Titanic Survival Prediction

In [66]:
```
import pandas as pd
import matplotlib as plt
import seaborn as sns
import numpy as np
```

In [3]:
```
dataset=pd.read_csv(r"D:\Arivu\Desktop\titanic3.csv")
```

In [10]:
```
dataset.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1310 entries, 0 to 1309
Data columns (total 14 columns):
 #   Column     Non-Null Count  Dtype  
---  ------     --------------  -----  
 0   pclass     1309 non-null   float64
 1   survived   1309 non-null   float64
 2   name       1309 non-null   object 
 3   sex        1309 non-null   object 
 4   age        1046 non-null   float64
 5   sibsp      1309 non-null   float64
 6   parch      1309 non-null   float64
 7   ticket     1309 non-null   object 
 8   fare       1308 non-null   float64
 9   cabin      295 non-null    object 
 10  embarked   1307 non-null   object 
 11  boat       486 non-null    object 
 12  body       121 non-null    float64
 13  home.dest  745 non-null    object 
dtypes: float64(7), object(7)
memory usage: 143.4+ KB
```

In [ ]:
```
```

In [4]:
```
print(dataset.columns)
```

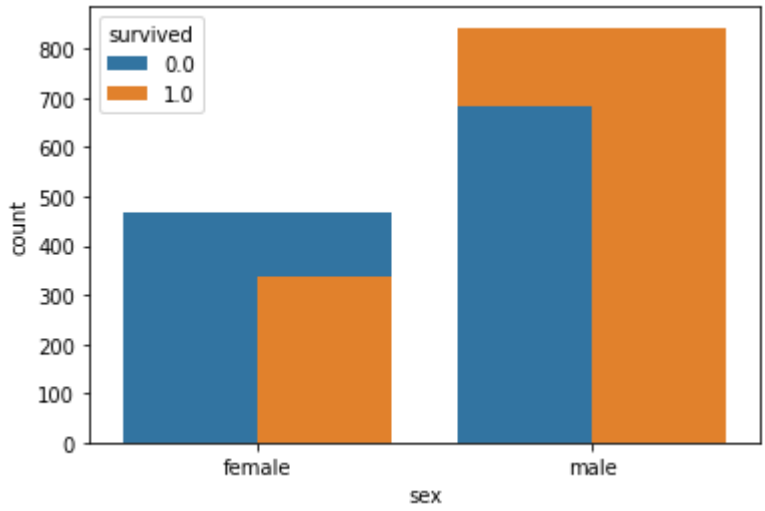
```
Index(['pclass', 'survived', 'name', 'sex', 'age', 'sibsp', 'parch', 'tick
et',
       'fare', 'cabin', 'embarked', 'boat', 'body', 'home.dest'],
      dtype='object')
```

In [6]:
```
#Column Selection/Field Selection
y=dataset[["survived"]]
print(y.head())
```

```
   survived
0       1.0
1       1.0
2       0.0
3       0.0
4       0.0
```

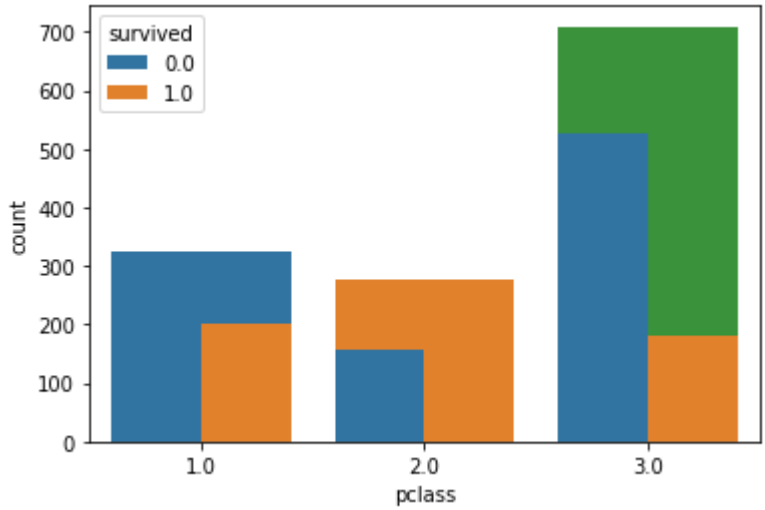
In [7]:
```
gender=dataset['sex']
sns.countplot(data=dataset,x='sex')
sns.countplot(data=dataset,x='sex',hue='survived')
```

Out[7]: <AxesSubplot:xlabel='sex', ylabel='count'>



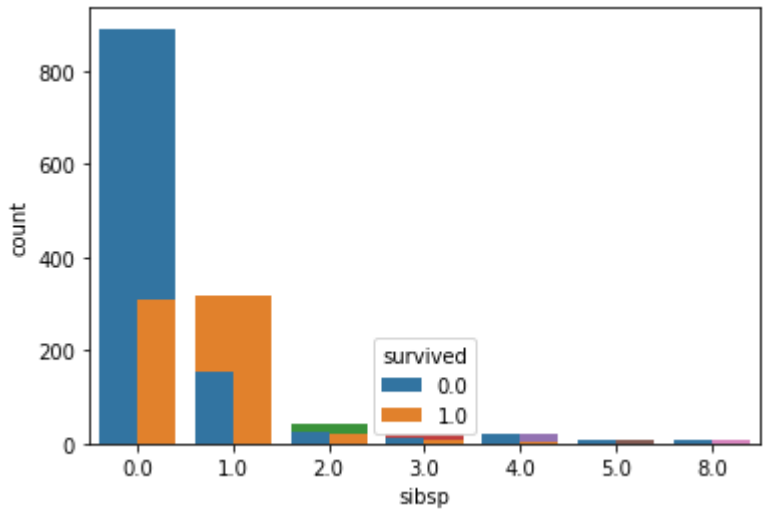
In [8]:
```
sns.countplot(data=dataset,x='pclass')
sns.countplot(data=dataset,x='pclass',hue='survived')
```

Out[8]: <AxesSubplot:xlabel='pclass', ylabel='count'>



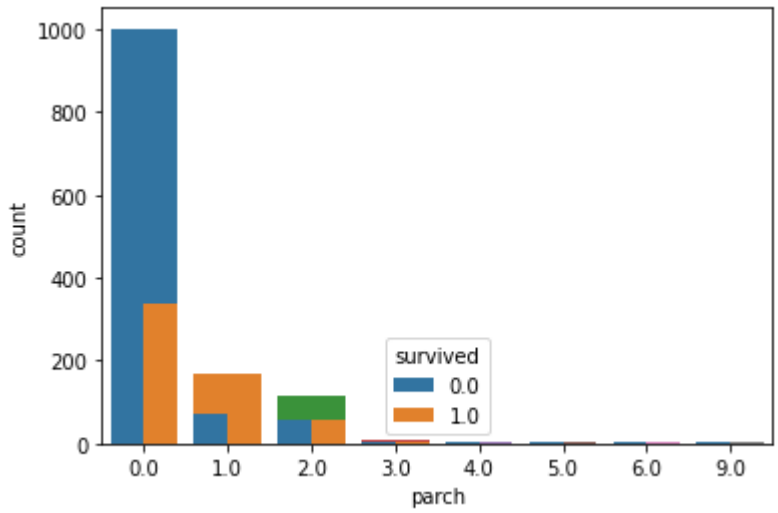
In [11]:
```
sns.countplot(data=dataset,x='sibsp')
sns.countplot(data=dataset,x='sibsp',hue='survived')
```

Out[11]: <AxesSubplot:xlabel='sibsp', ylabel='count'>



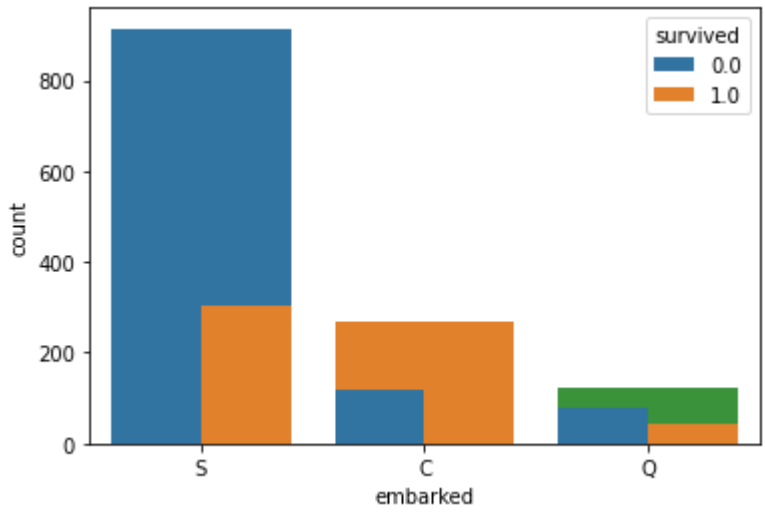
In [12]:
```
sns.countplot(data=dataset,x='parch')
sns.countplot(data=dataset,x='parch',hue='survived')
```

Out[12]: <AxesSubplot:xlabel='parch', ylabel='count'>



In [13]:
```
sns.countplot(data=dataset,x='embarked')
sns.countplot(data=dataset,x='embarked',hue='survived')
```

Out[13]: <AxesSubplot:xlabel='embarked', ylabel='count'>



In [14]:
```
dataset.drop("name",axis=1,inplace=True) #drop Name
dataset.drop("ticket",axis=1,inplace=True) #drop Ticket
dataset.drop("fare",axis=1,inplace=True) #drop Fare
dataset.drop("parch",axis=1,inplace=True) #drop parch
dataset.drop("cabin",axis=1,inplace=True) #drop Cabin
```

In [18]:
```
x=dataset[['pclass','sex','age','sibsp','embarked']]
print(x.isnull())
```

```
      pclass    sex    age  sibsp  embarked
0      False  False  False  False     False
1      False  False  False  False     False
2      False  False  False  False     False
3      False  False  False  False     False
4      False  False  False  False     False
...      ...    ...    ...    ...       ...
1305   False  False   True  False     False
1306   False  False  False  False     False
1307   False  False  False  False     False
1308   False  False  False  False     False
1309    True   True   True   True      True

[1310 rows x 5 columns]
```

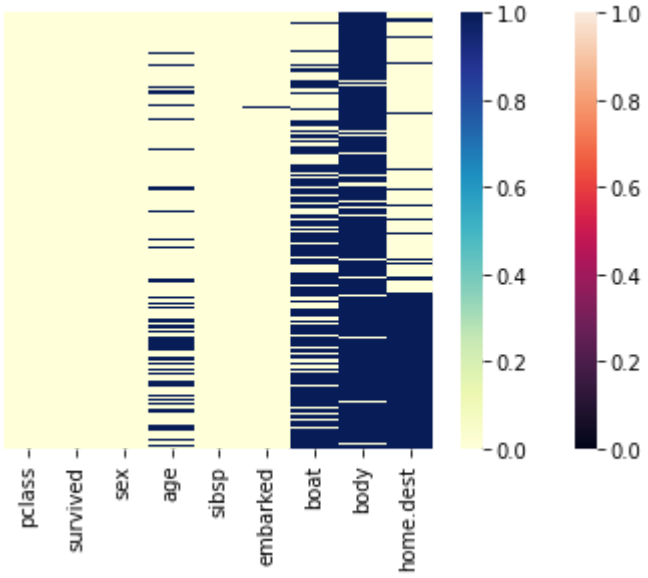
In [19]:
```
#Working with Null values
print(dataset.isnull())
sns.heatmap(dataset.isnull())
sns.heatmap(dataset.isnull() ,yticklabels=False , cmap="YlGnBu")
print(dataset[dataset["age"].isnull()]) #print null value
```

```
      pclass  survived    sex    age  sibsp  embarked   boat   body  home.
dest
0      False     False  False  False  False     False  False   True      F
alse
1      False     False  False  False  False     False  False   True      F
alse
2      False     False  False  False  False     False   True   True      F
alse
3      False     False  False  False  False     False   True  False      F
alse
4      False     False  False  False  False     False   True   True      F
alse
...      ...       ...    ...    ...    ...       ...    ...    ...
...
1305   False     False  False   True  False     False   True   True
True
1306   False     False  False  False  False     False   True  False
True
1307   False     False  False  False  False     False   True   True
True
1308   False     False  False  False  False     False   True   True
True
1309    True      True   True   True   True      True   True   True
True

[1310 rows x 9 columns]
      pclass  survived     sex  age  sibsp embarked boat  body  \
15       1.0       0.0    male  NaN    0.0        S  NaN   NaN
37       1.0       1.0    male  NaN    0.0        S    9   NaN
40       1.0       0.0    male  NaN    0.0        C  NaN   NaN
46       1.0       0.0    male  NaN    0.0        S  NaN   NaN
59       1.0       1.0  female  NaN    0.0        C    5   NaN
...      ...       ...     ...  ...    ...      ...  ...   ...
1297     3.0       0.0    male  NaN    0.0        S  NaN   NaN
1302     3.0       0.0    male  NaN    0.0        C  NaN   NaN
1303     3.0       0.0    male  NaN    0.0        C  NaN   NaN
1305     3.0       0.0  female  NaN    1.0        C  NaN   NaN
1309     NaN       NaN     NaN  NaN    NaN      NaN  NaN   NaN

            home.dest
15       New York, NY
37    Los Angeles, CA
40   Philadelphia, PA
46                NaN
59       New York, NY
...               ...
1297              NaN
1302              NaN
1303              NaN
1305              NaN
1309              NaN

[264 rows x 9 columns]
```



In [20]:
```
#here we find out the mean value using the perticular pclass
for i in range(1,4):
    age=int(dataset[dataset["pclass"]==i]['age'].dropna().mean())
    print(age)
```

```
39
29
24
```

In [23]:
```
#fill the Null value
def set_age(row):
    pclass=row[0]
    age=row[1]
    if np.isnan(age):
        if pclass==1:
            return 38
        elif pclass==2:
            return 29
        else:
            return 25
    else:
        return age
```

In [24]:
```
dataset[['pclass','age']].apply(set_age,axis=1)
```

Out[24]:
```
0       29.0000
1        0.9167
2        2.0000
3       30.0000
4       25.0000
         ...
1305    25.0000
1306    26.5000
1307    27.0000
1308    29.0000
1309    25.0000
Length: 1310, dtype: float64
```

In [25]:
```
# Working on Catagorical variable
#creating dummy variable
pclass=pd.get_dummies(dataset['pclass'],drop_first=True)
pclass.head()
```

Out[25]:

	2.0	3.0
0	0	0
1	0	0
2	0	0
3	0	0
4	0	0

In [26]:
```
sex=pd.get_dummies(dataset['sex'],drop_first=True)
sex.head()
```

Out[26]:

	male
0	0
1	1
2	0
3	1
4	0

In [27]:
```
sibsp=pd.get_dummies(dataset['sibsp'],drop_first=True)
sibsp.head()
```

Out[27]:

	1.0	2.0	3.0	4.0	5.0	8.0
0	0	0	0	0	0	0
1	1	0	0	0	0	0
2	1	0	0	0	0	0
3	1	0	0	0	0	0
4	1	0	0	0	0	0

In [28]:
```
embarked=pd.get_dummies(dataset['embarked'],drop_first=True)
embarked.head()
```

Out[28]:

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

In [30]:
```
#we have to drop catagorical variable
dataset.drop("pclass",axis=1,inplace=True)
dataset.drop("sex",axis=1,inplace=True)
dataset.drop("sibsp",axis=1,inplace=True)
dataset.drop("embarked",axis=1,inplace=True)
```

In [32]:
```
y=dataset[['survived']]
dataset.drop("survived",axis=1,inplace=True)
```

In [34]:
```
#use that variable that we have converted into dummy variable
dataset=pd.concat([pclass,sex,sibsp,embarked] , axis=1)
print(dataset.head())
```

```
   2.0  3.0  male  1.0  2.0  3.0  4.0  5.0  8.0  Q  S
0    0    0     0    0    0    0    0    0    0  0  1
1    0    0     1    1    0    0    0    0    0  0  1
2    0    0     0    1    0    0    0    0    0  0  1
3    0    0     1    1    0    0    0    0    0  0  1
4    0    0     0    1    0    0    0    0    0  0  1
```

In [35]:
```
#in some version of python feature name should be string
x.columns=x.columns.astype(str)
x.columns
```

Out[35]:
```
Index(['pclass', 'sex', 'age', 'sibsp', 'embarked'], dtype='object')
```

In [55]:
```
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split( x_encoded, y, test_size=0.2
```

In [56]:
```
print(x_train)
```

```
      pclass   age  sibsp  embarked  sex_female  sex_male
1024     3.0  25.0    0.0         2           0         1
467      2.0  24.0    1.0         2           1         0
1022     3.0   0.0    0.0         2           0         1
996      3.0  33.0    0.0         2           0         1
1300     3.0  15.0    1.0         0           1         0
...      ...   ...    ...       ...         ...       ...
1242     3.0   0.0    0.0         0           0         1
924      3.0  34.5    0.0         1           0         1
1247     3.0   0.0    1.0         2           1         0
271      1.0  24.0    1.0         2           0         1
474      2.0  31.0    0.0         2           0         1

[1048 rows x 6 columns]
```

In [38]:
```
print(y_train)
```

```
      survived
1024       0.0
467        1.0
1022       0.0
996        0.0
1300       1.0
...        ...
1242       0.0
924        0.0
1247       1.0
271        1.0
474        0.0

[1048 rows x 1 columns]
```

In [39]:
```
# Assuming 'Embarked' is a categorical variable
x_encoded = pd.get_dummies(x, columns=['embarked'])
```

In [42]:
```
from sklearn.preprocessing import LabelEncoder

label_encoder = LabelEncoder()
x['embarked'] = label_encoder.fit_transform(x['embarked'])
```

```
C:\Users\arivu\AppData\Local\Temp\ipykernel_5356\1609296811.py:4: SettingW
ithCopyWarning:
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-doc
s/stable/user_guide/indexing.html#returning-a-view-versus-a-copy (https://
pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-
view-versus-a-copy)
  x['embarked'] = label_encoder.fit_transform(x['embarked'])
```

In [45]:
```
# Assuming 'Sex' is a categorical variable
x_encoded = pd.get_dummies(x, columns=['sex'])
```

In [46]:
```
from sklearn.preprocessing import LabelEncoder

label_encoder = LabelEncoder()
x['sex'] = label_encoder.fit_transform(x['sex'])
```

```
C:\Users\arivu\AppData\Local\Temp\ipykernel_5356\1486867810.py:4: SettingW
ithCopyWarning:
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-doc
s/stable/user_guide/indexing.html#returning-a-view-versus-a-copy (https://
pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-
view-versus-a-copy)
  x['sex'] = label_encoder.fit_transform(x['sex'])
```

In [58]:
```
x_encoded.fillna(0, inplace=True)  # Replace NaN values with 0
y_train.fillna(0, inplace=True)
```

In [59]:
```
print(x_encoded.isnull().sum())
print(y_train.isnull().sum())
```

```
pclass        0
age           0
sibsp         0
embarked      0
sex_female    0
sex_male      0
dtype: int64
survived    0
dtype: int64
```

In [49]:
```
model=LogisticRegression()
```

In [60]:
```
model.fit(x_train,y_train)
```

```
C:\Users\arivu\AppData\Local\Programs\Python\Python310\lib\site-packages\s
klearn\utils\validation.py:993: DataConversionWarning: A column-vector y w
as passed when a 1d array was expected. Please change the shape of y to (n
_samples, ), for example using ravel().
  y = column_or_1d(y, warn=True)
```

Out[60]:
```
LogisticRegression()
```

In [61]:
```
print(model.coef_)
```

```
[[-0.89328966 -0.01041016 -0.24473532 -0.26075279  1.69867438 -0.9113117
6]]
```

In [63]:
```
y_pred=model.predict(x_test)
```

In [64]:
```
print(y_pred)
```

```
[1. 0. 1. 0. 0. 0. 1. 0. 0. 0. 0. 1. 1. 1. 1. 1. 0. 0. 0. 1. 1. 0. 0. 1.
 0. 0. 1. 1. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 1. 0. 0. 1. 1. 1. 0. 1. 1.
 0. 1. 0. 0. 0. 1. 0. 1. 0. 0. 0. 0. 0. 0. 1. 1. 1. 0. 0. 0. 0. 0. 1. 0.
 0. 1. 0. 1. 0. 0. 1. 0. 0. 0. 0. 0. 1. 1. 0. 1. 1. 0. 0. 1. 1. 1. 1. 0.
 0. 0. 0. 0. 1. 0. 1. 0. 0. 0. 0. 1. 1. 0. 0. 0. 1. 0. 0. 0. 0. 1. 1. 1.
 0. 0. 0. 0. 0. 0. 1. 1. 0. 1. 0. 0. 1. 0. 1. 0. 1. 0. 1. 0. 0. 0. 1. 1.
 0. 1. 0. 0. 0. 0. 0. 0. 0. 1. 1. 1. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0.
 0. 0. 0. 1. 0. 0. 0. 1. 0. 0. 1. 0. 1. 1. 1. 0. 0. 1. 0. 0. 0. 0. 0. 0.
 0. 0. 0. 0. 0. 1. 1. 1. 0. 0. 1. 0. 0. 0. 1. 0. 1. 0. 0. 1. 1. 0. 0. 1.
 1. 1. 1. 0. 0. 0. 1. 0. 0. 0. 0. 1. 0. 1. 0. 0. 0. 0. 0. 0. 1. 0. 0. 1.
 0. 0. 0. 1. 1. 0. 0. 1. 1. 0. 0. 0. 1. 1. 1. 0. 0. 0. 0. 0. 1. 0.]
```

In [65]:
```
#confusion matrix
from sklearn.metrics import confusion_matrix
print(confusion_matrix(y_test,y_pred))
```

```
[[131  21]
 [ 39  71]]
```

In [ ]:

In [ ]: