I wrote a Python code that reads the Titanic dataset and performs some data analysis and pre-processing, followed by training a logistic regression model and evaluating its performance on the test set.

Data Analysis

First, I decide to create a new "Minor" column in both the train and test data frames. This is done using the *apply()* function and a lambda function that returns 1 if the "Age" column is less than 18, indicating that the passenger is a minor, and 0 otherwise.

The number of passengers who survived for each sex is then calculated using the groupby() function. We can see that females had a way higher chance of survival compared to men.

```
Survived = 1 & Died = 0
Sex  Survived
female  0           81
        1          233
male    0          468
        1          109
dtype: int64
Percentage of men who survived: 19%
Percentage of women who survived: 74%
```

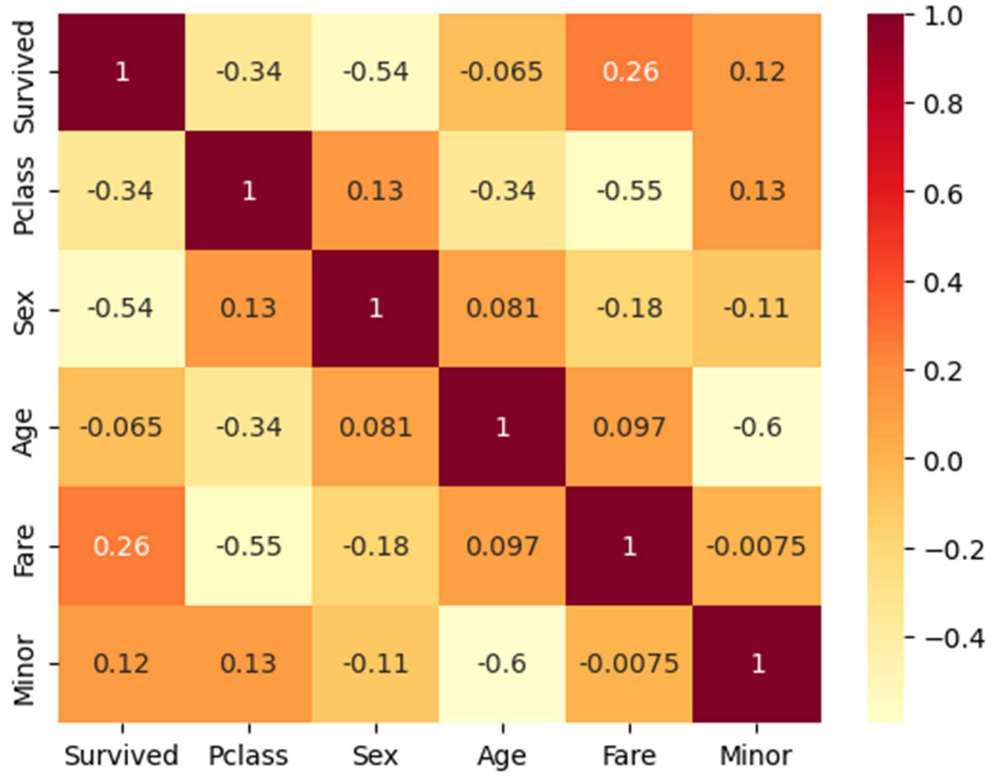
I then used the groupby() function to calculate the number of passengers who survived and did not survive for each class. The results suggest that passengers who paid a higher fare had a higher chance of survival.

```
Pclass  Survived
1       0           80
        1          136
2       0           97
        1           87
3       0          372
        1          119
dtype: int64
Percentage of third who survived: 25%
Percentage of second who survived: 42%
Percentage of first who survived: 43%
```

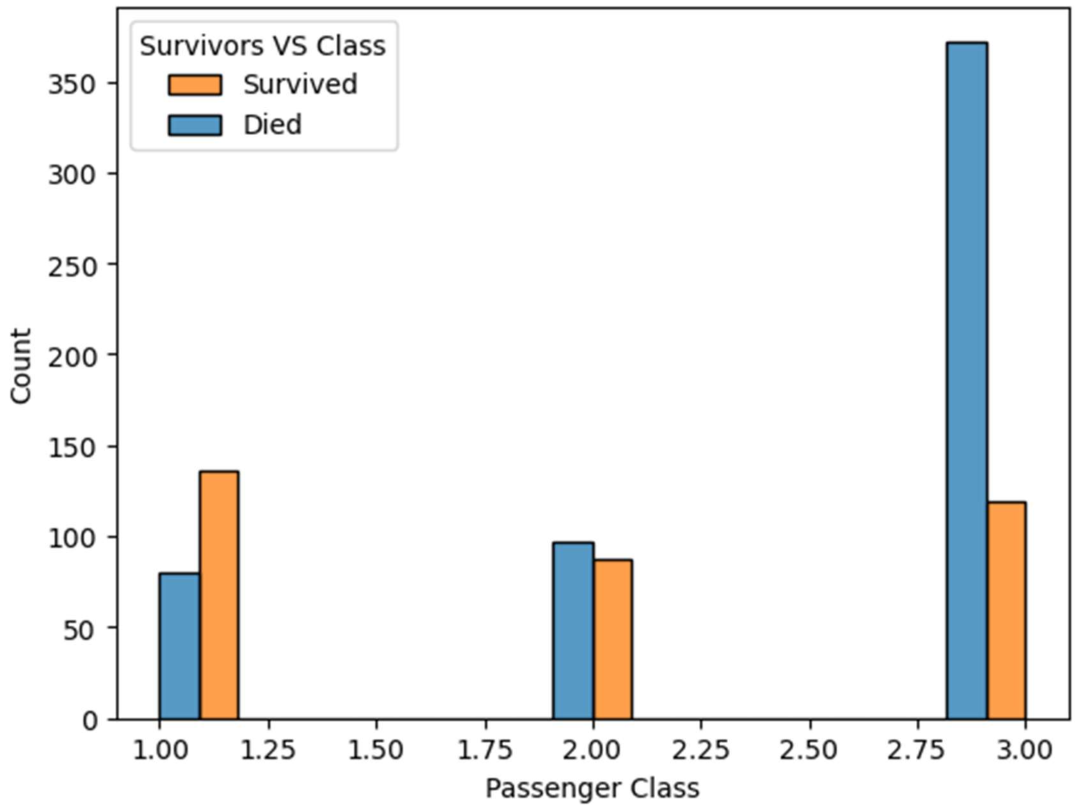
I continued by grouping the "Minor" and "Survived" columns. As expected, we can see that minors had a higher chance of surviving.

```
Pclass  Survived
1       0           80
        1          136
2       0           97
        1           87
3       0          372
        1          119
dtype: int64
Percentage of adults who survived: 36%
Percentage of minors who survived: 54%
```

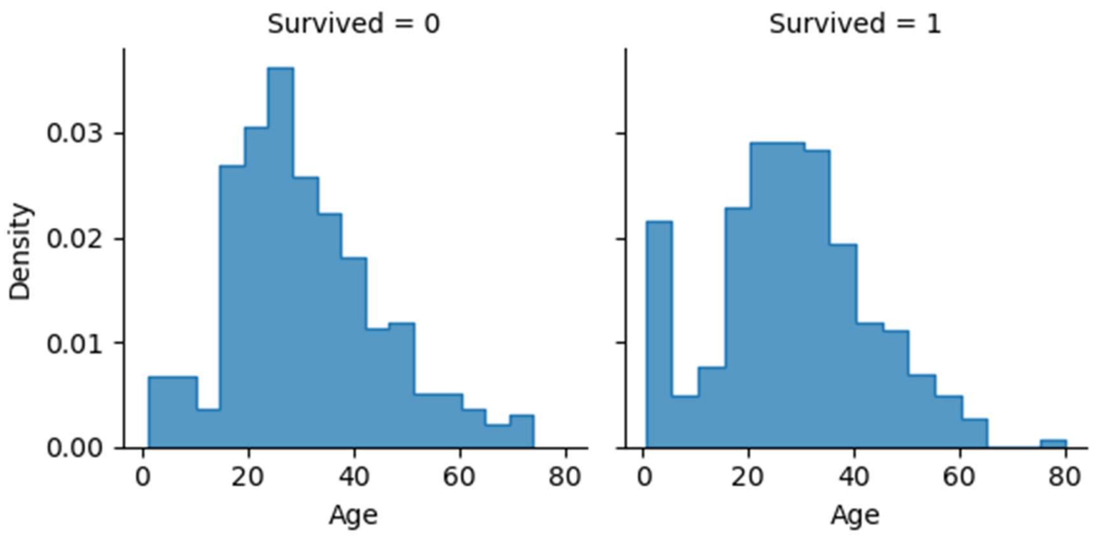
After, I decided to analyze the data using some graphs.
The first plot is a correlation matrix heatmap of the numerical features. This helps to visualize the correlation between the features and the target variable Survived.



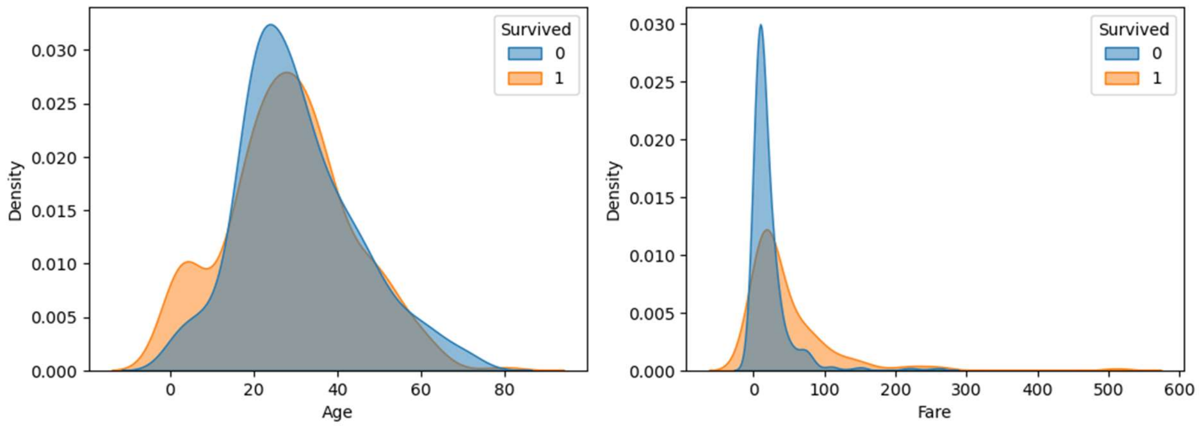
The next plot is a histogram of passenger class with the hue set to Survived. This plot shows the number of passengers who survived and died for each passenger class. It helps to visualize the effect of the passenger class on survival.



Next, I created a FacetGrid with Survived as the column variable and Age as the feature variable. It helps to visualize the distribution of ages for both survived and dead passengers.



Finally, I created two KDE plots of age and fare. These plots show the density estimate of the age and fare distributions for survived and dead passengers. The plot of age suggests that younger and richer passengers had a higher chance of survival.



Pre-processing

After the data analysis, the next step is to pre-process the data, which involves:

Removing irrelevant features such as the "PassengerId", "Name", "Ticket", and "Cabin" columns. We used the *drop()* function.

Handling missing values, for example the "Embarked" and "Fare" columns have missing values, which are filled with the *fillna()* function. The "Age" column in both the train and test frames also has missing values, which are replaced with the median age of the respective datasets.

Encoding categorical variables: the "Sex" and "Embarked" columns are encoded using the *LabelEncoder*.

Splitting the data into features and target variable: the "Survived" column is separated and stored in the y_train variable, while the rest of the columns are stored in the X_train variable. The test dataframe is similarly split into X_test and y_test variables.

Evaluating the classifier

After pre-processing the data, the next step is to train the logistic regression model. This is done using a pipeline that includes standardizing the features and training the logistic regression model using *LogisticRegression*. The hyperparameter C of the logistic regression model is optimized using *GridSearchCV*, which performs a grid search over a range of values for C and selects the best value based on cross-validation.

Evaluating the classifier

Finally, I evaluated the performance of the logistic regression model using the *accuracy_score(), precision_score(), recall_score(),* and *f1_score()* functions.

```
Accuracy: 0.9617224880382775
Precision: 0.9473684210526315
Recall: 0.9473684210526315
F1-score: 0.9473684210526315
```