In [2]:
```
#import library
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
```

In [3]:
```
# Load the Titanic dataset
titanic_data = pd.read_csv('C:/Users/squir/Downloads/tested.csv')
```

In [4]:
```
# Display the first few rows of the dataset to understand its structure
print(titanic_data.head())
```

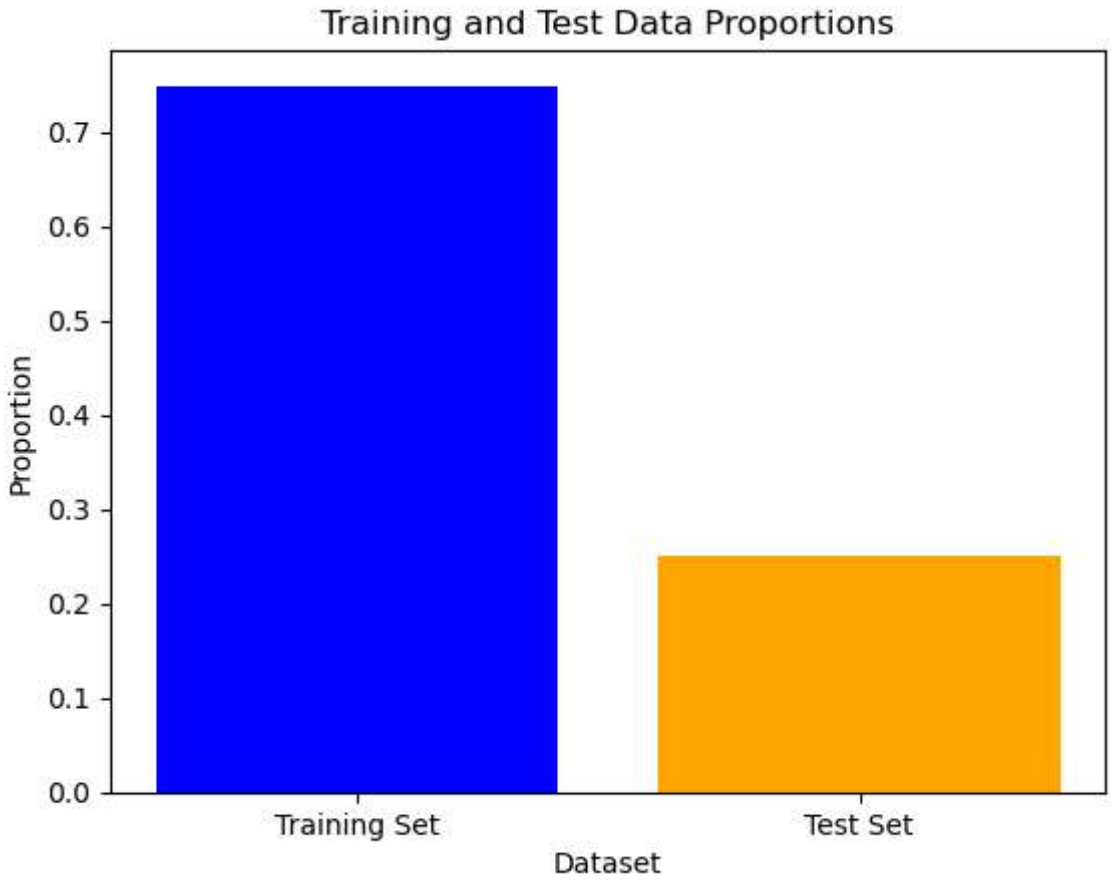
```
   PassengerId  Survived  Pclass  \
0          892         0       3
1          893         1       3
2          894         0       2
3          895         0       3
4          896         1       3

                                          Name     Sex   Age  SibSp  Parch
\
0                              Kelly, Mr. James    male  34.5      0      0
1              Wilkes, Mrs. James (Ellen Needs)  female  47.0      1      0
2                     Myles, Mr. Thomas Francis    male  62.0      0      0
3                              Wirz, Mr. Albert    male  27.0      0      0
4  Hirvonen, Mrs. Alexander (Helga E Lindqvist)  female  22.0      1      1

    Ticket     Fare Cabin Embarked
0   330911   7.8292   NaN        Q
1   363272   7.0000   NaN        S
2   240276   9.6875   NaN        Q
3   315154   8.6625   NaN        S
4  3101298  12.2875   NaN        S
```

In [5]:
```
# Define features (X) and target variable (y)
X = titanic_data.drop('Survived', axis=1)
y = titanic_data['Survived']
```

In [6]:
```
# Split the data into training and test sets (75% training, 25% test)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.25, rand
```

In [7]:
```
# Check the proportions and visualize using a bar graph
proportions = [len(X_train) / len(titanic_data), len(X_test) / len(titanic_dat
labels = ['Training Set', 'Test Set']
```

In [8]:
```
# Plotting the bar graph
plt.bar(labels, proportions, color=['blue', 'orange'])
plt.xlabel('Dataset')
plt.ylabel('Proportion')
plt.title('Training and Test Data Proportions')
plt.show()
```



In [9]:
```
# Display the total number of records in the training set
num_records_training_set = X_train.shape[0]
print("Total number of records in the training data set:", num_records_trainir
```

```
Total number of records in the training data set: 313
```

In [10]:
```
#Importing Library
from scipy import stats
```

In [13]:
```
# Calculate the mean of the 'Survived' column for training and test sets
mean_survived_train = y_train.mean()
mean_survived_test = y_test.mean()
```

In [14]:
```
# Perform a two-sample t-test
t_stat, p_value = stats.ttest_ind(y_train, y_test)
```

In [16]:
```
# Display the means and t-test results
print("Mean of 'Survived' in Training Set:", mean_survived_train)
print("Mean of 'Survived' in Test Set:", mean_survived_test)
print("\nT-test results:")
print("T-statistic:", t_stat)
print("P-value:", p_value)
```

```
Mean of 'Survived' in Training Set: 0.3706070287539936
Mean of 'Survived' in Test Set: 0.34285714285714286

T-test results:
T-statistic: 0.5104439211062601
P-value: 0.610011234153832
```

In [17]:
```
# Calculate the mean of the 'Survived' column for training and test sets
mean_survived_train = round(y_train.mean())
mean_survived_test = round(y_test.mean())

# Perform a two-sample t-test
t_stat, p_value = stats.ttest_ind(y_train, y_test)

# Display the means and t-test results as integers
print("Mean of 'Survived' in Training Set:", int(mean_survived_train))
print("Mean of 'Survived' in Test Set:", int(mean_survived_test))
print("\nT-test results:")
print("T-statistic:", t_stat)
print("P-value:", p_value)
```

```
Mean of 'Survived' in Training Set: 0
Mean of 'Survived' in Test Set: 0

T-test results:
T-statistic: 0.5104439211062601
P-value: 0.610011234153832
```