Assignment 18

Arsh Patial

Predict the survival of passenger from given dataset using Logistic Regression.

Note:

1. Clean the Testing Set as steps did for Training Set

2. Does not consider Name and Ticket. Skip also in Testing set.

3. Apply model in Training set and predict in Testing set.

4. Make column i.e Survival in Testing set to store the predicted values

In [342]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
```

In [343]:
```
train=pd.read_csv("titanic_train.csv")
```

In [344]:
```
train.head()
```

Out[344]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Ca
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	N
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	N
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	N

In [345]:
```
train.info()
```

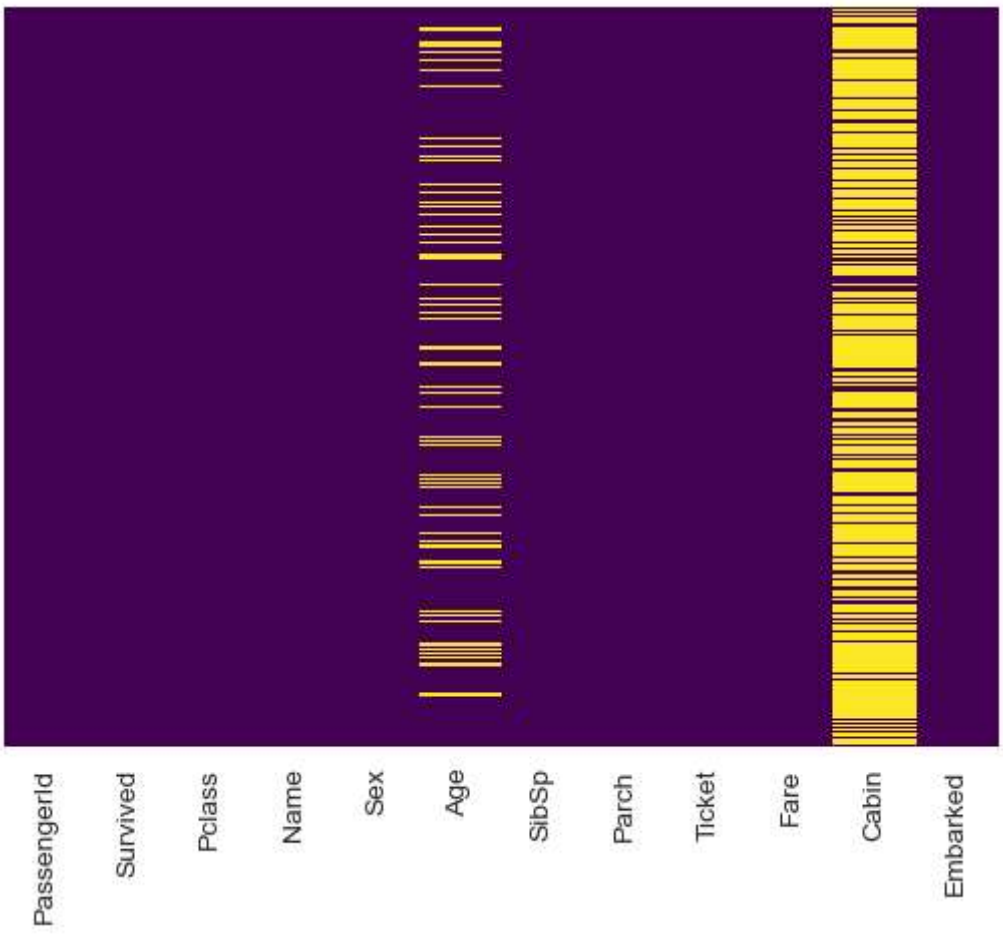
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 891 entries, 0 to 890
Data columns (total 12 columns):
 #   Column       Non-Null Count  Dtype  
---  ------       --------------  -----  
 0   PassengerId  891 non-null    int64  
 1   Survived     891 non-null    int64  
 2   Pclass       891 non-null    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 [346]:
```
train.isnull().sum()
```

Out[346]:
```
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
```

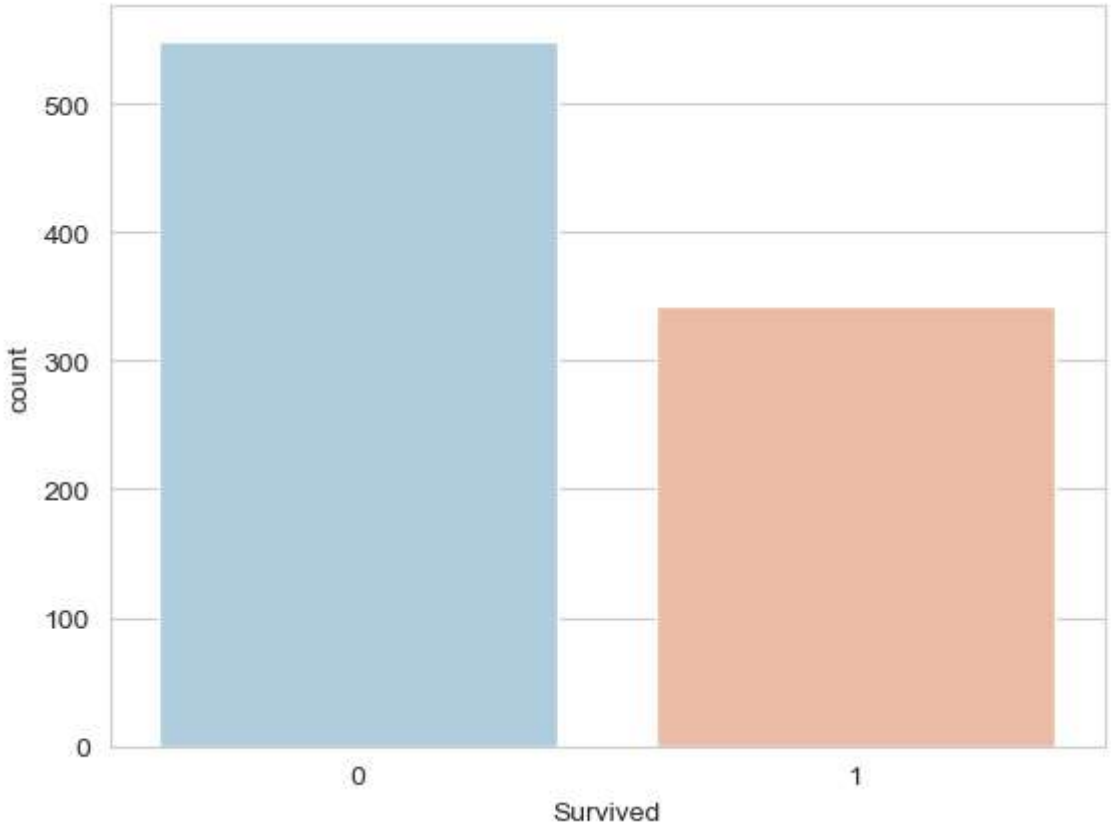
In [347]:
```
sns.heatmap(train.isnull(),yticklabels=False,cbar=False,cmap='viridis')
```

Out[347]:
```
<AxesSubplot:>
```



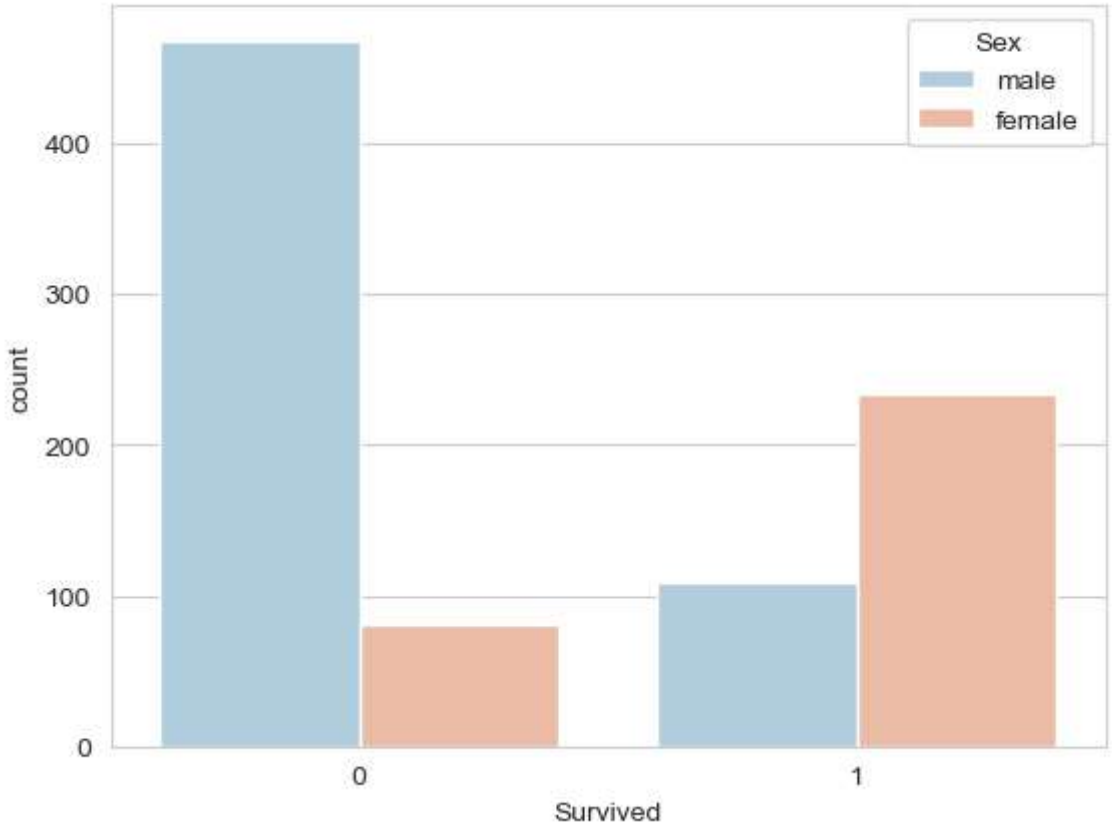
In [348]:
```
sns.set_style('whitegrid')
sns.countplot(x='Survived',data=train,palette='RdBu_r')
```

Out[348]: <AxesSubplot:xlabel='Survived', ylabel='count'>



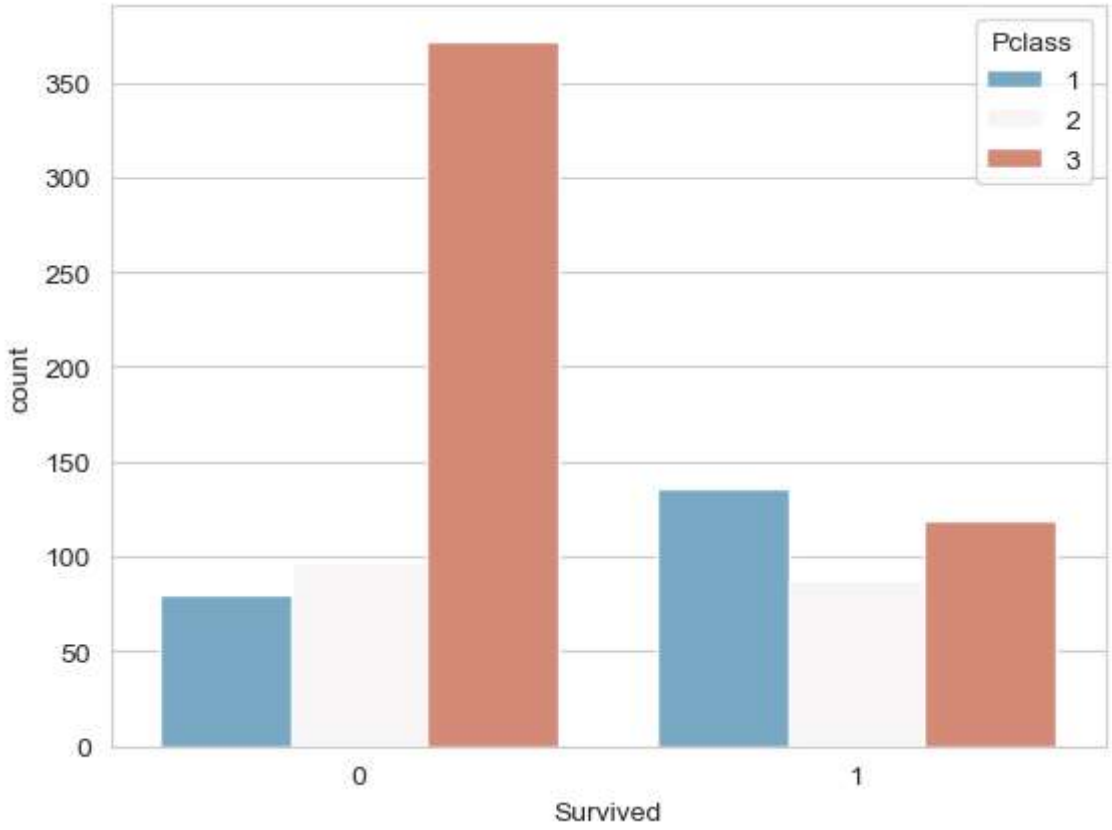
In [349]:
```
sns.countplot(x='Survived',hue='Sex',data=train,palette='RdBu_r')
```

Out[349]: <AxesSubplot:xlabel='Survived', ylabel='count'>



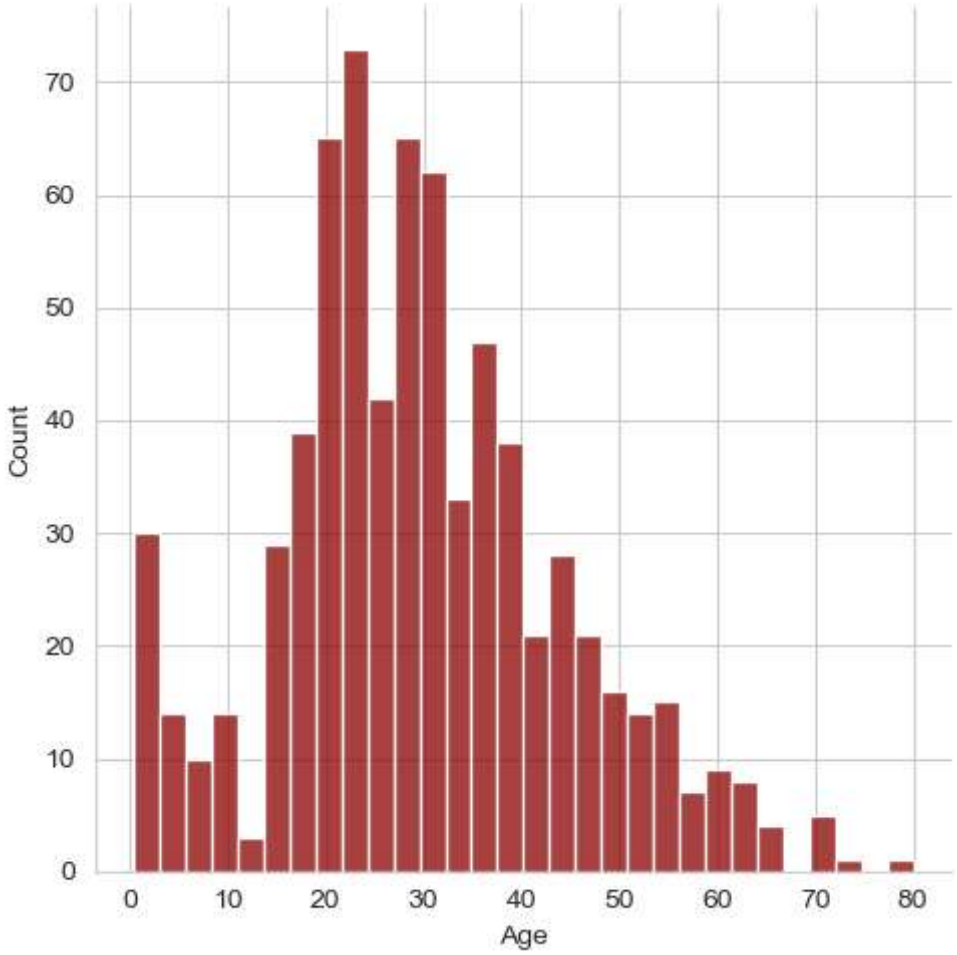
In [350]:
```
sns.set_style('whitegrid')
sns.countplot(x='Survived',hue='Pclass',data=train,palette='RdBu_r')
```

Out[350]: <AxesSubplot:xlabel='Survived', ylabel='count'>



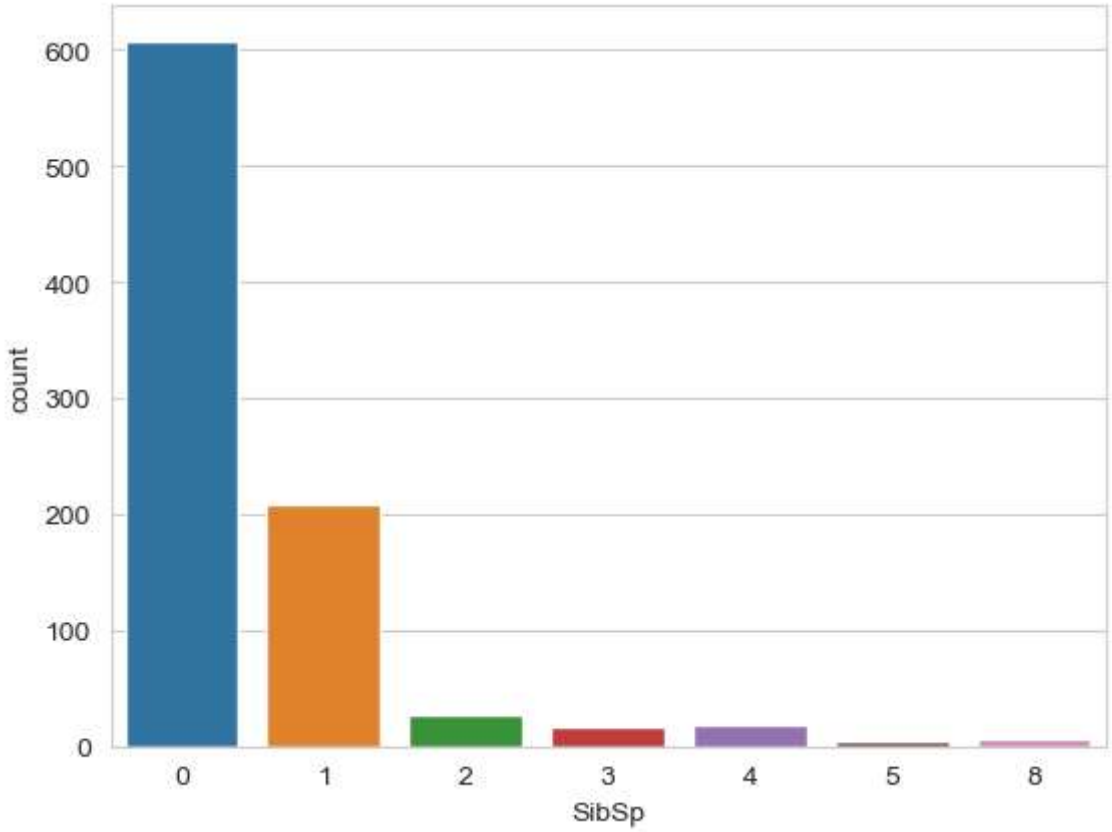
In [351]:
```
sns.displot(train['Age'].dropna(),kde=False,color='darkred',bins=30)
```

Out[351]:
```
<seaborn.axisgrid.FacetGrid at 0x2a82f708790>
```



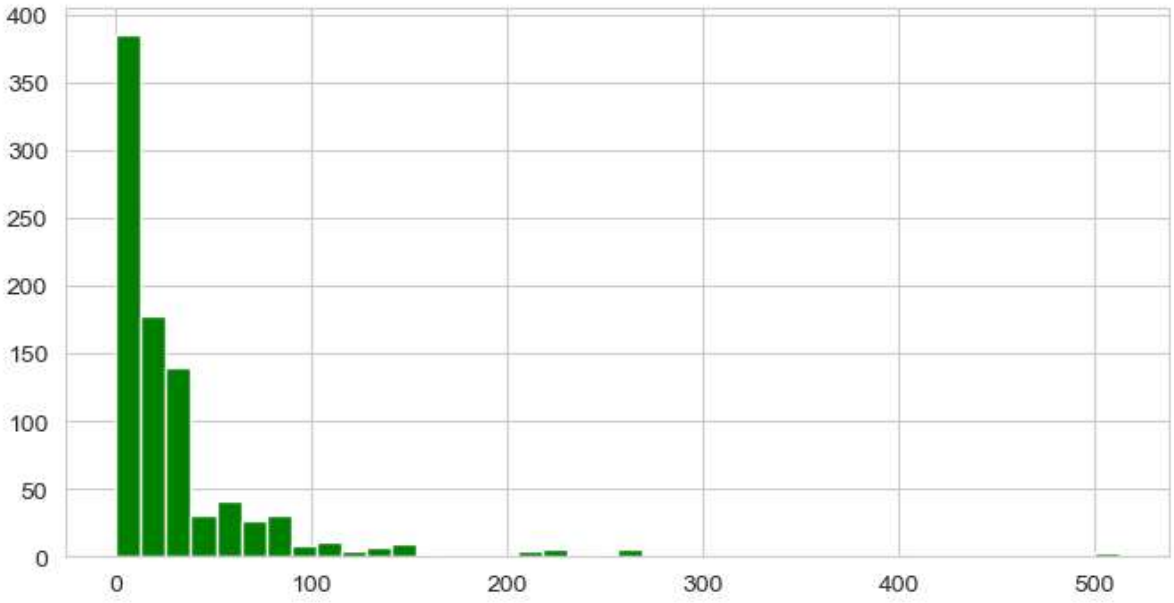
In [352]:
```
sns.countplot(x='SibSp',data=train)
```

Out[352]: <AxesSubplot:xlabel='SibSp', ylabel='count'>



In [353]:
```
train['Fare'].hist(color='green',bins=40,figsize=(8,4))
```

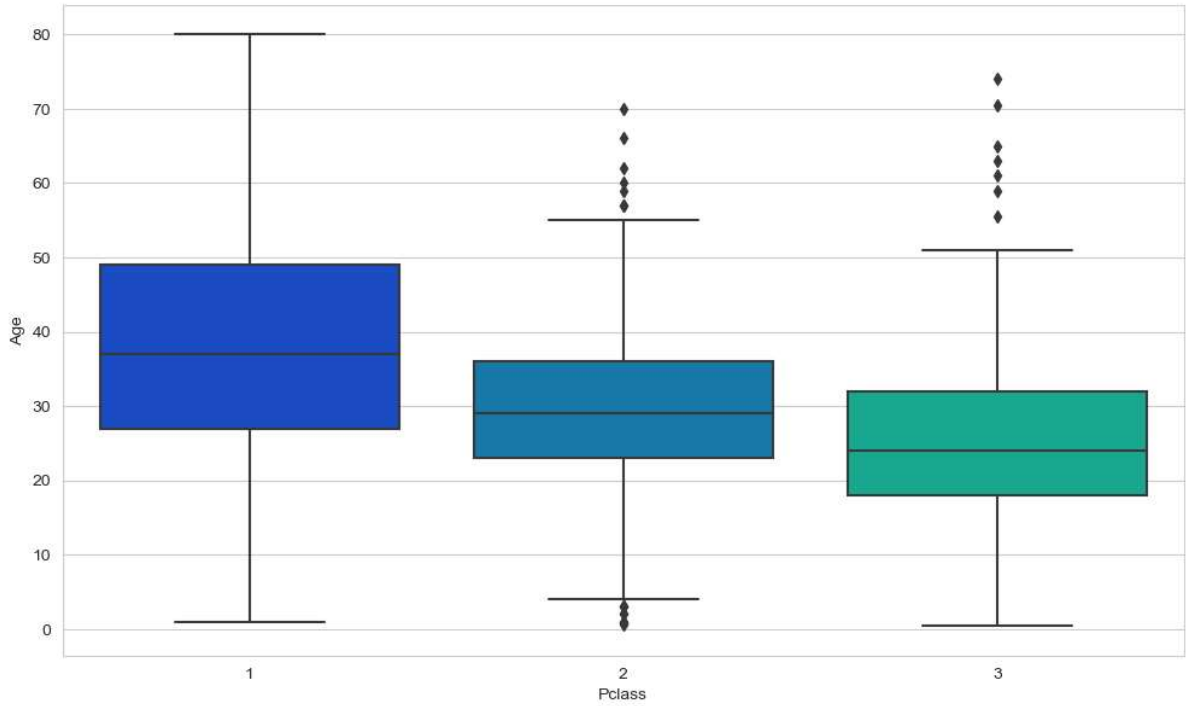
Out[353]: <AxesSubplot:>



In [354]:
```
#DataCleaning
```

In [355]:
```
plt.figure(figsize=(12,7))
sns.boxplot(x='Pclass',y='Age',data=train,palette='winter')
```

Out[355]: <AxesSubplot:xlabel='Pclass', ylabel='Age'>



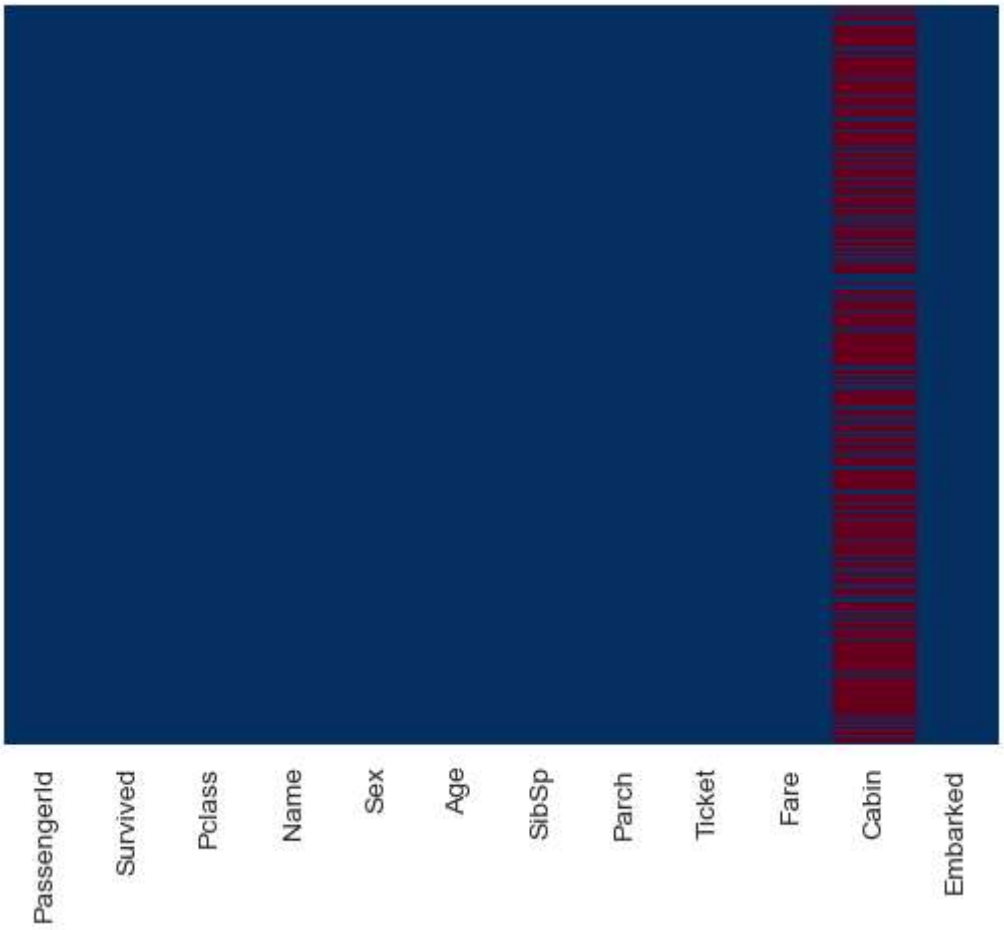
In [356]:
```
def add_age(cols):
    Age=cols[0]
    Pclass=cols[1]

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

In [357]:
```
train['Age']=train[['Age','Pclass']].apply(add_age,axis=1)
```

In [358]:
```
sns.heatmap(train.isna(),yticklabels=False,cbar=False,cmap='RdBu_r')
```

Out[358]: <AxesSubplot:>



In [359]:
```
train.drop('Cabin',axis=1,inplace=True)
```

In [360]:
```
train.head(2)
```

Out[360]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Embark
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	

In [361]:
```
train.dropna(inplace=True)
```

In [362]:
```
#Converting Categorical Feature
```

In [363]:
```
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 [364]:
```
train=pd.get_dummies(train,columns=['Sex','Embarked'])
```

In [365]:
```
train.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 889 entries, 0 to 890
Data columns (total 14 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   Age          889 non-null    float64
 5   SibSp        889 non-null    int64  
 6   Parch        889 non-null    int64  
 7   Ticket       889 non-null    object 
 8   Fare         889 non-null    float64
 9   Sex_female   889 non-null    uint8  
 10  Sex_male     889 non-null    uint8  
 11  Embarked_C   889 non-null    uint8  
 12  Embarked_Q   889 non-null    uint8  
 13  Embarked_S   889 non-null    uint8  
dtypes: float64(2), int64(5), object(2), uint8(5)
memory usage: 73.8+ KB
```

For test data

In [366]:
```
test=pd.read_csv("titanic_test.csv")
```

In [367]:
```
test.head()
```

Out[367]:

	PassengerId	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embark
0	892	3	Kelly, Mr. James	male	34.5	0	0	330911	7.8292	NaN	
1	893	3	Wilkes, Mrs. James (Ellen Needs)	female	47.0	1	0	363272	7.0000	NaN	
2	894	2	Myles, Mr. Thomas Francis	male	62.0	0	0	240276	9.6875	NaN	
3	895	3	Wirz, Mr. Albert	male	27.0	0	0	315154	8.6625	NaN	
4	896	3	Hirvonen, Mrs. Alexander (Helga E Lindqvist)	female	22.0	1	1	3101298	12.2875	NaN	

In [368]:
```
test.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 418 entries, 0 to 417
Data columns (total 11 columns):
 #   Column       Non-Null Count  Dtype  
---  ------       --------------  -----  
 0   PassengerId  418 non-null    int64  
 1   Pclass       418 non-null    int64  
 2   Name         418 non-null    object 
 3   Sex          418 non-null    object 
 4   Age          332 non-null    float64
 5   SibSp        418 non-null    int64  
 6   Parch        418 non-null    int64  
 7   Ticket       418 non-null    object 
 8   Fare         417 non-null    float64
 9   Cabin        91 non-null     object 
 10  Embarked     418 non-null    object 
dtypes: float64(2), int64(4), object(5)
memory usage: 36.0+ KB
```

In [369]:
```
test.describe()
```

Out[369]:

	PassengerId	Pclass	Age	SibSp	Parch	Fare
count	418.000000	418.000000	332.000000	418.000000	418.000000	417.000000
mean	1100.500000	2.265550	30.272590	0.447368	0.392344	35.627188
std	120.810458	0.841838	14.181209	0.896760	0.981429	55.907576
min	892.000000	1.000000	0.170000	0.000000	0.000000	0.000000
25%	996.250000	1.000000	21.000000	0.000000	0.000000	7.895800
50%	1100.500000	3.000000	27.000000	0.000000	0.000000	14.454200
75%	1204.750000	3.000000	39.000000	1.000000	0.000000	31.500000
max	1309.000000	3.000000	76.000000	8.000000	9.000000	512.329200

In [370]:
```
test.isnull().sum()
```

Out[370]:
```
PassengerId      0
Pclass           0
Name             0
Sex              0
Age             86
SibSp            0
Parch            0
Ticket           0
Fare             1
Cabin          327
Embarked         0
dtype: int64
```

data cleaning of test set

In [371]:
```
test['Age']=test[['Age','Pclass']].apply(add_age,axis=1)
```

In [372]:
```
test.drop('Cabin',axis=1,inplace=True)
```

In [373]:
```
test.head()
```

Out[373]:

	PassengerId	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Embarked
0	892	3	Kelly, Mr. James	male	34.5	0	0	330911	7.8292	Q
1	893	3	Wilkes, Mrs. James (Ellen Needs)	female	47.0	1	0	363272	7.0000	S
2	894	2	Myles, Mr. Thomas Francis	male	62.0	0	0	240276	9.6875	Q
3	895	3	Wirz, Mr. Albert	male	27.0	0	0	315154	8.6625	S
4	896	3	Hirvonen, Mrs. Alexander (Helga E Lindqvist)	female	22.0	1	1	3101298	12.2875	S

In [374]:
```
test.isnull().sum()
```

Out[374]:
```
PassengerId    0
Pclass         0
Name           0
Sex            0
Age            0
SibSp          0
Parch          0
Ticket         0
Fare           1
Embarked       0
dtype: int64
```

In [375]:
```
test.dropna(inplace=True)
```

In [376]:
```
test.isnull().sum()
```

Out[376]:
```
PassengerId    0
Pclass         0
Name           0
Sex            0
Age            0
SibSp          0
Parch          0
Ticket         0
Fare           0
Embarked       0
dtype: int64
```

In [377]:
```
test=pd.get_dummies(test,columns=['Sex','Embarked'])
```

In [378]:
```
test.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 417 entries, 0 to 417
Data columns (total 13 columns):
 #   Column       Non-Null Count  Dtype  
---  ------       --------------  -----  
 0   PassengerId  417 non-null    int64  
 1   Pclass       417 non-null    int64  
 2   Name         417 non-null    object 
 3   Age          417 non-null    float64
 4   SibSp        417 non-null    int64  
 5   Parch        417 non-null    int64  
 6   Ticket       417 non-null    object 
 7   Fare         417 non-null    float64
 8   Sex_female   417 non-null    uint8  
 9   Sex_male     417 non-null    uint8  
 10  Embarked_C   417 non-null    uint8  
 11  Embarked_Q   417 non-null    uint8  
 12  Embarked_S   417 non-null    uint8  
dtypes: float64(2), int64(4), object(2), uint8(5)
memory usage: 31.4+ KB
```

In [379]:
```
test.columns
```

Out[379]:
```
Index(['PassengerId', 'Pclass', 'Name', 'Age', 'SibSp', 'Parch', 'Ticket',
       'Fare', 'Sex_female', 'Sex_male', 'Embarked_C', 'Embarked_Q',
       'Embarked_S'],
      dtype='object')
```

importing the necessary libraries

In [400]:
```
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import LinearRegression
```

In [425]:
```
x_train=train[['Pclass', 'Age', 'SibSp', 'Parch','Fare', 'Sex_female', 'Sex_ma
        'Embarked_C', 'Embarked_Q',
       'Embarked_S']]
x_test=test[['Pclass', 'Age', 'SibSp', 'Parch','Fare', 'Sex_female', 'Sex_male
        'Embarked_C', 'Embarked_Q',
       'Embarked_S']]
y_train=train['Survived']
```

In [ ]:

In [426]:
```
model=LogisticRegression(max_iter=1000) # creating an object model in class li
```

In [427]:
```
model.fit(x_train,y_train) # fitting model on x train and y train
```

Out[427]:
```
LogisticRegression(max_iter=1000)
```

In [428]:
```
y_pred=model.predict(x_test) # predicting data on unseen x train
```

In [429]:
```
, # plotting y pred and y_train.. since we have no data on y_test./ its only op
```

Out[429]:
```
('#',
 'plotting',
 'y',
 'pred',
 'and',
 'y_train..',
 'since',
 'we',
 'have',
 'no',
 'data',
 'on',
 'y_test./',
 'its',
 'only',
 'option')
```

In [430]:
```
y_train  # this y training data
```

Out[430]:
```
0      0
1      1
2      1
3      1
4      0
      ..
886    0
887    1
888    0
889    1
890    0
Name: Survived, Length: 889, dtype: int64
```

the assignment is complete ..as we added survival column in x_test

In [431]:
```
x_test['survived']= y_pred
```

```
C:\Users\user\AppData\Local\Temp\ipykernel_8124\3310133094.py:1: SettingWithC
opyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/s
table/user_guide/indexing.html#returning-a-view-versus-a-copy (https://panda
s.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-ver
sus-a-copy)
  x_test['survived']= y_pred
```

In [432]:
```
x_test
```

Out[432]:

	Pclass	Age	SibSp	Parch	Fare	Sex_female	Sex_male	Embarked_C	Embarked_Q	E
0	3	34.5	0	0	7.8292	0	1	0	1	
1	3	47.0	1	0	7.0000	1	0	0	0	
2	2	62.0	0	0	9.6875	0	1	0	1	
3	3	27.0	0	0	8.6625	0	1	0	0	
4	3	22.0	1	1	12.2875	1	0	0	0	
...	...	...	...	...	...	...	...	...	...	
413	3	24.0	0	0	8.0500	0	1	0	0	
414	1	39.0	0	0	108.9000	1	0	1	0	
415	3	38.5	0	0	7.2500	0	1	0	0	
416	3	24.0	0	0	8.0500	0	1	0	0	
417	3	24.0	1	1	22.3583	0	1	1	0	

417 rows × 11 columns

In [433]:
```
y_pred
```

Out[433]:
```
array([0, 0, 0, 0, 1, 0, 1, 0, 1, 0, 0, 0, 1, 0, 1, 1, 0, 0, 1, 1, 0, 0,
       1, 1, 1, 0, 1, 0, 0, 0, 0, 0, 0, 1, 1, 0, 1, 1, 0, 0, 0, 0, 0, 1,
       1, 0, 0, 0, 1, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 1, 1,
       1, 0, 1, 1, 1, 0, 1, 1, 1, 1, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 1, 1,
       1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0,
       0, 1, 1, 1, 1, 0, 0, 1, 1, 1, 1, 0, 1, 0, 0, 1, 0, 1, 0, 0, 0, 0,
       1, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0,
       0, 1, 1, 0, 1, 1, 0, 1, 0, 0, 1, 0, 0, 1, 1, 0, 0, 0, 0, 0, 1, 1,
       0, 1, 1, 0, 0, 1, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0,
       1, 1, 0, 0, 1, 0, 1, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 1,
       0, 1, 0, 1, 1, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 0,
       0, 0, 0, 1, 0, 1, 1, 1, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 1, 0,
       0, 0, 0, 1, 0, 0, 0, 1, 1, 0, 1, 0, 0, 0, 0, 1, 0, 1, 1, 1, 0, 0,
       1, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0,
       0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 1,
       0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 1, 0, 1, 1, 0, 0,
       0, 1, 0, 1, 0, 0, 0, 0, 1, 1, 0, 1, 0, 0, 1, 1, 0, 0, 1, 0, 0, 1,
       1, 0, 0, 0, 0, 0, 0, 1, 1, 0, 1, 0, 0, 0, 0, 1, 1, 0, 0, 0, 1, 0,
       1, 0, 0, 1, 0, 1, 1, 0, 0, 0, 0, 1, 1, 1, 1, 1, 0, 1, 0, 0, 0],
      dtype=int64)
```

In [ ]:
```
# since we don't have y_test to check , we can use cross validation
```

In [434]:
```
from sklearn.model_selection import cross_val_predict
```

In [435]:
```
from sklearn.metrics import accuracy_score
```

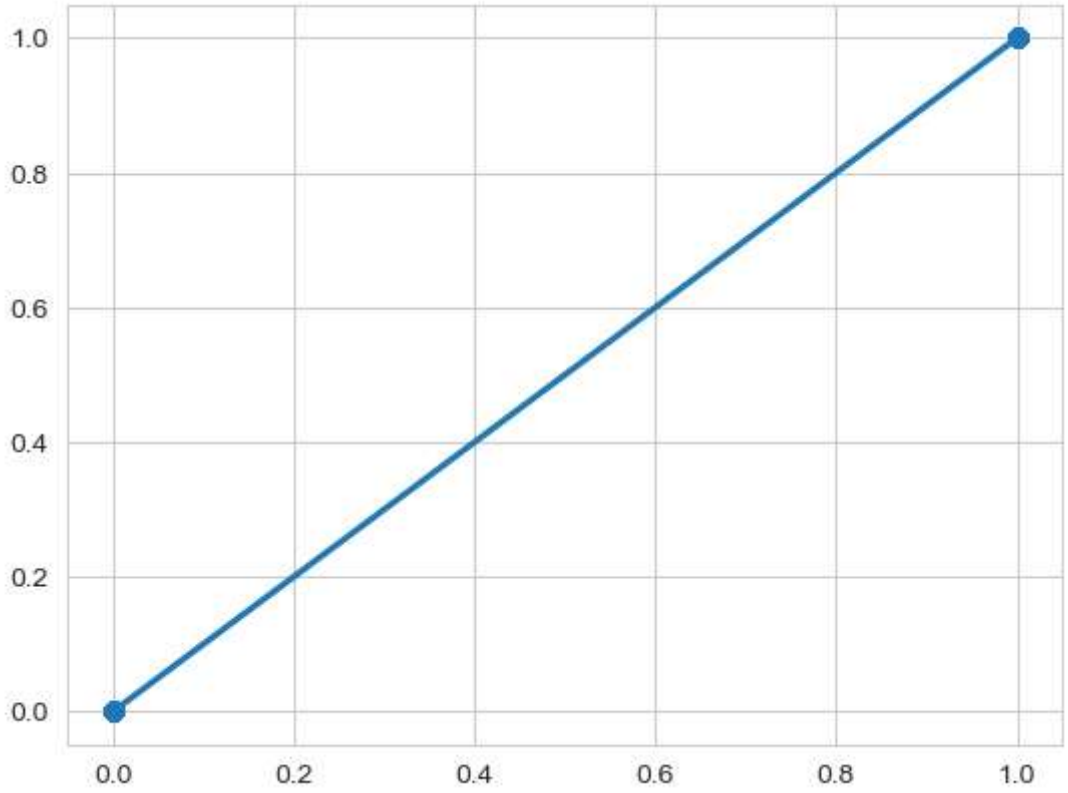
In [446]:
```
y_pred_cross= cross_val_predict(model,x_train,y_train,cv=5)

accuracy= accuracy_score(y_pred,y_pred_cross)
print('accuracy',accuracy)
```

```
accuracy 1.0
```

In [449]:
```
sns.regplot(x=y_pred,y=y_pred_cross)
```

Out[449]: <AxesSubplot:>



In [ ]:

In [ ]:

In [ ]: