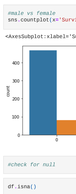
In [26]:
```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
```

In [23]:
```
df = pd.read_csv('train.csv')
```

In [ ]:
```
#data analysis
```

In [27]:
```
#survive vs survived
sns.countplot(x='Survived',data=df)
```

Out[27]:
<AxesSubplot:xlabel='Survived', ylabel='count'>



In [28]:
```
#male vs female
sns.countplot(x='Survived',data=df,hue = 'Sex')
```

Out[28]:
<AxesSubplot:xlabel='Survived', ylabel='count'>

In [ ]:
```
#check for null
```

In [29]:
```
df.isna()
```

Out[29]:

	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 [30]:
```
#how many null values
df.isna().sum()
```

Out[30]:
```
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 [31]:
```
#visualize null values
sns.heatmap(df.isna())
```

Out[31]:
<AxesSubplot:>

In [32]:
```
#find % of null values in age
100*(df['Age'].isna().sum()/len(df))
```

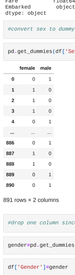
Out[32]:
```
19.865319865319865
```

In [33]:
```
#find % of null values in cabin
100*(df['Cabin'].isna().sum()/len(df))
```

Out[33]:
```
77.10437710437711
```

In [34]:
```
#find distribution of age column with exception to sex
sns.displot(x='Age',data= df,hue='Sex')
```

Out[34]:
<seaborn.axisgrid.FacetGrid at 0x26c790c49d0>



In [35]:
```
#Data cleaning
#start with age
```

In [37]:
```
df['Age'].mean()
```

Out[37]:
```
29.69911764705882
```

In [38]:
```
#now we fill
df['Age'].fillna(df['Age'].mean(),inplace=True)
```

In [39]:
```
#verify
df['Age'].isna().sum()
```

Out[39]:
```
0
```

In [40]:
```
#drop cabin
df.drop('Cabin',axis=1,inplace=True)
```

In [41]:
```
df.head()
```

Out[41]:

	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

In [42]:
```
#preparing Data for Model
```

In [43]:
```
#non-numerical
df.dtypes
```

Out[43]:
```
PassengerId      int64
Survived         int64
Pclass           int64
Name            object
Sex             object
Age            float64
SibSp            int64
Parch            int64
Ticket          object
Fare           float64
Embarked        object
dtype: object
```

In [44]:
```
#convert sex to dummy variables
```

In [45]:
```
pd.get_dummies(df['Sex'])
```

Out[45]:

	female	male
0	0	1
1	1	0
2	1	0
3	1	0
4	0	1
...	...	...
886	0	1
887	1	0
888	1	0
889	0	1
890	0	1

891 rows × 2 columns

In [46]:
```
#drop one column since esectially we really just need one since the opposite of siad value could be con
```

In [49]:
```
gender=pd.get_dummies(df['Sex'],drop_first=True)
```

In [50]:
```
df['Gender']=gender
```

In [51]:
```
#drop columns not required
```

In [52]:
```
df.drop(['Name','Sex','Ticket','Embarked'],axis=1,inplace=True)
```

In [53]:
```
df.head()
```

Out[53]:

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

In [54]:
```
#seperate Dependant and independent Variables
```

In [86]:
```
x=df[['PassengerId','Pclass','Age','SibSp','Parch','Fare','Gender']]
y=df['Survived']
```

In [88]:
```
#Data Modelling
```

In [89]:
```
#test and train the model
```

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

In [65]:
```
#train test split
```

In [69]:
```
x_train,x_test,y_train,y_test =train_test_split(x,y )
```

In [ ]:
```
#import Logistic Regression
```

In [70]:
```
from sklearn.linear_model import LogisticRegression
```

In [75]:
```
#fit Log regression
```

In [2]:
```
lr=LogisticRegression()
lr.fit(x_train,y_train)
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-2-431c11f325cd> in <module>
----> 1 lr=LogisticRegression()
      2 lr.fit(x_train,y_train)

NameError: name 'LogisticRegression' is not defined
```

In [73]:
```
#predict
predict=lr.predict(x_test)
```

In [74]:
```
#print confusion matrix
from sklearn.metrics import confusion_matrix
```

In [76]:
```
pd.DataFrame(confusion_matrix(y_test,predict),columns=['predicted no','predicted yes'],index=['Actual N
```

Out[76]:

	predicted no	predicted yes
Actual No	113	20
Actual Yes	26	64

In [78]:
```
from sklearn.metrics import classification_report
print(classification_report(y_test,predict))
```

```
              precision    recall  f1-score   support

           0       0.81      0.85      0.83       133
           1       0.76      0.71      0.74        90

    accuracy                           0.79       223
   macro avg       0.79      0.78      0.78       223
weighted avg       0.79      0.79      0.79       223
```

In [ ]:
```

```