```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, classification_report, confusion_matrix
```

In [2]:

```
data = pd.read_csv('titanic_dataset.csv')
data.columns
```

Out[2]:

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

In [3]:

```
data.info()
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 418 entries, 0 to 417
Data columns (total 12 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   PassengerId  418 non-null    int64
 1   Pclass       418 non-null    int64
 2   Name         418 non-null    object
 3   Sex          418 non-null    object
 4   Age          332 non-null    float64
 5   SibSp        418 non-null    int64
 6   Parch        418 non-null    int64
 7   Ticket       418 non-null    object
 8   Fare         417 non-null    float64
 9   Cabin        91 non-null     object
 10  Embarked     418 non-null    object
 11  Survived     418 non-null    int64
dtypes: float64(2), int64(5), object(5)
memory usage: 39.3+ KB
```

In [4]:

```
# Data preprocessing
# Handle missing data
data['Age'].fillna(data['Age'].mean(), inplace=True)
data['Fare'].fillna(data['Fare'].mean(), inplace=True)
data['Embarked'].fillna(data['Embarked'].mode()[0], inplace=True)
```

In [5]:

```
#Encoding variables
data = pd.get_dummies(data, columns=['Sex', 'Embarked'], drop_first=True)
```

In [6]:

```
# Create the feature matrix (X) and target variable (y)
X = data.drop(['Survived', 'Name', 'Ticket', 'Cabin'], axis=1)
y = data['Survived']
```

In [7]:

```
#Training Model
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=37)
```

In [8]:

```
# Standardize features (optional but can help with model convergence)
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)
```

In [9]:

```
# Create and train a logistic regression model
model = LogisticRegression()
```

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

Out[9]:

```
LogisticRegression
LogisticRegression()
```

In [10]:

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

In [11]:

```
accuracy = accuracy_score(y_test, y_pred)
print(f'Accuracy: {accuracy:.2f}')
```

```
Accuracy: 1.00
```

In [12]:

```
print(classification_report(y_test, y_pred))
print(confusion_matrix(y_test, y_pred))
```

```
              precision    recall  f1-score   support

           0       1.00      1.00      1.00        54
           1       1.00      1.00      1.00        30

    accuracy                           1.00        84
   macro avg       1.00      1.00      1.00        84
weighted avg       1.00      1.00      1.00        84

[[54  0]
 [ 0 30]]
```

In [ ]: