**Data Analysis using GenAI assignment**

Name - Chandrabhushan kumar Bharti

Batch - CPDA 5

DATA SET USED – TITANIC

Task List:

1 – Load the Data into ChatGPT

2 – Basic Statistics of 5 important columns (Mean, Median ,Mode, Std Dev, Min, Max)

3 – Check for missing values and Outliers

4 – Univariate Analysis (Visualize the distribution of a column)

5 – Bi-variate Analysis

**Task 2**

Prompt – Act as an expert data analyst and data scientist, do basic statics of 5 important columns and find mean median mode std dev min and max and present in tabular form for easy understanding also give google colab friendly python code for every action which i can use in colab.

**Output**

**The dataset is a version of the Titanic passenger list. Here are the main columns available:**

* **survived: 0 = No, 1 = Yes**
* **pclass: Ticket class (1 = 1st, 2 = 2nd, 3 = 3rd)**
* **sex: male or female**
* **age: Age in years**
* **sibsp: # of siblings / spouses aboard**
* **parch: # of parents / children aboard**
* **fare: Passenger fare**
* **embarked: Port of Embarkation (C = Cherbourg, Q = Queenstown, S = Southampton)**

### **📊 Let's analyze 5 important columns:**

* **age**
* **fare**
* **pclass**
* **sibsp**
* **parch**

**The basic statistical summary of 5 important columns from the Titanic dataset:**

| **Column** | **Mean** | **Median** | **Mode** | **Std Dev** | **Min** | **Max** |
| --- | --- | --- | --- | --- | --- | --- |
| **age** | **29.70** | **28.00** | **24.0** | **14.53** | **0.42** | **80.00** |
| **fare** | **32.20** | **14.45** | **8.05** | **49.69** | **0.00** | **512.33** |
| **pclass** | **2.31** | **3.00** | **3.00** | **0.84** | **1.00** | **3.00** |
| **sibsp** | **0.52** | **0.00** | **0.00** | **1.10** | **0.00** | **8.00** |
| **parch** | **0.38** | **0.00** | **0.00** | **0.81** | **0.00** | **6.00** |

**Python code for google colab**

import pandas as pd

from google.colab import files

import pandas as pd

# Upload the file

uploaded = files.upload()

# Load it into a DataFrame

# Replace 'your\_file.csv' with the actual uploaded filename

df = pd.read\_csv(next(iter(uploaded)))

df.head() # Display first 5 rows

# Select columns to analyze

columns\_to\_analyze = ['age', 'fare', 'pclass', 'sibsp', 'parch']

# Prepare summary

stats\_summary = {

'Column': [],

'Mean': [],

'Median': [],

'Mode': [],

'Std Dev': [],

'Min': [],

'Max': []

}

# Compute statistics

for col in columns\_to\_analyze:

stats\_summary['Column'].append(col)

stats\_summary['Mean'].append(df[col].mean())

stats\_summary['Median'].append(df[col].median())

stats\_summary['Mode'].append(df[col].mode().iloc[0] if not df[col].mode().empty else None)

stats\_summary['Std Dev'].append(df[col].std())

stats\_summary['Min'].append(df[col].min())

stats\_summary['Max'].append(df[col].max())

# Convert to DataFrame for display

stats\_df = pd.DataFrame(stats\_summary)

print(stats\_df)

**TASK 3**

PROMP –check the missing value and outliers in each column in which we have done basic statics and also give python code for every operations onwards.

**OUTPUT**

| **Column** | **Missing Values** | **Outliers Count** | **Lower Bound** | **Upper Bound** |
| --- | --- | --- | --- | --- |
| **age** | 177 | 11 | -6.69 | 64.81 |
| **fare** | 0 | 116 | -26.72 | 65.63 |
| **pclass** | 0 | 0 | 0.50 | 4.50 |
| **sibsp** | 0 | 46 | -1.50 | 2.50 |
| **parch** | 0 | 213 | 0.00 | 0.00 |

Note- Here some data is in negative which is practically not possible so where the data in the lower bound is negative we will consider that data as Zero.

### **🔍 Observations:**

* age has **177 missing values** and some minor outliers above 64.81.
* fare has **116 outliers**, mostly high fares.
* parch shows a lot of values outside the IQR bounds due to most values being 0.
* pclass has **no outliers**.
* sibsp has **46 outliers**, likely from large families.

**Python code**

import pandas as pd

# Load dataset

df = pd.read\_csv('titanic.csv') # Replace with your actual path

# Columns to analyze

columns\_to\_analyze = ['age', 'fare', 'pclass', 'sibsp', 'parch']

# Check missing values

missing\_values = df[columns\_to\_analyze].isnull().sum()

# Outlier detection using IQR

outlier\_info = {}

for col in columns\_to\_analyze:

col\_data = df[col].dropna()

Q1 = col\_data.quantile(0.25)

Q3 = col\_data.quantile(0.75)

IQR = Q3 - Q1

lower\_bound = Q1 - 1.5 \* IQR

upper\_bound = Q3 + 1.5 \* IQR

outliers = col\_data[(col\_data < lower\_bound) | (col\_data > upper\_bound)]

outlier\_info[col] = {

'Outliers Count': outliers.count(),

'Lower Bound': lower\_bound,

'Upper Bound': upper\_bound

}

# Create DataFrame to display results

outlier\_df = pd.DataFrame(outlier\_info).T

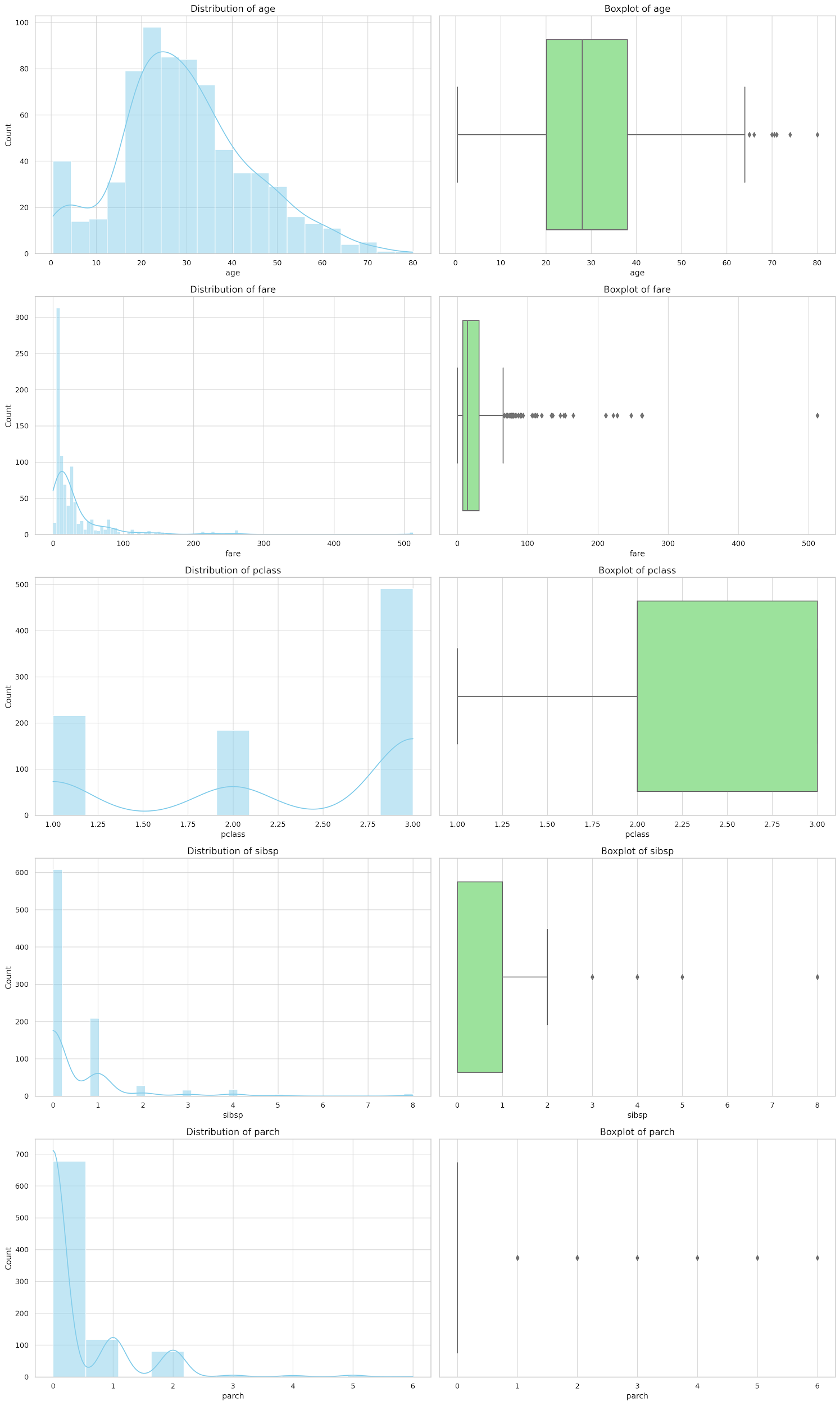
outlier\_df['Missing Values'] = missing\_values

print(outlier\_df)

**TASK 4**

###### **Prompt** – do univariate analysis of each of those 5 columns and make diagram bigger so i can clearly understand

OUTPUT

****

**🔍 Insights:**

1. **Age**
   * Distribution: Slightly right-skewed; most passengers are between 20 and 40.
   * Boxplot: Shows a few outliers above ~65 years.
2. **Fare**
   * Distribution: Highly right-skewed; most fares are below 100, but some are extremely high.
   * Boxplot: Lots of outliers on the higher end.
3. **Pclass**
   * Distribution: Discrete (1, 2, 3); most passengers are in 3rd class.
   * Boxplot: No outliers since it's categorical.
4. **SibSp (Siblings/Spouses Aboard)**
   * Distribution: Most people have 0 or 1, but a few have large family groups.
   * Boxplot: Many outliers above 2.
5. **Parch (Parents/Children Aboard)**
   * Distribution: Most passengers have 0; some have large families.
   * Boxplot: Outliers above 2 are visible.

Python code

import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

# Load your dataset

df = pd.read\_csv('titanic.csv') # Replace with your file path

# Columns to analyze

columns\_to\_analyze = ['age', 'fare', 'pclass', 'sibsp', 'parch']

# Set Seaborn and Matplotlib settings

sns.set(style="whitegrid")

plt.rcParams["figure.figsize"] = (12, 6)

# Create subplots: 5 rows (1 per column), 2 columns (histogram and boxplot)

fig, axes = plt.subplots(len(columns\_to\_analyze), 2, figsize=(18, 30))

for i, col in enumerate(columns\_to\_analyze):

# Histogram with KDE

sns.histplot(df[col].dropna(), kde=True, ax=axes[i, 0], color='skyblue')

axes[i, 0].set\_title(f'Distribution of {col}', fontsize=14)

axes[i, 0].set\_xlabel(col)

# Boxplot

sns.boxplot(x=df[col], ax=axes[i, 1], color='lightgreen')

axes[i, 1].set\_title(f'Boxplot of {col}', fontsize=14)

axes[i, 1].set\_xlabel(col)

# Adjust layout to prevent overlap

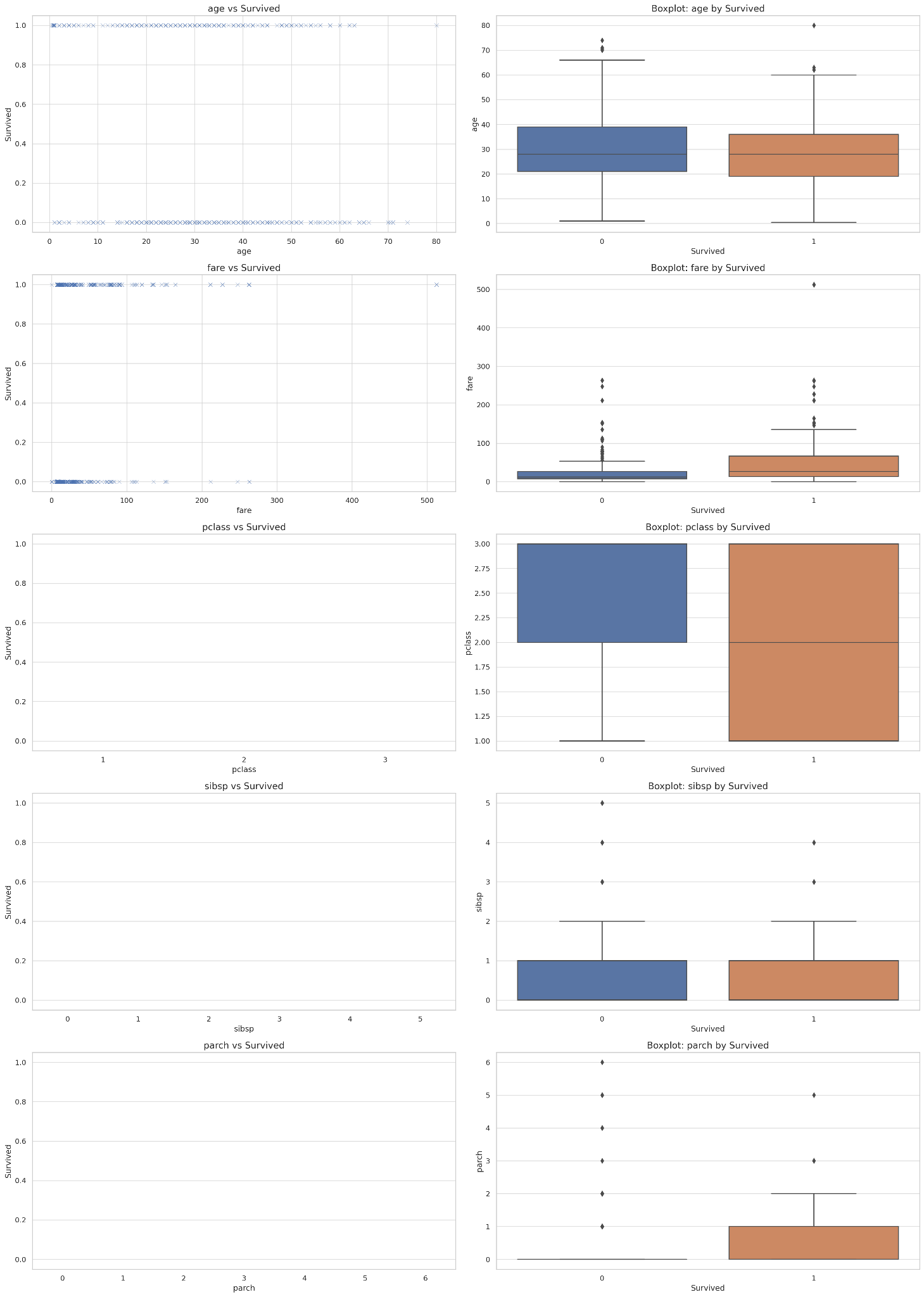
plt.tight\_layout()

plt.show()

**TASK 5**

**Prompt -** Do bivariate analysis of those 5 columns

OUTPUT



Python code

import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

df = pd.read\_csv('titanic.csv')

columns\_to\_analyze = ['age', 'fare', 'pclass', 'sibsp', 'parch']

df = df.dropna(subset=columns\_to\_analyze + ['survived'])

sns.set(style="whitegrid")

fig, axes = plt.subplots(len(columns\_to\_analyze), 2, figsize=(20, 28))

for i, col in enumerate(columns\_to\_analyze):

if df[col].nunique() < 15:

# Categorical or discrete variable

sns.stripplot(x=col, y='survived', data=df, ax=axes[i, 0], jitter=True)

else:

# Continuous variable

sns.scatterplot(x=df[col], y=df['survived'], ax=axes[i, 0])

axes[i, 0].set\_title(f'{col} vs Survived')

axes[i, 0].set\_xlabel(col)

axes[i, 0].set\_ylabel('Survived')

# Box plot always works

sns.boxplot(x='survived', y=col, data=df, ax=axes[i, 1])

axes[i, 1].set\_title(f'Boxplot: {col} by Survived')

axes[i, 1].set\_xlabel('Survived')

axes[i, 1].set\_ylabel(col)

plt.tight\_layout()

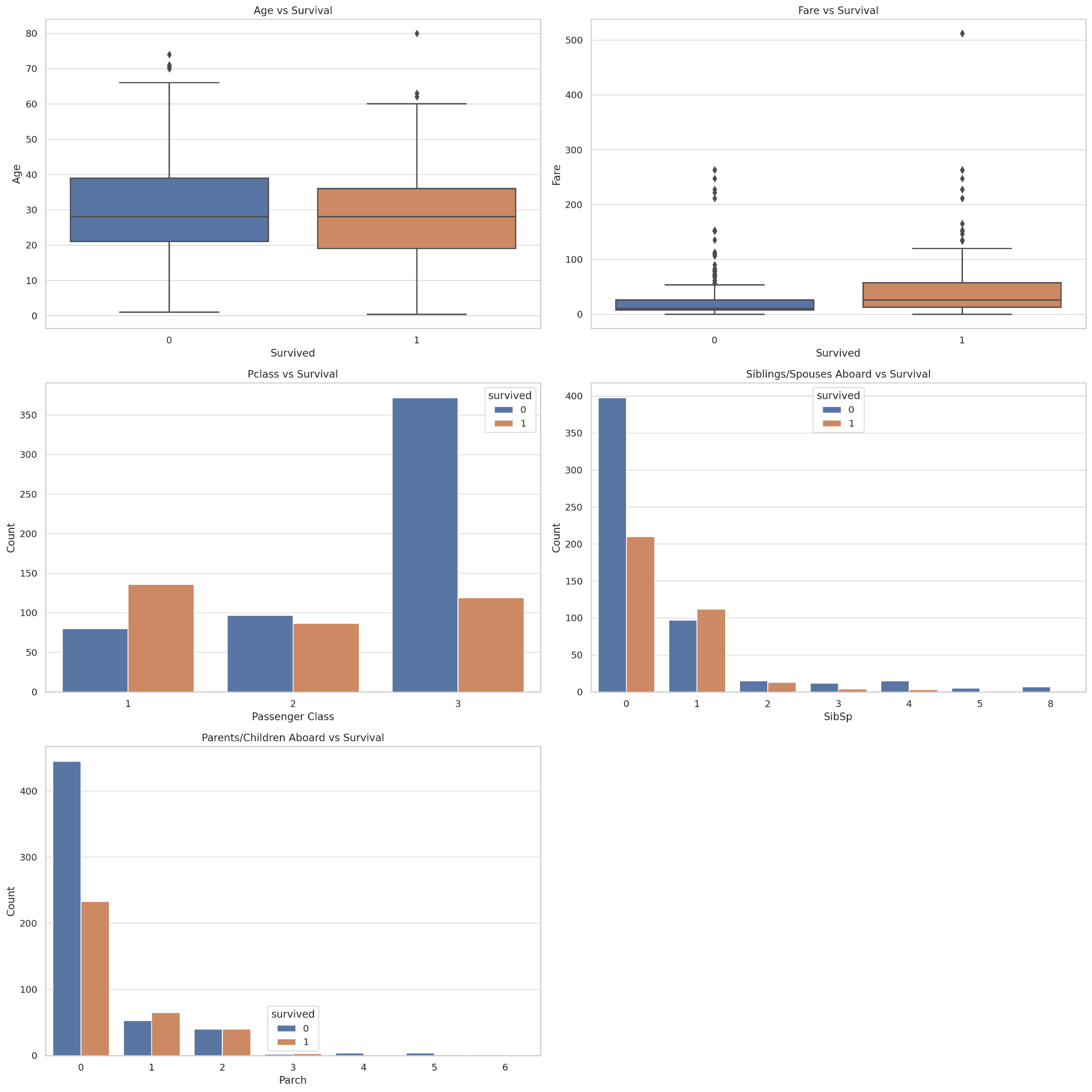
plt.show()

# # I have done a Bivariate analysis of survival vs those 5 columns and the results are more specific and interesting that's why I also included this here .

Prompt- This is a data set of passengers of titanic ship we have to do analysis in respect of survival. act as an expert data analyst and data scientist and perform bivariate analysis on each important 5 columns age ,fare , pclass ,sibsp ,parch and give clear graph also give a gogle friendly python code which i can use latter

Output-

**Target column**: survived (1 = survived, 0 = did not survive)



**📊 Bivariate Analysis Insights (with Survival)**

Here are observations from the visualizations:

1. **Age vs Survival**:  
   * Survivors tended to be slightly younger.
   * Many children survived, especially under age 10.
2. **Fare vs Survival**:  
   * Survivors generally paid higher fares.
   * Many high-fare passengers survived, indicating a class-based survival advantage.
3. **Pclass vs Survival**:  
   * 1st class had a much higher survival rate.
   * 3rd class had the lowest survival rate.
4. **SibSp vs Survival**:  
   * Passengers with 1–2 siblings/spouses had higher survival.
   * Solo travelers or those with many family members (3+) had lower survival.
5. **Parch vs Survival**:  
   * Passengers with 1–3 parents/children had better survival chances.
   * Too many dependents correlated with lower survival rates.

**Python code-**

# Titanic Bivariate Survival Analysis

import pandas as pd

import seaborn as sns

import matplotlib.pyplot as plt

# Load dataset

df = pd.read\_csv("titanic.csv") # Replace with your file path if needed

# Set style

sns.set(style="whitegrid")

# Create subplots

fig, axes = plt.subplots(3, 2, figsize=(18, 18))

axes = axes.flatten()

# Age vs Survival

sns.boxplot(x='survived', y='age', data=df, ax=axes[0])

axes[0].set\_title("Age vs Survival")

# Fare vs Survival

sns.boxplot(x='survived', y='fare', data=df, ax=axes[1])

axes[1].set\_title("Fare vs Survival")

# Pclass vs Survival

sns.countplot(x='pclass', hue='survived', data=df, ax=axes[2])

axes[2].set\_title("Pclass vs Survival")

# SibSp vs Survival

sns.countplot(x='sibsp', hue='survived', data=df, ax=axes[3])

axes[3].set\_title("Siblings/Spouses Aboard vs Survival")

# Parch vs Survival

sns.countplot(x='parch', hue='survived', data=df, ax=axes[4])

axes[4].set\_title("Parents/Children Aboard vs Survival")

# Remove empty plot

fig.delaxes(axes[5])

plt.tight\_layout()

plt.show()

**END**