Experiment No. 2
Analyze the Titanic Survival Dataset and apply appropriate regression technique
Date of Performance:
Date of Submission:

Aim: Analyze the Titanic Survival Dataset and apply appropriate Regression Technique.

Objective: Able to perform various feature engineering tasks, apply logistic regression on the given dataset and maximize the accuracy.

Theory:

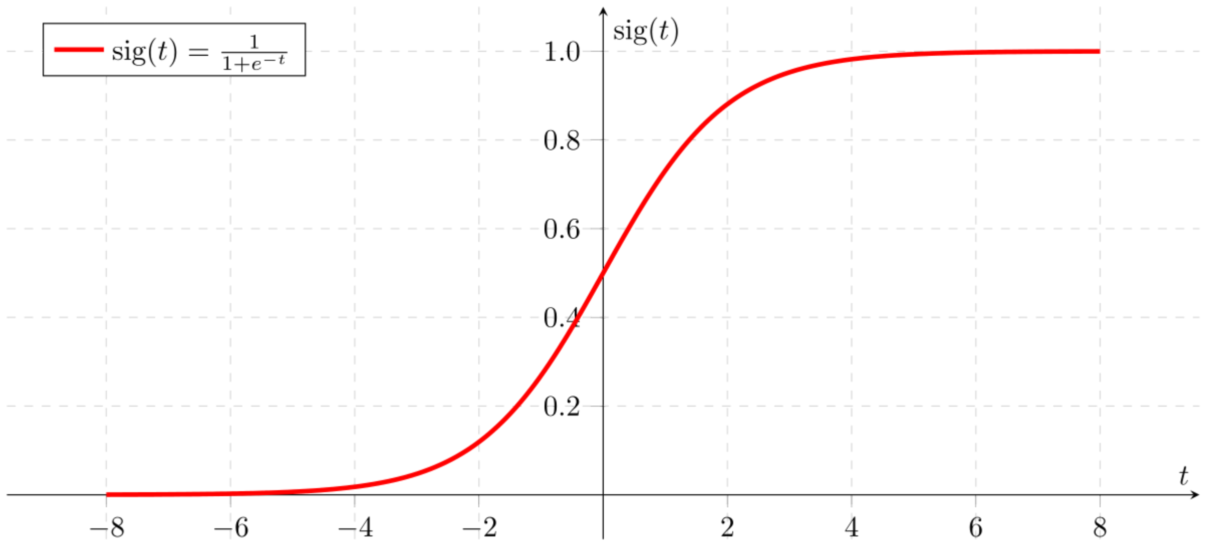
Logistic Regression was used in the biological sciences in early twentieth century. It was then used in many social science applications. Logistic Regression is used when the dependent variable(target) is categorical and is binary in nature. In order to perform binary classification the logistic regression techniques makes use of Sigmoid function.

For example,

To predict whether an email is spam (1) or (0)

Whether the tumor is malignant (1) or not (0)

Consider a scenario where we need to classify whether an email is spam or not. If we use linear regression for this problem, there is a need for setting up a threshold based on which classification can be done. Say if the actual class is malignant, predicted continuous value 0.4 and the threshold value is 0.5, the data point will be classified as not malignant which can lead to serious consequence in real time.



From this example, it can be inferred that linear regression is not suitable for classification problem. Linear regression is unbounded, and this brings logistic regression into picture. Their value strictly ranges from 0 to 1.

Dataset:

The sinking of the Titanic is one of the most infamous shipwrecks in history.

On April 15, 1912, during her maiden voyage, the widely considered "unsinkable" RMS Titanic sank after colliding with an iceberg. Unfortunately, there weren't enough lifeboats for everyone onboard, resulting in the death of 1502 out of 2224 passengers and crew.

While there was some element of luck involved in surviving, it seems some groups of people were more likely to survive than others.

In this challenge, we ask you to build a predictive model that answers the question: "what sorts of people were more likely to survive?" using passenger data (ie name, age, gender, socio-economic class, etc).

Variable	Definition	Key
survival	Survival	0 = No, 1 = Yes
pclass	Ticket class	1 = 1st, 2 = 2nd, 3 = 3rd
sex	Sex	
Age	Age in years	
sibsp	# of siblings / spouses aboard the Titanic	
parch	# of parents / children aboard the Titanic	
ticket	Ticket number	
fare	Passenger fare	
cabin	Cabin number	
embarked	Port of Embarkation	C = Cherbourg, Q = Queenstown, S = Southampton

Variable Notes

pclass: A proxy for socio-economic status (SES)

1st = Upper, 2nd = Middle, 3rd = Lower

age: Age is fractional if less than 1. If the age is estimated, is it in the form of xx.5

sibsp: The dataset defines family relations in this way...,

Sibling = brother, sister, stepbrother, stepsister

Spouse = husband, wife (mistresses and fiancés were ignored)

parch: The dataset defines family relations in this way...

Parent = mother, father

Child = daughter, son, stepdaughter, stepson

Some children travelled only with a nanny, therefore parch=0 for them.

Code:

Conclusion:

1. What features have been chosen to develop the model? Justify the features chosen to determine the survival of a passenger.

   Features chosen are as follows:-

   - Pclass and Fare reflect socio-economic status, affecting priority and resources.
   - Sex aligns with the "women and children first" protocol.
   - Age, SibSp, and Parch relate to vulnerabilities and group dynamics during evacuation.
   - Embarked could be linked to socio-economic factors.
   - Cabin, if available, might provide insight into location-based survival patterns.

2. Comment on the accuracy obtained.

   - Our model achieved an accuracy of approximately 70.27% on the Titanic survival dataset. While this result demonstrates that our model is performing better than random guessing, there's still potential for improvement. To enhance its performance, we could consider refining our feature selection, experimenting with different algorithms, optimizing hyperparameters, and addressing any data preprocessing issues. Exploring advanced techniques like ensemble methods or feature engineering might help us capture more nuanced patterns in the data. Evaluating the model's performance on both training and validation sets, along with potential cross-validation, can provide a more comprehensive understanding of its effectiveness.

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

from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, confusion_matrix, classification_report
from sklearn.model_selection import train_test_split
from sklearn import preprocessing
```

```
df = pd.read_csv('./Titanic-Dataset.csv')
df.head()
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.

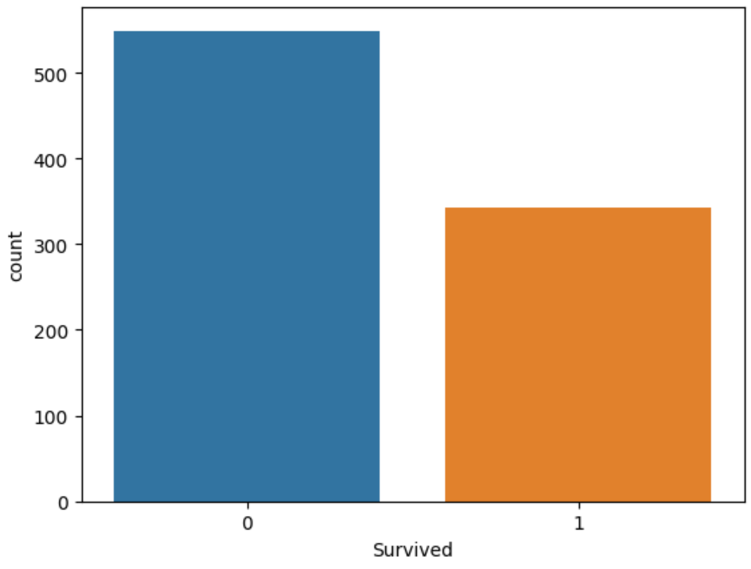
```
df.corr()
```

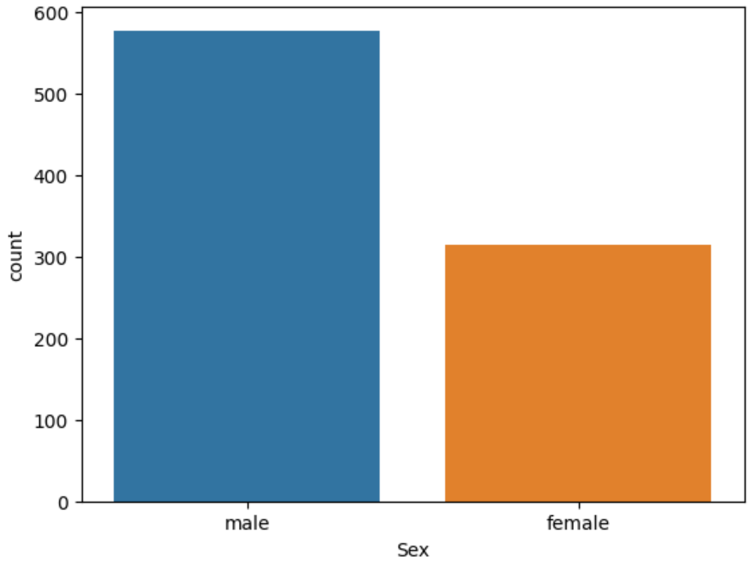
```
<ipython-input-3-2f6f6606aa2c>:1: FutureWarning: The default value of numeric_only i
  df.corr()
```

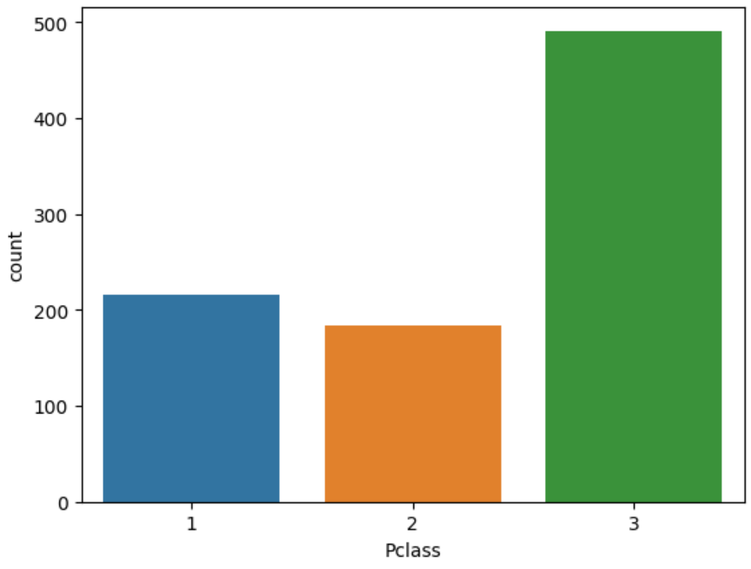
	PassengerId	Survived	Pclass	Age	SibSp	Parch	Far
PassengerId	1.000000	-0.005007	-0.035144	0.036847	-0.057527	-0.001652	0.01265
Survived	-0.005007	1.000000	-0.338481	-0.077221	-0.035322	0.081629	0.25730
Pclass	-0.035144	-0.338481	1.000000	-0.369226	0.083081	0.018443	-0.54950
Age	0.036847	-0.077221	-0.369226	1.000000	-0.308247	-0.189119	0.09606
SibSp	-0.057527	-0.035322	0.083081	-0.308247	1.000000	0.414838	0.15965
Parch	-0.001652	0.081629	0.018443	-0.189119	0.414838	1.000000	0.21622
Fare	0.012658	0.257307	-0.549500	0.096067	0.159651	0.216225	1.00000

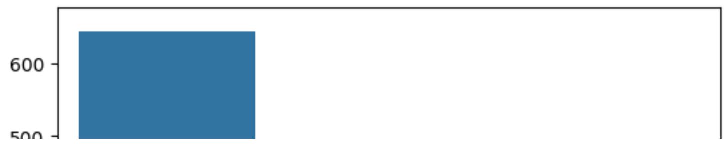
```
def count_plot(feature):
    # visualize
    sns.countplot(x=feature,data=df)
    plt.show()
    print("\n\n")
```

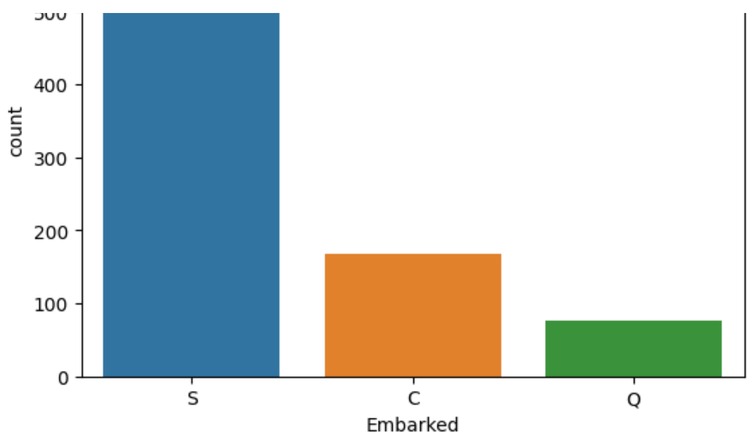
```
#choosing the vairables that will be visualized
categories = ["Survived","Sex","Pclass","Embarked","SibSp", "Parch"]
for x in categories:
    count_plot(x)
```

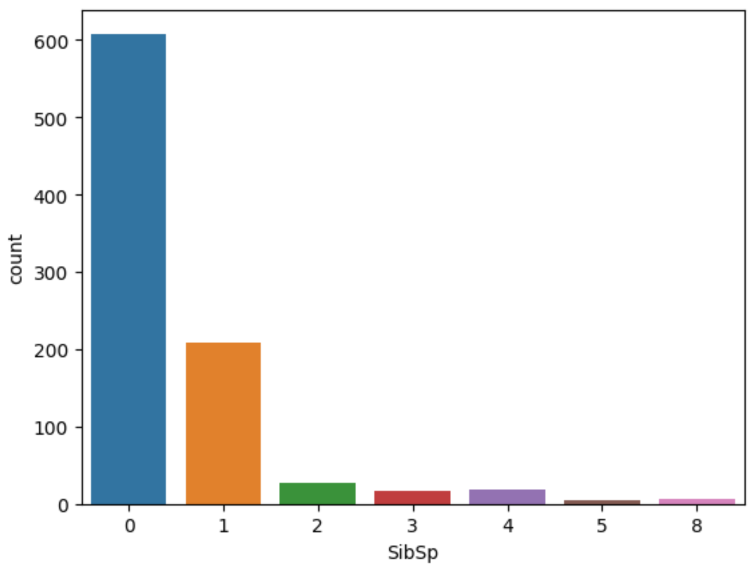

count
0
100
200
300
400
500
0
1
Survived


count
0
100
200
300
400
500
600
male
female
Sex


count
0
100
200
300
400
500
1
2
3
Pclass


600





```
# Null values
df.isnull().sum()

     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
```

```
# Drop columns
df.drop(['PassengerId', 'Name', 'Ticket', 'Cabin'], axis=1, inplace=True)


# Fill null values
df['Age'].fillna(df['Age'].mean(), inplace=True)


# Convert categorical values to numerical
label_encoder = preprocessing.LabelEncoder()

df['Embarked'] = label_encoder.fit_transform(df['Embarked'])
df['Sex'] = label_encoder.fit_transform(df['Sex'])


# drop null values
df.dropna(inplace=True)
df.isnull().sum()
```

```
Survived    0
Pclass      0
Sex         0
Age         0
SibSp       0
Parch       0
Fare        0
Embarked    0
dtype: int64
```

```
df.head()
```

	Survived	Pclass	Sex	Age	SibSp	Parch	Fare	Embarked
1	1	1	0	38.0	1	0	71.2833	0
3	1	1	0	35.0	1	0	53.1000	2
6	0	1	1	54.0	0	0	51.8625	2
10	1	3	0	4.0	1	1	16.7000	2
11	1	1	0	58.0	0	0	26.5500	2

```
# Split data
X = df.drop('Survived', axis=1)
y = df['Survived']

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2)
```

```
# Create model
model = LogisticRegression()

# Train model
model.fit(X_train, y_train)
```

```
▾ LogisticRegression
LogisticRegression()
```

```
# Predict
y_pred = model.predict(X_test)

print("ACCURACY")
print(accuracy_score(y_test, y_pred))

print("Confusion matrix")
print(confusion_matrix(y_test, y_pred))

print("Classification report")
print(classification_report(y_test, y_pred))
```

```
ACCURACY
0.7027027027027027
Confusion matrix
[[ 5  7]
 [ 4 21]]
Classification report
              precision    recall  f1-score   support

           0       0.56      0.42      0.48        12
           1       0.75      0.84      0.79        25

    accuracy                           0.70        37
   macro avg       0.65      0.63      0.63        37
weighted avg       0.69      0.70      0.69        37
```