Importing the packages

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

```
data = pd.read_csv(r"train.csv")
data
```

	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
...	...	...	...	...	...	...	...	...	...	...	...	...
886	887	0	2	Montvila, Rev. Juozas	male	27.0	0	0	211536	13.0000	NaN	S
887	888	1	1	Graham, Miss. Margaret Edith	female	19.0	0	0	112053	30.0000	B42	S
888	889	0	3	Johnston, Miss. Catherine Helen "Carrie"	female	NaN	1	2	W./C. 6607	23.4500	NaN	S

Next steps: Generate code with data | View recommended plots | New interactive sheet

Display Top 5 Rows of The Dataset

```
data.head(5)
```

	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	female	35.0	1	0	113803	53.1000	C123	S

Next steps: Generate code with data | View recommended plots | New interactive sheet

Display the Last 3 Rows of The Dataset

```
data.tail(3)
```

	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

Find Shape of Our Dataset (Number of Rows & Number of Columns)

```
data.shape
```

```
(891, 12)
```

```
print("Total Number of Rows", data.shape[0])
print("Total Number of Columns", data.shape[1])
```

```
Total Number of Rows 891
Total Number of Columns 12
```

Get Information About Our Dataset Like Total Number Rows, Total Number of Columns, Datatypes of Each Column And Memory Requirement

```
data.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
```

Get Overall Statistics About The Dataframe

```
data.describe()
```

	PassengerId	Survived	Pclass	Age	SibSp	Parch	Fare
count	891.000000	891.000000	891.000000	714.000000	891.000000	891.000000	891.000000
mean	446.000000	0.383838	2.308642	29.699118	0.523008	0.381594	32.204208
std	257.353842	0.486592	0.836071	14.526497	1.102743	0.806057	49.693429
min	1.000000	0.000000	1.000000	0.420000	0.000000	0.000000	0.000000
25%	223.500000	0.000000	2.000000	20.125000	0.000000	0.000000	7.910400
50%	446.000000	0.000000	3.000000	28.000000	0.000000	0.000000	14.454200
75%	668.500000	1.000000	3.000000	38.000000	1.000000	0.000000	31.000000
max	891.000000	1.000000	3.000000	80.000000	8.000000	6.000000	512.329200

```
data.describe(include = 'all')
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
count	891.000000	891.000000	891.000000	891	891	714.000000	891.000000	891.000000	891	891.000000	204	889
unique	NaN	NaN	NaN	891	2	NaN	NaN	NaN	681	NaN	147	3
top	NaN	NaN	NaN	Dooley, Mr. Patrick	male	NaN	NaN	NaN	347082	NaN	G6	S
freq	NaN	NaN	NaN	1	577	NaN	NaN	NaN	7	NaN	4	644
mean	446.000000	0.383838	2.308642	NaN	NaN	29.699118	0.523008	0.381594	NaN	32.204208	NaN	NaN
std	257.353842	0.486592	0.836071	NaN	NaN	14.526497	1.102743	0.806057	NaN	49.693429	NaN	NaN
min	1.000000	0.000000	1.000000	NaN	NaN	0.420000	0.000000	0.000000	NaN	0.000000	NaN	NaN
25%	223.500000	0.000000	2.000000	NaN	NaN	20.125000	0.000000	0.000000	NaN	7.910400	NaN	NaN
50%	446.000000	0.000000	3.000000	NaN	NaN	28.000000	0.000000	0.000000	NaN	14.454200	NaN	NaN
75%	668.500000	1.000000	3.000000	NaN	NaN	38.000000	1.000000	0.000000	NaN	31.000000	NaN	NaN

Data Filtering

```
data.columns
```

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

```
data["Name"]
```

	Name
0	Braund, Mr. Owen Harris
1	Cumings, Mrs. John Bradley (Florence Briggs Th...
2	Heikkinen, Miss. Laina
3	Futrelle, Mrs. Jacques Heath (Lily May Peel)
4	Allen, Mr. William Henry
...	...
886	Montvila, Rev. Juozas
887	Graham, Miss. Margaret Edith
888	Johnston, Miss. Catherine Helen "Carrie"
889	Behr, Mr. Karl Howell
890	Dooley, Mr. Patrick

891 rows × 1 columns

dtype: object

```
data[["Name", "Age"]]
```

	Name	Age
0	Braund, Mr. Owen Harris	22.0
1	Cumings, Mrs. John Bradley (Florence Briggs Th...	38.0
2	Heikkinen, Miss. Laina	26.0
3	Futrelle, Mrs. Jacques Heath (Lily May Peel)	35.0
4	Allen, Mr. William Henry	35.0
...	...	...
886	Montvila, Rev. Juozas	27.0
887	Graham, Miss. Margaret Edith	19.0
888	Johnston, Miss. Catherine Helen "Carrie"	NaN
889	Behr, Mr. Karl Howell	26.0
890	Dooley, Mr. Patrick	32.0

891 rows × 2 columns

Display how many people Survieved

```
sum(data["Survived"] == 1)
```

```
342
```

```
data[data["Survived"] ==1] #Data of the peoples who survieved
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
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
8	9	1	3	Johnson, Mrs. Oscar W (Elisabeth Vilhelmina Berg)	female	27.0	0	2	347742	11.1333	NaN	S
9	10	1	2	Nasser, Mrs. Nicholas (Adele Achem)	female	14.0	1	0	237736	30.0708	NaN	C
...	...	...	...	...	...	...	...	...	...	...	...	...
875	876	1	3	Najib, Miss. Adele Kiamie "Jane"	female	15.0	0	0	2667	7.2250	NaN	C
879	880	1	1	Potter, Mrs. Thomas Jr (Lily Alexenia Wilson)	female	56.0	0	1	11767	83.1583	C50	C

Check Null Values In The Dataset

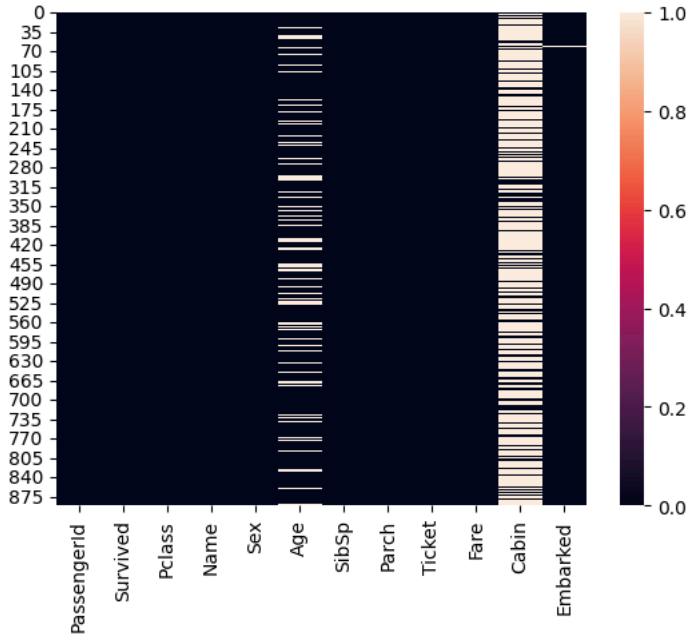
```
data.isnull().sum()
```

	0
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

```
#Will check null values through the heatmap
sns.heatmap(data.isnull())
```

```
<Axes: >
```



data.isnull(): This creates a boolean DataFrame of the same shape as data, where each cell is True if the corresponding cell in data isnull(missing) and False otherwise. sns.heatmap(): This function from the Seaborn library generates a heatmap. It takes the boolean DataFrame from data.isnull() as input. Missing values (represented by True) will be shown as a different color on the heatmap, making it easy to see which columns have missing data and how many missing values they contain relative to other columns.

```
#Now We will check how many % missing values are there in which columns
per = data.isnull().sum() * 100 / len(data)
per
```

	0
PassengerId	0.000000
Survived	0.000000
Pclass	0.000000
Name	0.000000
Sex	0.000000
Age	19.865320
SibSp	0.000000
Parch	0.000000
Ticket	0.000000
Fare	0.000000
Cabin	77.104377
Embarked	0.224467

dtype: float64

We Can see 'Cabin' Column having 77% missing values. So we will drop this columns

```
data.drop('Cabin', axis = 1, inplace = True)
```

axis = 1: This specifies that we are dropping a column inplace = True: This argument modifies the DataFrame directly in place. If it were False (the default), the drop method would return a new DataFrame with the 'Cabin' column removed, and the original data DataFrame would remain unchanged.

```
data.columns
```

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

Handle Missing Values

```
data['Embarked'].mode() #The mode is the value that appears most frequently in a dataset.
```

	Embarked
0	S

dtype: object

So 'S' is the appears most frequently in a dataset. So no we will replace 'Embarked' null values with 'S'

```
data['Embarked'].fillna('S', inplace = True)
```

```
/tmp/ipython-input-21-3433622103.py:1: FutureWarning: A value is trying to be set on a copy of a DataFrame or Series through chained
The behavior will change in pandas 3.0. This inplace method will never work because the intermediate object on which we are setting

For example, when doing 'df[col].method(value, inplace=True)', try using 'df.method({col: value}, inplace=True)' or df[col] = df[col

  data['Embarked'].fillna('S', inplace = True)
```

```
data.isnull().sum()
```

	0
PassengerId	0
Survived	0
Pclass	0
Name	0
Sex	0
Age	177
SibSp	0
Parch	0
Ticket	0
Fare	0
Embarked	0

dtype: int64

Now we will replace 'Age' missing values with Age Avg value

```
data['Age'].fillna(data['Age'].mean(), inplace = True)
```

```
/tmp/ipython-input-23-3104821419.py:1: FutureWarning: A value is trying to be set on a copy of a DataFrame or Series through chained
The behavior will change in pandas 3.0. This inplace method will never work because the intermediate object on which we are setting

For example, when doing 'df[col].method(value, inplace=True)', try using 'df.method({col: value}, inplace=True)' or df[col] = df[col

  data['Age'].fillna(data['Age'].mean(), inplace = True)
```

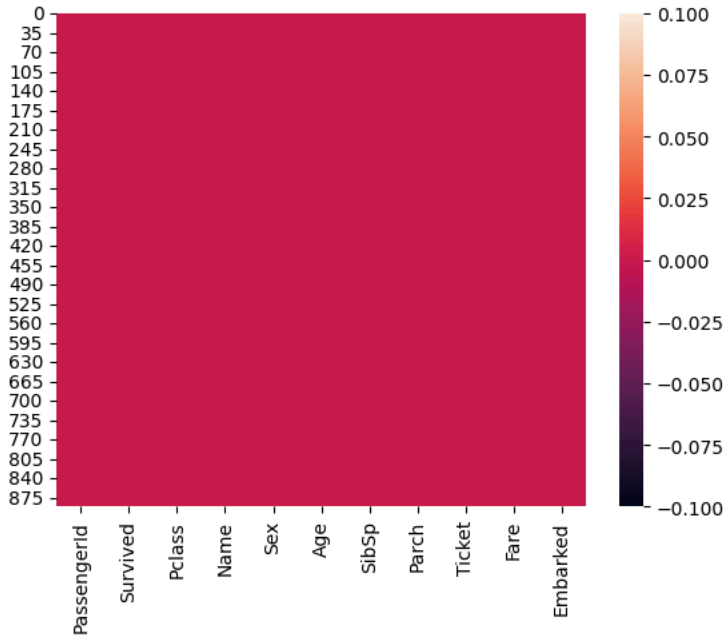
```
data.isnull().sum()
```

	0
PassengerId	0
Survived	0
Pclass	0
Name	0
Sex	0
Age	0
SibSp	0
Parch	0
Ticket	0
Fare	0
Embarked	0

dtype: int64

```
sns.heatmap(data.isnull())
```

```
<Axes: >
```



Now we dont have any missng values

Categorical Data Encoding

Categorical data encoding is the process of converting categorical data (data that can be grouped into categories) into a numerical format that machine learning models can understand and use

```
data.head()
```

	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	female	35.0	1	0	113803	53.1000	S

Next steps: Generate code with data | View recommended plots | New interactive sheet

```
data['Sex'].unique()
```

```
array(['male', 'female'], dtype=object)
```

```
data['Gender'] = data['Sex'].map({'male': 1, 'female': 0})
```

```
data.head(2)
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Embarked	Gender
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	S	1

Next steps: Generate code with data | View recommended plots | New interactive sheet

```
data['Embarked'].unique()
```

```
array(['S', 'C', 'Q'], dtype=object)
```

```
pd.get_dummies(data, columns = ['Embarked'])
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Gender	Embarked_C	Embarked_Q	Emb
0	1	0	3	Braund, Mr. Owen Harris	male	22.000000	1	0	A/5 21171	7.2500	1	False	False	
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.000000	1	0	PC 17599	71.2833	0	True	False	
2	3	1	3	Heikkinen, Miss. Laina	female	26.000000	0	0	STON/O2. 3101282	7.9250	0	False	False	
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.000000	1	0	113803	53.1000	0	False	False	
4	5	0	3	Allen, Mr. William	male	35.000000	0	0	373450	8.0500	1	False	False	

```
data1 = pd.get_dummies(data, columns = ['Embarked'] , drop_first = True)
```

```
data1
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Gender	Embarked_Q	Embarked_S
0	1	0	3	Braund, Mr. Owen Harris	male	22.000000	1	0	A/5 21171	7.2500	1	False	True
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.000000	1	0	PC 17599	71.2833	0	False	False
2	3	1	3	Heikkinen, Miss. Laina	female	26.000000	0	0	STON/O2. 3101282	7.9250	0	False	True
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.000000	1	0	113803	53.1000	0	False	True

Next steps: Generate code with `data1` View recommended plots New interactive sheet

Univariate Analysis

How Many People Survived And How Many Died?

```
data.columns
```

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

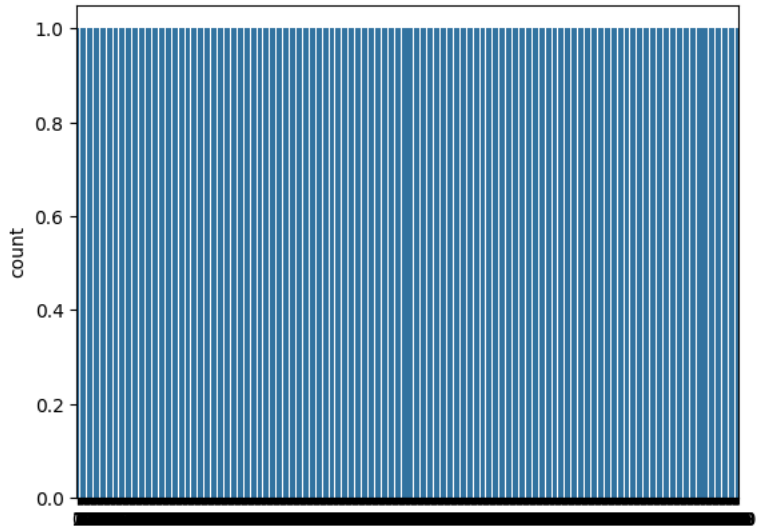
```
data['Survived'].value_counts()
```

Survived	count
0	549
1	342

dtype: int64

```
sns.countplot(data['Survived'])
```

```
<Axes: ylabel='count'>
```



How Many Passengers Were In First Class, Second Class, and Third Class?

```
data.columns
```

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

```
data['Pclass'].value_counts()
```

Pclass	count
3	491
1	216
2	184

dtype: int64

No of Male and Female passengers

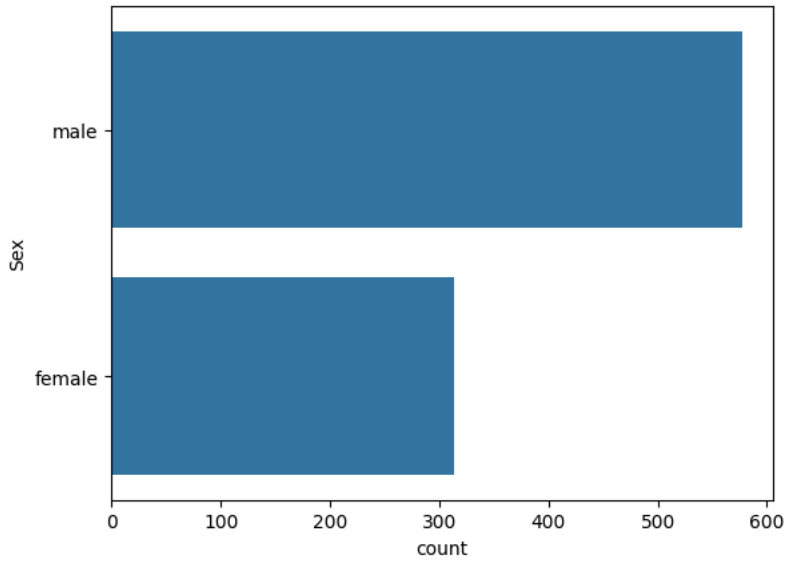
```
data['Sex'].value_counts()
```

Sex	count
male	577
female	314

dtype: int64

```
sns.countplot(data['Sex'])
```

```
<Axes: xlabel='count', ylabel='Sex'>
```

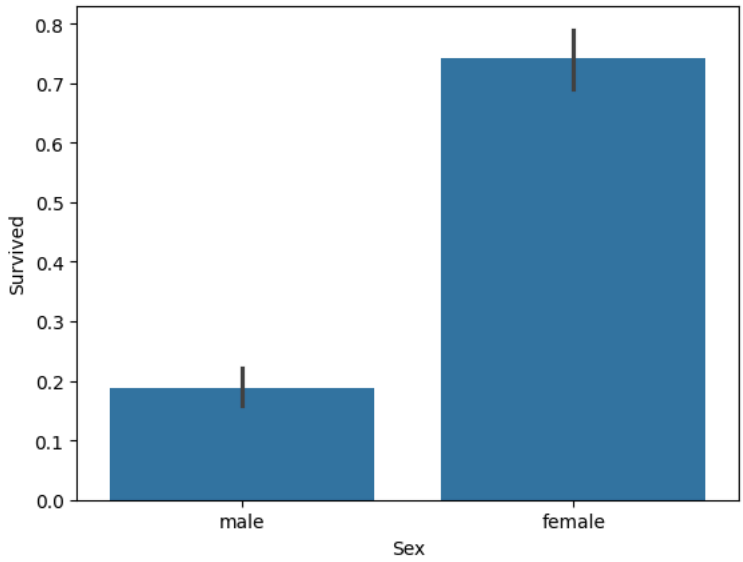


Bivariate Analysis

Who has Better Chance of Survival Male or Female?

```
sns.barplot(x = 'Sex', y = 'Survived', data = data)
```

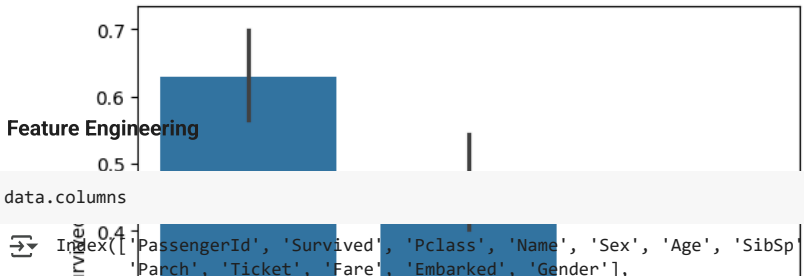
```
<Axes: xlabel='Sex', ylabel='Survived'>
```



Which Passenger Class Has Better Chance of Survival (First, Second, Or Third Class)?

```
sns.barplot(x = 'Pclass', y = 'Survived', data = data)
```

```
<Axes: xlabel='Pclass', ylabel='Survived'>
```



Feature Engineering

```
data.columns
```

```
Index(['PassengerId', 'Survived', 'Pclass', 'Name', 'Sex', 'Age', 'SibSp',
       'Parch', 'Ticket', 'Fare', 'Embarked', 'Gender'],
```