In [9]:
```
import numpy as np
import pandas as pd

#visualization libraries
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline

#ignore warnings
import warnings
warnings.filterwarnings('ignore')
```

In [12]:
```
test = pd.read_csv(r"C:\Users\neetu27\Downloads\archive (1)\tested.csv")

#take a look at the training data
test.describe(include="all")
```

Out[12]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
count	418.000000	418.000000	418.000000	418	418	332.000000	418.000000	418.000000	418	417.000000	91	418
unique	NaN	NaN	NaN	418	2	NaN	NaN	NaN	363	NaN	76	3
top	NaN	NaN	NaN	Kelly, Mr. James	male	NaN	NaN	NaN	PC 17608	NaN	B57 B59 B63 B66	S
freq	NaN	NaN	NaN	1	266	NaN	NaN	NaN	5	NaN	3	270
mean	1100.500000	0.363636	2.265550	NaN	NaN	30.272590	0.447368	0.392344	NaN	35.627188	NaN	NaN
std	120.810458	0.481622	0.841838	NaN	NaN	14.181209	0.896760	0.981429	NaN	55.907576	NaN	NaN
min	892.000000	0.000000	1.000000	NaN	NaN	0.170000	0.000000	0.000000	NaN	0.000000	NaN	NaN
25%	996.250000	0.000000	1.000000	NaN	NaN	21.000000	0.000000	0.000000	NaN	7.895800	NaN	NaN
50%	1100.500000	0.000000	3.000000	NaN	NaN	27.000000	0.000000	0.000000	NaN	14.454200	NaN	NaN
75%	1204.750000	1.000000	3.000000	NaN	NaN	39.000000	1.000000	0.000000	NaN	31.500000	NaN	NaN
max	1309.000000	1.000000	3.000000	NaN	NaN	76.000000	8.000000	9.000000	NaN	512.329200	NaN	NaN

In [13]:
```
print(test.columns)
```

```
Index(['PassengerId', 'Survived', 'Pclass', 'Name', 'Sex', 'Age', 'SibSp',
       'Parch', 'Ticket', 'Fare', 'Cabin', 'Embarked'],
      dtype='object')
```

In [14]:
```
test.sample(5)
```

Out[14]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
336	1228	0	2	de Brito, Mr. Jose Joaquim	male	32.0	0	0	244360	13.0000	NaN	S
70	962	1	3	Mulvihill, Miss. Bertha E	female	24.0	0	0	382653	7.7500	NaN	Q
189	1081	0	2	Veal, Mr. James	male	40.0	0	0	28221	13.0000	NaN	S
85	977	0	3	Khalil, Mr. Betros	male	NaN	1	0	2660	14.4542	NaN	C
238	1130	1	2	Hiltunen, Miss. Marta	female	18.0	1	1	250650	13.0000	NaN	S

In [15]:
```
test.describe(include = "all")
```

Out[15]:

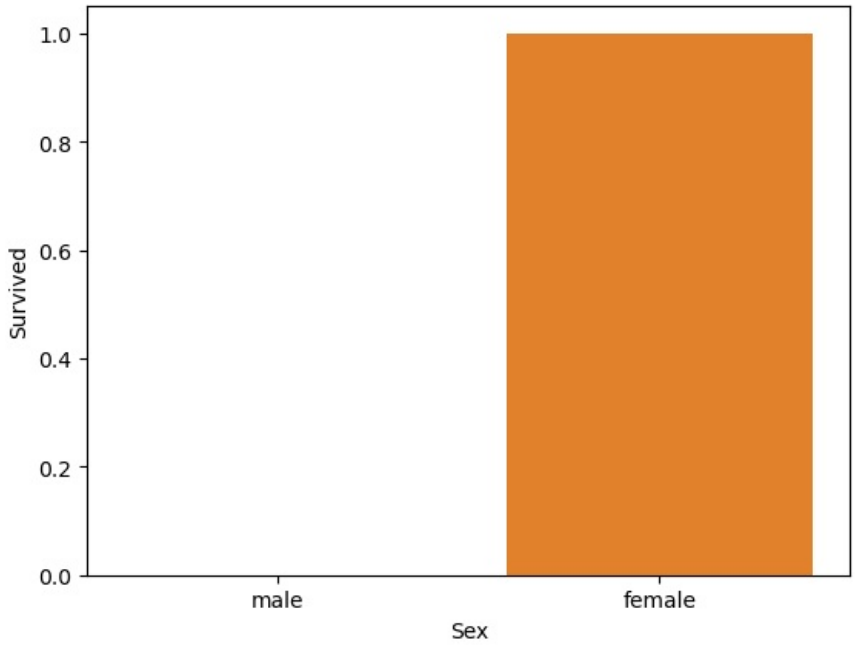
	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
count	418.000000	418.000000	418.000000	418	418	332.000000	418.000000	418.000000	418	417.000000	91	418
unique	NaN	NaN	NaN	418	2	NaN	NaN	NaN	363	NaN	76	3
top	NaN	NaN	NaN	Kelly, Mr. James	male	NaN	NaN	NaN	PC 17608	NaN	B57 B59 B63 B66	S
freq	NaN	NaN	NaN	1	266	NaN	NaN	NaN	5	NaN	3	270
mean	1100.500000	0.363636	2.265550	NaN	NaN	30.272590	0.447368	0.392344	NaN	35.627188	NaN	NaN
std	120.810458	0.481622	0.841838	NaN	NaN	14.181209	0.896760	0.981429	NaN	55.907576	NaN	NaN
min	892.000000	0.000000	1.000000	NaN	NaN	0.170000	0.000000	0.000000	NaN	0.000000	NaN	NaN
25%	996.250000	0.000000	1.000000	NaN	NaN	21.000000	0.000000	0.000000	NaN	7.895800	NaN	NaN
50%	1100.500000	0.000000	3.000000	NaN	NaN	27.000000	0.000000	0.000000	NaN	14.454200	NaN	NaN
75%	1204.750000	1.000000	3.000000	NaN	NaN	39.000000	1.000000	0.000000	NaN	31.500000	NaN	NaN
max	1309.000000	1.000000	3.000000	NaN	NaN	76.000000	8.000000	9.000000	NaN	512.329200	NaN	NaN

In [17]:
```
print(pd.isnull(test).sum())
```

```
PassengerId      0
Survived         0
Pclass           0
Name             0
Sex              0
Age             86
SibSp            0
Parch            0
Ticket           0
Fare             1
Cabin          327
Embarked         0
dtype: int64
```

In [22]:
```
sns.barplot(x="Sex", y="Survived", data=test)
```

Out[22]:
```
<Axes: xlabel='Sex', ylabel='Survived'>
```

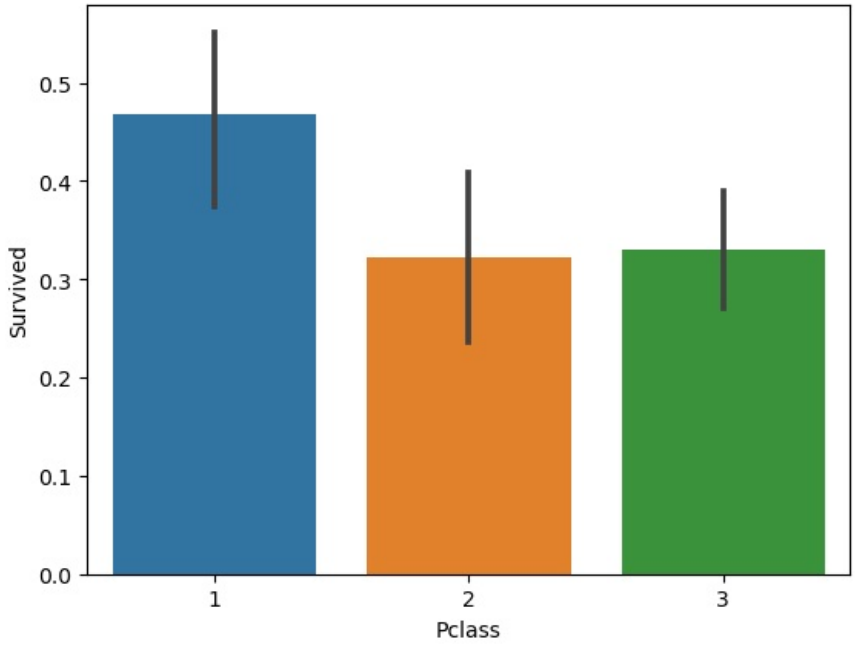


In [23]:
```
print("Percentage of females who survived:", test["Survived"][test["Sex"] == 'female'].value_counts(normalize =
```

```
Percentage of females who survived: 100.0
```

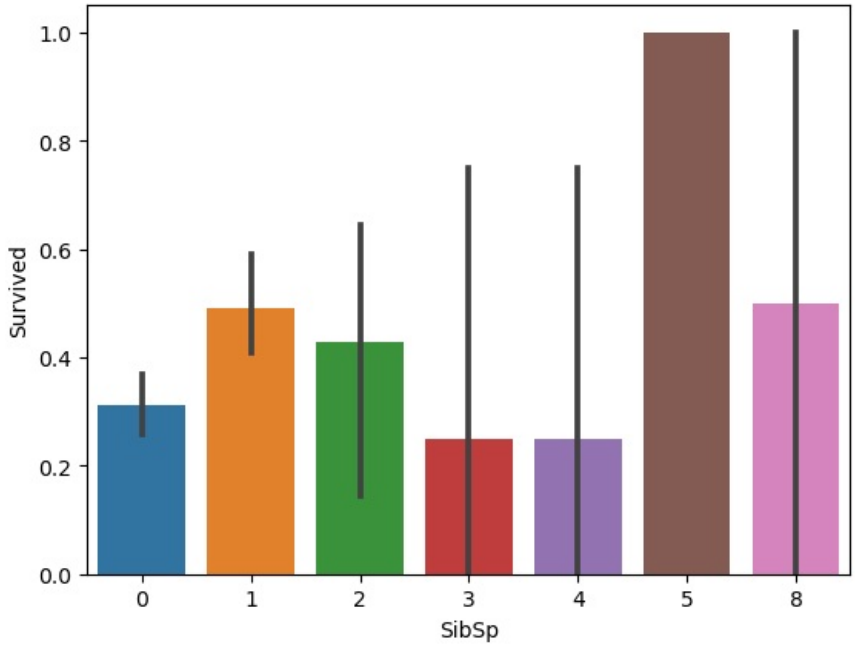
In [28]:
```
sns.barplot(x="Pclass", y="Survived", data=test)

#print percentage of people by Pclass that survived
print("Percentage of Pclass = 1 who survived:", test["Survived"][test["Pclass"] == 1].value_counts(normalize =

print("Percentage of Pclass = 2 who survived:", test["Survived"][test["Pclass"] == 2].value_counts(normalize =

print("Percentage of Pclass = 3 who survived:", test["Survived"][test["Pclass"] == 3].value_counts(normalize =
```

```
Percentage of Pclass = 1 who survived: 46.728971962616825
Percentage of Pclass = 2 who survived: 32.25806451612903
Percentage of Pclass = 3 who survived: 33.02752293577982
```

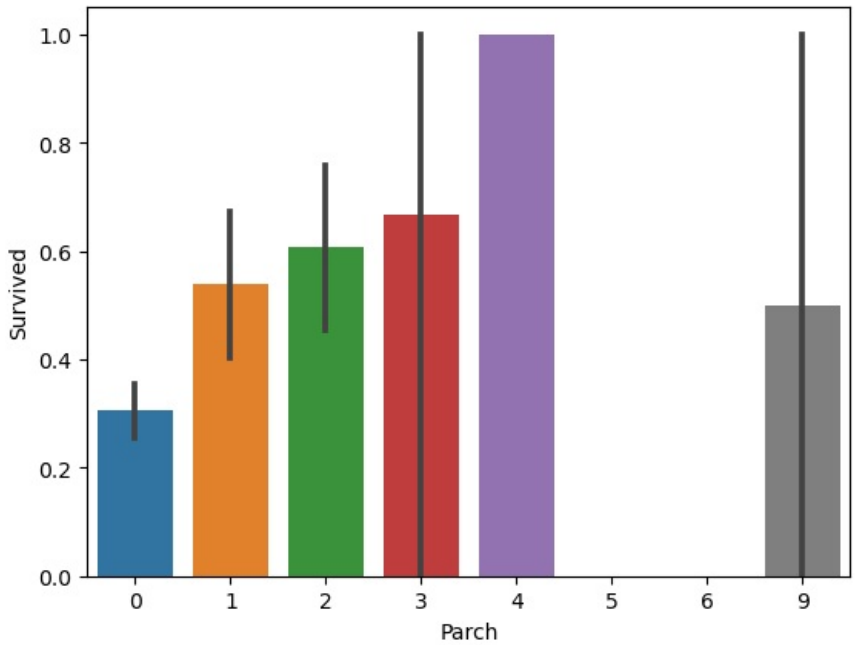


In [29]:
```
sns.barplot(x="SibSp", y="Survived", data=test)

#I won't be printing individual percent values for all of these.
print("Percentage of SibSp = 0 who survived:", test["Survived"][test["SibSp"] == 0].value_counts(normalize = Tr

print("Percentage of SibSp = 1 who survived:", test["Survived"][test["SibSp"] == 1].value_counts(normalize = Tr

print("Percentage of SibSp = 2 who survived:", test["Survived"][test["SibSp"] == 2].value_counts(normalize = Tr
```

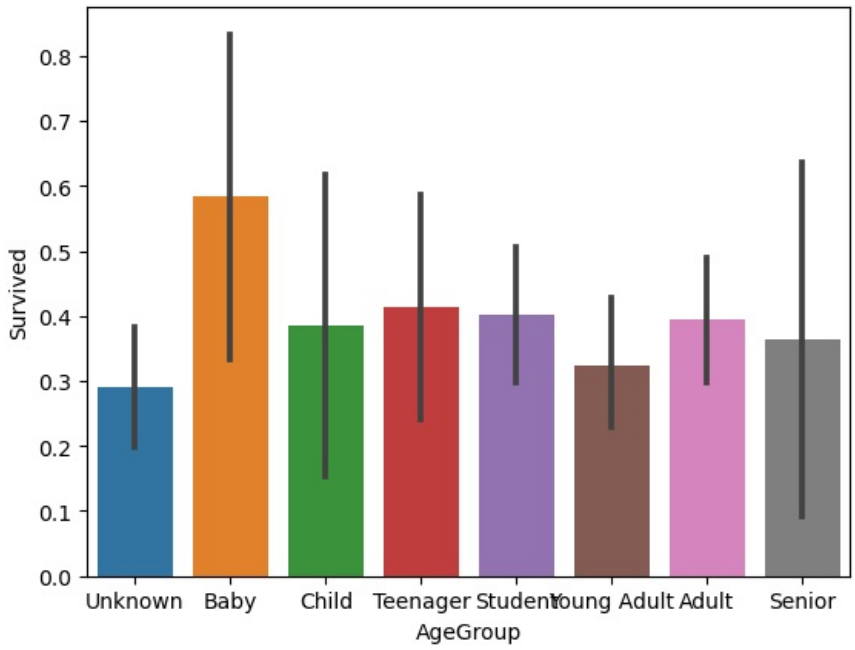
```
Percentage of SibSp = 0 who survived: 31.09540636042403
Percentage of SibSp = 1 who survived: 49.09090909090909
Percentage of SibSp = 2 who survived: 42.857142857142854
```



In [30]:
```
sns.barplot(x="Parch", y="Survived", data=test)
plt.show()
```

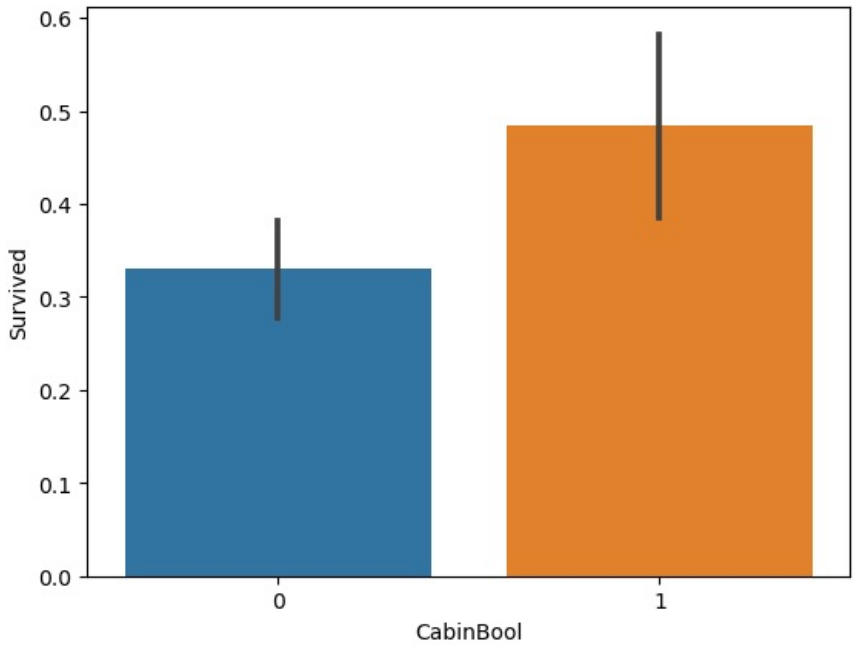


In [31]:
```
test["Age"] = test["Age"].fillna(-0.5)
test["Age"] = test["Age"].fillna(-0.5)
bins = [-1, 0, 5, 12, 18, 24, 35, 60, np.inf]
labels = ['Unknown', 'Baby', 'Child', 'Teenager', 'Student', 'Young Adult', 'Adult', 'Senior']
test['AgeGroup'] = pd.cut(test["Age"], bins, labels = labels)
test['AgeGroup'] = pd.cut(test["Age"], bins, labels = labels)

#draw a bar plot of Age vs. survival
sns.barplot(x="AgeGroup", y="Survived", data=test)
plt.show()
```



In [32]:
```
test["CabinBool"] = (test["Cabin"].notnull().astype('int'))
test["CabinBool"] = (test["Cabin"].notnull().astype('int'))

#calculate percentages of CabinBool vs. survived
print("Percentage of CabinBool = 1 who survived:", test["Survived"][test["CabinBool"] == 1].value_counts(normal

print("Percentage of CabinBool = 0 who survived:", test["Survived"][test["CabinBool"] == 0].value_counts(normal
#draw a bar plot of CabinBool vs. survival
sns.barplot(x="CabinBool", y="Survived", data=test)
plt.show()
```

```
Percentage of CabinBool = 1 who survived: 48.35164835164835
Percentage of CabinBool = 0 who survived: 33.02752293577982
```



In [38]:
```
print("Number of people embarking in Southampton (S):")
southampton = test[test["Embarked"] == "S"].shape[0]
print(southampton)

print("Number of people embarking in Cherbourg (C):")
cherbourg = test[test["Embarked"] == "C"].shape[0]
print(cherbourg)

print("Number of people embarking in Queenstown (Q):")
queenstown = test[test["Embarked"] == "Q"].shape[0]
print(queenstown)
```

```
Number of people embarking in Southampton (S):
270
Number of people embarking in Cherbourg (C):
102
Number of people embarking in Queenstown (Q):
46
```

In [39]:
```
test = test.fillna({"Embarked": "S"})
```

In [40]:
```
combine = [test]

#extract a title for each Name in the train and test datasets
for dataset in combine:
    dataset['Title'] = dataset.Name.str.extract(' ([A-Za-z]+)\.', expand=False)

pd.crosstab(test['Title'], test['Sex'])
```

Out[40]:

Sex	female	male
Title		
Col	0	2
Dona	1	0
Dr	0	1
Master	0	21
Miss	78	0
Mr	0	240
Mrs	72	0
Ms	1	0
Rev	0	2

In [41]:
```
for dataset in combine:
    dataset['Title'] = dataset['Title'].replace(['Lady', 'Capt', 'Col',
    'Don', 'Dr', 'Major', 'Rev', 'Jonkheer', 'Dona'], 'Rare')

    dataset['Title'] = dataset['Title'].replace(['Countess', 'Lady', 'Sir'], 'Royal')
    dataset['Title'] = dataset['Title'].replace('Mlle', 'Miss')
    dataset['Title'] = dataset['Title'].replace('Ms', 'Miss')
    dataset['Title'] = dataset['Title'].replace('Mme', 'Mrs')

test[['Title', 'Survived']].groupby(['Title'], as_index=False).mean()
```

Out[41]:

	Title	Survived
0	Master	0.000000
1	Miss	1.000000
2	Mr	0.000000
3	Mrs	1.000000
4	Rare	0.166667

In [42]:
```
title_mapping = {"Mr": 1, "Miss": 2, "Mrs": 3, "Master": 4, "Royal": 5, "Rare": 6}
for dataset in combine:
    dataset['Title'] = dataset['Title'].map(title_mapping)
    dataset['Title'] = dataset['Title'].fillna(0)

test.head()
```

Out[42]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Embarked	AgeGroup	CabinBool	Title
0	892	0	3	Kelly, Mr. James	male	34.5	0	0	330911	7.8292	Q	Young Adult	0	1
1	893	1	3	Wilkes, Mrs. James (Ellen Needs)	female	47.0	1	0	363272	7.0000	S	Adult	0	3
2	894	0	2	Myles, Mr. Thomas Francis	male	62.0	0	0	240276	9.6875	Q	Senior	0	1
3	895	0	3	Wirz, Mr. Albert	male	27.0	0	0	315154	8.6625	S	Young Adult	0	1
4	896	1	3	Hirvonen, Mrs. Alexander (Helga E Lindqvist)	female	22.0	1	1	3101298	12.2875	S	Student	0	3

In [45]:
```
mr_age = test[test["Title"] == 1]["AgeGroup"].mode() #Young Adult
miss_age = test[test["Title"] == 2]["AgeGroup"].mode() #Student
mrs_age = test[test["Title"] == 3]["AgeGroup"].mode() #Adult
master_age = test[test["Title"] == 4]["AgeGroup"].mode() #Baby
royal_age = test[test["Title"] == 5]["AgeGroup"].mode() #Adult
rare_age = test[test["Title"] == 6]["AgeGroup"].mode() #Adult

age_title_mapping = {1: "Young Adult", 2: "Student", 3: "Adult", 4: "Baby", 5: "Adult", 6: "Adult"}

#I tried to get this code to work with using .map(), but couldn't.
#I've put down a less elegant, temporary solution for now.
#train = train.fillna({"Age": train["Title"].map(age_title_mapping)})
```

```
#test = test.fillna({"Age": test["Title"].map(age_title_mapping)})

for x in range(len(test["AgeGroup"])):
    if test["AgeGroup"][x] == "Unknown":
        test["AgeGroup"][x] = age_title_mapping[test["Title"][x]]

for x in range(len(test["AgeGroup"])):
    if test["AgeGroup"][x] == "Unknown":
        test["AgeGroup"][x] = age_title_mapping[test["Title"][x]]
```

In [50]:
```
sex_mapping = {"male": 0, "female": 1}
test['Sex'] = test['Sex'].map(sex_mapping)
test['Sex'] = test['Sex'].map(sex_mapping)
test.head()
```

Out[50]:

	PassengerId	Survived	Pclass	Name	Sex	SibSp	Parch	Ticket	Fare	Embarked	AgeGroup	CabinBool	Title
0	892	0	3	Kelly, Mr. James	NaN	0	0	330911	7.8292	Q	NaN	0	1
1	893	1	3	Wilkes, Mrs. James (Ellen Needs)	NaN	1	0	363272	7.0000	S	NaN	0	3
2	894	0	2	Myles, Mr. Thomas Francis	NaN	0	0	240276	9.6875	Q	NaN	0	1
3	895	0	3	Wirz, Mr. Albert	NaN	0	0	315154	8.6625	S	NaN	0	1
4	896	1	3	Hirvonen, Mrs. Alexander (Helga E Lindqvist)	NaN	1	1	3101298	12.2875	S	NaN	0	3

In [51]:
```
embarked_mapping = {"S": 1, "C": 2, "Q": 3}
test['Embarked'] = test['Embarked'].map(embarked_mapping)
test['Embarked'] = test['Embarked'].map(embarked_mapping)

test.head()
```

Out[51]:

	PassengerId	Survived	Pclass	Name	Sex	SibSp	Parch	Ticket	Fare	Embarked	AgeGroup	CabinBool	Title
0	892	0	3	Kelly, Mr. James	NaN	0	0	330911	7.8292	NaN	NaN	0	1
1	893	1	3	Wilkes, Mrs. James (Ellen Needs)	NaN	1	0	363272	7.0000	NaN	NaN	0	3
2	894	0	2	Myles, Mr. Thomas Francis	NaN	0	0	240276	9.6875	NaN	NaN	0	1
3	895	0	3	Wirz, Mr. Albert	NaN	0	0	315154	8.6625	NaN	NaN	0	1
4	896	1	3	Hirvonen, Mrs. Alexander (Helga E Lindqvist)	NaN	1	1	3101298	12.2875	NaN	NaN	0	3

In [67]:
```
from sklearn.model_selection import train_test_split

predictors = test.drop(['Survived', 'PassengerId'], axis=1)
target = test["Survived"]
x_test, x_val, y_test, y_val = train_test_split(predictors, target, test_size = 0.22, random_state = 0)
```

In [71]:
```
y.shape
```

Out[71]:
```
(105,)
```