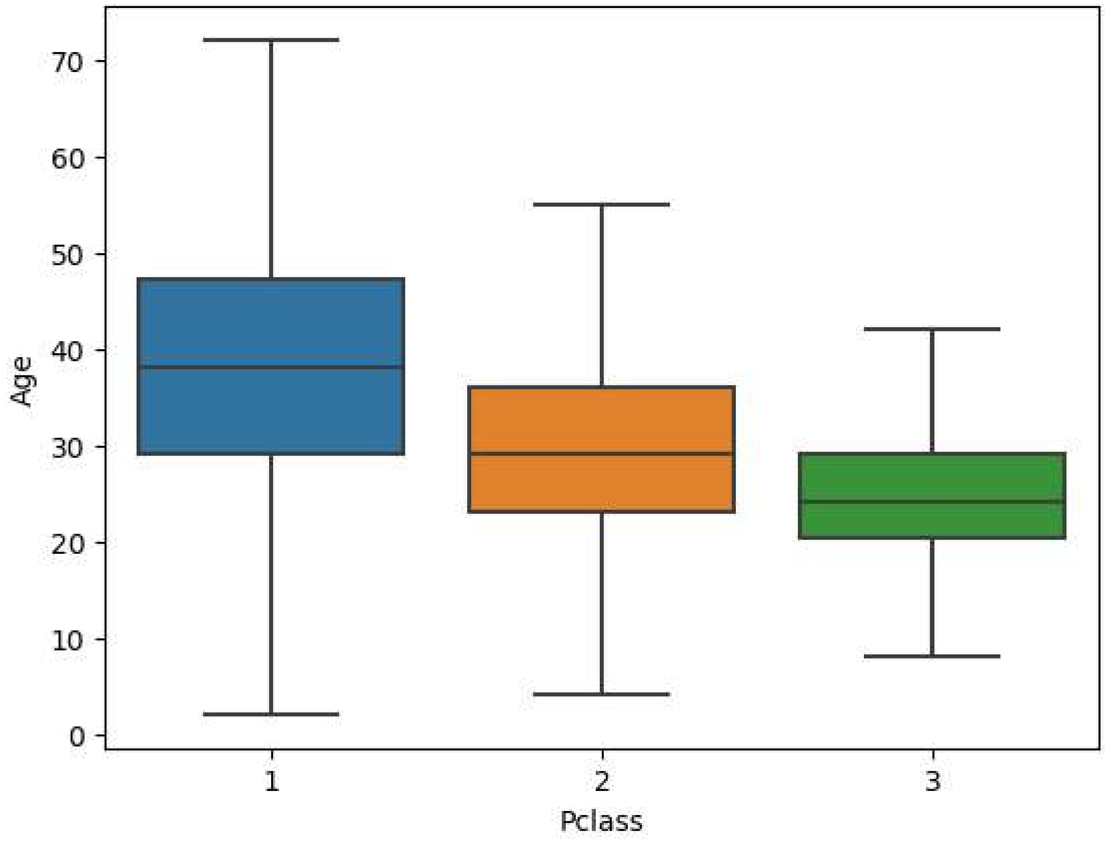
	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	F
483	484	1	3	Turkula, Mrs. (Hedwig)	female	63.0	0	0	4134	9.5
592	593	0	3	Elsbury, Mr. William James	male	47.0	0	0	A/5 3902	7.2
597	598	0	3	Johnson, Mr. Alfred	male	49.0	0	0	LINE	0.0
603	604	0	3	Torber, Mr. Ernst William	male	44.0	0	0	364511	8.0
631	632	0	3	Lundahl, Mr. Johan Svensson	male	51.0	0	0	347743	7.0
668	669	0	3	Cook, Mr. Jacob	male	43.0	0	0	A/5 3536	8.0
678	679	0	3	Goodwin, Mrs. Frederick (Augusta Tyler)	female	43.0	1	6	CA 2144	46.9
696	697	0	3	Kelly, Mr. James	male	44.0	0	0	363592	8.0
736	737	0	3	Ford, Mrs. Edward (Margaret Ann Watson)	female	48.0	1	3	W./C. 6608	34.3
771	772	0	3	Jensen, Mr. Niels Peder	male	48.0	0	0	350047	7.8
818	819	0	3	Holm, Mr. John Fredrik Alexander	male	43.0	0	0	C 7075	6.4
851	852	0	3	Svensson, Mr. Johan	male	74.0	0	0	347060	7.7
873	874	0	3	Vander Cruyssen, Mr. Victor	male	47.0	0	0	345765	9.0

In [43]:

```
df.loc[[94,116,129,132,152,160,167,203,222,276,280,326,338,362,406,414,482
```

In [44]:

```
sns.boxplot(data=df,x="Pclass",y="Age");
```



In [45]:

```
feature=df.iloc[:,[2,4,5]]
target=df["Survived"]
```

In [46]:

```
feature
```

Out[46]:

```
     Pclass     Sex   Age
  0       3    male  22.0
  1       1  female  38.0
  2       3  female  26.0
  3       1  female  35.0
  4       3    male  35.0
...     ...     ...   ...
886       2    male  27.0
887       1  female  19.0
888       3  female  24.0
889       1    male  26.0
890       3    male  32.0

891 rows × 3 columns
```

In [47]:

```
target
```

Out[47]:

```
0      0
1      1
2      1
3      1
4      0
      ..
886    0
887    1
888    0
889    1
890    0
Name: Survived, Length: 891, dtype: int64
```

In [48]:

```
target.shape
```

Out[48]:

```
(891,)
```

In [49]:

```
feature.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 891 entries, 0 to 890
Data columns (total 3 columns):
 #   Column  Non-Null Count  Dtype  
---  ------  --------------  -----  
 0   Pclass  891 non-null    int64  
 1   Sex     891 non-null    object 
 2   Age     891 non-null    float64
dtypes: float64(1), int64(1), object(1)
memory usage: 21.0+ KB
```

In [50]:

```
catcol=feature.select_dtypes("object").columns
```

In [51]:

```
catcol
```

Out[51]:

```
Index(['Sex'], dtype='object')
```

In [52]:

```
#Encoding input data using Ordinal Encoder
#Encoding the Gender column of input data using Ordinal Encoder

from sklearn.preprocessing import OrdinalEncoder
oe=OrdinalEncoder()
feature=oe.fit_transform(feature)
```

In [53]:

```
print(feature)
```

```
[[ 2.  1. 21.]
 [ 0.  0. 44.]
 [ 2.  0. 27.]
 ...
 [ 2.  0. 24.]
 [ 0.  1. 27.]
 [ 2.  1. 35.]]
```

Train and fit the model

In [54]:

```
from sklearn.model_selection import train_test_split
xtrain,xtest,ytrain,ytest=train_test_split(feature,target,test_size=0.2,ra
```

In [55]:

```
print(xtrain.shape)
print(ytrain.shape)
print(xtest.shape)
print(ytest.shape)
```

```
(712, 3)
(712,)
(179, 3)
(179,)
```

In [56]:

```
# -------import the model using K Nearest Neighbors (KNN) Classification--

from sklearn.neighbors import KNeighborsClassifier

knn=KNeighborsClassifier(n_neighbors=3)

knn.fit(xtrain,ytrain)

ypred=knn.predict(xtest)
```

Model Evaluation

In [57]:

```python
from sklearn.metrics import accuracy_score,confusion_matrix,classification

acc=accuracy_score(ytest,ypred)
cm=confusion_matrix(ytest,ypred)
cr=classification_report(ytest,ypred)

print(f"Accuracy :- {acc}\n{cm}\n{cr}")
```

```
Accuracy :- 0.770949720670391
[[95 15]
 [26 43]]
              precision    recall  f1-score   support

           0       0.79      0.86      0.82       110
           1       0.74      0.62      0.68        69

    accuracy                           0.77       179
   macro avg       0.76      0.74      0.75       179
weighted avg       0.77      0.77      0.77       179
```

Check Accuracy

In [58]:

```python
# KNN :-Training score and testing score
trainacc = knn.score(xtrain, ytrain)
testacc = knn.score(xtest, ytest)

print(f"Training Accuracy -: {trainacc}\nTesting Accuracy -: {testacc}")
```

```
Training Accuracy -: 0.8328651685393258
Testing Accuracy -: 0.770949720670391
```

Hyperparameter Tunning

In [59]:

```
# Getting the best K - value

final_k=[]

for i in range(1,31):
    knn=KNeighborsClassifier(n_neighbors=i)
    knn.fit(xtrain,ytrain)
    pred=knn.predict(xtest)
    k=accuracy_score(ytest,pred,normalize=True)*float(100)
    final_k.append(k)
    print('\n ytest accuracy for k=%d is %d'%(i,k))
```

```
ytest accuracy for k=1 is 74
ytest accuracy for k=2 is 75
ytest accuracy for k=3 is 77
ytest accuracy for k=4 is 73
ytest accuracy for k=5 is 77
ytest accuracy for k=6 is 78
ytest accuracy for k=7 is 74
ytest accuracy for k=8 is 74
ytest accuracy for k=9 is 74
ytest accuracy for k=10 is 72
ytest accuracy for k=11 is 75
ytest accuracy for k=12 is 77
ytest accuracy for k=13 is 76
ytest accuracy for k=14 is 74
ytest accuracy for k=15 is 75
ytest accuracy for k=16 is 75
ytest accuracy for k=17 is 75
ytest accuracy for k=18 is 73
ytest accuracy for k=19 is 73
ytest accuracy for k=20 is 71
ytest accuracy for k=21 is 73
ytest accuracy for k=22 is 72
ytest accuracy for k=23 is 72
ytest accuracy for k=24 is 73
ytest accuracy for k=25 is 70
ytest accuracy for k=26 is 69
ytest accuracy for k=27 is 69
ytest accuracy for k=28 is 69
ytest accuracy for k=29 is 69

ytest accuracy for k=30 is 69
```

In [60]:

```python
optimal_k = final_k.index(max(final_k))
print(optimal_k)
```

```
5
```

In [61]:

```python
# taking K value as 5
from sklearn.neighbors import KNeighborsClassifier

knn=KNeighborsClassifier(n_neighbors=5)

knn.fit(xtrain,ytrain)

ypred=knn.predict(xtest)
```

In [62]:

```python
acc=accuracy_score(ytest,ypred)
cm=confusion_matrix(ytest,ypred)
cr=classification_report(ytest,ypred)

print(f"Accuracy :- {acc}\n{cm}\n{cr}")
```

```
Accuracy :- 0.776536312849162
[[100  10]
 [ 30  39]]
              precision    recall  f1-score   support

           0       0.77      0.91      0.83       110
           1       0.80      0.57      0.66        69

    accuracy                           0.78       179
   macro avg       0.78      0.74      0.75       179
weighted avg       0.78      0.78      0.77       179
```

In [63]:

```python
# Applying LogisticRegression
from sklearn.linear_model import LogisticRegression
logreg = LogisticRegression()
logreg.fit(xtrain, ytrain)
ypred = logreg.predict(xtest)
```

In [64]:

```python
test=[[2,0,24]]
yp=logreg.predict(test)
```

In [65]:

```
#-------model prediction--------
if(yp==1):
    print("Survived")
else:
    print("Not survived")
```

```
Survived
```

In [66]:

```
from sklearn.metrics import accuracy_score,confusion_matrix,classification
acc=accuracy_score(ytest,ypred)
cm=confusion_matrix(ytest,ypred)
cr=classification_report(ytest,ypred)
print(f"Accuraccy :{acc}\nConfusion Matrix\n{cm}\n{cr}")
```

```
Accuraccy :0.7932960893854749
Confusion Matrix
[[93 17]
 [20 49]]
              precision    recall  f1-score   support

           0       0.82      0.85      0.83       110
           1       0.74      0.71      0.73        69

    accuracy                           0.79       179
   macro avg       0.78      0.78      0.78       179
weighted avg       0.79      0.79      0.79       179
```

In [67]:

```
trainingacc=logreg.score(xtrain,ytrain)
testingacc=logreg.score(xtest,ytest)

print("Training Score=",trainingacc)
print("Testing Score=",testingacc)
```

```
Training Score= 0.7949438202247191
Testing Score= 0.7932960893854749
```

In [68]:

```
#low bias + low variance => best fit
```

In [ ]: