Pandas Plots

Plot 1: Age Distribution

Objective:
This plot displays the **distribution of age** among the passengers in the Titanic dataset. It allows us to observe the frequency of passengers in different age groups.

Description:

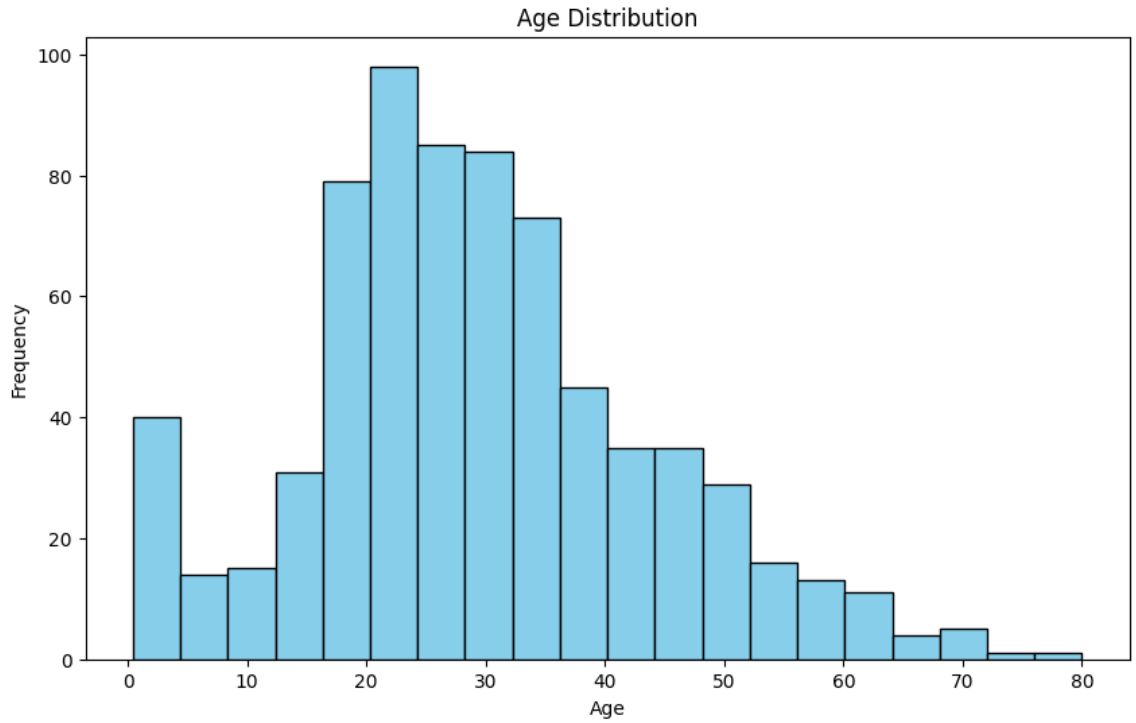
- A **histogram** is used to visualize the distribution of the **Age** feature.
- The x-axis represents **Age**, and the y-axis represents the **Frequency** (number of passengers within each age range).

```
df['Age'].plot(kind='hist', bins=20, figsize=(10, 6), color='skyblue', edgecolor='black')
plt.title('Age Distribution')
plt.xlabel('Age')
plt.ylabel('Frequency')
plt.show()
```

Explanation of the Code:

- The plot() method is used with kind='hist' to generate a **histogram** of the **Age** column.
- The **bins=20** argument divides the data into 20 bins to visualize the distribution in groups.
- We use **skyblue** for the bar color and **black** for the edge color.
- Titles and labels are added to make the plot clear, and plt.show() renders the plot.

The plot:



Plot 2: Survival Count by Pclass

Objective:
This plot shows the number of passengers who survived and did not survive, grouped by **Passenger Class (Pclass)**. It helps in understanding the survival distribution across different classes.

Description:

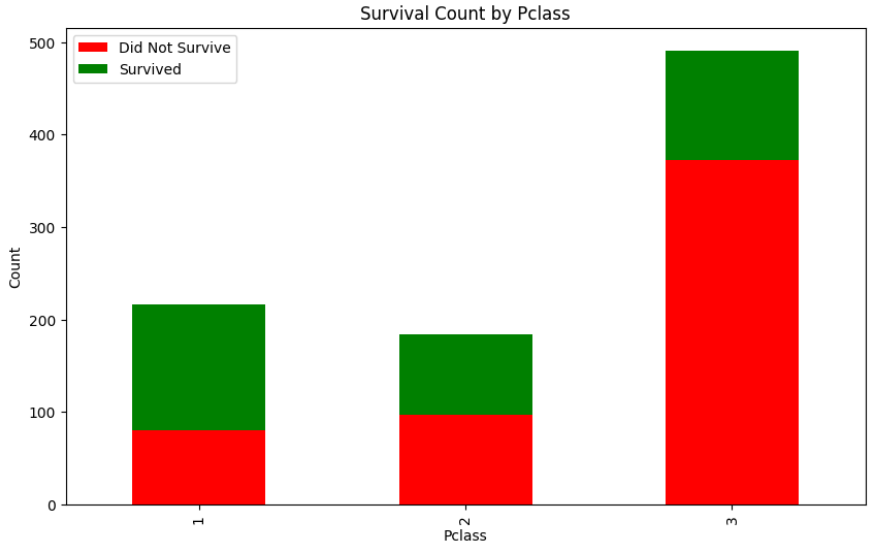
- A **stacked bar chart** is used to show the **count of survivors and non-survivors** in each passenger class.
- The x-axis represents the **Passenger Class**, and the y-axis represents the **Count** of passengers in each category.
- Two colors are used: **red** for those who did not survive and **green** for those who survived.

```
df.groupby(['Pclass', 'Survived']).size().unstack().plot(
    kind='bar', stacked=True, figsize=(10, 6), color=['red', 'green']
)
plt.title('Survival Count by Pclass')
plt.ylabel('Count')
plt.legend(['Did Not Survive', 'Survived'])
plt.show()
```

Explanation of the Code:

- We first **group** the data by **Pclass** and **Survived** using groupby(), and then use .size() to count the occurrences.
- The **unstack()** method reshapes the data for plotting, separating survival counts for each class.
- A **stacked bar chart** is created using the kind='bar' argument, with **red** and **green** representing non-survivors and survivors, respectively.
- Labels, titles, and a legend are added for clarity.

The plot:



Plot 3: Fare vs Age

Objective:
This scatter plot visualizes the relationship between **Fare** and **Age** of the passengers. It helps us understand how the fare varies with age and if there are any visible trends.

Description:

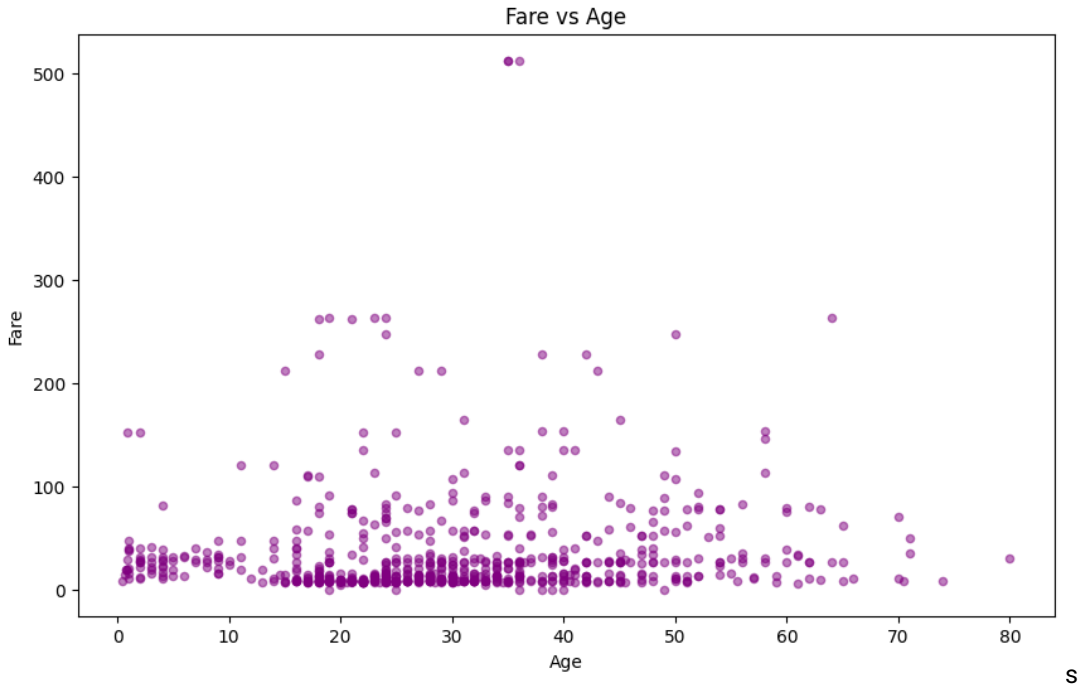
- A **scatter plot** is used to show the **relationship between Age and Fare**.
- The x-axis represents **Age**, and the y-axis represents **Fare** (the amount paid for the ticket).
- The **alpha=0.5** makes the points semi-transparent, and the color is set to **purple** for better visualization.

```
df.plot(kind='scatter', x='Age', y='Fare', alpha=0.5, color='purple', figsize=(10, 6))
plt.title('Fare vs Age')
plt.xlabel('Age')
plt.ylabel('Fare')
plt.show()
```

Explanation of the Code:

- The plot() method is used with kind='scatter' to create a scatter plot between **Age** and **Fare**.
- The **alpha=0.5** parameter makes the points semi-transparent, allowing us to better see any overlaps in the data points.
- The plot is sized with figsize=(10, 6) for better readability, and **purple** is chosen for the point color.
- Titles and labels are added for clarity, and plt.show() renders the plot.

The plot:



s

Matplotlib Plots

Plot 1: Survival Rate by Age

Objective:
This plot shows the **relationship between age and survival rate** in the Titanic dataset. It helps us understand if age played a role in the chances of survival.

Description:

- A **line plot** is used to show how survival rates vary across different ages.
- The x-axis represents **Age**, and the y-axis represents the **Survival Rate** (percentage of people who survived at each age).

```
survival_rate_by_age = df.groupby("Age")["Survived"].mean()

plt.plot(survival_rate_by_age.index, survival_rate_by_age.values, "r")
plt.xlabel("Age")
plt.ylabel("Survival Rate")
plt.title("Survival Rate by Age")
plt.show()
```

Explanation of the Code:

- We first group the data by **age** and calculate the **average survival rate** for each age group.
- We use the **plot function** to create a **red line graph**.
- **Labels** and **title** are added for clarity, and plt.show() renders the plot.

The plot:

Plot 2: Survival Rate by Age and Class

Objective:

This plot shows the **relationship between age and survival rate** in one subplot, and the **survival rate by passenger class** in another subplot. It helps compare survival trends by age and class at once.

Description:

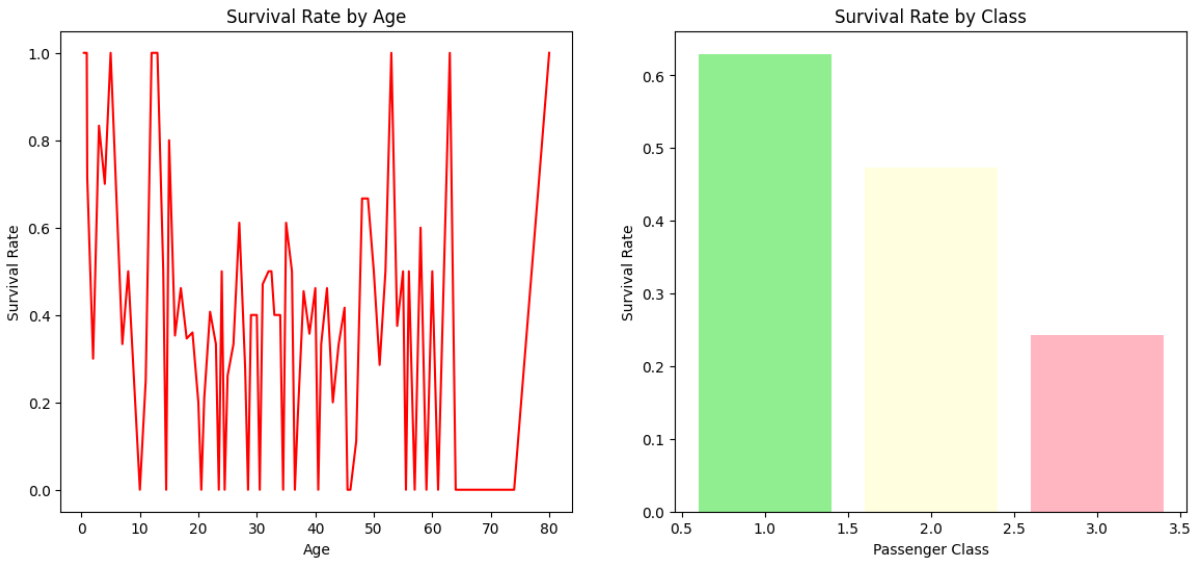
- The first subplot is a **line plot** of survival rate by age, similar to Plot 1.
- The second subplot is a **bar plot** comparing the survival rate across the three passenger classes (1st, 2nd, 3rd).

```
fig1, axs1 = plt.subplots(1, 2, figsize=(14, 6))

survival_rate_by_age = df.groupby("Age")["Survived"].mean()
axs1[0].plot(survival_rate_by_age.index, survival_rate_by_age.values, "r")
axs1[0].set_xlabel("Age")
axs1[0].set_ylabel("Survival Rate")
axs1[0].set_title("Survival Rate by Age")

survival_rate_by_class = df.groupby("Pclass")["Survived"].mean()
axs1[1].bar(survival_rate_by_class.index, survival_rate_by_class.values, color=["lightgreen", "lightyellow", "lightpink"])
axs1[1].set_xlabel("Passenger Class")
axs1[1].set_ylabel("Survival Rate")
axs1[1].set_title("Survival Rate by Class")
```

Explanation of the Code:

- plt.subplots(1, 2) creates a **1-row, 2-column grid** to accommodate two subplots.
- **Plot 1 (left):** We plot **survival rate by age** as a **red line**.
- **Plot 2 (right):** We plot **survival rate by passenger class** as a **bar plot**.
- Labels, titles, and the final plt.show() to render the plots.

The plot:



Plot 3: Multiple Factors Affecting Survival

Objective:
This plot helps analyze the survival rate with respect to **gender, passenger class, number of passengers by gender, and number of passengers by class**. By combining these four visualizations, we get a broader view of the factors that influenced survival rates.

Description:

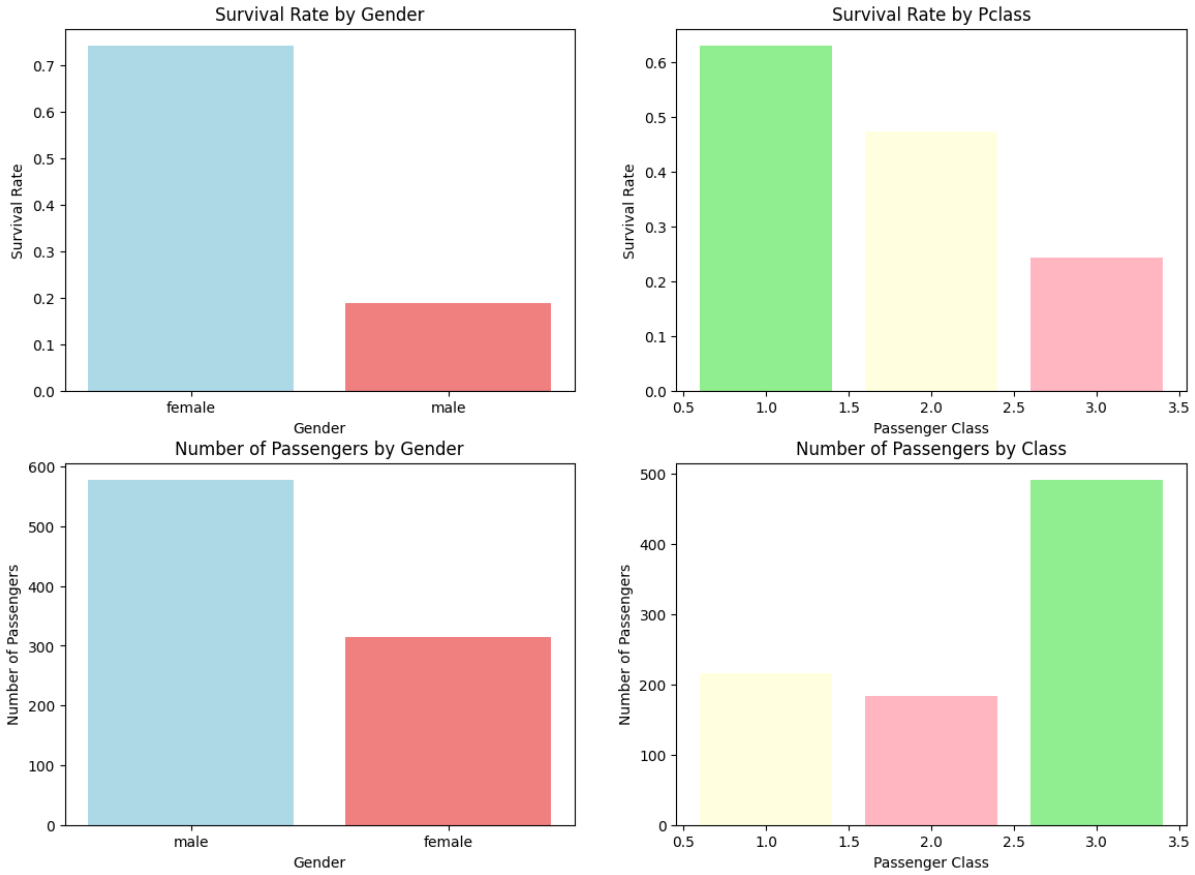
- The **first subplot** shows survival rate by gender.
- The **second subplot** compares survival rates by class.
- The **third subplot** shows the number of passengers by gender.
- The **fourth subplot** shows the number of passengers by class.

```
fig2, axs2 = plt.subplots(2, 2, figsize=(14, 10))

survival_rate_by_gender = df.groupby("Sex")["Survived"].mean()
axs2[0, 0].bar(survival_rate_by_gender.index, survival_rate_by_gender.values, color=["lightblue", "lightcoral"])
axs2[0, 0].set_xlabel("Gender")
axs2[0, 0].set_ylabel("Survival Rate")
axs2[0, 0].set_title("Survival Rate by Gender")

survival_rate_by_class = df.groupby("Pclass")["Survived"].mean()
axs2[0, 1].bar(survival_rate_by_class.index, survival_rate_by_class.values, color=["lightgreen", "lightyellow", "lightpink"])
axs2[0, 1].set_xlabel("Passenger Class")
axs2[0, 1].set_ylabel("Survival Rate")
axs2[0, 1].set_title("Survival Rate by Pclass")

passenger_count_by_gender = df["Sex"].value_counts()
axs2[1, 0].bar(passenger_count_by_gender.index, passenger_count_by_gender.values, color=["lightblue", "lightcoral"])
axs2[1, 0].set_xlabel("Gender")
axs2[1, 0].set_ylabel("Number of Passengers")
axs2[1, 0].set_title("Number of Passengers by Gender")

passenger_count_by_class = df["Pclass"].value_counts()
axs2[1, 1].bar(passenger_count_by_class.index, passenger_count_by_class.values, color=["lightgreen", "lightyellow", "lightpink"])
axs2[1, 1].set_xlabel("Passenger Class")
axs2[1, 1].set_ylabel("Number of Passengers")
axs2[1, 1].set_title("Number of Passengers by Class")

plt.show()
```

Explanation of the Code:

- plt.subplots(2, 2) creates a **2-row, 2-column grid** for four subplots.
- **Plot 1 (top-left):** Shows **survival rate by gender** using a **bar plot**.
- **Plot 2 (top-right):** Shows **survival rate by passenger class** using a **bar plot**.
- **Plot 3 (bottom-left):** Shows the **number of passengers by gender**.
- **Plot 4 (bottom-right):** Shows the **number of passengers by class**.
- plt.tight_layout() ensures that the plots do not overlap.

The plot:



Seaborn Plots:

Plot 1: Age Distribution by Gender (Violin Plot)

Objective: This plot visualizes the age distribution of passengers based on gender. It provides insights into the spread and density of ages within each gender group.

Description:

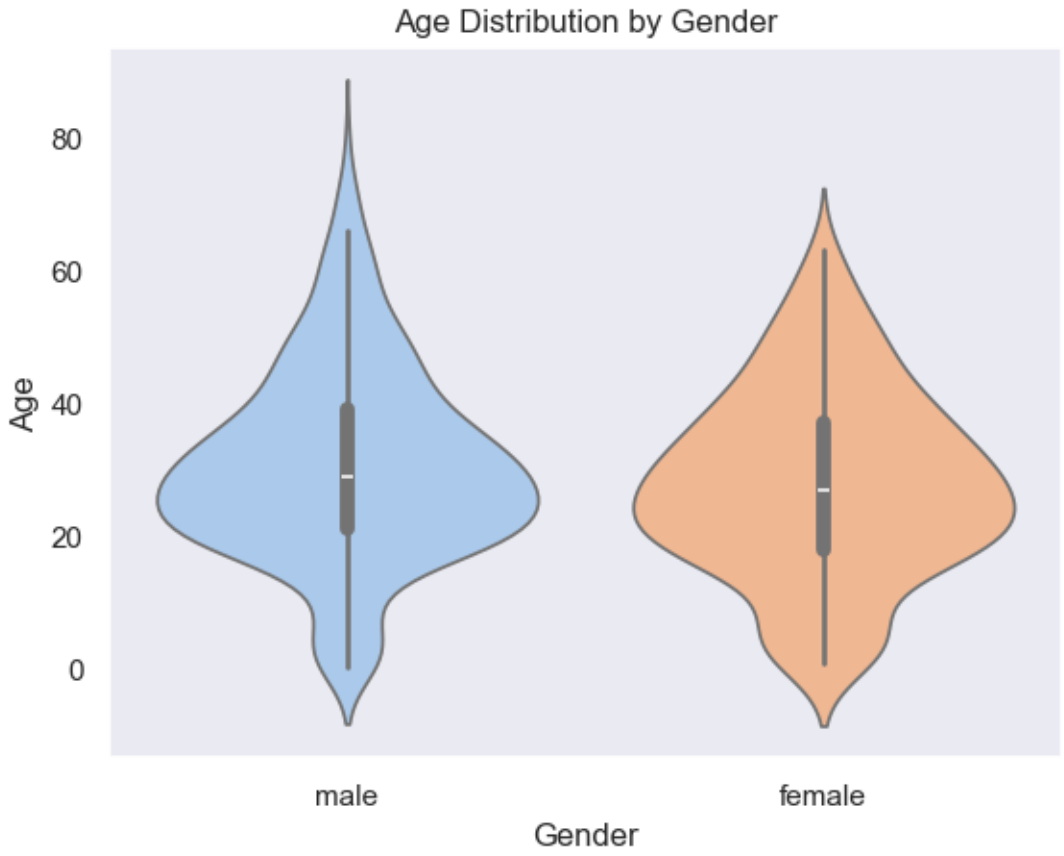
- A violin plot combines a box plot and a kernel density estimation (KDE), showing the distribution of age values for each gender.
- It helps identify differences in age distribution between male and female passengers.
- Wider sections indicate a higher concentration of passengers at a given age.

```
sns.violinplot(x="Sex", y="Age", data=df, palette="pastel")
plt.title("Age Distribution by Gender")
plt.xlabel("Gender")
plt.ylabel("Age")
plt.show()
```

Explanation of the Code:

- sns.violinplot(...) creates the violin plot with Sex on the x-axis and Age on the y-axis.

- palette="pastel" sets the color scheme.
- plt.title(), plt.xlabel(), and plt.ylabel() define the plot title and axis labels.
- plt.show() displays the plot.



Plot 2: Survival Count by Passenger Class (Count Plot)

Objective:
This plot shows how survival rates vary across passenger classes, helping to analyze whether class influenced survival chances.

Description:

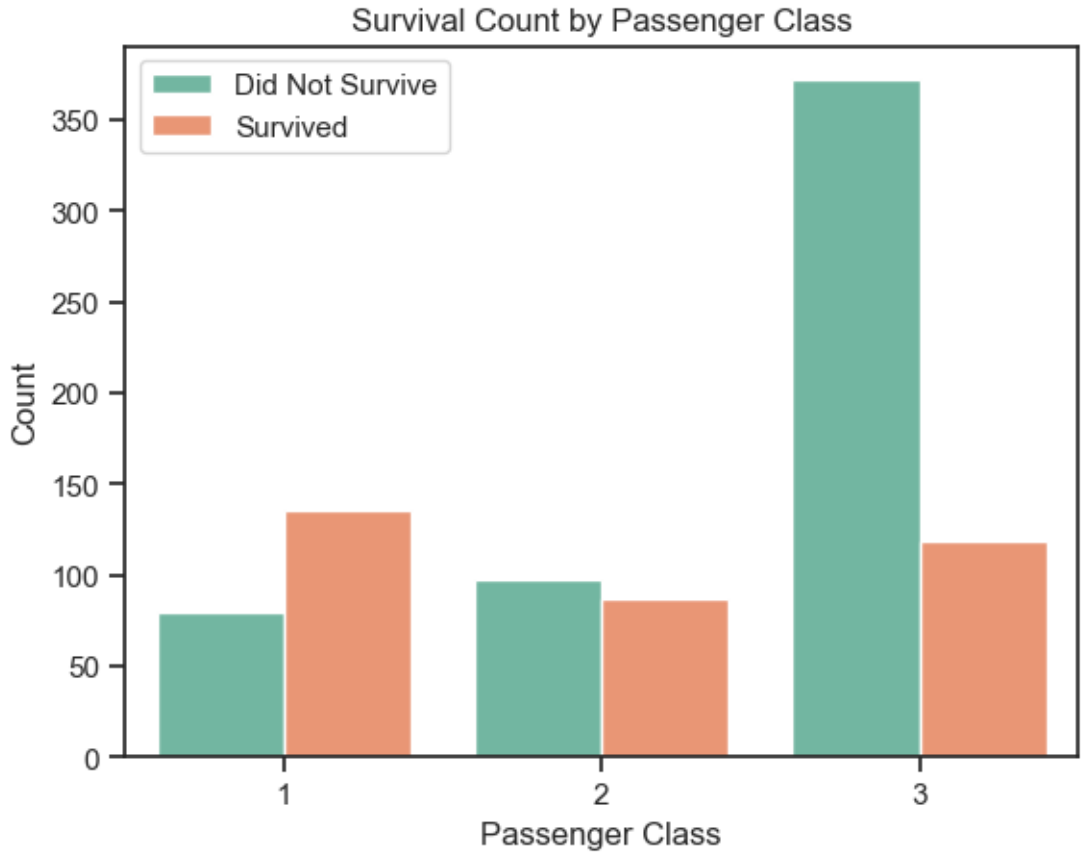
- A count plot displays the number of passengers in each class, divided by survival status.
- The hue argument differentiates between survivors and non-survivors using different colors.
- It highlights trends such as a higher survival rate in first-class passengers.

```
sns.countplot(x="Pclass", hue="Survived", data=df, palette="Set2")
plt.title("Survival Count by Passenger Class")
plt.xlabel("Passenger Class")
plt.ylabel("Count")
plt.legend(["Did Not Survive", "Survived"])
plt.show()
```

Explanation of the Code:

- sns.countplot(...) creates the count plot with Pclass on the x-axis and passenger count on the y-axis.
- hue="Survived" colors the bars based on survival status.
- palette="Set2" applies a color scheme.
- plt.legend() assigns labels to the survival status colors.

The plot:



Plot 3: Fare Distribution by Passenger Class (Box Plot)

Objective:
This plot illustrates fare price distribution for each passenger class, showing variability and outliers.

Description:

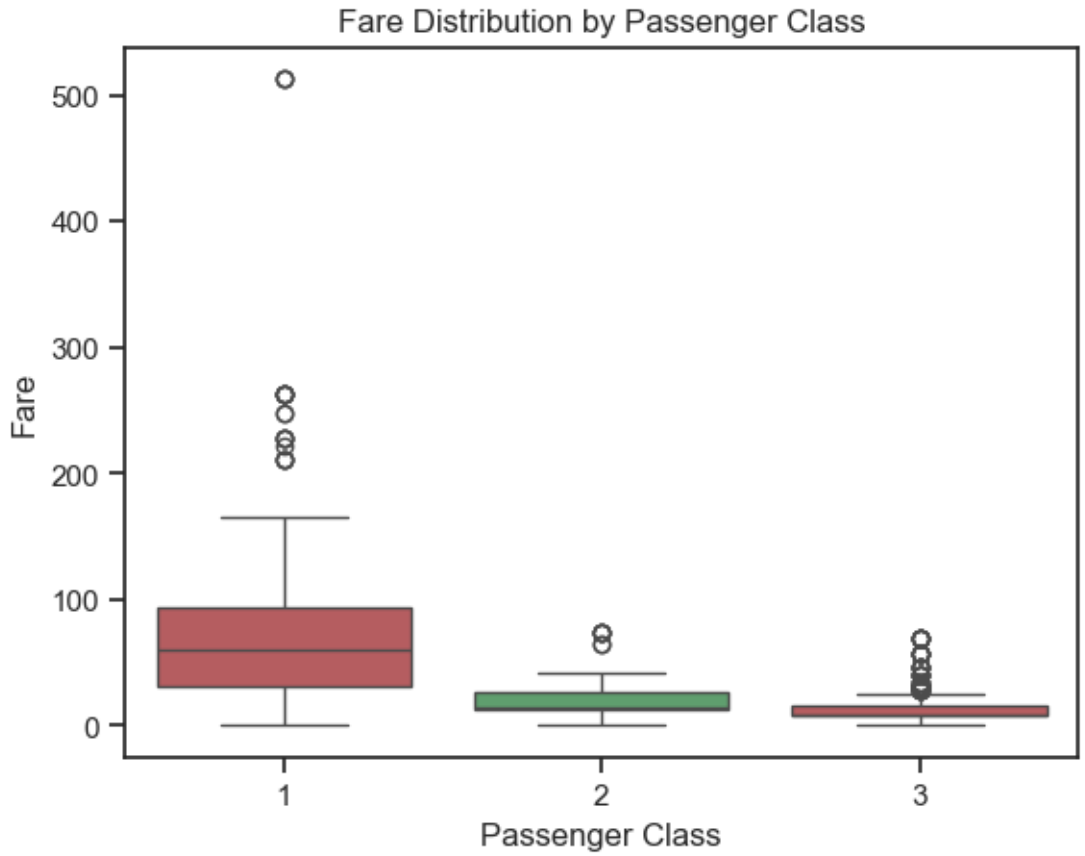
- A box plot represents the median, interquartile range (IQR), and potential outliers in fare prices for each class.
- It provides insights into how fares varied between classes.
- The first-class passengers generally paid higher fares compared to lower classes.

```
sns.boxplot(x="Pclass", y="Fare", data=df, palette=["r", "g"])
plt.title("Fare Distribution by Passenger Class")
plt.xlabel("Passenger Class")
plt.ylabel("Fare")
plt.show()
```

Explanation of the Code:

- sns.boxplot(...) generates the box plot with Pclass on the x-axis and Fare on the y-axis.
- palette=["r", "g"] assigns colors to different classes.
- plt.title(), plt.xlabel(), and plt.ylabel() add the title and labels.

The plot:



Plot 4: Feature Correlation Heatmap

Objective:
This heatmap visualizes the correlation between numerical features, helping identify relationships between variables.

Description:

- It shows correlation coefficients for all numeric columns.

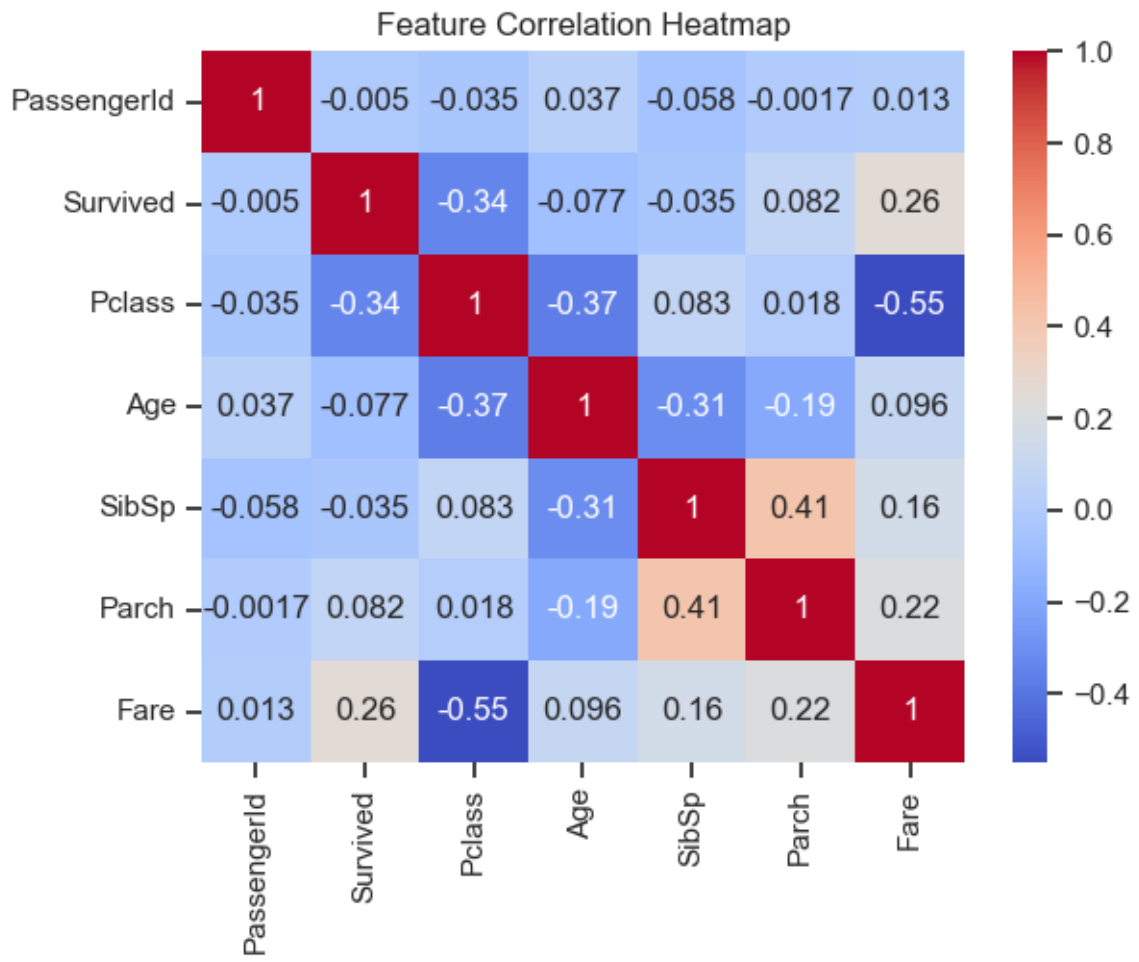
- A high positive value (close to 1) indicates a strong direct relationship, while a negative value (close to -1) shows an inverse relationship.
- This analysis helps in feature selection for machine learning models.

```
numeric_df = df.select_dtypes(include=["number"])
sns.heatmap(numeric_df.corr(), annot=True, cmap="coolwarm", cbar=True)
plt.title("Feature Correlation Heatmap")
plt.show()
```

Explanation of the Code:

- df.select_dtypes(include=["number"]) selects only numeric columns.
- numeric_df.corr() computes the correlation matrix.
- sns.heatmap(...) plots the heatmap with a color gradient (cmap="coolwarm").
- annot=True displays the correlation values inside the heatmap cells.

The plot:



Plot 5: Age vs Fare by Survival Status (Swarm Plot)

Objective:
This plot examines how fare price relates to survival status and whether gender played a role in survival.

Description:

- A swarm plot places data points side by side to show distributions without overlap.
- The hue="Sex" parameter distinguishes male and female passengers.
- This visualization highlights trends in fare amounts paid by survivors and non-survivors.

```
sns.swarmplot(x="Survived", y="Fare", data=df, hue="Sex", palette="muted", dodge=True)
plt.title("Age vs Fare by Survival Status")
plt.xlabel("Survival Status")
plt.ylabel("Fare")
plt.legend(title= "Gender")
plt.show()
```

Explanation of the Code:

- sns.swarmplot(...) creates a swarm plot with survival status (Survived) on the x-axis and Fare on the y-axis.
- hue="Sex" assigns colors based on gender.
- dodge=True ensures non-overlapping points for better clarity.

The plot: