Problem Statement:perform data cleaning and exploratory data analysis on the given titanic dataset from kaggle. Explore the relationship between variables and identify patterns and trends in the data. Dataset : Titanic dataset About the dataset: The Titanic dataset provided contains information on passengers aboard the RMS Titanic, including details such as their names, ages, genders, ticket class, number of siblings/spouses aboard, number of parents/children aboard, ticket number, fare, cabin number, and embarkation point. This dataset is often used for predictive modeling and analysis tasks, such as predicting survival outcomes based on various factors. It includes a mix of categorical and numerical data, providing a comprehensive view of the passengers and their circumstances during the ill-fated voyage in 1912.

In [ ]:
```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
# Load the Students performance dataset
file_path = 'c:\\Users\\Admin\\Downloads\\Titanic.csv'
data = pd.read_csv(file_path)
print(data)
```

```
     PassengerId  Survived  Pclass  \
0            892         0       3
1            893         1       3
2            894         0       2
3            895         0       3
4            896         1       3
..           ...       ...     ...
413         1305         0       3
414         1306         1       1
415         1307         0       3
416         1308         0       3
417         1309         0       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
..                                            ...     ...   ...    ...    ...
413                            Spector, Mr. Woolf    male   NaN      0      0
414                  Oliva y Ocana, Dona. Fermina  female  39.0      0      0
415                  Saether, Mr. Simon Sivertsen    male  38.5      0      0
416                           Ware, Mr. Frederick    male   NaN      0      0
417                     Peter, Master. Michael J    male   NaN      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
..                  ...       ...   ...      ...
413           A.5. 3236    8.0500   NaN        S
414            PC 17758  108.9000  C105        C
415  SOTON/O.Q. 3101262    7.2500   NaN        S
416              359309    8.0500   NaN        S
417                2668   22.3583   NaN        C

[418 rows x 12 columns]
```

In [ ]:

```
# Basic EDA
# Dimensions of the dataset
print(f"The dataset contains {data.shape[0]} rows and {data.shape[1]} columns.")
# Data types and missing values
data.info()
```

```
The dataset contains 418 rows and 12 columns.
<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   Survived     418 non-null    int64  
 2   Pclass       418 non-null    int64  
 3   Name         418 non-null    object 
 4   Sex          418 non-null    object 
 5   Age          332 non-null    float64
 6   SibSp        418 non-null    int64  
 7   Parch        418 non-null    int64  
 8   Ticket       418 non-null    object 
 9   Fare         417 non-null    float64
 10  Cabin        91 non-null     object 
 11  Embarked     418 non-null    object 
dtypes: float64(2), int64(5), object(5)
memory usage: 39.3+ KB
```

In [ ]:
```
# Check for missing values
missing_data = data.isnull().sum()
print("Missing Data:\n", missing_data)
```

```
Missing Data:
 PassengerId      0
Survived         0
Pclass           0
Name             0
Sex              0
Age             86
SibSp            0
Parch            0
Ticket           0
Fare             1
Cabin          327
Embarked         0
dtype: int64
```

Here Age,Fare and Cabin have missing values

In [ ]:
```
#Handle Missing Values
# Fill missing 'Age' with median
data['Age'].fillna(data['Age'].median(), inplace=True)

# Fill missing 'Fare' with mean
data['Fare'].fillna(data['Fare'].mean(), inplace=True)


# Drop 'Cabin' column as it is not relevant for analysis
data.drop(columns=['Cabin'], inplace=True)
```

In [ ]:
```
# Check again for any remaining missing values
print(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
```

In [ ]:
```
# EDA
# Summary statistics
summary_stats = data.describe()
print("Summary Statistics:\n", summary_stats)
```

```
Summary Statistics:
        PassengerId    Survived      Pclass         Age       SibSp  \
count   418.000000  418.000000  418.000000  418.000000  418.000000
mean   1100.500000    0.363636    2.265550   29.599282    0.447368
std     120.810458    0.481622    0.841838   12.703770    0.896760
min     892.000000    0.000000    1.000000    0.170000    0.000000
25%     996.250000    0.000000    1.000000   23.000000    0.000000
50%    1100.500000    0.000000    3.000000   27.000000    0.000000
75%    1204.750000    1.000000    3.000000   35.750000    1.000000
max    1309.000000    1.000000    3.000000   76.000000    8.000000

            Parch        Fare
count  418.000000  418.000000
mean     0.392344   35.627188
std      0.981429   55.840500
min      0.000000    0.000000
25%      0.000000    7.895800
50%      0.000000   14.454200
75%      0.000000   31.500000
max      9.000000  512.329200
```

In [ ]:
```
#correlation matrix
# Selecting numerical columns
data_numerical = data.select_dtypes(include=[np.number])
print(data_numerical)
```

```
     PassengerId  Survived  Pclass   Age  SibSp  Parch      Fare
0            892         0       3  34.5      0      0    7.8292
1            893         1       3  47.0      1      0    7.0000
2            894         0       2  62.0      0      0    9.6875
3            895         0       3  27.0      0      0    8.6625
4            896         1       3  22.0      1      1   12.2875
..           ...       ...     ...   ...    ...    ...       ...
413         1305         0       3  27.0      0      0    8.0500
414         1306         1       1  39.0      0      0  108.9000
415         1307         0       3  38.5      0      0    7.2500
416         1308         0       3  27.0      0      0    8.0500
417         1309         0       3  27.0      1      1   22.3583

[418 rows x 7 columns]
```

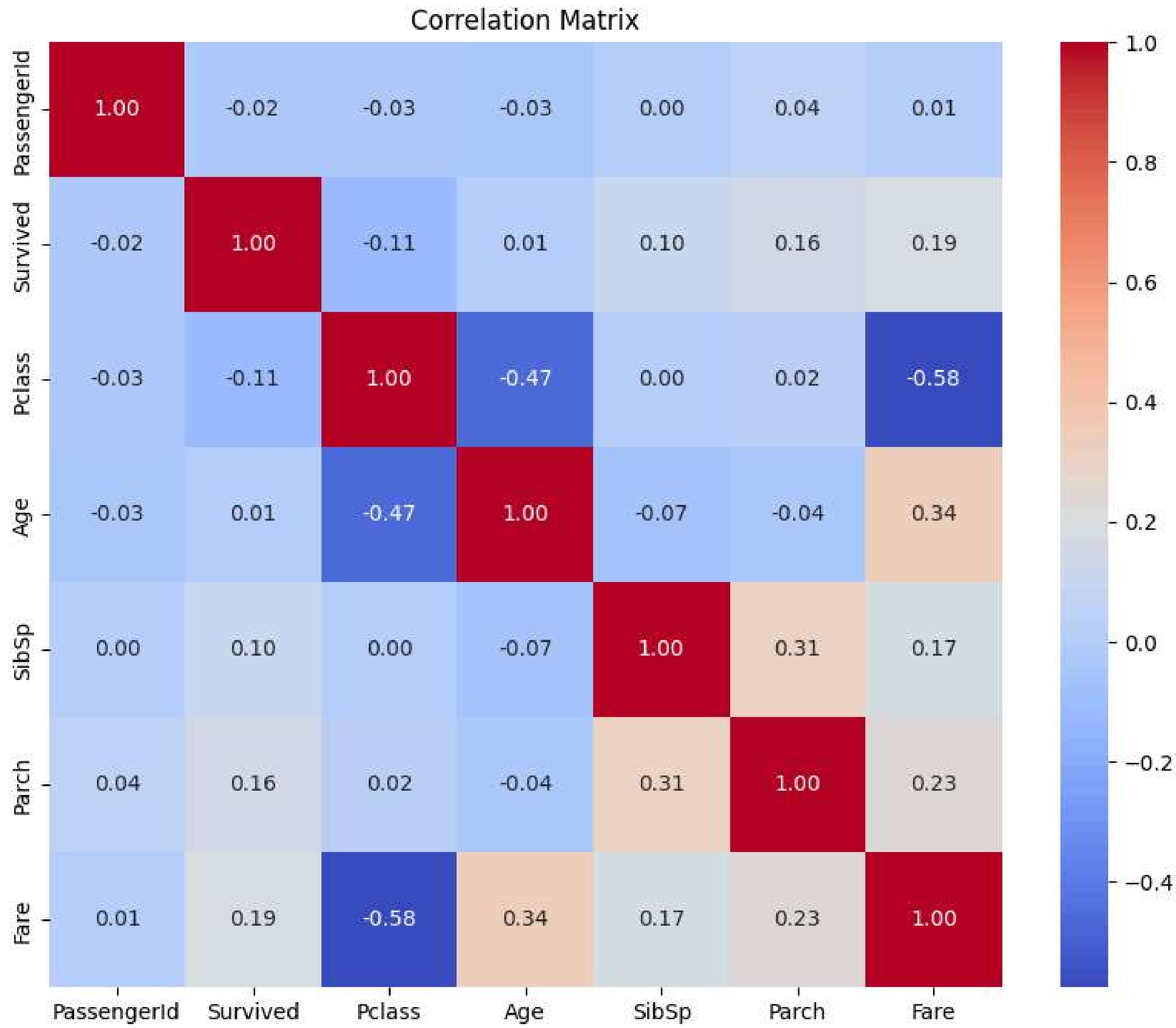
In [ ]:

```python
# Correlation matrix
correlation_matrix = data_numerical.corr()
print(correlation_matrix)
```

```
             PassengerId  Survived    Pclass       Age     SibSp     Parch  \
PassengerId     1.000000 -0.023245 -0.026751 -0.031447  0.003818  0.043080
Survived       -0.023245  1.000000 -0.108615  0.008035  0.099943  0.159120
Pclass         -0.026751 -0.108615  1.000000 -0.467853  0.001087  0.018721
Age            -0.031447  0.008035 -0.467853  1.000000 -0.071197 -0.043731
SibSp           0.003818  0.099943  0.001087 -0.071197  1.000000  0.306895
Parch           0.043080  0.159120  0.018721 -0.043731  0.306895  1.000000
Fare            0.008209  0.191382 -0.576619  0.344627  0.171488  0.230001

                 Fare
PassengerId  0.008209
Survived     0.191382
Pclass      -0.576619
Age          0.344627
SibSp        0.171488
Parch        0.230001
Fare         1.000000
```

In [ ]:

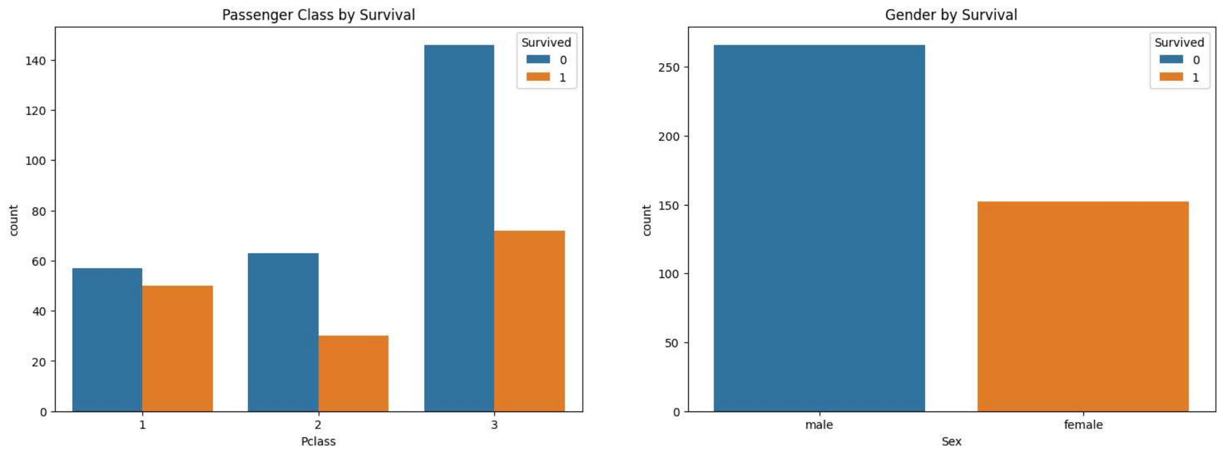
```python
# Heatmap
plt.figure(figsize=(10, 8))
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', fmt=".2f")
plt.title('Correlation Matrix')
plt.show()
```




The heatmap indicates that Pclass has a significant inverse relationship with both Fare and Age, and Fare shows a moderate positive correlation with Age. Other features show weaker correlations with each other.

In [ ]:

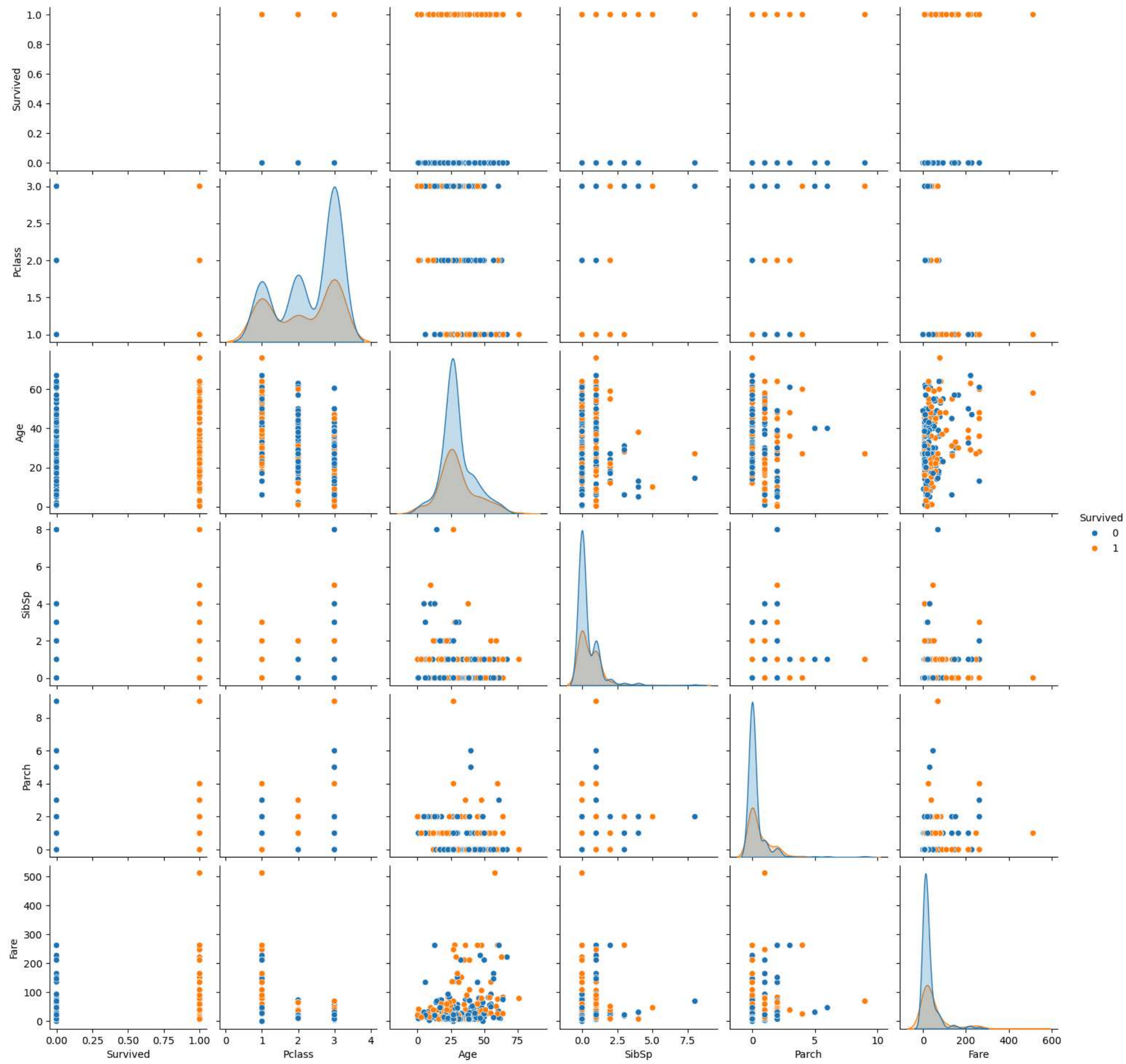
```
fig, axes = plt.subplots(nrows=1, ncols=2, figsize=(18, 6))

# Bar plot for 'Pclass' by 'Survived'
sns.countplot(data=data, x='Pclass', hue='Survived', ax=axes[0])
axes[0].set_title('Passenger Class by Survival')

# Bar plot for 'Sex' by 'Survived'
sns.countplot(data=data, x='Sex', hue='Survived', ax=axes[1])
axes[1].set_title('Gender by Survival')

plt.show()
```



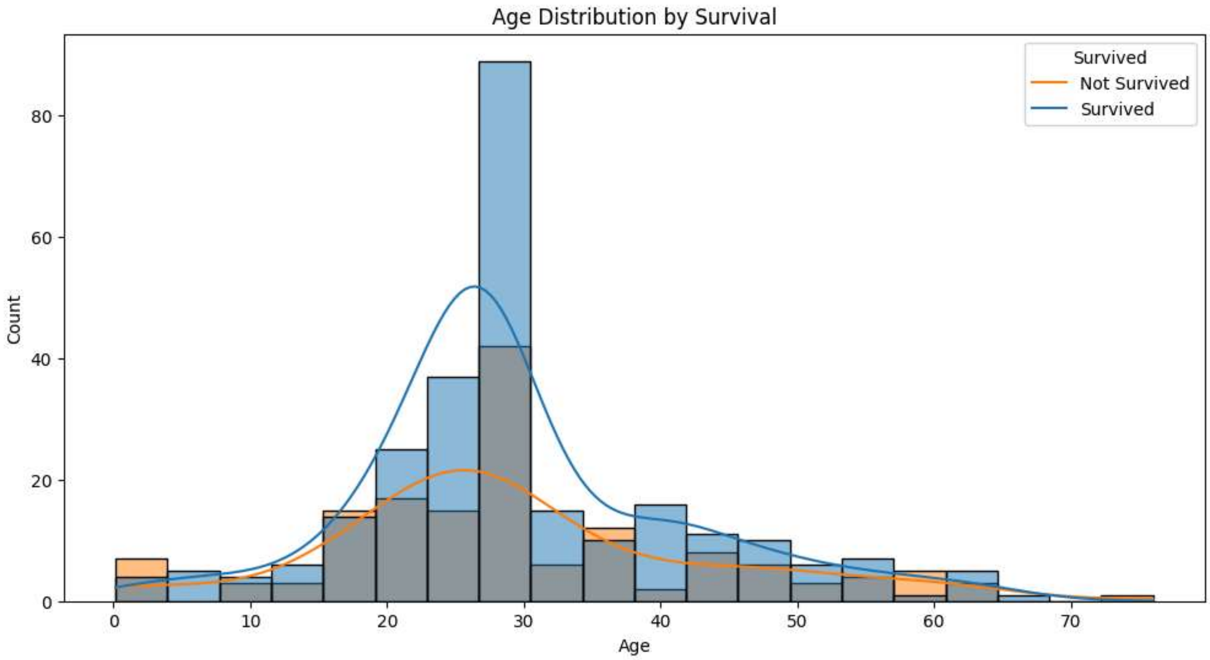

Higher survival rates are observed in first class, while the lowest survival rates are in third class. Female passengers had a much higher survival rate compared to male passengers.

In [ ]:
```
# Pairplot
sns.pairplot(data=data, vars=["Survived", "Pclass", "Age", "SibSp", "Parch", "Fare"
plt.show()
```

```
c:\Users\Admin\Desktop\TRIAL\.venv\lib\site-packages\seaborn\axisgrid.py:123: UserWa
rning: The figure layout has changed to tight
  self._figure.tight_layout(*args, **kwargs)
```




In [ ]:
```
import matplotlib.pyplot as plt
import seaborn as sns

# Create the histogram for 'Age' by 'Survived'
plt.figure(figsize=(12, 6))
sns.histplot(data=data, x='Age', hue='Survived', bins=20, kde=True)
plt.title('Age Distribution by Survival')
plt.xlabel('Age')
plt.ylabel('Count')
plt.legend(title='Survived', labels=['Not Survived', 'Survived'])
plt.show()
```




The histogram shows that younger passengers, especially children, had higher survival rates. Passengers in their 20s and 30s formed the largest group and had a balanced distribution of survival and non-survival. Older passengers had lower survival rates overall, but there are still some survivors in the older age groups.

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]: