DS_MAJOR_AUGUST_DS_08_SPB7

October 4, 2022

1 MAJOR PROJECT

Take any Dataset of your choice , perform EDA (Exploratory Data Analysis) and apply a suitable Classifier, Regressor or Clusterer and calculate the accuracy of the model.

Dataset used = Titanic Train Dataset

Applied LOGISTIC REGRESSION

2 PERFORMING EDA ON Titanic Dataset

[11]:
```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline
```

[12]:
```
df = pd.read_csv('/train.csv')
```

3 UNDERSTANDING THE DATASET

[13]:
```
df.head()
```

[13]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th…	female	38.0	1
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1
4	5	0	3	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
```

[93]:
```
df.tail()
```

[93]:
```
     PassengerId  Survived  Pclass   Age  SibSp  Parch   Fare  male  Q  S
886          887         0       2  27.0      0      0  13.00     1  0  1
887          888         1       1  19.0      0      0  30.00     0  0  1
888          889         0       3  24.0      1      2  23.45     0  0  1
889          890         1       1  26.0      0      0  30.00     1  0  0
890          891         0       3  32.0      0      0   7.75     1  1  0
```

[92]:
```
df.describe
```

[92]:
```
<bound method NDFrame.describe of      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
..           …       …     …   …    …    …       …   … .. ..
886          887         0       2  27.0      0      0  13.0000     1  0  1
887          888         1       1  19.0      0      0  30.0000     0  0  1
888          889         0       3  24.0      1      2  23.4500     0  0  1
889          890         1       1  26.0      0      0  30.0000     1  0  0
890          891         0       3  32.0      0      0   7.7500     1  1  0

[889 rows x 10 columns]>
```

[91]:
```
df.shape
```

[91]:
```
(889, 10)
```

[90]:
```
df.columns
```

[90]:
```
Index(['PassengerId', 'Survived', 'Pclass', 'Age', 'SibSp', 'Parch', 'Fare',
       'male', 'Q', 'S'],
      dtype='object')
```

[89]:
```
df.nunique()
```

[89]:
```
PassengerId    889
Survived         2
Pclass           3
Age             88
SibSp            7
Parch            7
Fare           247
male             2
Q                2
S                2
dtype: int64
```

4 Exploratory Data Analysis

Missing Data

[15]:
```
df.isnull()
```

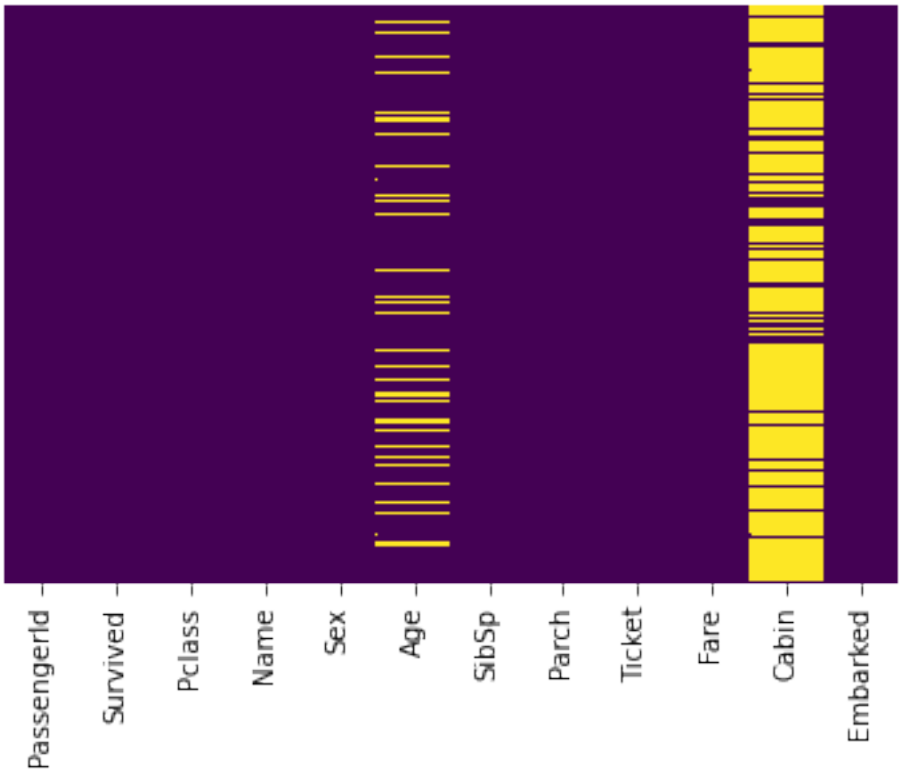
[15]:

	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 x 12 columns]

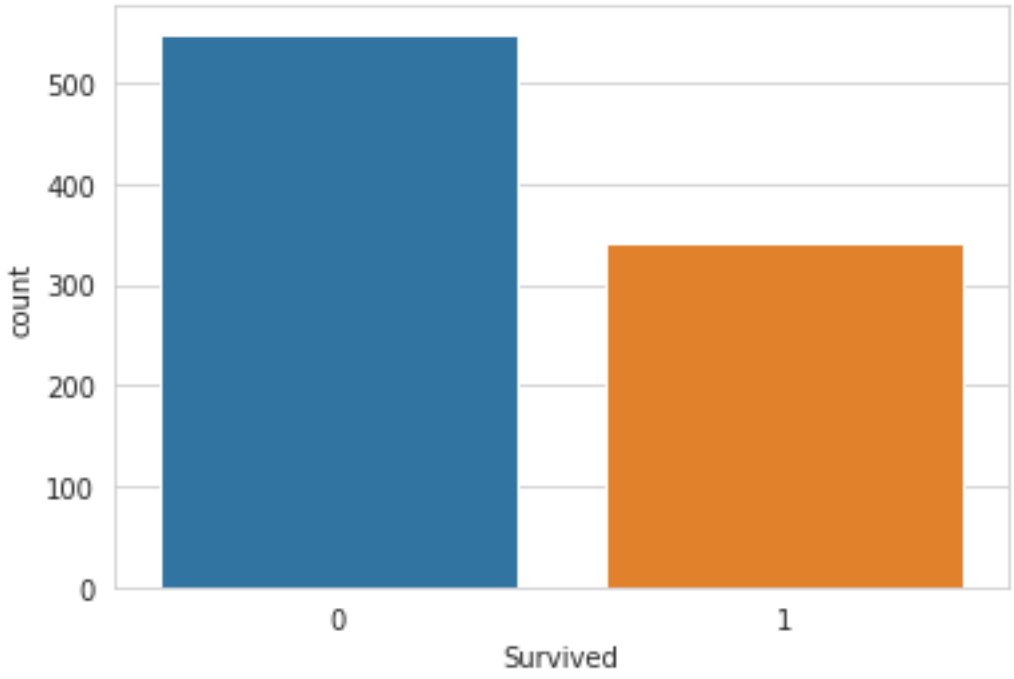
[23]:
```
sns.heatmap(df.isnull(),yticklabels=False,cbar=False,cmap='viridis')
```

[23]: <matplotlib.axes._subplots.AxesSubplot at 0x7f0414df1e50>



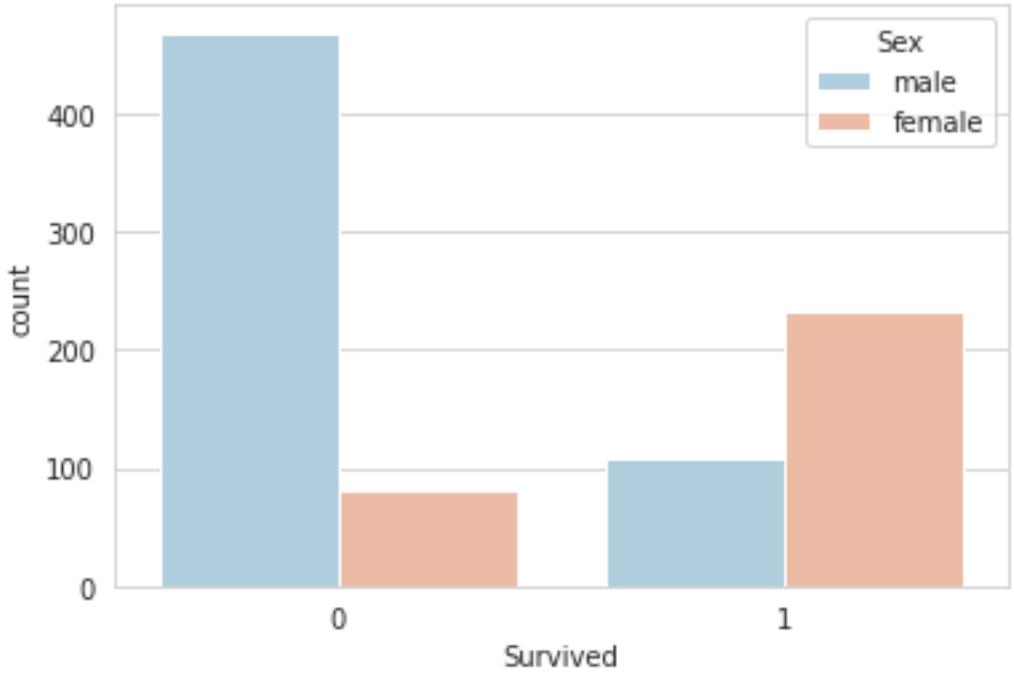
[24]:
```
sns.set_style('whitegrid')
sns.countplot(x='Survived',data=df)
```

[24]: <matplotlib.axes._subplots.AxesSubplot at 0x7f0412508b90>



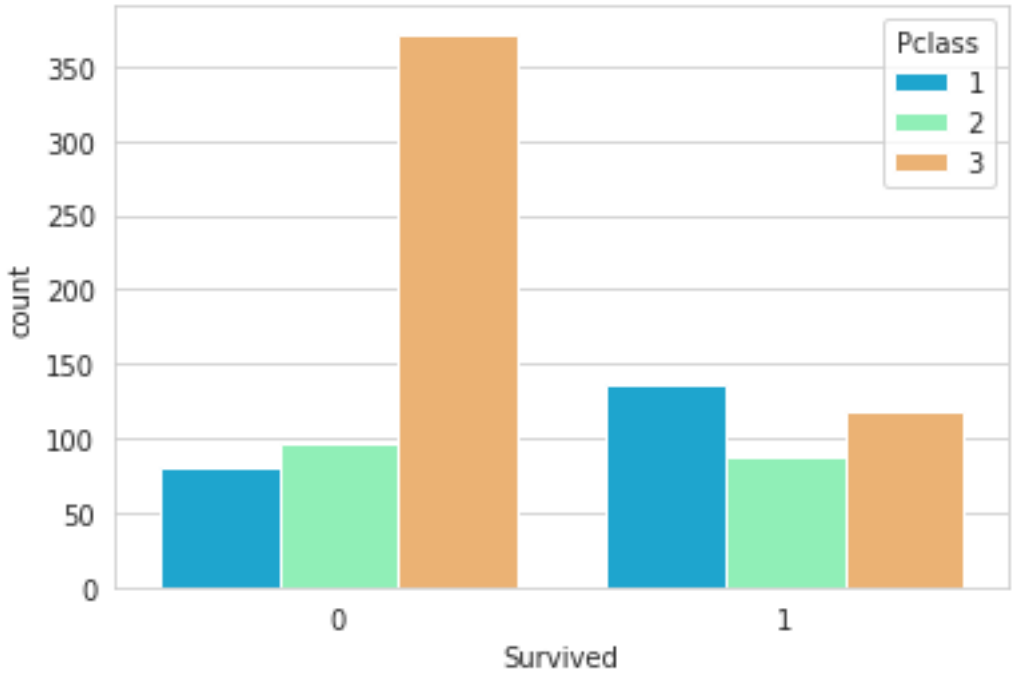
[26]:
```
sns.set_style('whitegrid')
sns.countplot(x='Survived',hue='Sex',data=df,palette='RdBu_r')
```

[26]: <matplotlib.axes._subplots.AxesSubplot at 0x7f0412015ad0>



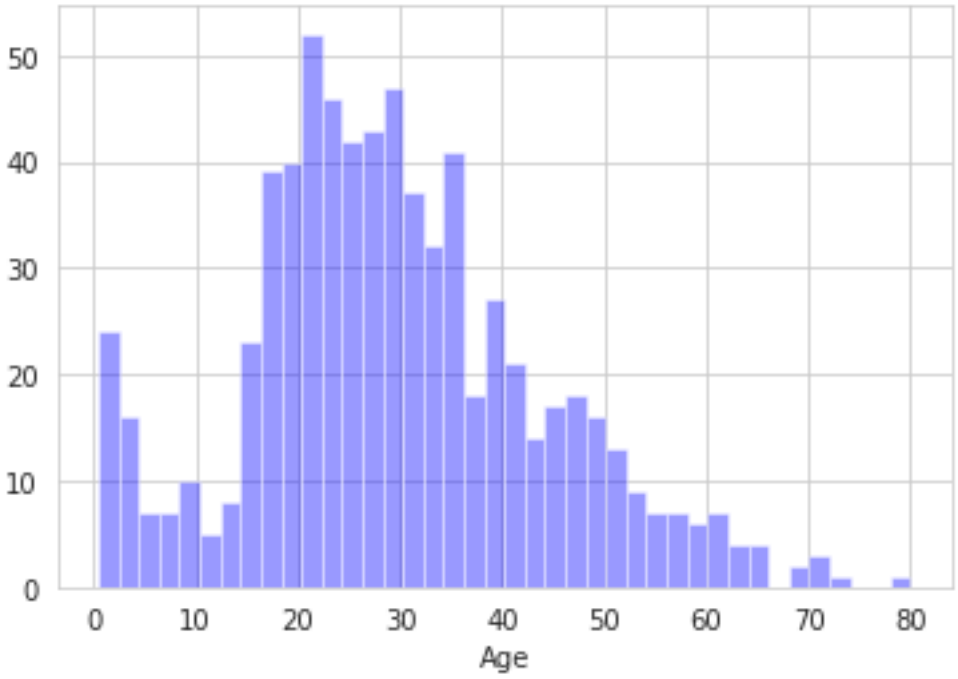
[27]:
```
sns.set_style('whitegrid')
sns.countplot(x='Survived',hue='Pclass',data=df,palette='rainbow')
```

[27]: <matplotlib.axes._subplots.AxesSubplot at 0x7f0411fb8fd0>



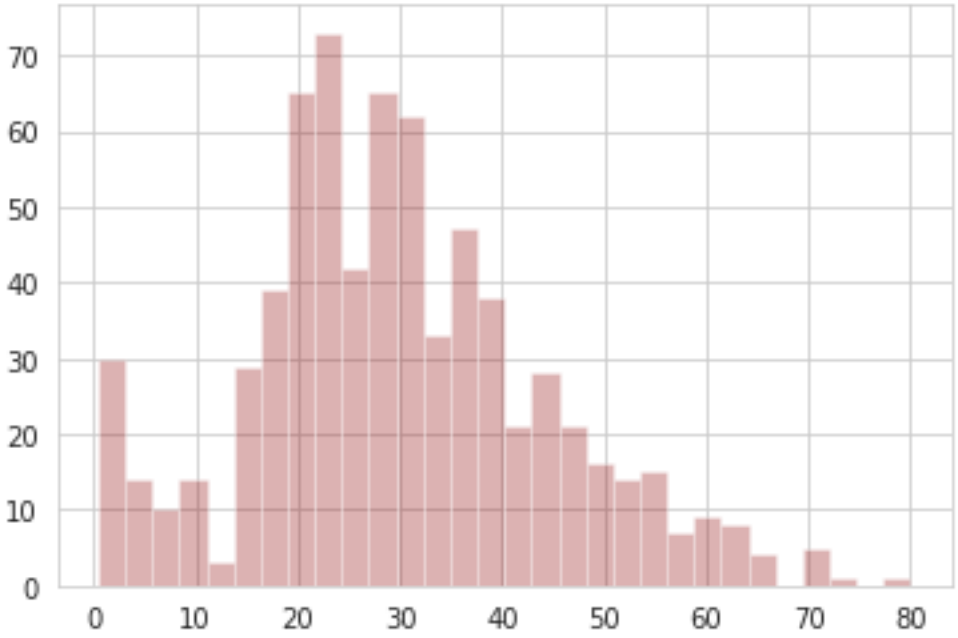
[33]:
```
sns.distplot(df['Age'].dropna(),kde=False,color='blue',bins=40)
```

[33]: <matplotlib.axes._subplots.AxesSubplot at 0x7f0411d0d1d0>



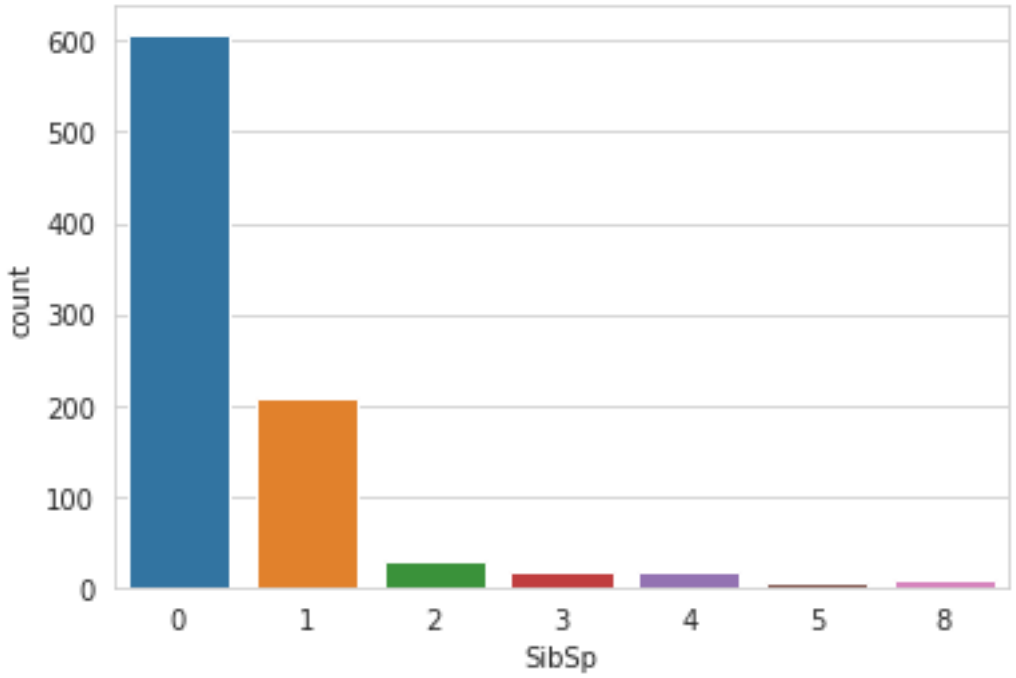
[31]:
```
df['Age'].hist(bins=30,color='darkred',alpha=0.3)
```

[31]: <matplotlib.axes._subplots.AxesSubplot at 0x7f0411fb4950>



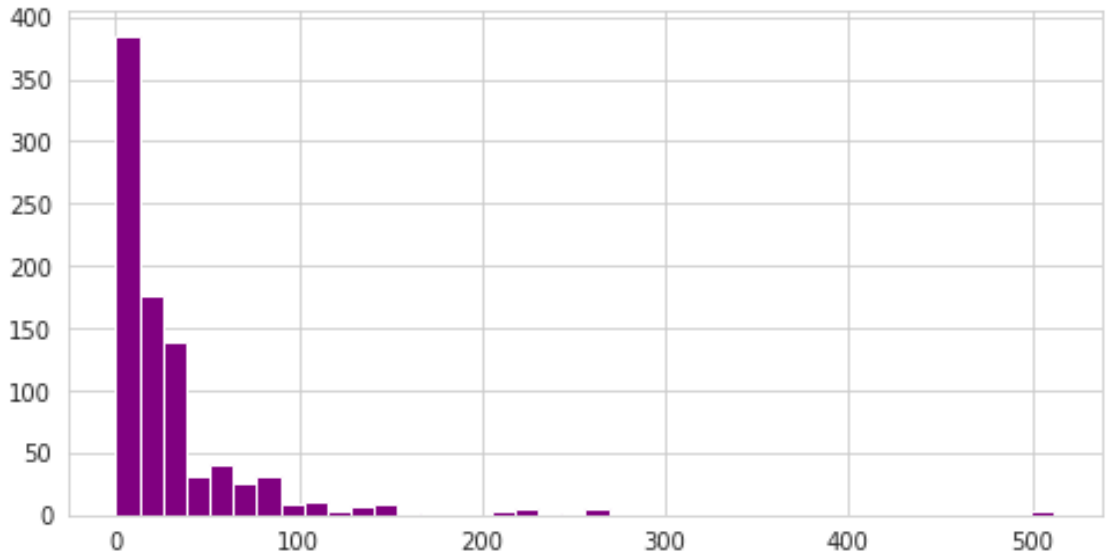
[34]:
```
sns.countplot(x='SibSp',data=df)
```

[34]: <matplotlib.axes._subplots.AxesSubplot at 0x7f0411bb0610>



[37]:
```
df['Fare'].hist(color='purple',bins=40,figsize=(8,4))
```

[37]: <matplotlib.axes._subplots.AxesSubplot at 0x7f04119becd0>



Cufflinks for plots

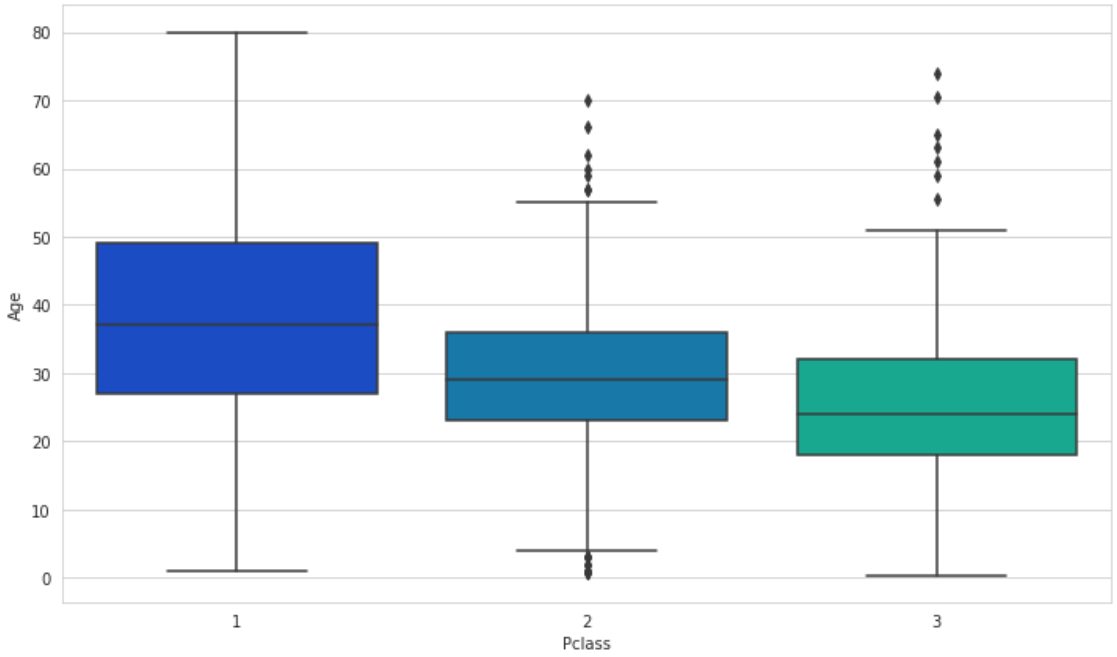
[38]:
```
import cufflinks as cf
cf.go_offline()
```

[39]:
```
df['Fare'].iplot(kind='hist',bins=30,color='green')
```

Data Cleaning

[41]:
```
plt.figure(figsize=(12,7))
sns.boxplot(x='Pclass',y='Age',data=df,palette='winter')
```

[41]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f040b00bc50>
```



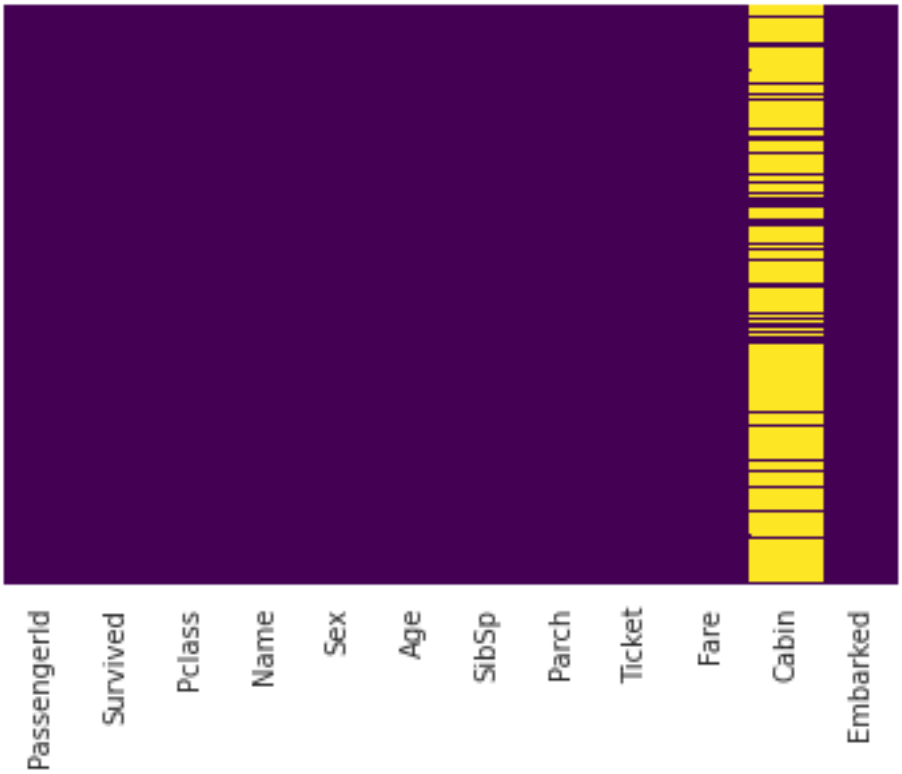
[49]:
```
def impute_age(cols):
  Age = cols[0]
  Pclass = cols[1]

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

```
    return 24
  else:
    return Age
```

[50]:
```
df['Age'] = df[['Age','Pclass']].apply(impute_age,axis=1)
```

[54]:
```
sns.heatmap(df.isnull(),yticklabels=False,cbar=False,cmap='viridis')
```

[54]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f040ab78590>
```



[55]:
```
df.drop('Cabin',axis=1,inplace=True)
```

[56]:
```
df.head()
```

[56]:

	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 Embarked
0      0         A/5 21171   7.2500        S
1      0          PC 17599  71.2833        C
2      0  STON/O2. 3101282   7.9250        S
3      0            113803  53.1000        S
4      0            373450   8.0500        S
```

[57]:
```
df.dropna(inplace=True)
```

Converting Categorical Features

[58]:
```
df.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
```

[59]:
```
pd.get_dummies(df['Embarked'],drop_first=True).head()
```

[59]:
```
   Q  S
0  0  1
1  0  0
2  0  1
3  0  1
4  0  1
```

[62]:
```
sex = pd.get_dummies(df['Sex'],drop_first=True)
embark = pd.get_dummies(df['Embarked'],drop_first =True)
```

[63]:
```
df.drop(['Sex','Embarked','Name','Ticket'],axis=1,inplace=True)
```

[65]:
```
df.head()
```

[65]:

	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

[66]:
```
df = pd.concat([df,sex,embark],axis=1)
```

[67]:
```
df.head()
```

[67]:

	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 Building a LOGISTIC REGRESSION MODEL

Train Test Split

[68]:
```
df.drop('Survived',axis=1).head()
```

[68]:

	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

[69]:
```
df['Survived'].head()
```

[69]:
```
0    0
1    1
2    1
3    1
4    0
Name: Survived, dtype: int64
```

[71]:
```
from sklearn.model_selection import train_test_split
```

[80]:
```
x_train, x_test, y_train, y_test = train_test_split(df.
↪drop('Survived',axis=1),df['Survived'],test_size=0.30,random_state=101)
```

[112]:
```
x_train
```

[112]:

	PassengerId	Pclass	Age	SibSp	Parch	Fare	male	Q	S
807	808	3	18.0	0	0	7.7750	0	0	1
651	652	2	18.0	0	1	23.0000	0	0	1
2	3	3	26.0	0	0	7.9250	0	0	1
690	691	1	31.0	1	0	57.0000	1	0	1
196	197	3	24.0	0	0	7.7500	1	1	0
..	…	…	…	…	…	…	…	..	..
576	577	2	34.0	0	0	13.0000	0	0	1
840	841	3	20.0	0	0	7.9250	1	0	1
338	339	3	45.0	0	0	8.0500	1	0	1
524	525	3	24.0	0	0	7.2292	1	0	0
865	866	2	42.0	0	0	13.0000	0	0	1

[622 rows x 9 columns]

[118]:
```
x_train.values
```

[118]:
```
array([[808.,   3.,  18., …,   0.,   0.,   1.],
       [652.,   2.,  18., …,   0.,   0.,   1.],
       [  3.,   3.,  26., …,   0.,   0.,   1.],
       …,
       [339.,   3.,  45., …,   1.,   0.,   1.],
       [525.,   3.,  24., …,   1.,   0.,   0.],
       [866.,   2.,  42., …,   0.,   0.,   1.]])
```

[111]:
```
y_train
```

[111]:
```
807    0
651    1
2      1
690    1
196    0
      ..
576    1
840    0
338    1
524    0
865    1
Name: Survived, Length: 622, dtype: int64
```

[119]:
```
y_train.values
```

[119]:
```
array([0, 1, 1, 1, 0, 0, 1, 1, 0, 1, 0, 0, 1, 1, 1, 0, 1, 0, 1, 0, 0, 1,
       1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 1,
       1, 1, 0, 0, 1, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0,
       0, 1, 1, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 1, 1, 0, 1, 0, 1, 0, 1, 1,
       1, 1, 0, 1, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 1,
       1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 1, 0, 0, 0, 1, 0, 0,
       0, 0, 0, 0, 1, 0, 0, 1, 1, 0, 1, 0, 1, 1, 0, 0, 1, 0, 0, 1, 1, 0,
       1, 1, 1, 0, 0, 1, 0, 1, 1, 1, 0, 0, 0, 1, 0, 1, 0, 0, 0, 1, 1, 1,
       1, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 0, 0, 0,
       1, 1, 0, 1, 0, 0, 0, 0, 1, 0, 1, 1, 0, 0, 1, 0, 0, 1, 0, 0, 1, 0,
       1, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 1, 0,
       1, 0, 1, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 1,
       0, 1, 1, 0, 1, 0, 0, 1, 0, 1, 1, 0, 0, 0, 1, 0, 1, 0, 1, 0, 1, 1,
       1, 0, 1, 0, 0, 1, 0, 0, 1, 1, 1, 0, 0, 1, 1, 0, 0, 1, 0, 0, 0, 0,
       1, 1, 1, 1, 0, 0, 1, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 1, 0, 0, 1, 1,
       0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0,
       1, 0, 1, 0, 0, 0, 1, 0, 0, 1, 0, 1, 0, 0, 1, 0, 1, 0, 1, 1, 0, 0,
       0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 1, 1, 1, 1, 0, 0, 1, 0, 1, 0, 1, 1,
       0, 1, 0, 0, 0, 0, 0, 1, 0, 1, 1, 1, 1, 1, 0, 1, 1, 0, 0, 1, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 1, 0, 0,
       0, 0, 0, 1, 0, 1, 0, 1, 0, 0, 0, 1, 1, 0, 0, 1, 0, 0, 0, 1, 0, 1,
       0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 1, 0, 0, 1, 1, 1,
       1, 1, 1, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 1,
       0, 0, 1, 0, 1, 1, 0, 0, 0, 1, 1, 1, 0, 0, 0, 1, 1, 0, 0, 1, 0, 0,
       0, 0, 0, 1, 0, 1, 0, 0, 1, 1, 1, 0, 0, 0, 0, 1, 0, 0, 0, 1, 1, 0,
       0, 0, 0, 0, 0, 0, 1, 0, 1, 1, 0, 0, 1, 1, 0, 1, 0, 0, 0, 1, 1, 0,
       0, 0, 1, 1, 0, 0, 0, 1, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 1,
       0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0,
       1, 1, 0, 1, 0, 1])
```

[120]:
```
x_test
```

[120]:

	PassengerId	Pclass	Age	SibSp	Parch	Fare	male	Q	S
511	512	3	24.0	0	0	8.05	1	0	1
613	614	3	24.0	0	0	7.75	1	1	0
615	616	2	24.0	1	2	65.00	0	0	1
337	338	1	41.0	0	0	134.50	0	0	0
718	719	3	24.0	0	0	15.50	1	1	0
..	…	…	…	…	…	…	..	..	
792	793	3	24.0	8	2	69.55	0	0	1
828	829	3	24.0	0	0	7.75	1	1	0
732	733	2	29.0	0	0	0.00	1	0	1
669	670	1	37.0	1	0	52.00	0	0	1
634	635	3	9.0	3	2	27.90	0	0	1

```
[267 rows x 9 columns]
```

[121]:
```
x_test.values
```

[121]:
```
array([[512.,   3.,  24., …,   1.,   0.,   1.],
       [614.,   3.,  24., …,   1.,   1.,   0.],
       [616.,   2.,  24., …,   0.,   0.,   1.],
       …,
       [733.,   2.,  29., …,   1.,   0.,   1.],
       [670.,   1.,  37., …,   0.,   0.,   1.],
       [635.,   3.,   9., …,   0.,   0.,   1.]])
```

[122]:
```
y_test
```

[122]:
```
511    0
613    0
615    1
337    1
718    0
      ..
792    0
828    1
732    0
669    1
634    0
Name: Survived, Length: 267, dtype: int64
```

[123]:
```
y_test.values
```

[123]:
```
array([0, 0, 1, 1, 0, 0, 0, 0, 0, 1, 1, 1, 0, 1, 0, 0, 1, 1, 1, 0, 0, 0,
       0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 1,
       0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 0, 0, 0, 1, 0, 1, 1, 1, 1, 0, 1, 0,
       0, 1, 0, 0, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 1, 0, 1,
       0, 1, 0, 1, 0, 0, 1, 1, 1, 1, 0, 1, 1, 1, 0, 1, 0, 1, 1, 0, 0, 1,
       0, 0, 1, 1, 0, 1, 0, 0, 1, 1, 0, 0, 1, 0, 0, 1, 1, 0, 1, 1, 0, 1,
       1, 1, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 1, 0, 1,
       0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0,
       1, 0, 1, 0, 1, 0, 1, 1, 0, 1, 1, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 1, 0, 1, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 1, 1, 0, 0, 1,
       1, 0, 0, 1, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 1, 0, 1, 0, 0, 0,
       1, 1, 1, 0, 0, 1, 0, 1, 0, 1, 0, 0, 0, 0, 0, 1, 0, 1, 1, 0, 0, 1,
       0, 1, 0])
```

Training and Predicting

[81]:
```
from sklearn.linear_model import LogisticRegression
```

[ ]:
```
model=LogisticRegression()
model.fit(x_train,y_train)
```

[95]:
```
predictions = logmodel.predict(x_test)
```

[98]:
```
from sklearn.metrics import confusion_matrix
```

[100]:
```
accuracy=confusion_matrix(y_test,predictions)
accuracy
```

[100]:
```
array([[149,  14],
       [ 39,  65]])
```

[107]:
```
predictions
```

[107]:
```
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, 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, 0, 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])
```

#Model Evaluation

Accuracy Score

[101]:
```
from sklearn.metrics import accuracy_score
```

[130]:
```
# accuracy on training data
x_train_prediction = model.predict(x_train)
training_data_accuracy = accuracy_score(x_train_prediction, y_train)
```

[131]:
```
print('Accuracy on Training data : ', training_data_accuracy)
```

```
Accuracy on Training data :  0.7893890675241158
```

[133]:
```
# accuracy on test data
x_test_prediction = model.predict(x_test)
test_data_accuracy = accuracy_score(x_test_prediction, y_test)
```

[134]:
```
print('Accuracy score on Test Data : ', test_data_accuracy)
```

```
Accuracy score on Test Data :  0.8014981273408239
```

[139]:
```
# accuracy score on the model
accuracy=accuracy_score(y_test,predictions)
print('Accuracy score of the model : ', accuracy)
```

```
Accuracy score of the model :  0.8014981273408239
```