sns.barplot(x=titanic\_data.columns, y=titanic\_data.isnull().sum().values)

plt.xticks(rotation=45)

plt.show()-> To see missing values in the plot

Then calling ***.sum()*** off of this gives us back a Series telling us how many true (missing values) were in each column. Recall that True is an alias for 1, which is why we can take the sum of True False columns.

titanic\_data**.**isnull()**.**sum()

For Age, our best bet would be to impute any missing values with the mean age. We can do this very quickly with pandas ***.apply(func)***. This will apply any function to every value along a column. If you're not familiar with lambda functions, you can create a normal python function that accepts the age and mean\_age, and returns the mean age if age is null, or the age itself if it's not null. Then you can supply that function to ***.apply(func)***. So here we're reassigning the titanic\_data['Age'] column to titanic\_data['Age'] after our function has been applied on it, which will essentially fill any missing age values with the mean age calculated.

mean\_age **=** int(titanic\_data['Age']**.**mean())

titanic\_data['Age'] **=** titanic\_data['Age']**.**apply(**lambda** age : mean\_age **if** pd**.**isnull(age) **else** age)

Our next step is to handle categorical variables since machine learning algorithms can only understand the **number**

***pd.get\_dummies(data, columns)***

So if a specific observation is female, we will place a 1 in sex\_female and 0 in sex\_male. One important note is that you should always add an additional drop\_first=True parameter when using get\_dummies. This will drop one of the columns created in the dummy process, since keeping all of them will result in multicollinearity.

titanic\_data **=** pd**.**get\_dummies(data**=**titanic\_data, columns**=**['Sex', 'Embarked'], drop\_first**=True**)

titanic\_data**.**drop(labels**=**['Name','Ticket'], axis**=**1, inplace**=True**)