In [2]:
```
#Importing libraries
```

In [35]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import missingno as msno
from sklearn.preprocessing import LabelEncoder
from sklearn import preprocessing
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import confusion_matrix,accuracy_score,classification_report,roc_auc_score,roc_curve
from sklearn.model_selection import train_test_split,cross_val_score
import warnings
warnings.filterwarnings("ignore",category=DeprecationWarning)
warnings.filterwarnings("ignore",category=FutureWarning)
```

In [2]:
```
df = pd.read_csv('train.csv')
```

In [3]:
```
df.head()
```

Out[3]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	S
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN	S

In [4]:
```
df.tail()
```

Out[4]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
886	887	0	2	Montvila, Rev. Juozas	male	27.0	0	0	211536	13.00	NaN	S
887	888	1	1	Graham, Miss. Margaret Edith	female	19.0	0	0	112053	30.00	B42	S

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
888	889	0	3	Johnston, Miss. Catherine Helen "Carrie"	female	NaN	1	2	W./C. 6607	23.45	NaN	S
889	890	1	1	Behr, Mr. Karl Howell	male	26.0	0	0	111369	30.00	C148	C
890	891	0	3	Dooley, Mr. Patrick	male	32.0	0	0	370376	7.75	NaN	Q

In [5]:
```
df.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 [6]:
```
df.describe().T
```

Out[6]:

	count	mean	std	min	25%	50%	75%	max
PassengerId	891.0	446.000000	257.353842	1.00	223.5000	446.0000	668.5	891.0000
Survived	891.0	0.383838	0.486592	0.00	0.0000	0.0000	1.0	1.0000
Pclass	891.0	2.308642	0.836071	1.00	2.0000	3.0000	3.0	3.0000
Age	714.0	29.699118	14.526497	0.42	20.1250	28.0000	38.0	80.0000
SibSp	891.0	0.523008	1.102743	0.00	0.0000	0.0000	1.0	8.0000
Parch	891.0	0.381594	0.806057	0.00	0.0000	0.0000	0.0	6.0000

	count	mean	std	min	25%	50%	75%	max
Fare	891.0	32.204208	49.693429	0.00	7.9104	14.4542	31.0	512.3292

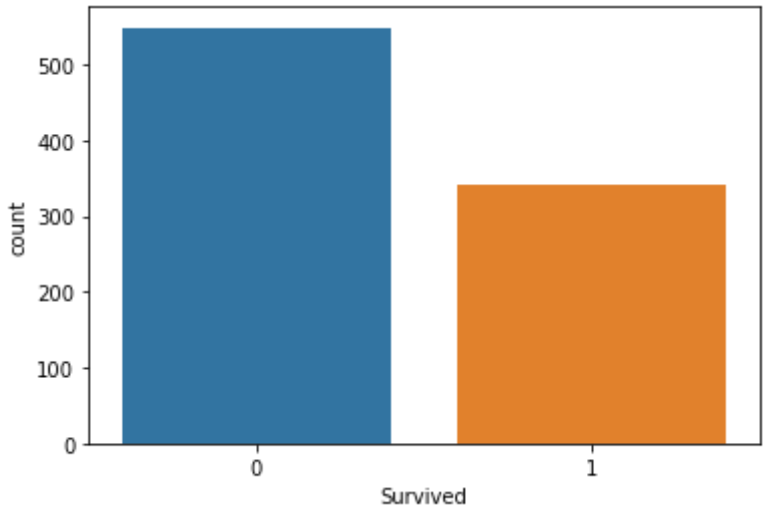
In [7]:
```
df['Survived'].value_counts()
```

Out[7]:
```
0    549
1    342
Name: Survived, dtype: int64
```

In [8]:
```
#Visualizations
```

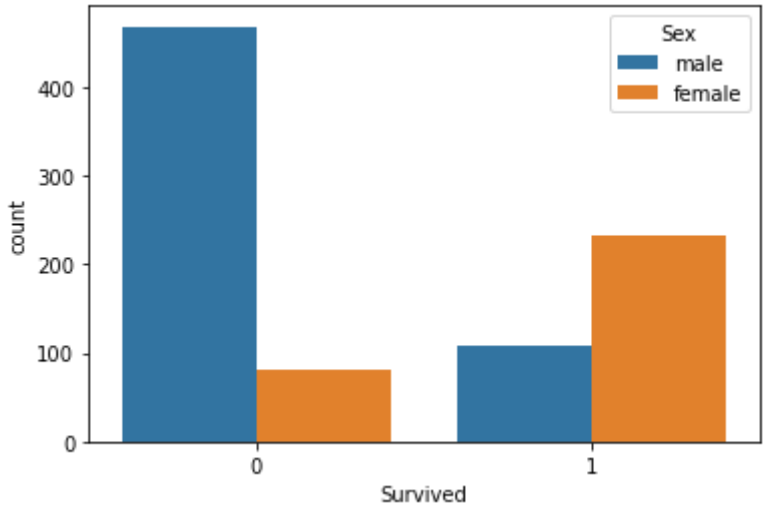
In [9]:
```
sns.countplot(df['Survived'])
```

Out[9]:
```
<AxesSubplot:xlabel='Survived', ylabel='count'>
```



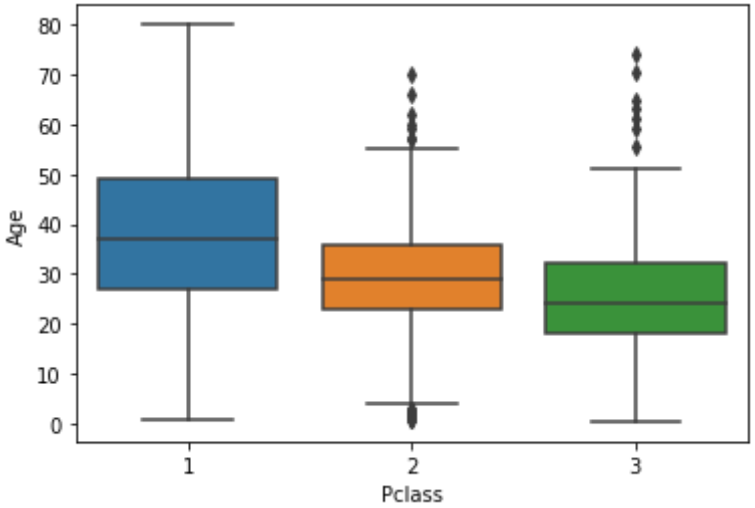
In [10]:
```
sns.countplot(df['Survived'],hue=df['Sex'],data = df)
```

Out[10]:
```
<AxesSubplot:xlabel='Survived', ylabel='count'>
```



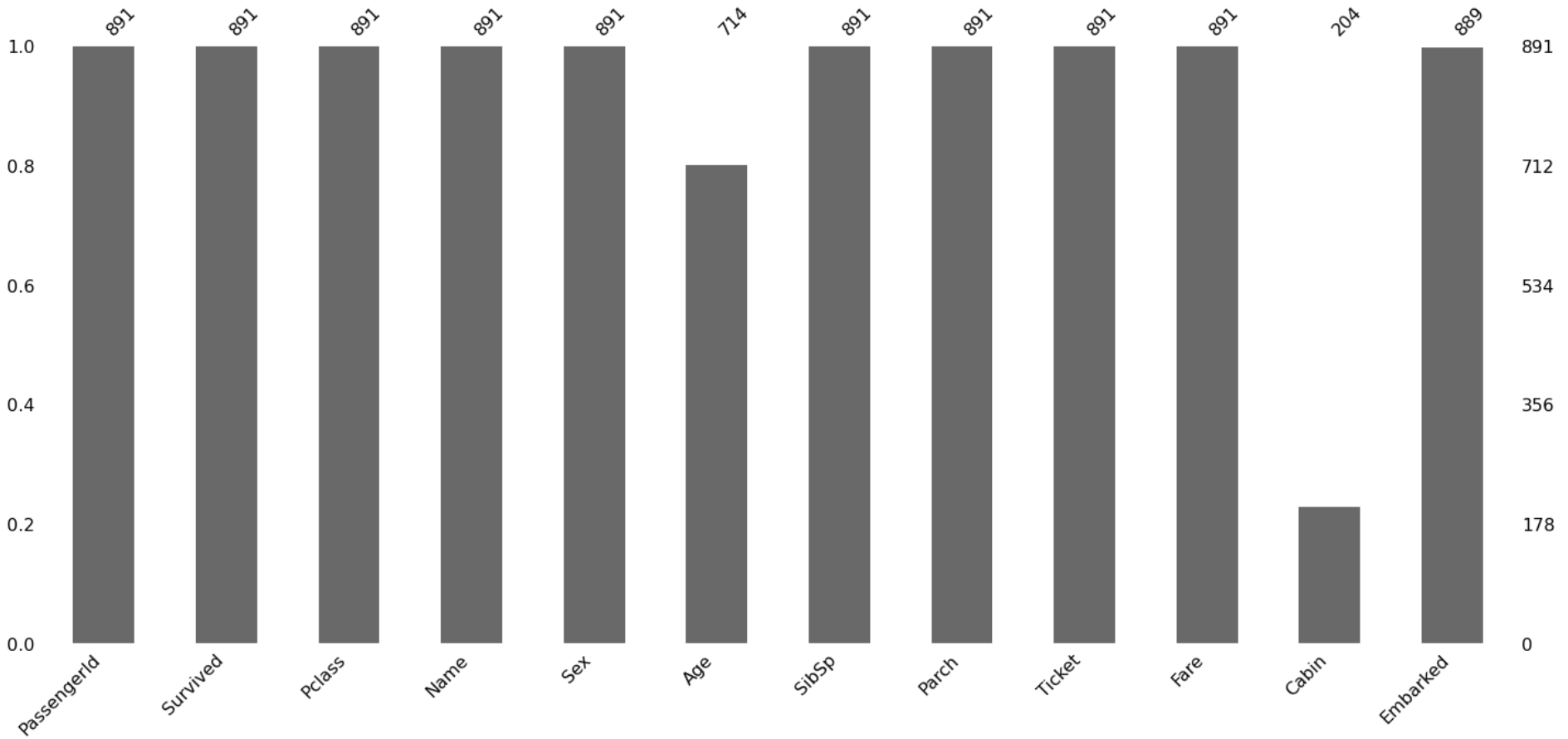
In [11]:
```
sns.boxplot(x="Pclass", y="Age", data=df)
```

Out[11]:
```
<AxesSubplot:xlabel='Pclass', ylabel='Age'>
```



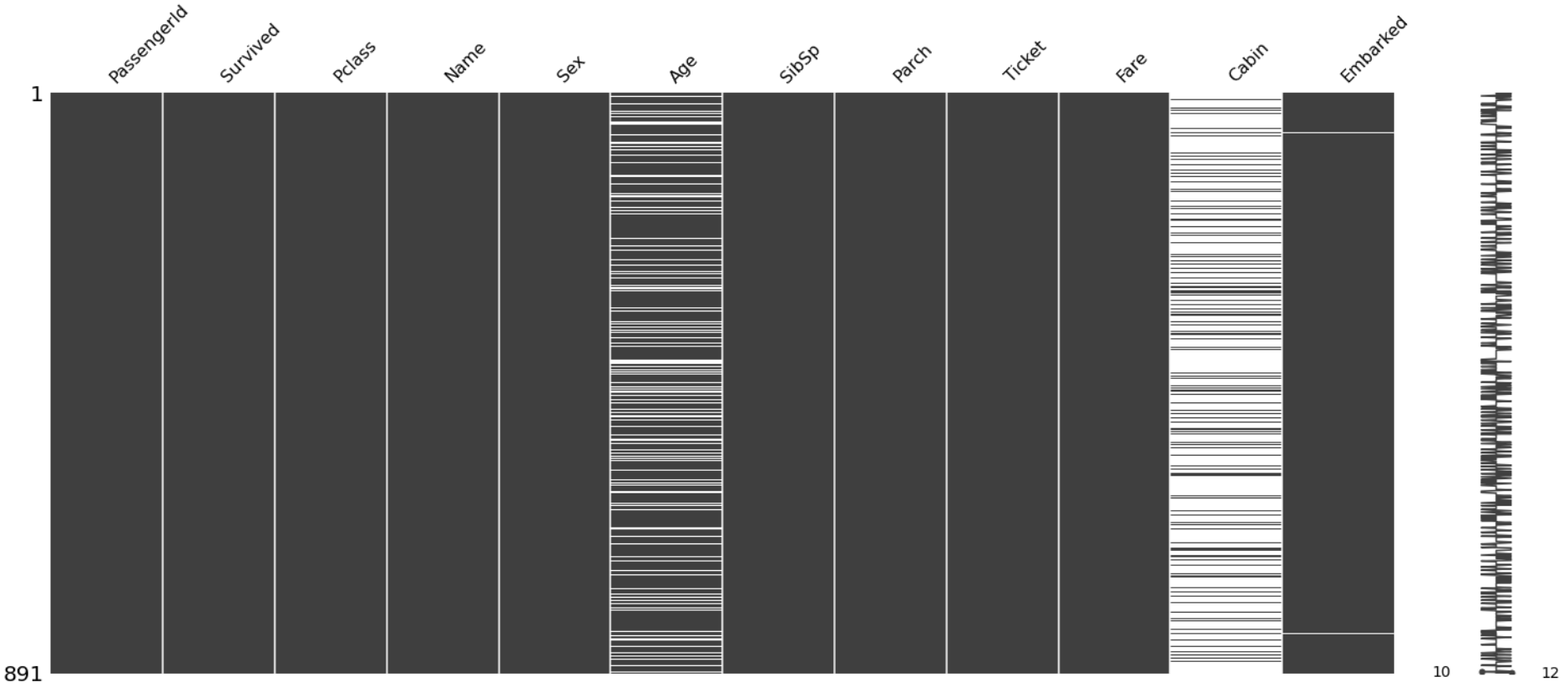
In [12]:
```
msno.bar(df)
```

Out[12]:
```
<AxesSubplot:>
```



In [13]:
```
msno.matrix(df)
```

Out[13]: <AxesSubplot:>



In [14]:
```
df.isnull()
```

Out[14]:

	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

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
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 [15]:
```
df.isnull().sum()
```

Out[15]:
```
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 [16]:
```
df.drop(['Name','Ticket','Cabin'],inplace = True,axis = 1)
```

In [17]:
```
lf = preprocessing.LabelEncoder()
```

In [18]:
```
df['Sex'] = lf.fit_transform(df['Sex'])
```

In [19]:
```
df
```

Out[19]:

	PassengerId	Survived	Pclass	Sex	Age	SibSp	Parch	Fare	Embarked
0	1	0	3	1	22.0	1	0	7.2500	S
1	2	1	1	0	38.0	1	0	71.2833	C
2	3	1	3	0	26.0	0	0	7.9250	S

	PassengerId	Survived	Pclass	Sex	Age	SibSp	Parch	Fare	Embarked
3	4	1	1	0	35.0	1	0	53.1000	S
4	5	0	3	1	35.0	0	0	8.0500	S
...	...	...	...	...	...	...	...	...	...
886	887	0	2	1	27.0	0	0	13.0000	S
887	888	1	1	0	19.0	0	0	30.0000	S
888	889	0	3	0	NaN	1	2	23.4500	S
889	890	1	1	1	26.0	0	0	30.0000	C
890	891	0	3	1	32.0	0	0	7.7500	Q

891 rows × 9 columns

In [20]:
```
df.drop(['Embarked','Age'],inplace = True,axis = 1)
```

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

In [41]:
```
df
```

Out[41]:

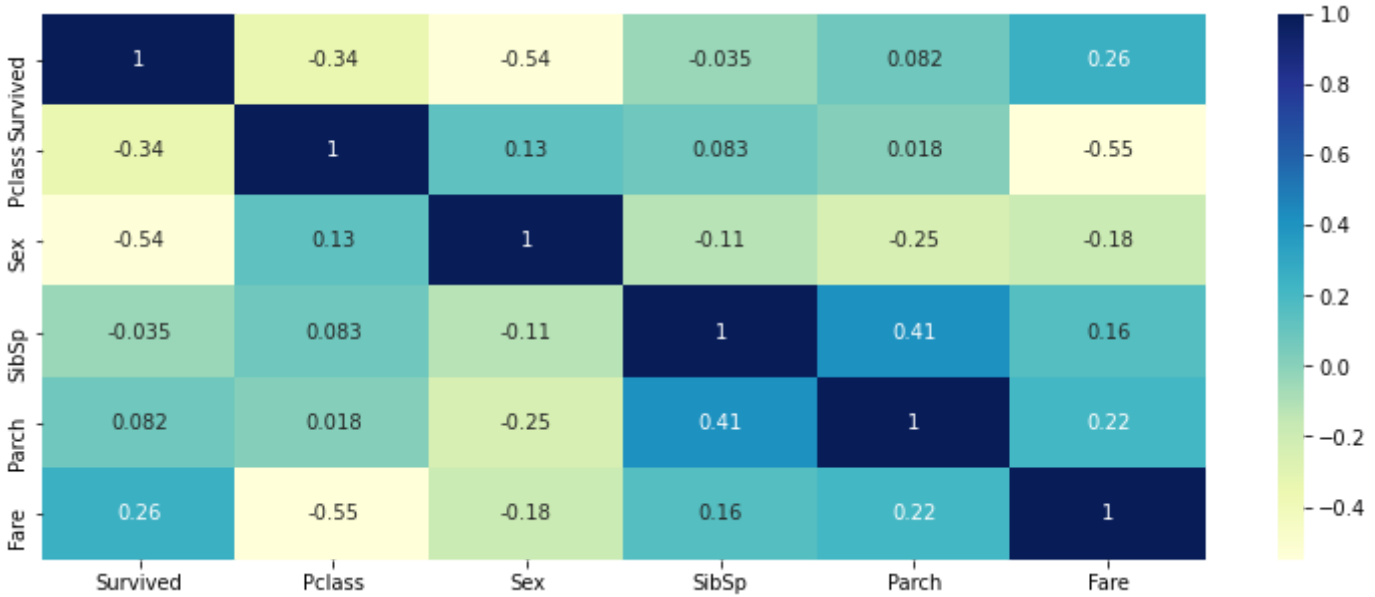
	Survived	Pclass	Sex	SibSp	Parch	Fare
0	0	3	1	1	0	7.2500
1	1	1	0	1	0	71.2833
2	1	3	0	0	0	7.9250
3	1	1	0	1	0	53.1000
4	0	3	1	0	0	8.0500
...	...	...	...	...	...	...
886	0	2	1	0	0	13.0000
887	1	1	0	0	0	30.0000
888	0	3	0	1	2	23.4500

	Survived	Pclass	Sex	SibSp	Parch	Fare
889	1	1	1	0	0	30.0000
890	0	3	1	0	0	7.7500

891 rows × 6 columns

In [42]:
```
corr_mat=df.corr()
```

In [68]:
```
plt.figure(figsize=(13,5))
sns_plot=sns.heatmap(data=corr_mat, annot=True, cmap='YlGnBu')
plt.show()
```



In [44]:
```
#Splitting train and test data
```

In [45]:
```
X = df.drop("Survived", axis=1)
y = df["Survived"]
```

In [46]:
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=1)
```

In [47]:
```
log_model=LogisticRegression()
```

In [48]:
```
log_model.fit(X_train,y_train)
```

Out[48]:
```
LogisticRegression()
```

In [49]:
```
log_model.intercept_
```

Out[49]:
```
array([3.18505413])
```

In [50]:
```
log_model.coef_
```

Out[50]:
```
array([[-9.05889925e-01, -2.70438304e+00, -2.62124389e-01,
         7.61902204e-02,  9.00840304e-04]])
```

In [53]:
```
predictions = log_model.predict(X_test)
predictions[:10]
```

Out[53]:
```
array([1, 0, 1, 1, 1, 0, 0, 1, 0, 1], dtype=int64)
```

In [54]:
```
print(classification_report(y_test,predictions))
```

```
              precision    recall  f1-score   support

           0       0.76      0.87      0.81       153
           1       0.78      0.63      0.70       115

    accuracy                           0.76       268
   macro avg       0.77      0.75      0.75       268
weighted avg       0.77      0.76      0.76       268
```

In [57]:
```
log_model.predict_proba(X)[0:10]
```

Out[57]:
```
array([[0.92362086, 0.07637914],
       [0.11093216, 0.88906784],
       [0.38354615, 0.61645385],
       [0.11255801, 0.88744199],
       [0.90288917, 0.09711083],
       [0.90285692, 0.09714308],
       [0.59351103, 0.40648897],
       [0.94922282, 0.05077718],
       [0.34755722, 0.65244278],
       [0.24264385, 0.75735615]])
```

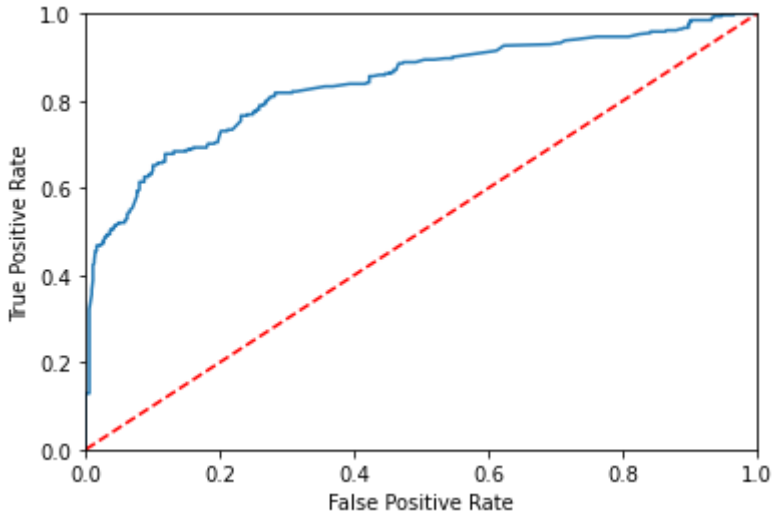
In [55]:
```
confusion_matrix(y_test, predictions)
```

Out[55]:
```
array([[133,  20],
       [ 43,  72]], dtype=int64)
```

In [56]:
```
 accuracy_score(y_test,predictions)
```

Out[56]:
```
0.7649253731343284
```

In [65]:
```
logit_roc_auc = roc_auc_score(y,log_model.predict(X))
fpr,tpr,thresholds=roc_curve(y,log_model.predict_proba(X)[:,1])
plt.figure()
plt.plot(fpr,tpr,label="AUC (area=%0.2f)" % logit_roc_auc)
plt.plot([0,1],[0,1],"r--")
plt.xlim([0.0,1.0])
plt.ylim([0.0,1.0])
plt.xlabel("False Positive Rate")
plt.ylabel("True Positive Rate")
plt.savefig("Log_ROC")
plt.show()
```



In [ ]:
```

```