Task 2

Perform data cleaning and data exploratory data analysis (EDA) on a dataset of your choice, such as the Titanic datasets from kaggle. Explore the relationship between variables and identify patterns and trends in the data.

In [1]:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

Load Dataset :

In [2]:

```
data = pd.read_csv('titanic.csv')
data.head()
```

Out[2]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	S
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN	S

Exploratory Data Analysis (EDA):

In [3]:
```
data.info()                    #Getting information of data
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 891 entries, 0 to 890
Data columns (total 12 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   PassengerId  891 non-null    int64
 1   Survived     891 non-null    int64
 2   Pclass       891 non-null    int64
 3   Name         891 non-null    object
 4   Sex          891 non-null    object
 5   Age          714 non-null    float64
 6   SibSp        891 non-null    int64
 7   Parch        891 non-null    int64
 8   Ticket       891 non-null    object
 9   Fare         891 non-null    float64
 10  Cabin        204 non-null    object
 11  Embarked     889 non-null    object
dtypes: float64(2), int64(5), object(5)
memory usage: 83.7+ KB
```

In [4]:
```
data.describe().T                    #Statistical description of data
```

Out[4]:

	count	mean	std	min	25%	50%	75%	max
PassengerId	891.0	446.000000	257.353842	1.00	223.5000	446.0000	668.5	891.0000
Survived	891.0	0.383838	0.486592	0.00	0.0000	0.0000	1.0	1.0000
Pclass	891.0	2.308642	0.836071	1.00	2.0000	3.0000	3.0	3.0000
Age	714.0	29.699118	14.526497	0.42	20.1250	28.0000	38.0	80.0000

Parch	891.0	0.381594	0.806057	0.00	0.0000	0.0000	0.0	6.0000
Fare	891.0	32.204208	49.693429	0.00	7.9104	14.4542	31.0	512.3292

From above description we get that:

1. There are total 891 passenger record in our dataset.
2. Average age of passenger is around 30.
3. Average Fare price is around 32.20 (in dollars) and maximum fare price is around 512.32 (in dollars).

In [5]:

```
data.isna().sum()
```

Out[5]:

```
PassengerId      0
Survived         0
Pclass           0
Name             0
Sex              0
Age            177
SibSp            0
Parch            0
Ticket           0
Fare             0
Cabin          687
Embarked         2
dtype: int64
```

In [6]:

```
# filling age column with mean value of age column
data['Age'].fillna(data['Age'].mean(), inplace=True)

# filling Embark column with mode value of the column
data['Embarked'].fillna(data['Embarked'].mode()[0], inplace=True)
```

In [7]:

```python
# dropping not necessary column
data.drop(columns = ['PassengerId', 'Name', 'Cabin', 'Ticket'], axis=1, inplace=True)
data
```

Out[7]:

	Survived	Pclass	Sex	Age	SibSp	Parch	Fare	Embarked
0	0	3	male	22.000000	1	0	7.2500	S
1	1	1	female	38.000000	1	0	71.2833	C
2	1	3	female	26.000000	0	0	7.9250	S
3	1	1	female	35.000000	1	0	53.1000	S
4	0	3	male	35.000000	0	0	8.0500	S
...	...	...	...	...	...	...	...	...
886	0	2	male	27.000000	0	0	13.0000	S
887	1	1	female	19.000000	0	0	30.0000	S
888	0	3	female	29.699118	1	2	23.4500	S
889	1	1	male	26.000000	0	0	30.0000	C
890	0	3	male	32.000000	0	0	7.7500	Q

891 rows × 8 columns

In [8]:

```python
data.info()
```

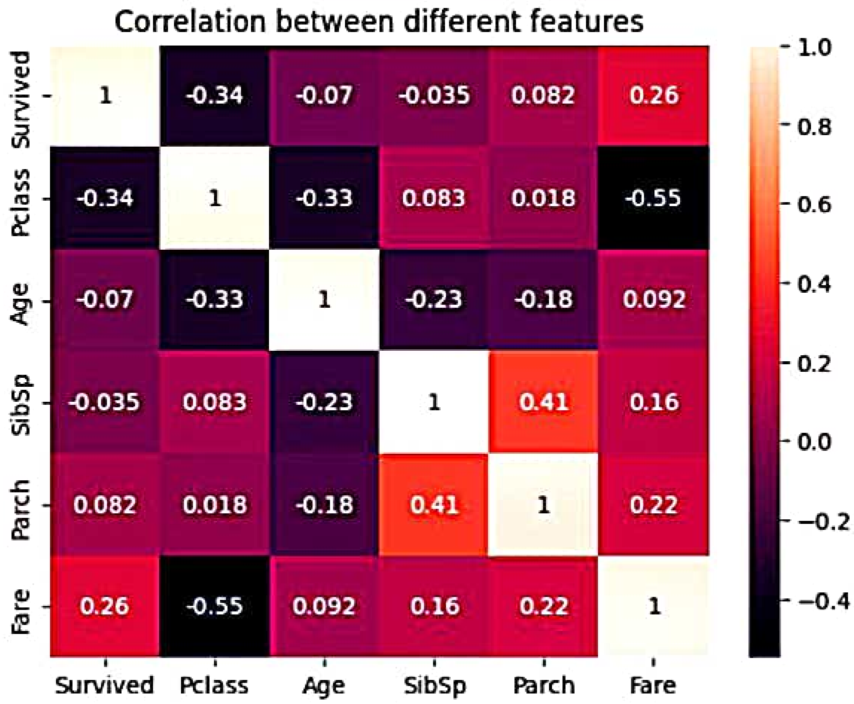
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 891 entries, 0 to 890
Data columns (total 8 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   Survived  891 non-null    int64
```

In [9]:
```
data.duplicated().sum()   #checking for null values
```
Out[9]:
```
111
```
In [10]:
```
data.nunique()            #checking unique values present in our data frame
```
Out[10]:
```
Survived      2
Pclass        3
Sex           2
Age          89
SibSp         7
Parch         7
Fare        248
Embarked      3
dtype: int64
```
In [11]:
```
#checking stastical correlation between numeric columns
data.corr(numeric_only=True)
```
Out[11]:

	Survived	Pclass	Age	SibSp	Parch	Fare
Survived	1.000000	-0.338481	-0.069809	-0.035322	0.081629	0.257307
Pclass	-0.338481	1.000000	-0.331339	0.083081	0.018443	-0.549500
Age	-0.069809	-0.331339	1.000000	-0.232625	-0.179191	0.091566
SibSp	-0.035322	0.083081	-0.232625	1.000000	0.414838	0.159651
Parch	0.081629	0.018443	-0.179191	0.414838	1.000000	0.216225
Fare	0.257307	-0.549500	0.091566	0.159651	0.216225	1.000000

In [12]:

```
# plotting correlation matrix by using heatmap
sns.heatmap(data.corr(numeric_only=True), annot=True)
plt.title('Correlation between different features')
plt.show()
```



In [13]:

```
data.head()
```

Out[13]:

	Survived	Pclass	Sex	Age	SibSp	Parch	Fare	Embarked
0	0	3	male	22.0	1	0	7.2500	S

Data Visualization

In [14]:

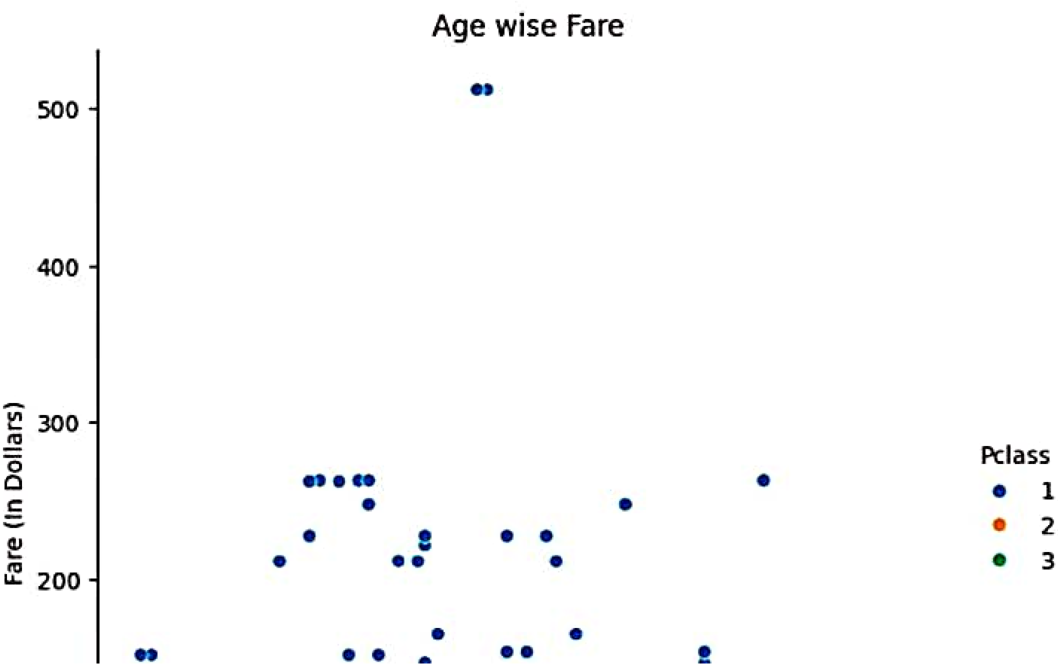
```
sns.displot(data, x = 'Fare', kde=True, aspect=2, color = 'Green')
plt.title("Distribution of Fare Prices")
plt.xlabel('Fare Price')
plt.ylabel('Count')
plt.grid(axis='y', ls=':', alpha=0.4, color='b')
plt.show()
```

Observation:

From above dstribustion we can say most of the ticket are sold in price range of 1-50 dollars and from this we can determine that fare column is having high skewness

In [22]:

```
# ploting scatter point
sns.relplot(data, y='Fare', x='Age', kind='scatter', hue='Pclass', palette='colorblind', height=6)
plt.title('Age wise Fare')
plt.xlabel('Age')
plt.ylabel('Fare (In Dollars)')
plt.show()
```

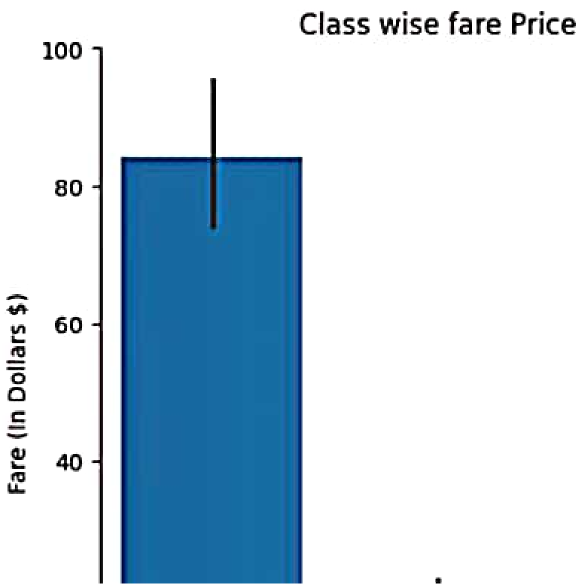


Observation :

1. Most tickets are sold from 3rd class

2. as expected 1st class tickets are costlier than class 2 and class

In [36]:
```
# Ploting bar plot
sns.catplot(data, x='Pclass', y='Fare', kind='bar')
plt.title('Class wise fare Price')
plt.xlabel('Class')
plt.ylabel('Fare (In Dollars $)')
plt.show()
```



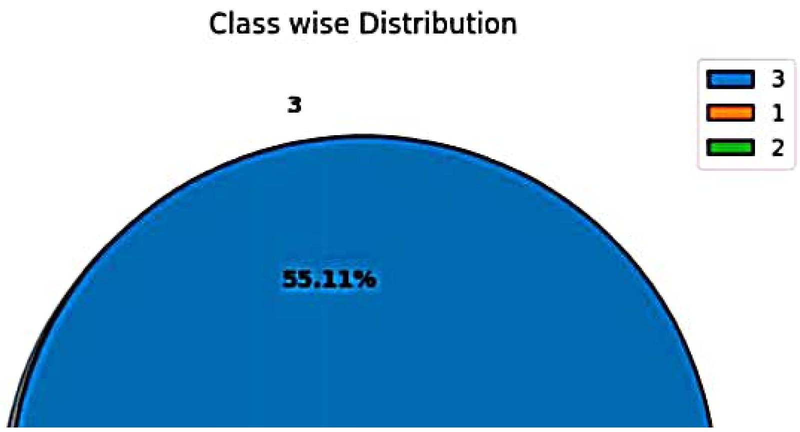
Observation :

1. 1st class having highest fare price.
2. 3rd class having lowest fare price.

In [27]:
```
class_count = data['Pclass'].value_counts()              # counting values for each class
```

In [29]:
```
# setting figure size
plt.figure(figsize=(10,7))

# ploting pie chart for class counts
plt.pie(class_count, labels=class_count.index, autopct='%.2f%%', shadow=True,
       wedgeprops={'linewidth':1, 'edgecolor':'black'},
       textprops={'weight':'bold'})
plt.legend()
plt.title('Class wise Distribution')
plt.show()
```



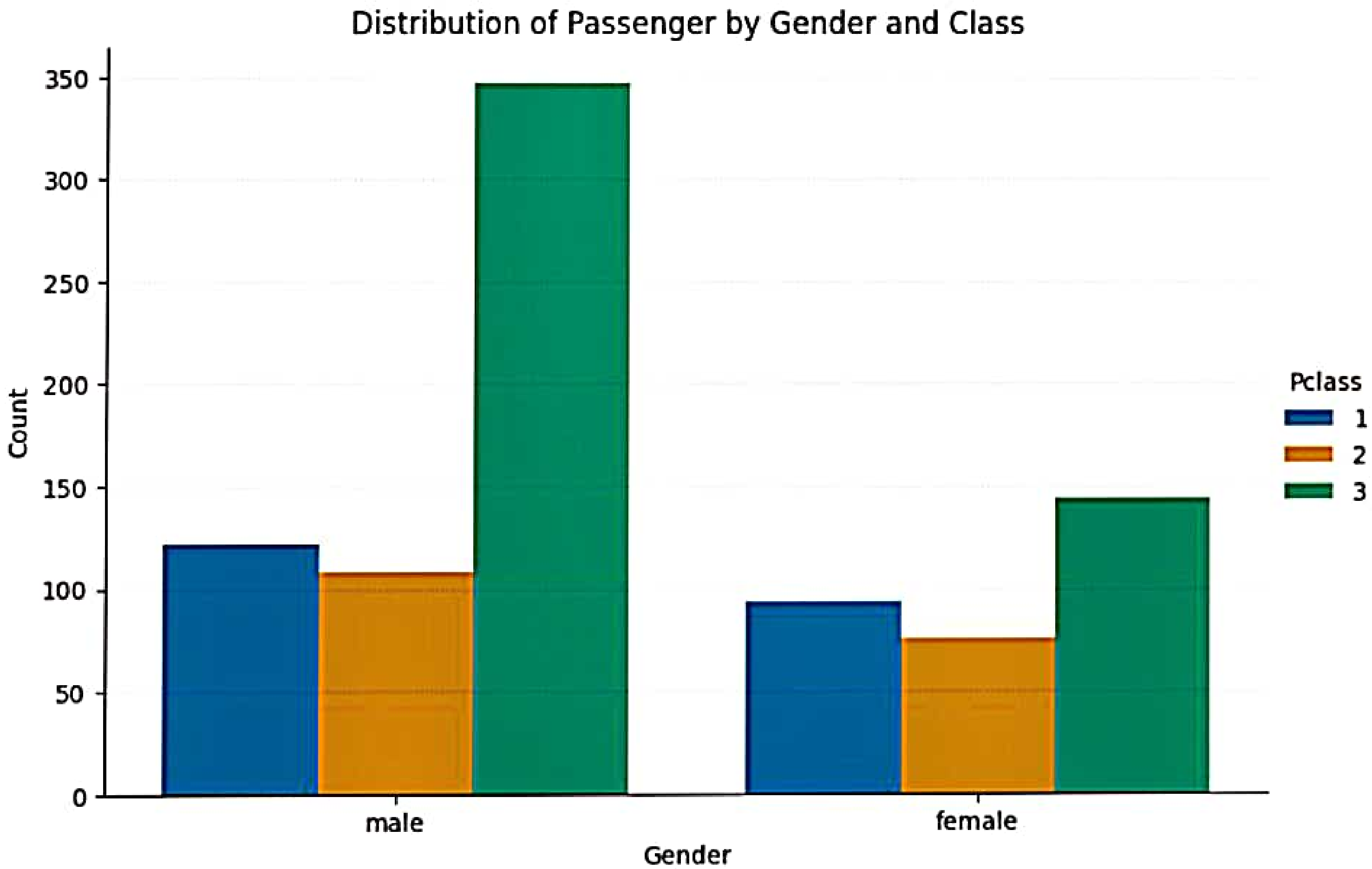
Obseevation:

1. From above pie chart we can observed that 3rd class tickets sold highest and is about 55.11%

2. Lowest sale tickets are from 2nd class and about 20.65%

In [40]:

```
# Ploting countplot
sns.catplot(data, x='Sex', kind='count', hue='Pclass', palette='colorblind', aspect=1.5, height=5)
plt.title('Distribution of Passenger by Gender and Class')
plt.xlabel('Gender')
plt.ylabel('Count')
plt.grid(axis='y', ls=':', color='b', alpha=0.2)
plt.show()
```

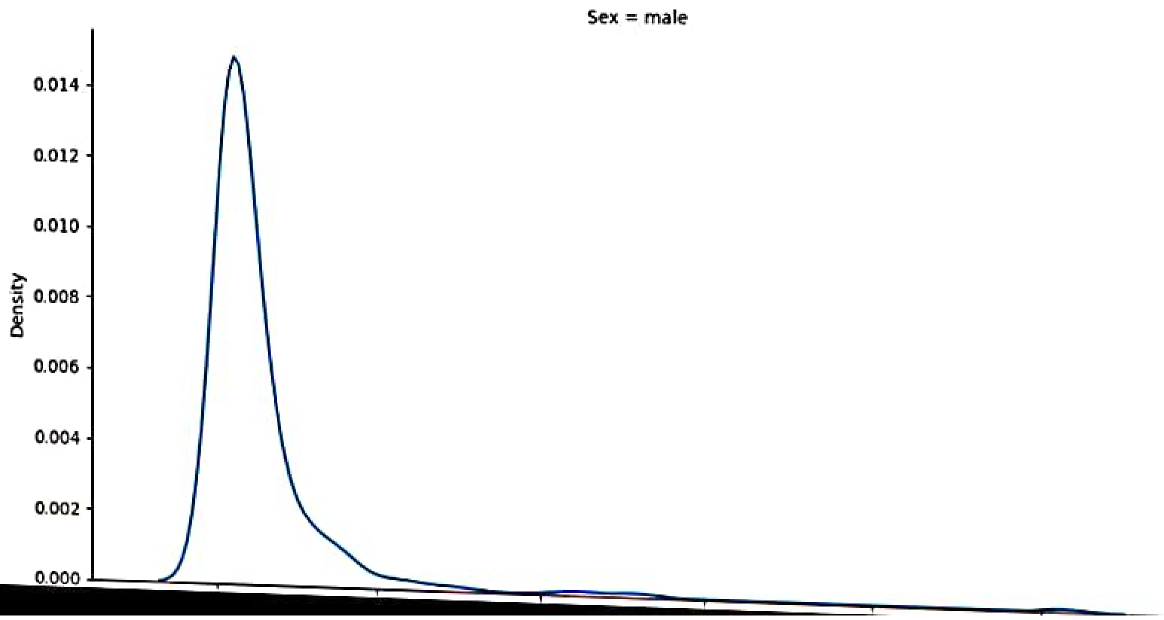


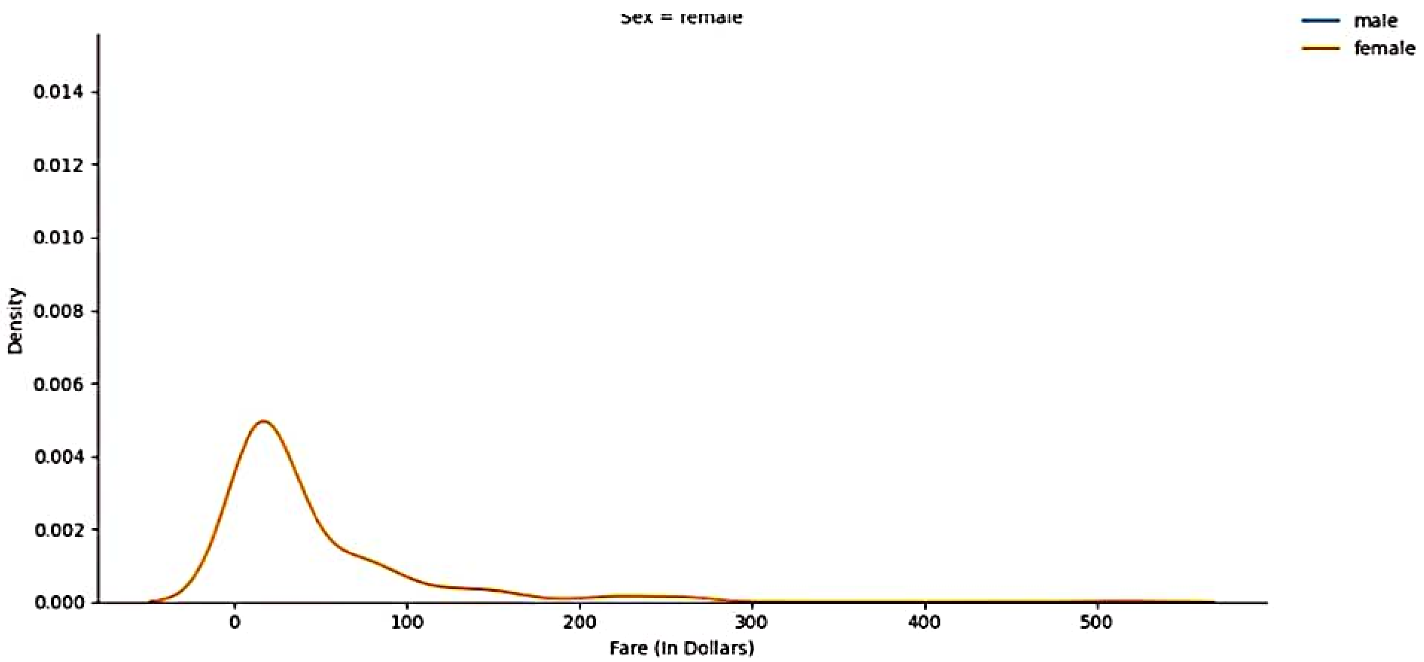
Observations :

1. From above obseravation we can determine that most male and female travels from 3rd class.

In [42]:

```
# Plotting kernal density estimation (KDE) plot.
sns.displot(data, x='Fare', hue='Sex', kind='kde', row='Sex', palette='colorblind', aspect=2)
plt.xlabel('Fare (In Dollars)')
plt.ylabel('Density')
plt.show()
```



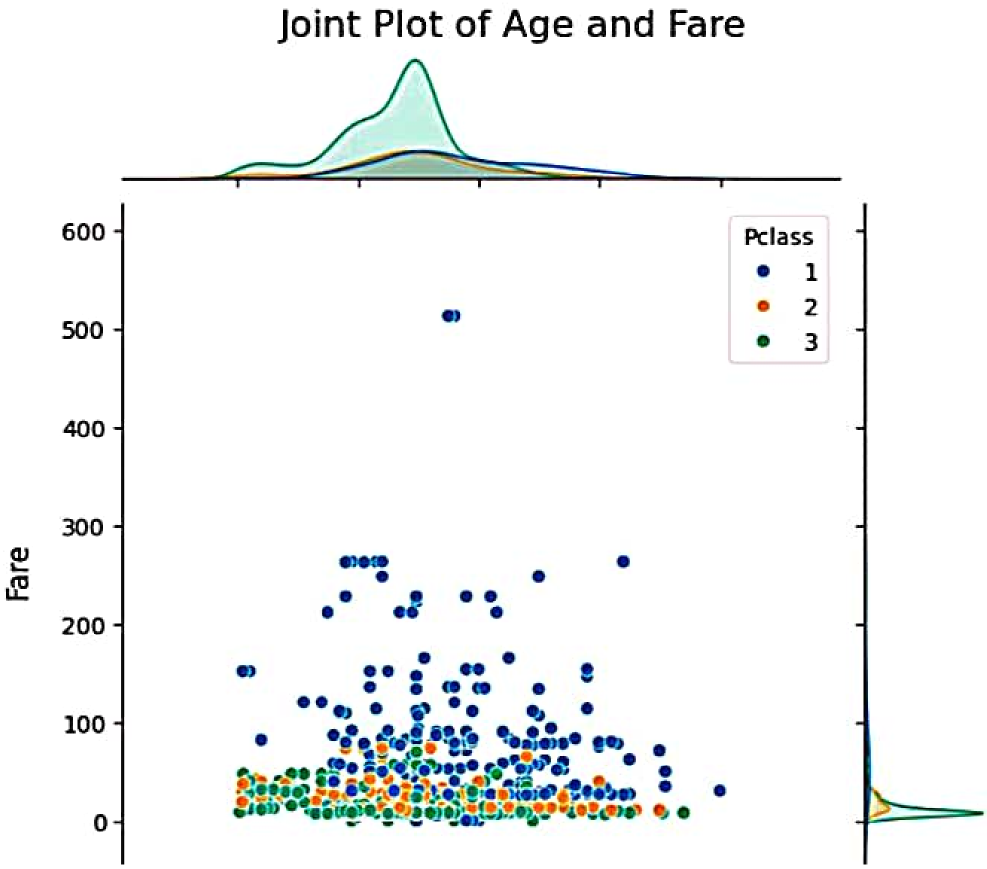


Observation:

Based on the analysis, it can be inferred that the number of male passenger of male passenger is greater than the number of female passenger

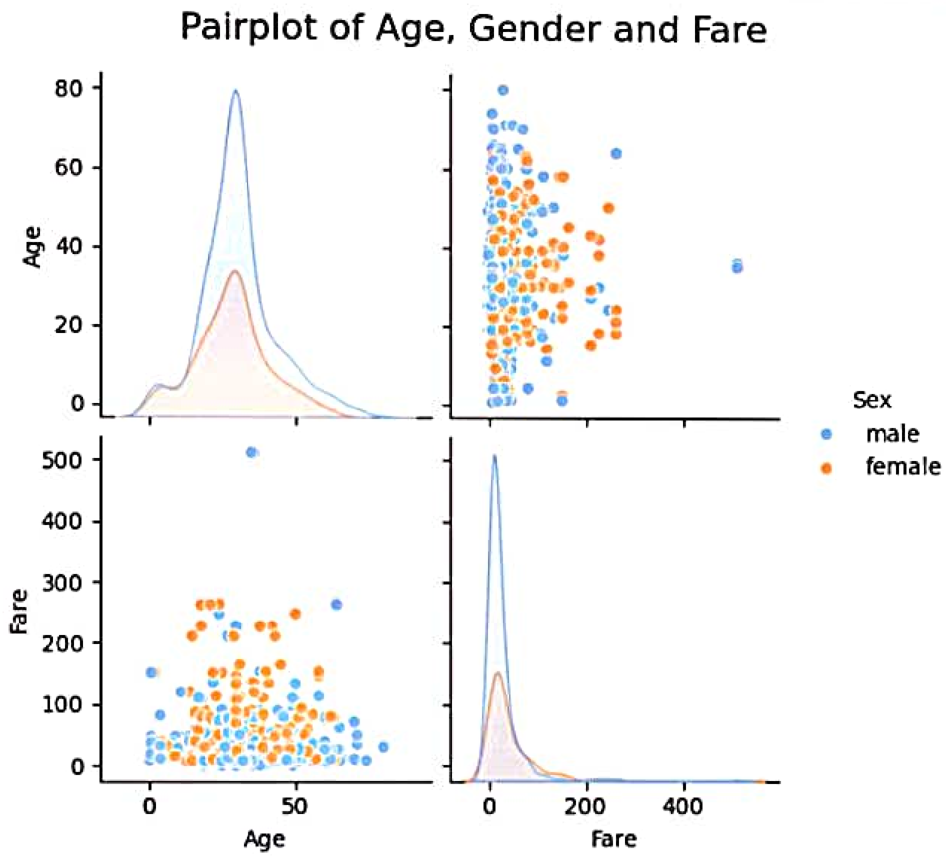
In [45]:

```
# plotiing joint plot
plot = sns.jointplot(data, x="Age", y='Fare', hue='Pclass', palette='colorblind')
plot.set_axis_labels('Age', 'Fare', fontsize=12)
plot.fig.suptitle('Joint Plot of Age and Fare', y=1.02, fontsize=16) # Giving title to plot
plt.show()
```



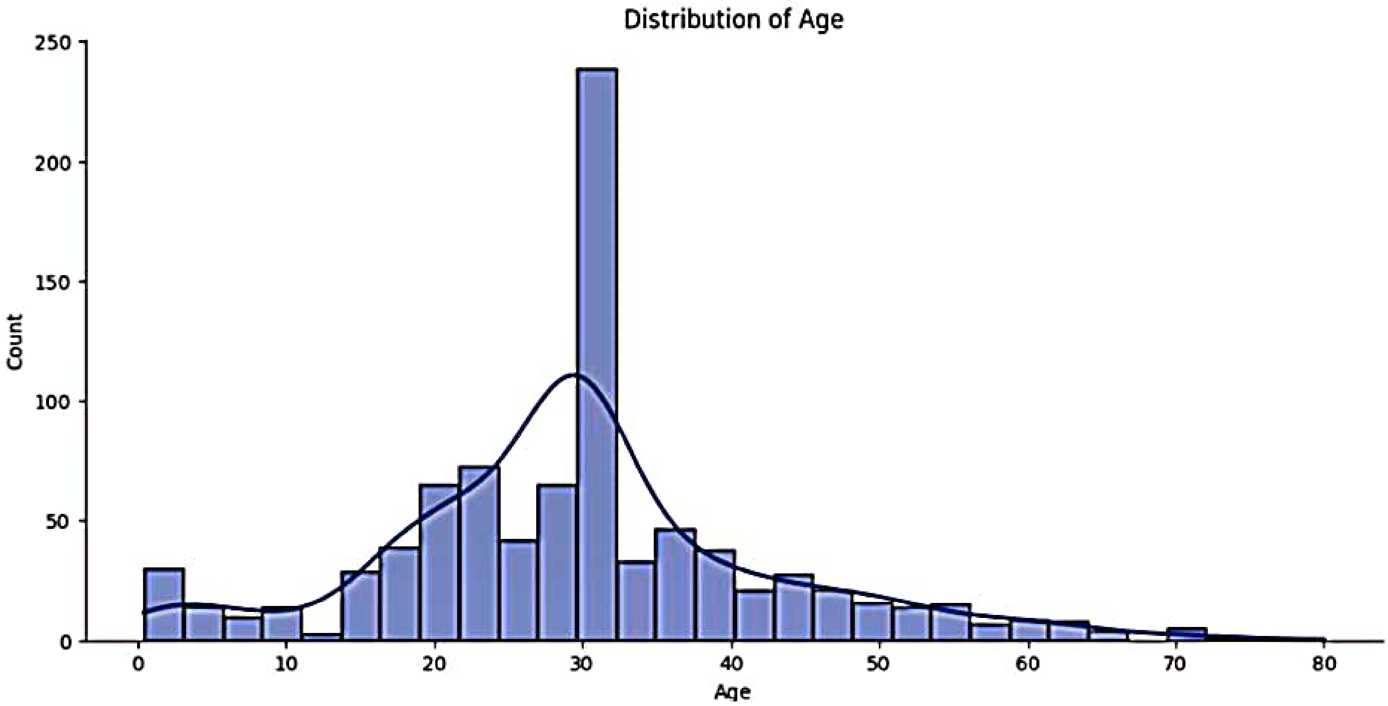
In [46]:

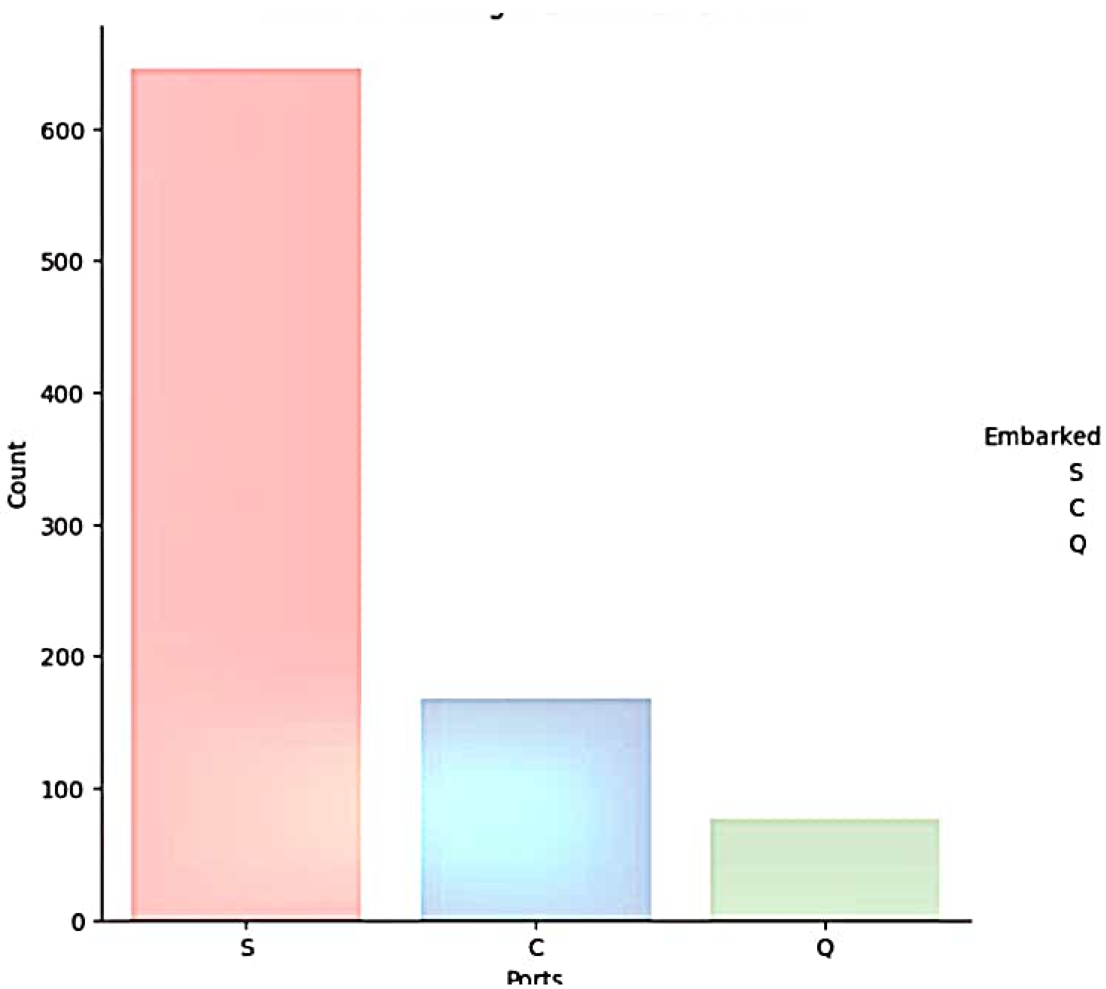
```
sns.pairplot(data, vars=['Age','Fare'], hue='Sex', palette='pastel')
plt.suptitle('Pairplot of Age, Gender and Fare', y=1.05, fontsize=16)
plt.show()
```

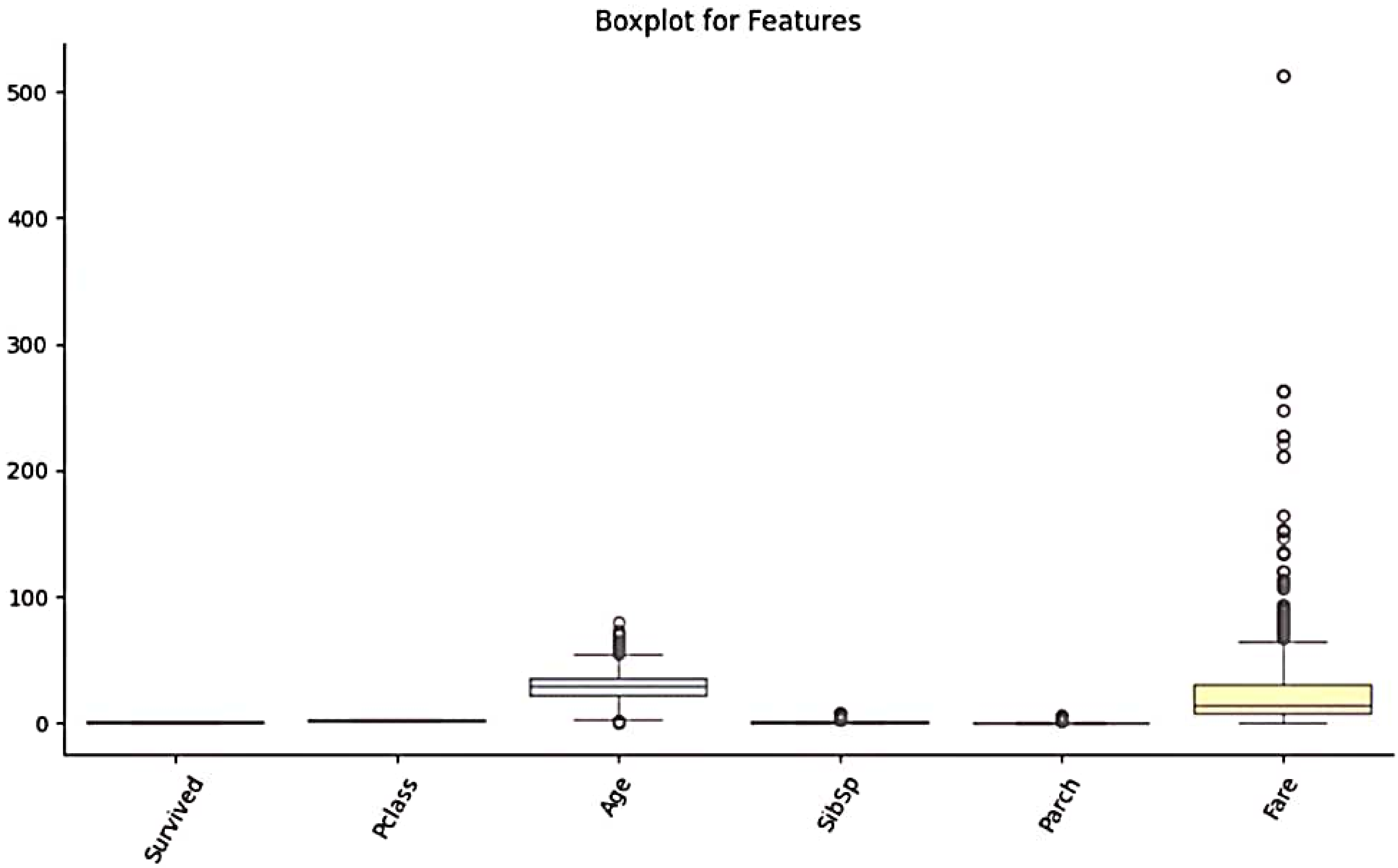


In [48]:

```
# ploting histogram to check age distribution
sns.displot(data, x='Age', kde=True, color=sns.color_palette('dark')[0], line_kws={'linewidth':2}, aspect=2)
plt.title('Distribution of Age')
plt.xlabel('Age')
plt.ylabel('Count')
plt.show()
```




600
500
400
300
200
100
0
Count
S
C
Q
Embarked
S
C
Q


Boxplot for Features
500
400
300
200
100
0
Survived
Pclass
Age
SibSp
Parch
Fare