Name- Radheya Deshmukh

Elevate Labs Internship Task 5

Dataset Used: test.csv (File Attached in the Repository)

Objective: The main goal is to explore a dataset using Python tools such as Pandas, Matplotlib, and Seaborn. By doing this, we aim to find patterns, trends, insights, and any unusual points in the data through exploration and visualizations.

1. **Read the dataset**
   df = pd.read_csv('test.csv')

```
In [3]: df = pd.read_csv('test.csv')

In [4]: df
Out[4]:
```

	PassengerId	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	892	3	Kelly, Mr. James	male	34.5	0	0	330911	7.8292	NaN	Q
1	893	3	Wilkes, Mrs. James (Ellen Needs)	female	47.0	1	0	363272	7.0000	NaN	S
2	894	2	Myles, Mr. Thomas Francis	male	62.0	0	0	240276	9.6875	NaN	Q
3	895	3	Wirz, Mr. Albert	male	27.0	0	0	315154	8.6625	NaN	S
4	896	3	Hirvonen, Mrs. Alexander (Helga E Lindqvist)	female	22.0	1	1	3101298	12.2875	NaN	S
...	...	...	...	...	...	...	...	...	...	...	...
413	1305	3	Spector, Mr. Woolf	male	NaN	0	0	A.5. 3236	8.0500	NaN	S
414	1306	1	Oliva y Ocana, Dona. Fermina	female	39.0	0	0	PC 17758	108.9000	C105	C
415	1307	3	Saether, Mr. Simon Sivertsen	male	38.5	0	0	SOTON/O.Q. 3101262	7.2500	NaN	S
416	1308	3	Ware, Mr. Frederick	male	NaN	0	0	359309	8.0500	NaN	S
417	1309	3	Peter, Master. Michael J	male	NaN	1	1	2668	22.3583	NaN	C

418 rows × 11 columns

2. **Preprocessing**
   df.isnull().sum()

```
In [8]: df.isnull().sum()

Out[8]: PassengerId      0
        Pclass           0
        Name             0
        Sex              0
        Age             86
        SibSp            0
        Parch            0
        Ticket           0
        Fare             1
        Cabin          327
        Embarked         0
        dtype: int64
```

3. **Basic Functions**

- df.info()

```
In [6]: df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 418 entries, 0 to 417
Data columns (total 11 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   PassengerId  418 non-null    int64
 1   Pclass       418 non-null    int64
 2   Name         418 non-null    object
 3   Sex          418 non-null    object
 4   Age          332 non-null    float64
 5   SibSp        418 non-null    int64
 6   Parch        418 non-null    int64
 7   Ticket       418 non-null    object
 8   Fare         417 non-null    float64
 9   Cabin        91 non-null     object
 10  Embarked     418 non-null    object
dtypes: float64(2), int64(4), object(5)
memory usage: 36.0+ KB
```

- df.describe()

```
In [7]: df.describe()
```

Out[7]:

	PassengerId	Pclass	Age	SibSp	Parch	Fare
count	418.000000	418.000000	332.000000	418.000000	418.000000	417.000000
mean	1100.500000	2.265550	30.272590	0.447368	0.392344	35.627188
std	120.810458	0.841838	14.181209	0.896760	0.981429	55.907576
min	892.000000	1.000000	0.170000	0.000000	0.000000	0.000000
25%	996.250000	1.000000	21.000000	0.000000	0.000000	7.895800
50%	1100.500000	3.000000	27.000000	0.000000	0.000000	14.454200
75%	1204.750000	3.000000	39.000000	1.000000	0.000000	31.500000
max	1309.000000	3.000000	76.000000	8.000000	9.000000	512.329200

- if 'Sex' in df.columns:
  print(df['Sex'].value_counts())

```
In [11]: if 'Sex' in df.columns:
             print(df['Sex'].value_counts())
```

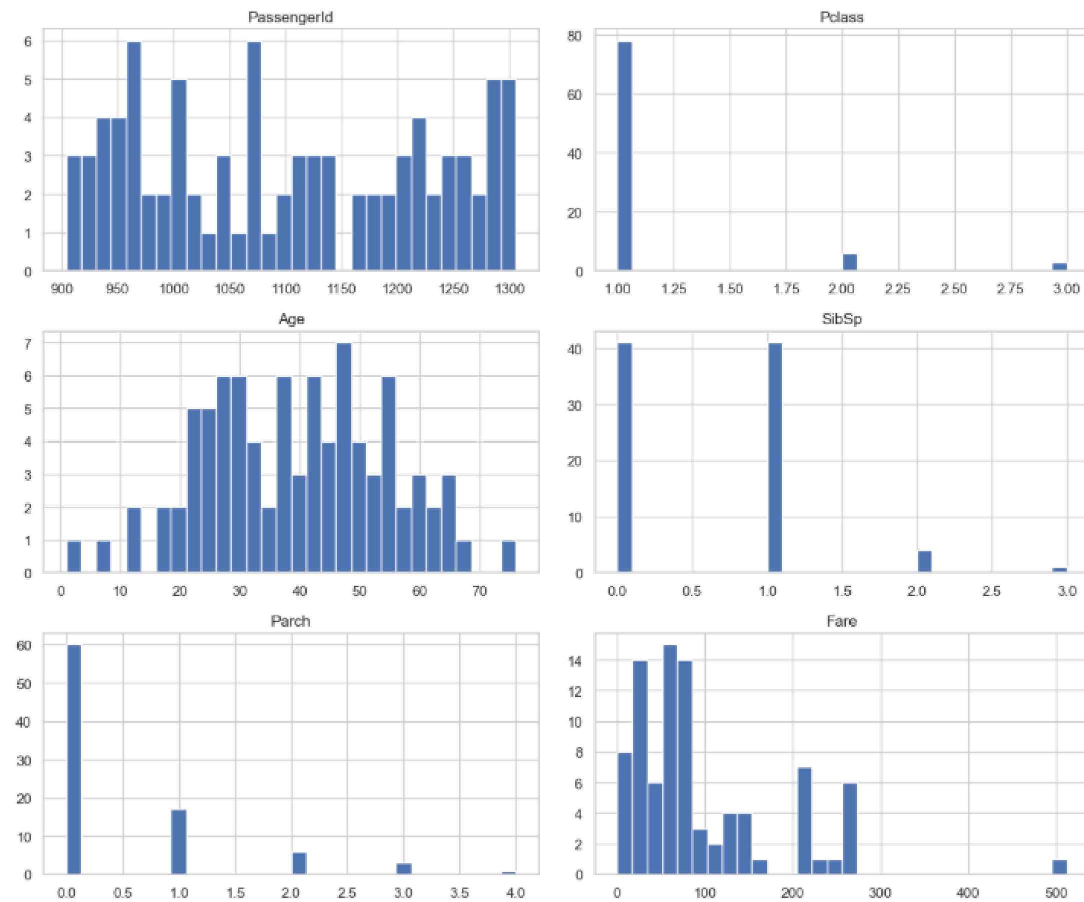
```
female    44
male      43
Name: Sex, dtype: int64
```

4. **Visualization**

- **Histograms** df.hist(figsize=(12, 10), bins=30)
  plt.tight_layout()
  plt.show()

In [17]:

```
df.hist(figsize=(12, 10), bins=30)
plt.tight_layout()
plt.show()
```

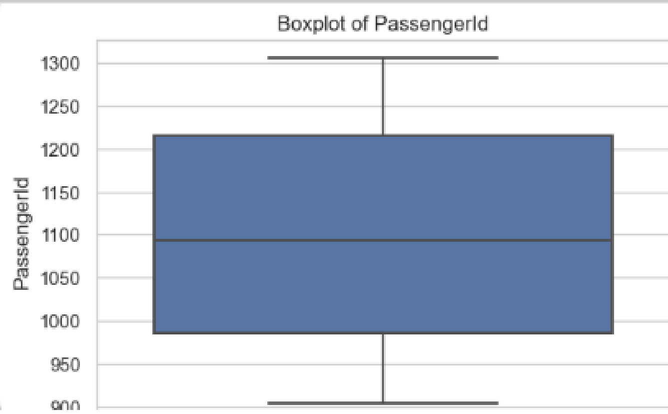


Observation: In the histograms, we observed the overall distribution of numeric features across the dataset. Most of the variables like 'Age', 'Fare', and others showed uneven distributions. For instance, 'Age' displayed a concentration around 20-40 years, indicating that most passengers belonged to this age group. The 'Fare' column was highly skewed towards the lower side, showing that the majority of passengers paid lower ticket prices, with very few paying extremely high fares. Some features like 'Pclass' exhibited a distinct distribution, confirming that they represent categorical-like data. These histograms help in understanding the spread and skewness of the data.

- **Boxplot:** numeric_cols = df.select_dtypes(include=np.number).columns
  for col in numeric_cols:
  plt.figure(figsize=(6,4))
  sns.boxplot(y=df[col])
  plt.title(f'Boxplot of {col}')

plt.show()

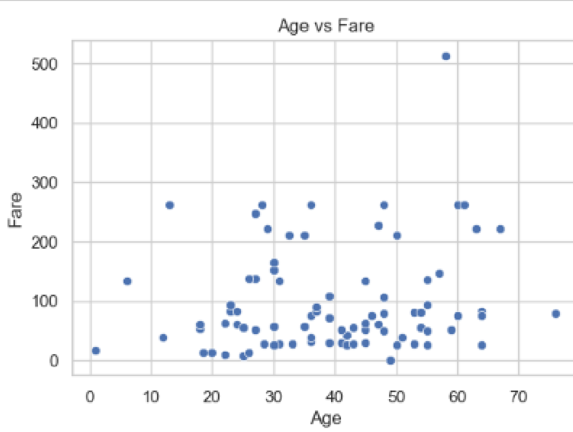
```
In [18]: numeric_cols = df.select_dtypes(include=np.number).columns
         for col in numeric_cols:
             plt.figure(figsize=(6,4))
             sns.boxplot(y=df[col])
             plt.title(f'Boxplot of {col}')
             plt.show()
```



Observation The boxplots provided clear insights into the presence of outliers in the dataset. Significant outliers were detected in the 'Fare' column, where a few passengers paid much higher amounts compared to others. The 'Age' column also showed slight outliers at the lower and higher ends of the age range. In general, most numeric features had their data points concentrated within the interquartile range (the box area), but a few extreme values were plotted as individual points outside the whiskers. These observations indicate that proper outlier handling or special treatment may be necessary before applying statistical models.

- Scatterplot: if 'Age' in df.columns and 'Fare' in df.columns:
  plt.figure(figsize=(6,4))
  sns.scatterplot(x='Age', y='Fare', data=df)
  plt.title('Age vs Fare')
  plt.show()

```
In [19]: if 'Age' in df.columns and 'Fare' in df.columns:
             plt.figure(figsize=(6,4))
             sns.scatterplot(x='Age', y='Fare', data=df)
             plt.title('Age vs Fare')
             plt.show()
```

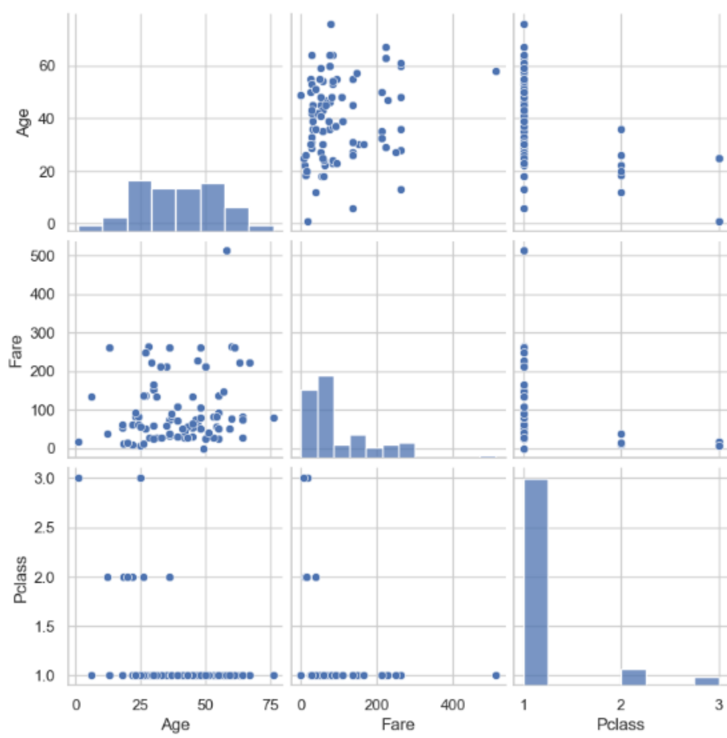


Observation: The scatter plot between 'Age' and 'Fare' revealed that there was no strong relationship between the two variables. Passengers of all ages tended to pay a wide range of fares, mostly concentrated towards lower values. A few older individuals paid exceptionally high fares, but overall, no consistent trend or correlation was visible. This observation suggests that age alone does not influence the fare significantly and highlights the need to look at other factors (like 'Pclass') for better insights into fare determination.

- **Pairplot:** selected_cols = ['Age', 'Fare', 'Pclass'] # Edit according to your dataset
  available_cols = [col for col in selected_cols if col in df.columns]
  if len(available_cols) > 1:
   sns.pairplot(df[available_cols])
   plt.show()

In [20]:
```
selected_cols = ['Age', 'Fare', 'Pclass']  # Edit according to your dataset
available_cols = [col for col in selected_cols if col in df.columns]

if len(available_cols) > 1:
    sns.pairplot(df[available_cols])
    plt.show()
```



Observation: The pairplot gave a combined view of the relationships between 'Age', 'Fare', and 'Pclass'. It clearly showed that 'Pclass' and 'Fare' are related — passengers from first class paid higher fares, while those from lower classes paid less. The distribution of 'Age' across different classes and fares was more scattered, indicating no strong direct influence of class or fare on age. These plots helped identify visible clusters and separation, especially for the 'Pclass' feature, making it an important variable in survival prediction or passenger segmentation analysis.

- **Heatmap:** plt.figure(figsize=(10,8))
  corr = df.corr()
  sns.heatmap(corr, annot=True, cmap="coolwarm", fmt='.2f')
  plt.title('Correlation Heatmap')
  plt.show()

In [21]:
```
plt.figure(figsize=(10,8))
corr = df.corr()
sns.heatmap(corr, annot=True, cmap="coolwarm", fmt='.2f')
plt.title('Correlation Heatmap')
plt.show()
```

Observation: The correlation heatmap provided a comprehensive view of how different numeric features are related. The strongest observation was a negative correlation between 'Pclass' and 'Fare', meaning higher classes (lower 'Pclass' number) generally had passengers paying higher fares. Other features like 'Age' showed very weak or no correlation with most variables. Importantly, the heatmap revealed no strong multicollinearity among variables, suggesting that each numeric feature carries mostly independent information useful for further modeling or analysis.