In [1]:
```
# [IMPORTING ALL THE IMPORTANT LIBRARIES]


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

EDA WITH PYTHON AND APPLYING LOGISTIC REGRESSION MODEL

DATA TAKEN FROM [TITANIC DATA SET OF KAGGLE] . WILL TRY TO PREDICT A CLASSIFICATION OF SURVIVAL OR DECEASED. WE WILL CLEAN THE DATA IN THIS DUE PROCESS AS WELL.

In [2]:
```
train = pd.read_csv("titanic_train.csv")
```

In [9]:
```
train.head(20) #READING THE DATA SET#
```

Out[9]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	S
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN	S
5	6	0	3	Moran, Mr. James	male	NaN	0	0	330877	8.4583	NaN	Q
6	7	0	1	McCarthy, Mr. Timothy J	male	54.0	0	0	17463	51.8625	E46	S
7	8	0	3	Palsson, Master. Gosta Leonard	male	2.0	3	1	349909	21.0750	NaN	S
8	9	1	3	Johnson, Mrs. Oscar W (Elisabeth Vilhelmina Berg)	female	27.0	0	2	347742	11.1333	NaN	S
9	10	1	2	Nasser, Mrs. Nicholas (Adele Achem)	female	14.0	1	0	237736	30.0708	NaN	C
10	11	1	3	Sandstrom, Miss. Marguerite Rut	female	4.0	1	1	PP 9549	16.7000	G6	S
11	12	1	1	Bonnell, Miss. Elizabeth	female	58.0	0	0	113783	26.5500	C103	S
12	13	0	3	Saundercock, Mr. William Henry	male	20.0	0	0	A/5. 2151	8.0500	NaN	S
13	14	0	3	Andersson, Mr. Anders Johan	male	39.0	1	5	347082	31.2750	NaN	S
14	15	0	3	Vestrom, Miss. Hulda Amanda Adolfina	female	14.0	0	0	350406	7.8542	NaN	S
15	16	1	2	Hewlett, Mrs. (Mary D Kingcome)	female	55.0	0	0	248706	16.0000	NaN	S
16	17	0	3	Rice, Master. Eugene	male	2.0	4	1	382652	29.1250	NaN	Q
17	18	1	2	Williams, Mr. Charles Eugene	male	NaN	0	0	244373	13.0000	NaN	S
18	19	0	3	Vander Planke, Mrs. Julius (Emelia Maria Vande...	female	31.0	1	0	345763	18.0000	NaN	S
19	20	1	3	Masselmani, Mrs. Fatima	female	NaN	0	0	2649	7.2250	NaN	C

```
In [ ]:
```

MISSING DATA

WE CAN USE SEABORN TO FIND OUT WHICH DATA ARE MISSING FROM THE TABLE, WHICH COLUMN HAS LEAST AMOUNT OF DATA, SO WE CAN ELIMINATE IT IN FUTURE AND WE CAN CLEAN THE DATA BUT GETTING RID OF UNNECESSARY COLUMNS.

```
In [4]: train.isnull()
```

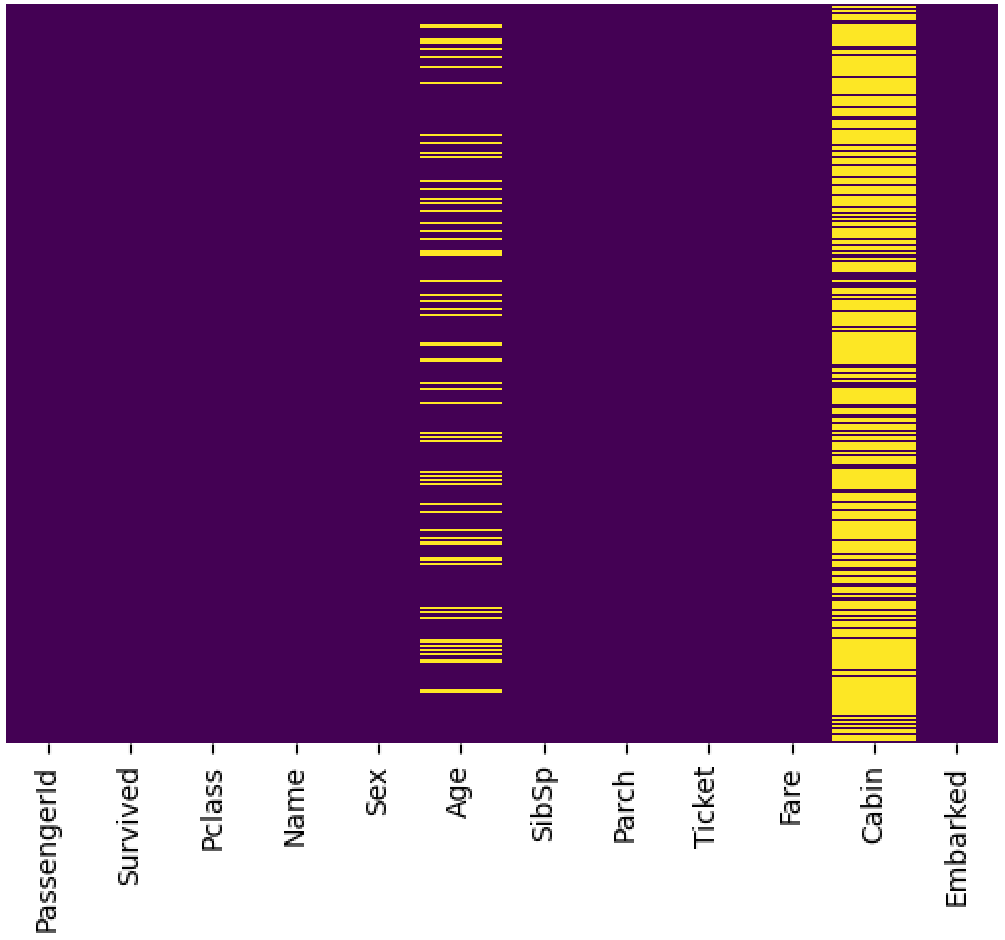
Out[4]:

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

891 rows × 12 columns

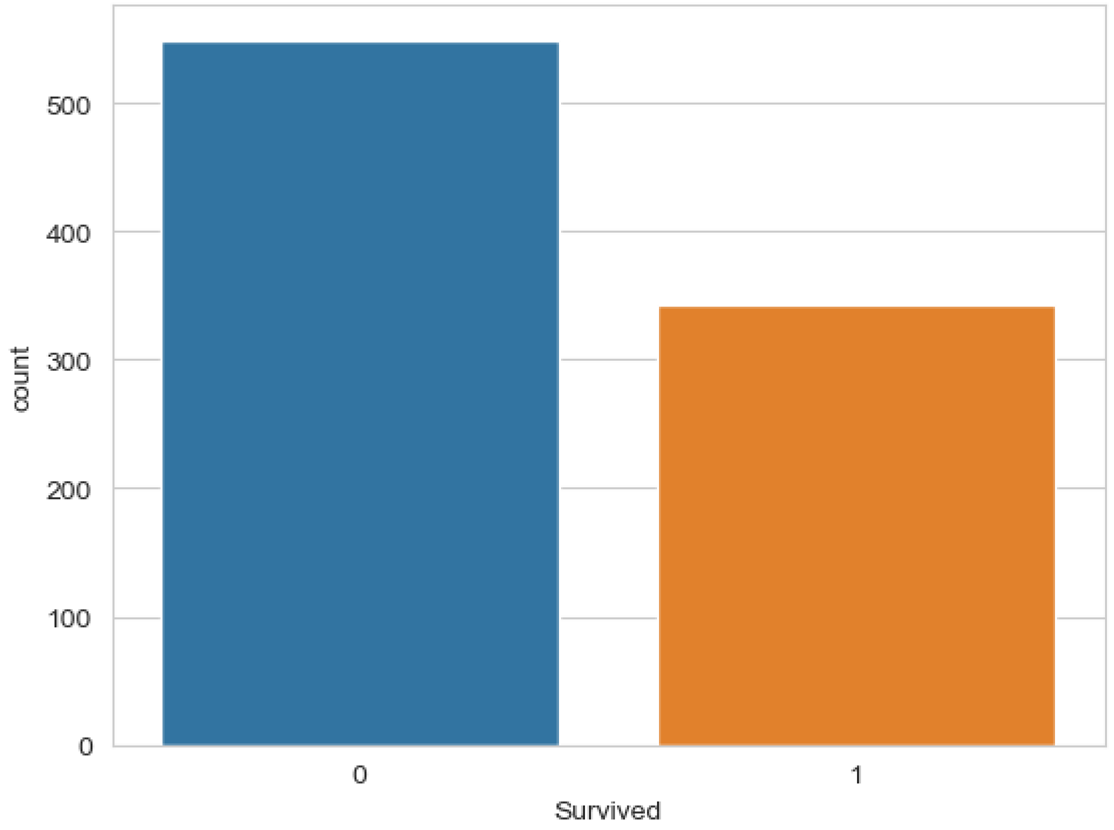
In [5]:
```
sns.heatmap(train.isnull(),yticklabels=False,cbar=False,cmap='viridis')
```

Out[5]:
```
<Axes: >
```



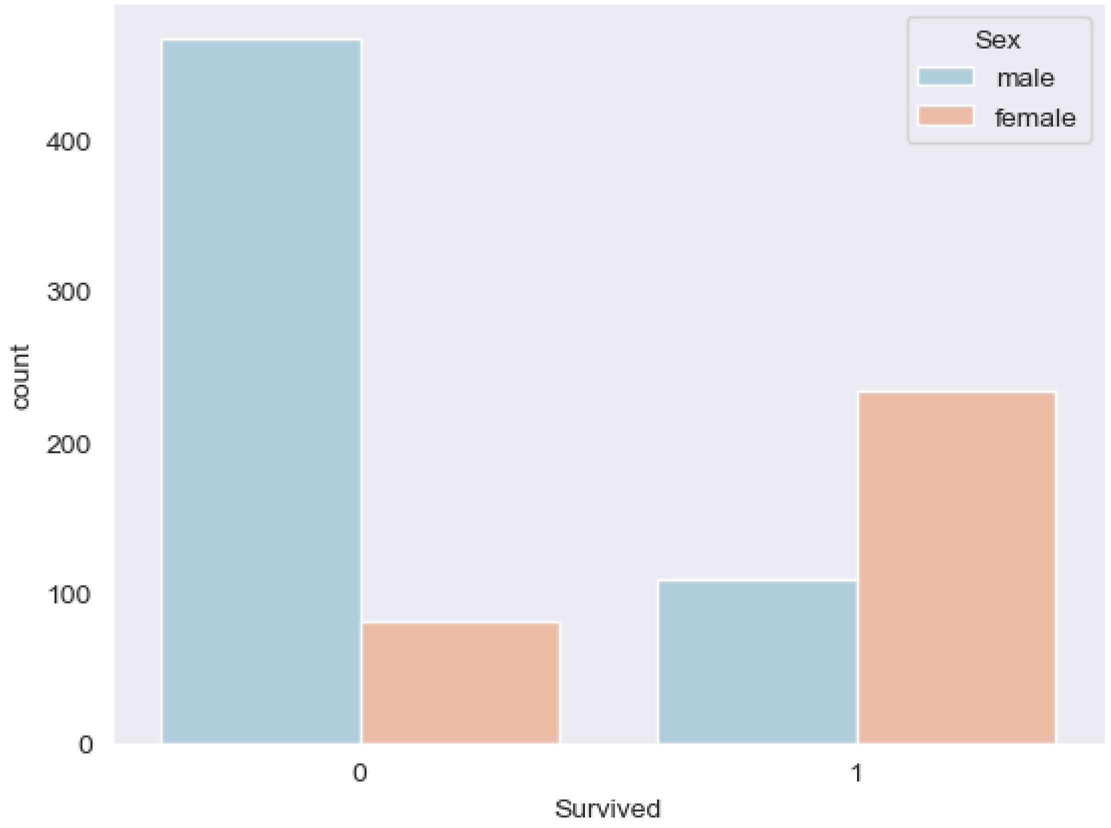
In [6]:
```
sns.set_style('whitegrid')
sns.countplot(x="Survived",data= train)
```

Out[6]:
```
<Axes: xlabel='Survived', ylabel='count'>
```



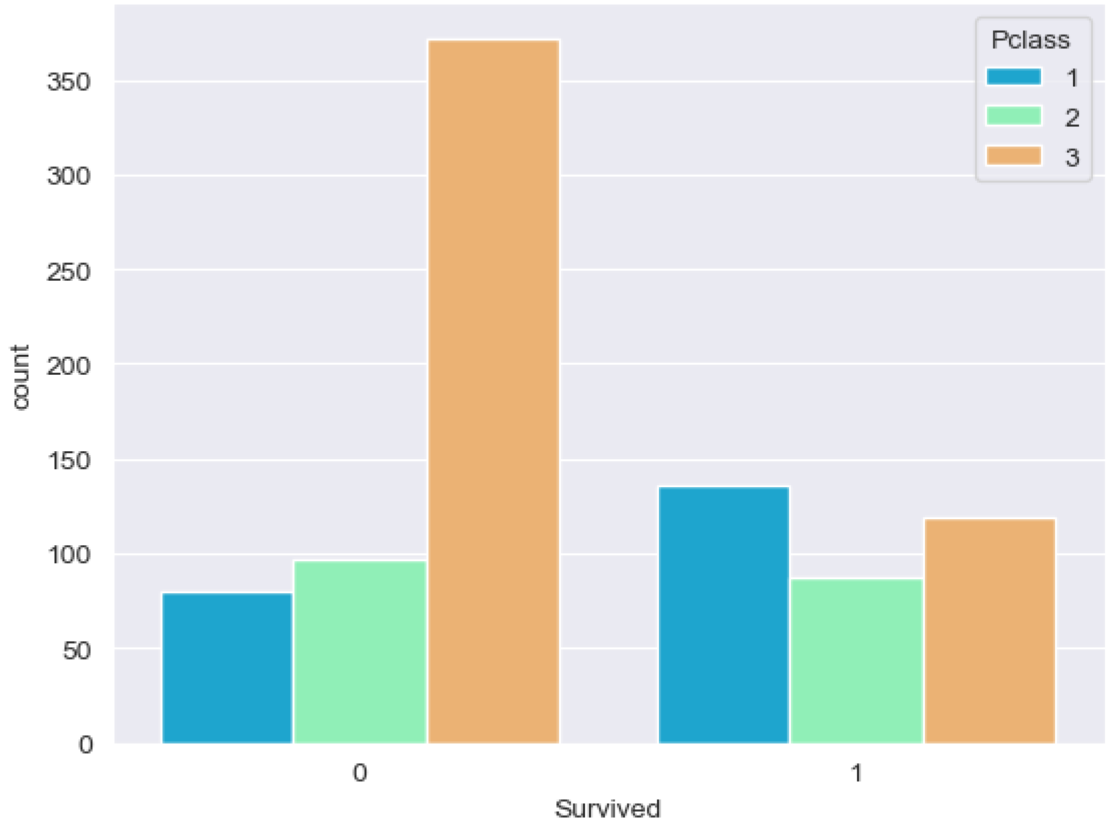
In [11]:
```
sns.set_style("dark")
sns.countplot(x="Survived",hue="Sex",data= train,palette="RdBu_r")
```

Out[11]:
```
<Axes: xlabel='Survived', ylabel='count'>
```



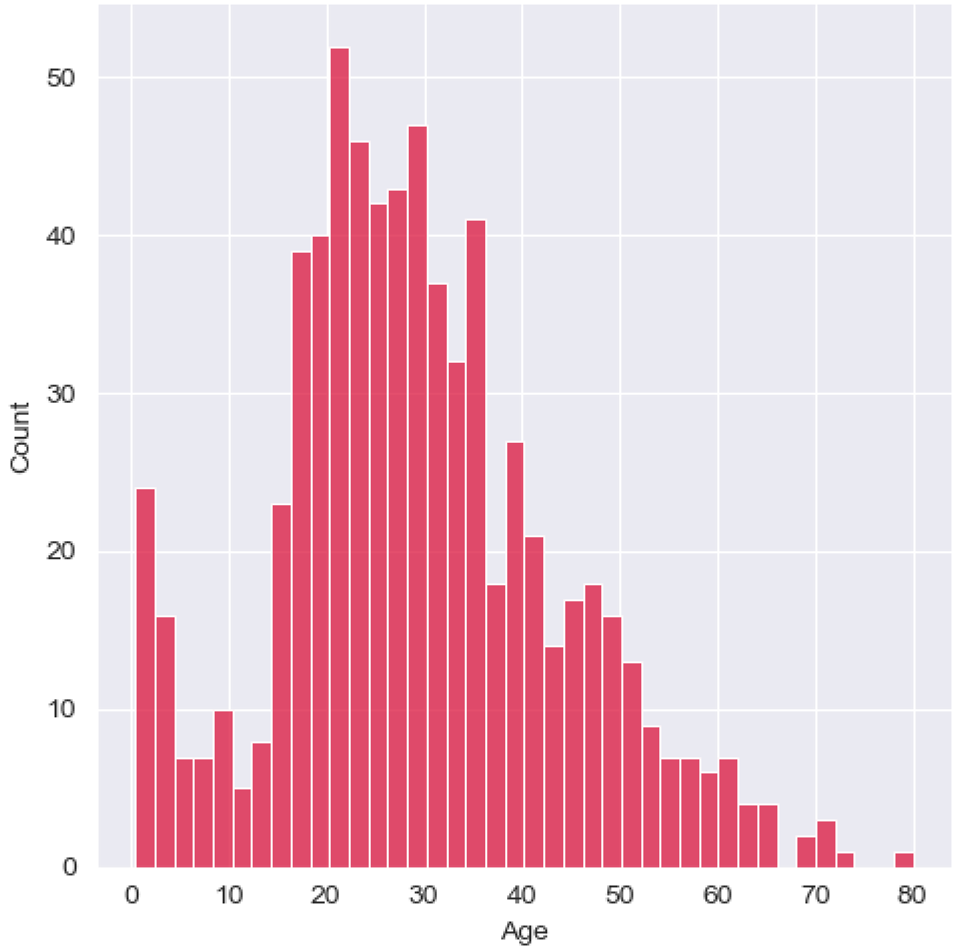
In [13]:
```
sns.set_style("darkgrid")
sns.countplot(x="Survived",hue="Pclass",data= train,palette="rainbow" )
```

Out[13]:
```
<Axes: xlabel='Survived', ylabel='count'>
```



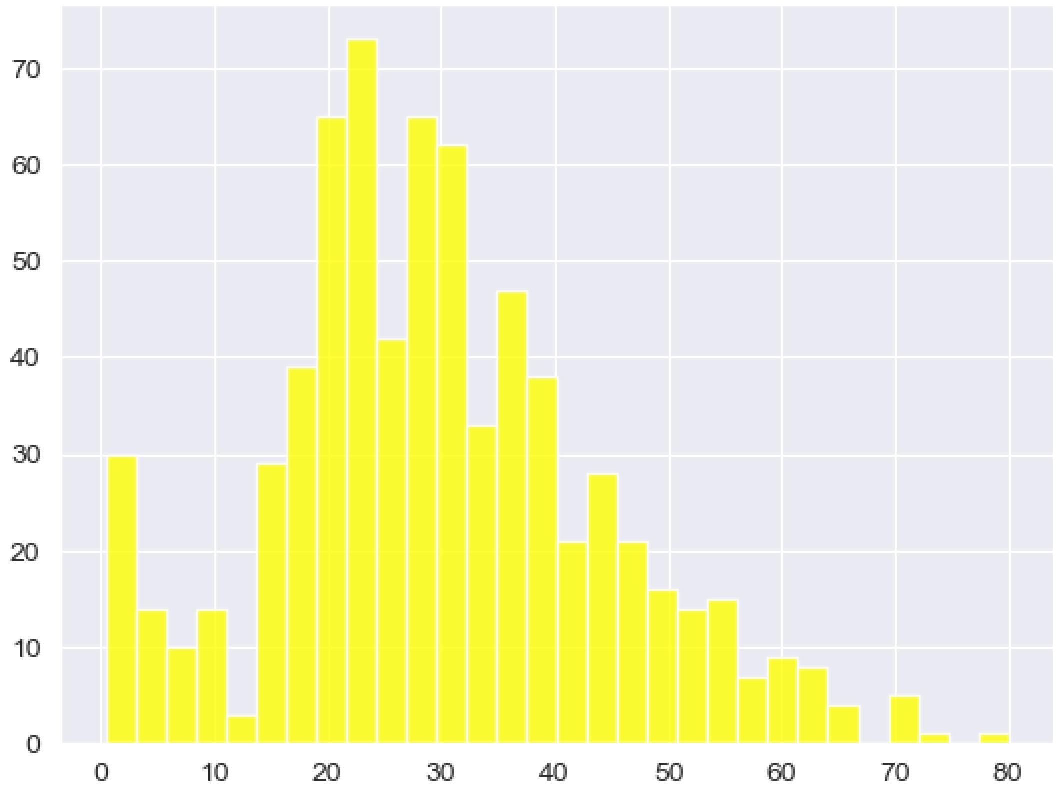
```
In [18]: sns.displot(train['Age'].dropna(),kde=False,color='crimson',bins=40)

Out[18]: <seaborn.axisgrid.FacetGrid at 0x217deef73a0>
```



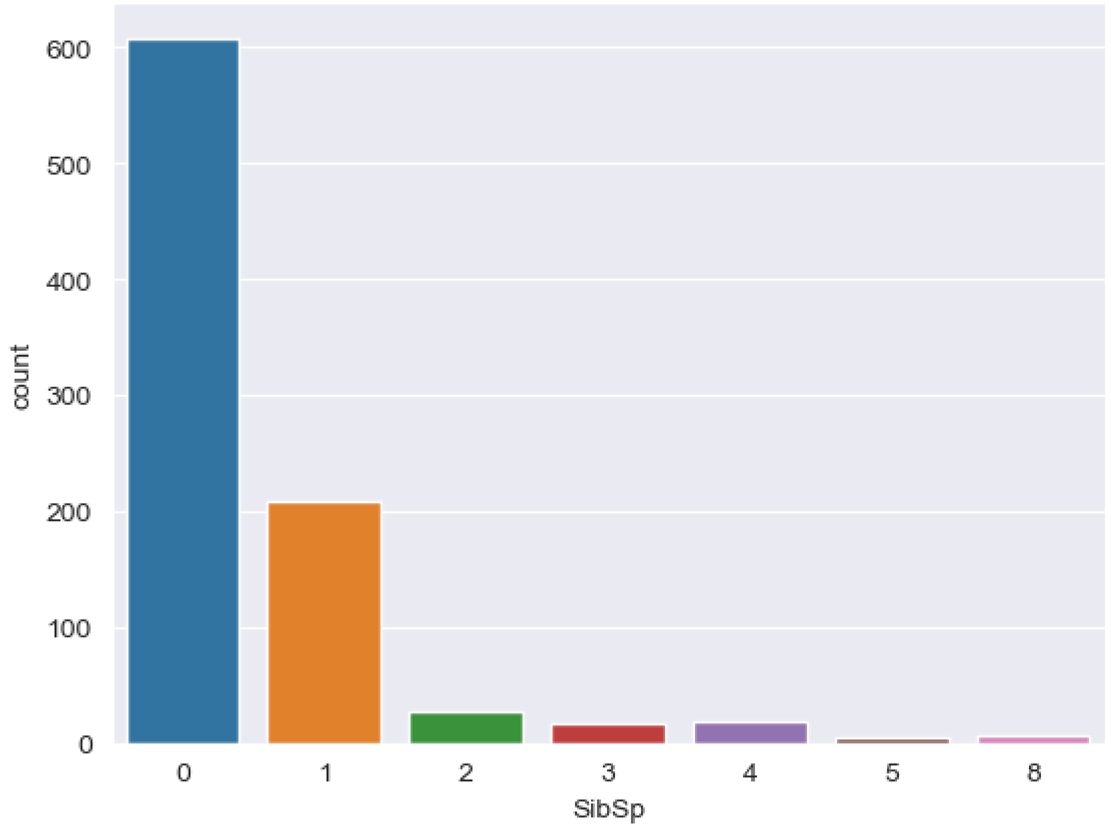
```
In [19]: train['Age'].hist(bins=30,color= "yellow",alpha= 0.8)
```

```
Out[19]: <Axes: >
```



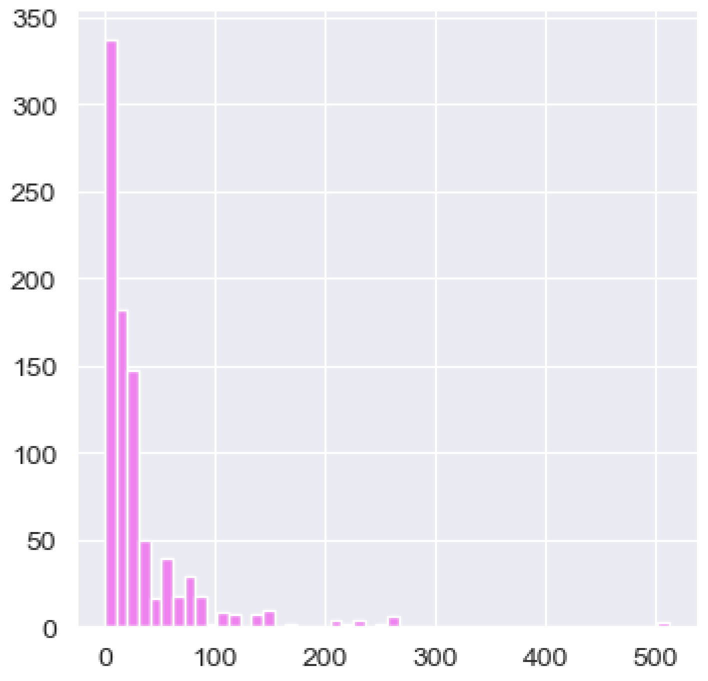
In [20]:
```
sns.countplot(x='SibSp',data=train)
```

Out[20]:
```
<Axes: xlabel='SibSp', ylabel='count'>
```



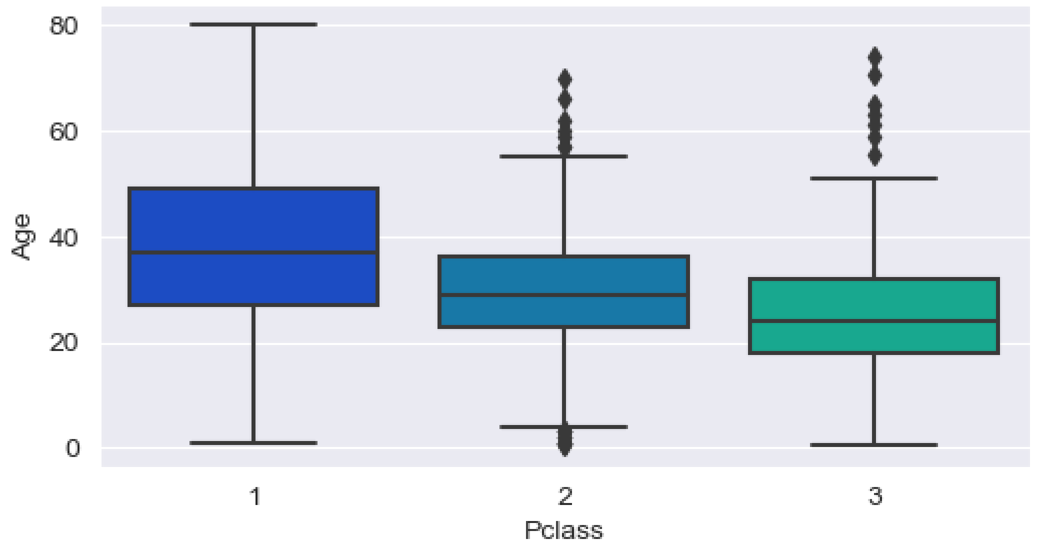
```
In [26]: train["Fare"].hist(color='violet',bins=50,figsize=(4,4))

Out[26]: <Axes: >
```



In [32]:
```
plt.figure(figsize=(6,3))
sns.boxplot(x='Pclass',y='Age',data=train,palette='winter')
```

Out[32]:
```
<Axes: xlabel='Pclass', ylabel='Age'>
```



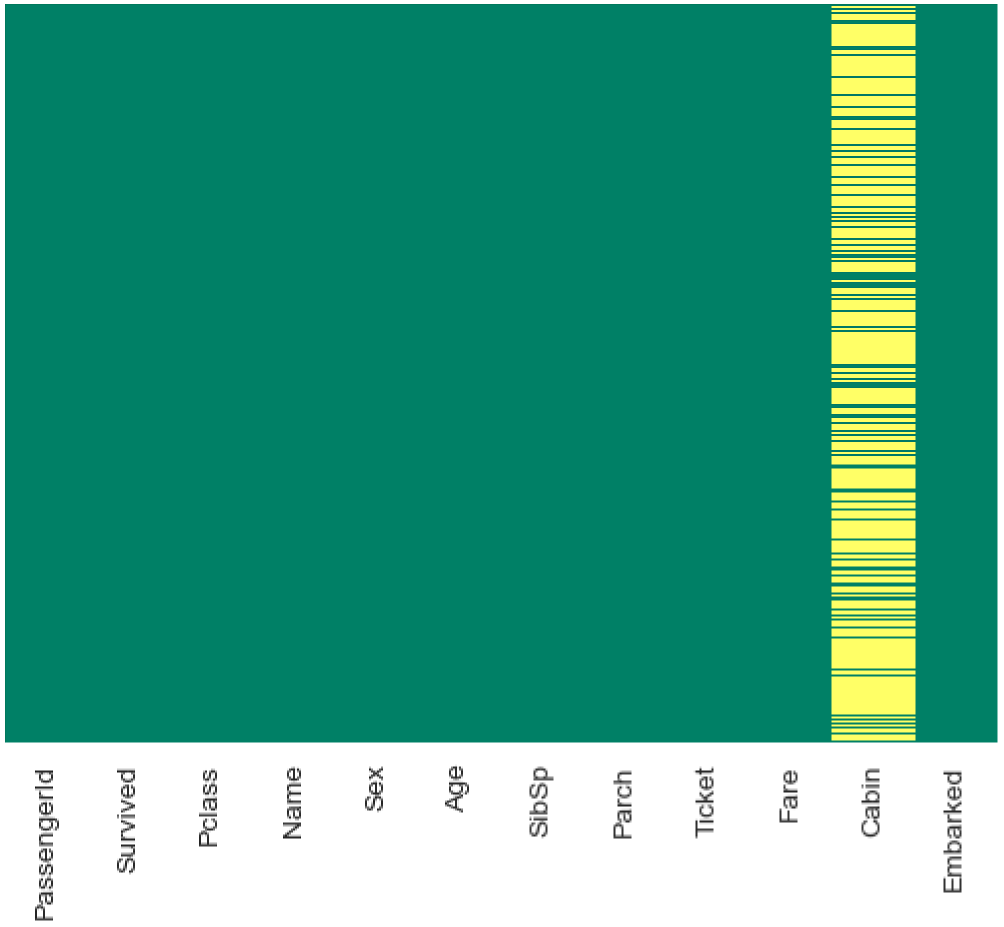
In [34]:
```
def impute_age(cols):
    Age= cols[0]
    Pclass= cols[1]

    if pd.isnull(Age):
        if Pclass == 1:
            return 37
        elif Pclass==2:
            return 29
        else:
            return 24
    else:
        return Age
```

In [36]:
```
train['Age'] = train[['Age','Pclass']].apply(impute_age,axis=1)
#axis 1 means through out the whole column
and # axis 0 means throughout the whole row
```

In [44]:
```
sns.heatmap(train.isnull(),yticklabels=False,cbar=False,cmap='summer')
```

Out[44]:
```
<Axes: >
```



In [45]:
```
train.drop("Cabin",axis=1,inplace=True)
```

In [46]:
```
train.head(10)
```

Out[46]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Embarked
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	S
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	S
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	S
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	S
5	6	0	3	Moran, Mr. James	male	24.0	0	0	330877	8.4583	Q
6	7	0	1	McCarthy, Mr. Timothy J	male	54.0	0	0	17463	51.8625	S
7	8	0	3	Palsson, Master. Gosta Leonard	male	2.0	3	1	349909	21.0750	S
8	9	1	3	Johnson, Mrs. Oscar W (Elisabeth Vilhelmina Berg)	female	27.0	0	2	347742	11.1333	S
9	10	1	2	Nasser, Mrs. Nicholas (Adele Achem)	female	14.0	1	0	237736	30.0708	C

In [47]:
```
train.dropna(inplace=True)
```

In [48]:
```
train.head(12)
```

Out[48]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Embarked
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	S
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	S
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	S
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	S
5	6	0	3	Moran, Mr. James	male	24.0	0	0	330877	8.4583	Q
6	7	0	1	McCarthy, Mr. Timothy J	male	54.0	0	0	17463	51.8625	S
7	8	0	3	Palsson, Master. Gosta Leonard	male	2.0	3	1	349909	21.0750	S
8	9	1	3	Johnson, Mrs. Oscar W (Elisabeth Vilhelmina Berg)	female	27.0	0	2	347742	11.1333	S
9	10	1	2	Nasser, Mrs. Nicholas (Adele Achem)	female	14.0	1	0	237736	30.0708	C
10	11	1	3	Sandstrom, Miss. Marguerite Rut	female	4.0	1	1	PP 9549	16.7000	S
11	12	1	1	Bonnell, Miss. Elizabeth	female	58.0	0	0	113783	26.5500	S

In [49]:
```
train.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 889 entries, 0 to 890
Data columns (total 11 columns):
 #   Column       Non-Null Count  Dtype  
---  ------       --------------  -----  
 0   PassengerId  889 non-null    int64  
 1   Survived     889 non-null    int64  
 2   Pclass       889 non-null    int64  
 3   Name         889 non-null    object 
 4   Sex          889 non-null    object 
 5   Age          889 non-null    float64
 6   SibSp        889 non-null    int64  
 7   Parch        889 non-null    int64  
 8   Ticket       889 non-null    object 
 9   Fare         889 non-null    float64
 10  Embarked     889 non-null    object 
dtypes: float64(2), int64(5), object(4)
memory usage: 83.3+ KB
```

In [51]:
```
pd.get_dummies(train['Embarked'],drop_first=True).head()
```

Out[51]:

	Q	S
0	0	1
1	0	0
2	0	1
3	0	1
4	0	1

In [55]:
```
sex=pd.get_dummies(train['Sex'],drop_first=True)
embark=pd.get_dummies(train['Embarked'],drop_first=True)
```

In [56]:
```
train.drop(['Sex','Embarked','Name','Ticket'],axis=1,inplace=True)
```

In [57]:
```
train.head()
```

Out[57]:

	PassengerId	Survived	Pclass	Age	SibSp	Parch	Fare
0	1	0	3	22.0	1	0	7.2500
1	2	1	1	38.0	1	0	71.2833
2	3	1	3	26.0	0	0	7.9250
3	4	1	1	35.0	1	0	53.1000
4	5	0	3	35.0	0	0	8.0500

In [58]:
```
 train['Survived'].head()
```

Out[58]:
```
0    0
1    1
2    1
3    1
4    0
Name: Survived, dtype: int64
```

In [59]:
```
train=pd.concat([train,sex,embark],axis=1)
```

In [60]:
```
train.head(10)
```

Out[60]:

	PassengerId	Survived	Pclass	Age	SibSp	Parch	Fare	male	Q	S
0	1	0	3	22.0	1	0	7.2500	1	0	1
1	2	1	1	38.0	1	0	71.2833	0	0	0
2	3	1	3	26.0	0	0	7.9250	0	0	1
3	4	1	1	35.0	1	0	53.1000	0	0	1
4	5	0	3	35.0	0	0	8.0500	1	0	1
5	6	0	3	24.0	0	0	8.4583	1	1	0
6	7	0	1	54.0	0	0	51.8625	1	0	1
7	8	0	3	2.0	3	1	21.0750	1	0	1
8	9	1	3	27.0	0	2	11.1333	0	0	1
9	10	1	2	14.0	1	0	30.0708	0	0	0

In [ ]:
```
#the data is ready now have to divide the data in dependent and independent features, the survive column is dependent
```

BUILDING A LOGISTIC REGRESSION MODEL

```
LET'S  START BY SPLITTING THE DATA IN TWO PART TRAIN.CSV AND TEST.CSV
```

[TRAIN TEST SPLIT]

In [61]:
```
train.drop('Survived',axis=1).head()
```

Out[61]:

	PassengerId	Pclass	Age	SibSp	Parch	Fare	male	Q	S
0	1	3	22.0	1	0	7.2500	1	0	1
1	2	1	38.0	1	0	71.2833	0	0	0
2	3	3	26.0	0	0	7.9250	0	0	1
3	4	1	35.0	1	0	53.1000	0	0	1
4	5	3	35.0	0	0	8.0500	1	0	1

In [62]:
```
train['Survived'].head()
```

Out[62]:
```
0    0
1    1
2    1
3    1
4    0
Name: Survived, dtype: int64
```

In [63]:
```
from sklearn.model_selection import train_test_split
```

In [64]:
```
x_train,x_test,y_train,y_test= train_test_split(train.drop('Survived',axis=1),train['Survived'],test_size=0.30,random_
```

Training and Predicting

In [65]:
```
from sklearn.linear_model import LogisticRegression
```

In [66]:
```
logmodel=LogisticRegression()
logmodel.fit(x_train,y_train)
```

```
C:\Users\User\anaconda3\lib\site-packages\sklearn\linear_model\_logistic.py:458: ConvergenceWarning: lbfgs failed to
converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html (https://scikit-learn.org/stable/modules/preprocessin
g.html)
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression (https://scikit-learn.org/stable/mo
dules/linear_model.html#logistic-regression)
  n_iter_i = _check_optimize_result(
```

Out[66]:
```
LogisticRegression()
```

In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

In [67]:
```
predictions = logmodel.predict(x_test)
```

In [68]:
```
from sklearn.metrics import confusion_matrix
```

In [69]:
```
accuracy= confusion_matrix(y_test,predictions)
```

In [70]:
```
accuracy
```

Out[70]:
```
array([[148,  15],
       [ 39,  65]], dtype=int64)
```

In [71]:
```
from sklearn.metrics import accuracy_score
```

In [72]:
```
accuracy = accuracy_score(y_test,predictions)
accuracy
```

Out[72]:
```
0.797752808988764
```

In [73]:
```
predictions
```

Out[73]:
```
array([0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 1,
       1, 0, 1, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 1, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 1, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1,
       0, 1, 1, 1, 0, 0, 0, 1, 1, 0, 0, 1, 0, 1, 0, 0, 1, 0, 1, 0, 0, 0,
       0, 0, 1, 1, 0, 1, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 1, 0, 1, 1, 0, 0,
       1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 1, 1, 1,
       0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0,
       0, 0, 1, 0, 1, 0, 0, 1, 0, 1, 1, 0, 0, 0, 0, 1, 1, 0, 0, 0, 1, 0,
       0, 1, 0, 1, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 1, 0, 0, 0,
       1, 0, 0, 0, 1, 0, 0, 1, 0, 0, 1, 0, 0, 0, 1, 1, 1, 0, 1, 0, 0, 0,
       0, 0, 1, 0, 0, 0, 1, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0,
       0, 1, 1], dtype=int64)
```

In [ ]: