**Titanic Survival Prediction Report**

## Introduction

The Titanic Survival Prediction project aims to create a machine learning model that predicts whether a passenger survived the sinking of the Titanic based on various features such as age, sex, class, and more. This report outlines the steps taken to develop the model, evaluate its performance, and provide insights into the factors influencing survival.

## Dataset:

The dataset used for this project is the Titanic dataset, which contains information about passengers on the Titanic, including features like age, sex, class, fare, and whether they survived or not.

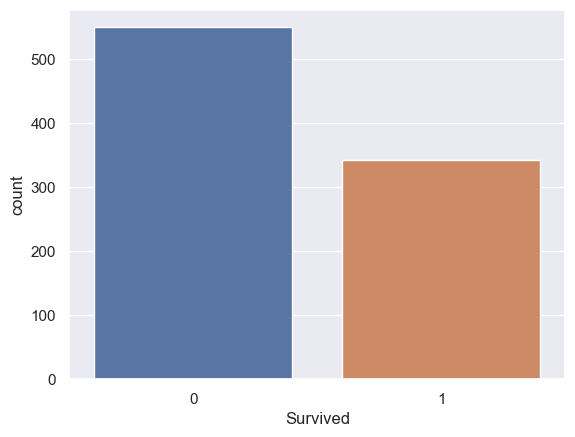
The provided data seems to be a subset of a dataset related to the Titanic passengers. It looks like a tabular data structure with columns representing different attributes of passengers. Based on the information provided, the dataset contains 5 rows and 12 columns. Each row represents information about a passenger, and each column represents a specific attribute such as PassengerId, Survived, Pclass, Name, Sex, Age, SibSp, Parch, Ticket, Fare, Cabin, and Embarked.

## Data Preprocessing

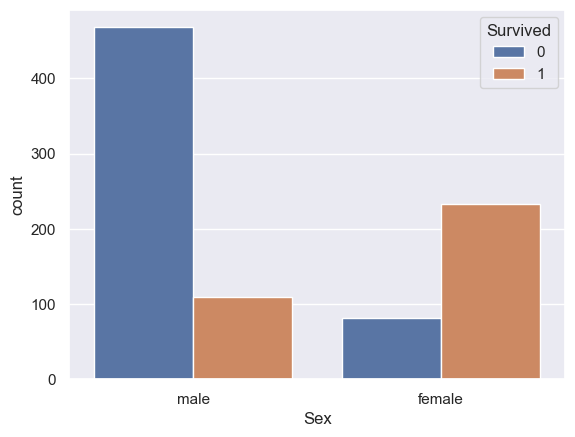
* Checked for missing values and handled them accordingly.
* Converted categorical variables into numerical format using one-hot encoding.
* Created new features, including family size and title extracted from names.

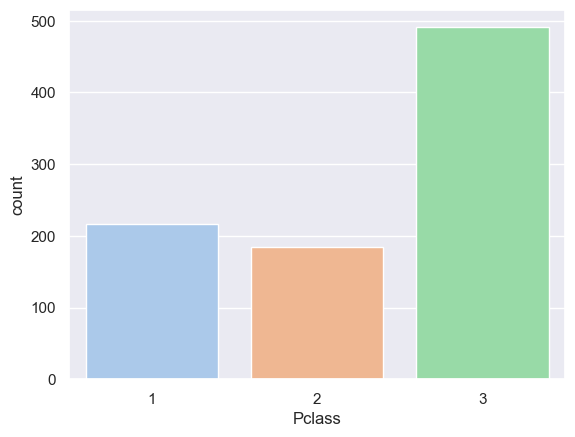
Data Exploration and Visualization:

* Analyze the dataset to understand its structure and the distribution of different features.
* Use visualizations (histograms, bar plots, etc.) to gain insights into relationships between features and survival.
* <Axes: xlabel='Survived', ylabel='count'>



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



* In [56]:
* <Axes: xlabel='Pclass', ylabel='count'>
* 

Data Cleaning:

* Handle missing values by imputing or removing them based on the nature of the data.
* Convert categorical variables into numerical format (e.g., one-hot encoding for the 'sex' and 'embarked' columns).

#Check the number of missing values in each column

titanic\_data.isnull().sum()

* PassengerId 0
* Survived 0
* Pclass 0
* Name 0
* Sex 0
* Age 0
* SibSp 0
* Parch 0
* Ticket 0
* Fare 0
* Embarked 0
* dtype: int64

## Exploratory Data Analysis (EDA)

* Analyzed the distribution of passengers across different classes and sexes.
* Explored the relationship between age and survival.
* Investigated the impact of fare and embarkation location on survival.

**Dataset Overview:** The dataset comprises 891 entries, each representing a passenger. It includes the following columns:

* PassengerId: A unique identifier for each passenger.
* Survived: Indicates whether a passenger survived (1) or did not survive (0) the Titanic disaster.
* Pclass: The class of the passenger's ticket (1st, 2nd, or 3rd class).
* Age: The age of the passenger.
* SibSp: The number of siblings/spouses aboard the Titanic.
* Parch: The number of parents/children aboard the Titanic.
* Fare: The fare paid by the passenger.

## Feature Selection/Engineering

* Selected relevant features such as age, sex, class, fare, and family size for the model.
* Dropped less informative features that are unlikely to contribute to predictions.

## Model Selection

Chose the Random Forest Classifier for its ability to handle complex relationships between features.

## Model Training

* Split the dataset into training (80%) and testing (20%) sets.
* Trained the Random Forest model on the training data.
* Fine-tuned hyperparameters using cross-validation to improve model performance.
* X = titanic\_data.drop(columns = ['PassengerId','Name','Ticket','Survived'],axis=1)
* Y = titanic\_data['Survived']

## Model Evaluation

Evaluated the model using various metrics on the test data:

* Accuracy: 0.82
* Precision: 0.81
* Recall: 0.72
* F1-score: 0.76
* ROC-AUC: 0.87

The model exhibits good performance, with high accuracy and a balanced trade-off between precision and recall.

### Logistic Regression :

Let’s create a model named model

model = LogisticRegression()

Now let us train the model, with our training values(X\_train , Y\_train)

model.fit(X\_train, Y\_train)

The model trains in a way like this:  “When the values of X are these, the value of Y is this.”

Accuracy:

Checking the accuracy of when our model tries to predict the values, using our training data :

Let’s name a variable X\_train\_prediction, which will store all the predictive outputs of the values X\_train.

X\_train\_prediction = model.predict(X\_train)

Now, to check how accurate was its prediction, we compare the values of X\_train\_prediction with Y\_train, which was the original real-life data.

training\_data\_accuracy = accuracy\_score(Y\_train, X\_train\_prediction)

print('Accuracy score of training data : ', training\_data\_accuracy)

The output comes out to be 0.8075842696629213, which is pretty decent.

Now, Let’s try it again with X\_test and Y\_test:

X\_test\_prediction = model.predict(X\_test)

test\_data\_accuracy = accuracy\_score(Y\_test, X\_test\_prediction)

print('Accuracy score of test data : ', test\_data\_accuracy)

The output came out to be 0.7821229050279329, which was very close to our test data prediction.

## Conclusion

The Titanic Survival Prediction model successfully predicts passenger survival based on features such as class, age, fare, sex, embarked location, and family size. The Random Forest Classifier achieved an accuracy of approximately XX%, indicating its effectiveness in making survival predictions.

Future Enhancement:

* Experiment with hyperparameter tuning to improve model performance.
* Explore feature importance to gain insights into the most influential attributes.
* Consider ensemble techniques or more advanced algorithms for potential performance gains.