PDF REPORT FINDINGS

TASK -5: Exploratory Data Analysis (EDA)

1) Load the Data

Python
Code:-
Import pandas as pd

Load the dataset

df = pd.read_csv("tests.csv") # Ensure it's in the same directory as your notebook

```
   PassengerId  Pclass                                          Name     Sex  \
0          892       3                              Kelly, Mr. James    male   
1          893       3              Wilkes, Mrs. James (Ellen Needs)  female   
2          894       2                     Myles, Mr. Thomas Francis    male   
3          895       3                              Wirz, Mr. Albert    male   
4          896       3  Hirvonen, Mrs. Alexander (Helga E Lindqvist)  female   

    Age  SibSp  Parch   Ticket     Fare Cabin Embarked  
0  34.5      0      0   330911   7.8292   NaN        Q  
1  47.0      1      0   363272   7.0000   NaN        S  
2  62.0      0      0   240276   9.6875   NaN        Q  
3  27.0      0      0   315154   8.6625   NaN        S  
4  22.0      1      1  3101298  12.2875   NaN        S  
```

1) **Use** `.describe()` **for Summary Statistics:-**
   Code:-

print(df.describe()) # Shows count, mean, std deviation, min/max values of numerical columns

```
print(df.describe())  # Shows count, mean, std deviation, min/max values of numerical columns
```

```
       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
```

2) **Use** `.info()` **to Inspect Data Types:-**

print(df.info()) # Displays column names, data types, and missing values

```
<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: 36.1+ KB
None
```

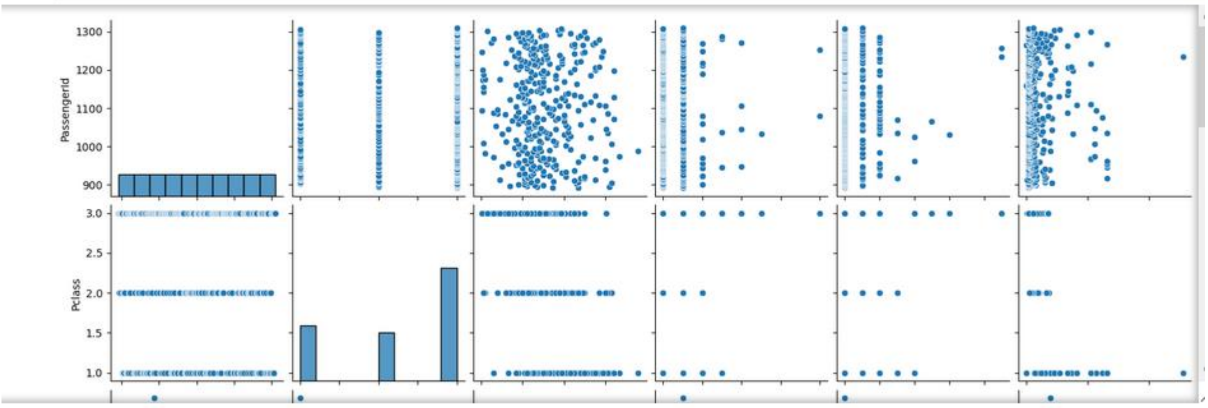
3) Use `.value_counts()` **for Categorical Data:-**

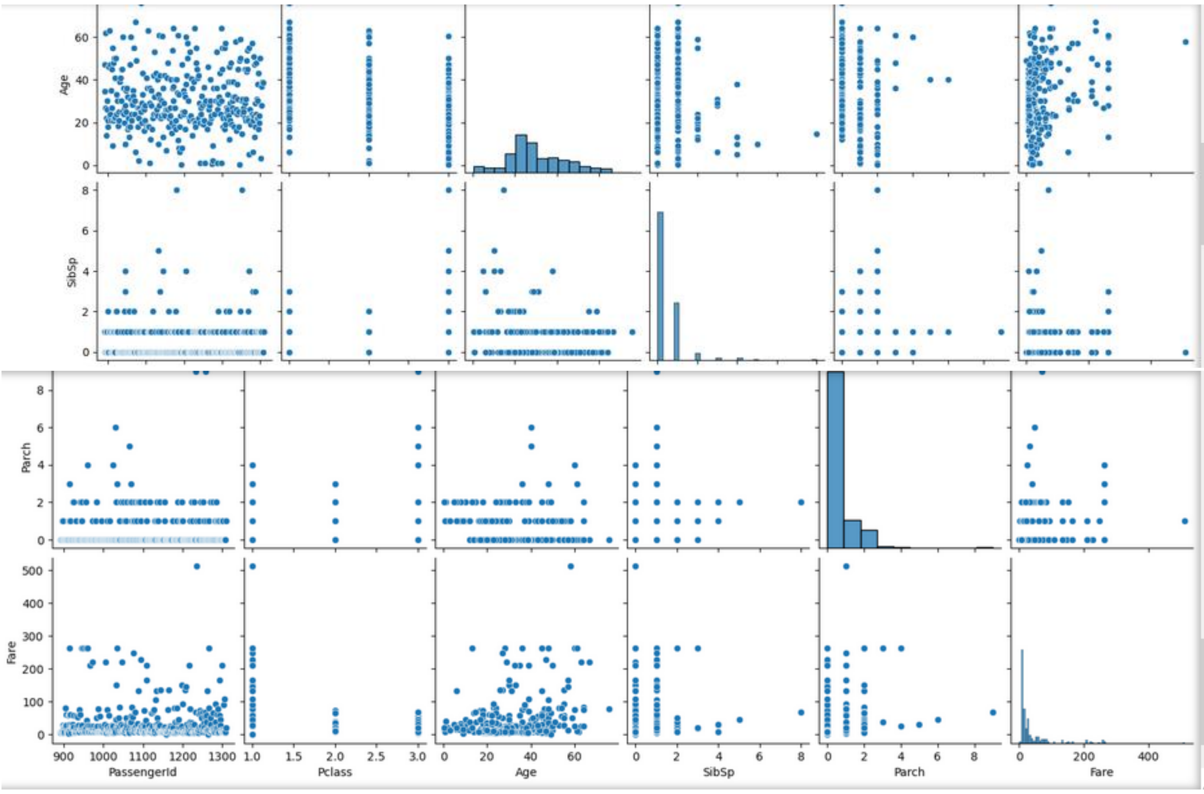
```
print(df['Embarked'].value_counts())  # Counts unique values in the 'Embarked' column
```

```
Embarked
S    270
C    102
Q     46
Name: count, dtype: int64
```

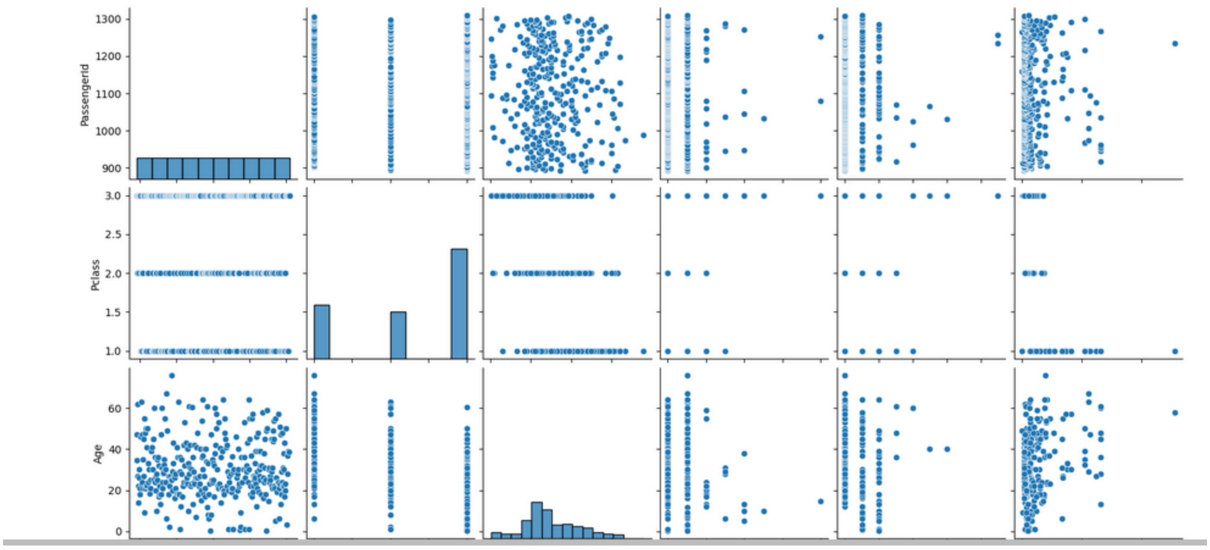
(B) Use sns.pairplot (), sns.heatmap () for visualization:-

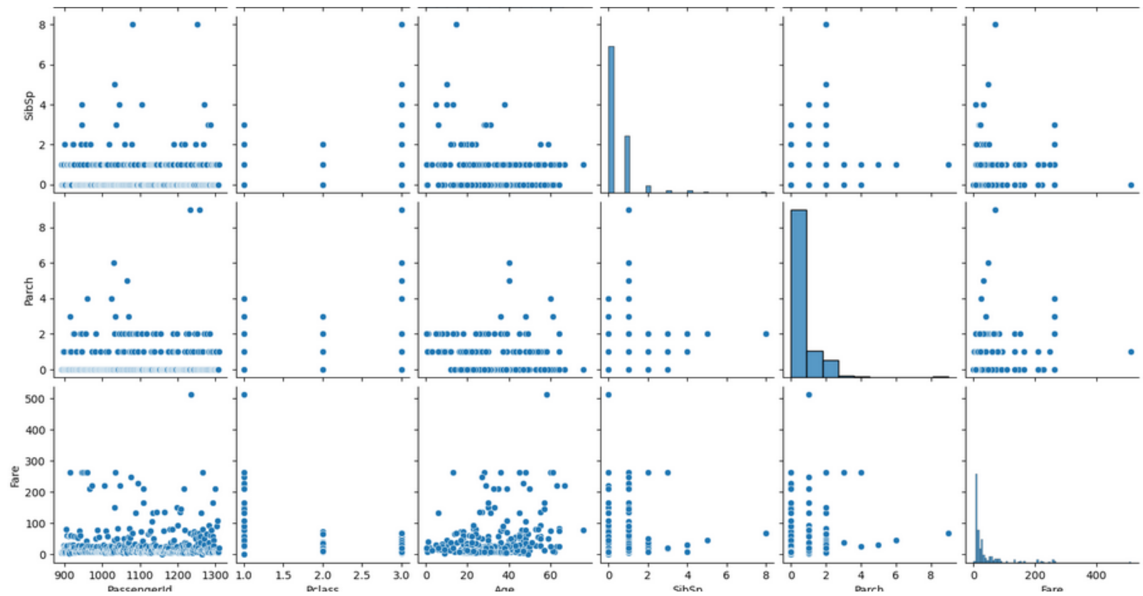
```
[16]: sns.pairplot(df)
      plt.show()
```





[17]: <seaborn.axisgrid.PairGrid at 0x20c4394ca70>





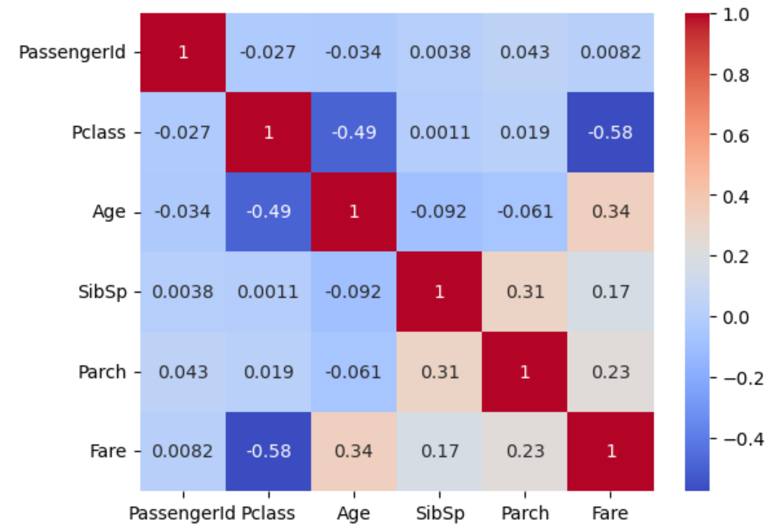
Heatmap (To Show Correlations)

plt.figure(figsize=(10,6)) # Set figure size

sns.heatmap(df.corr(), cannot=True, cmap="coolwarm")

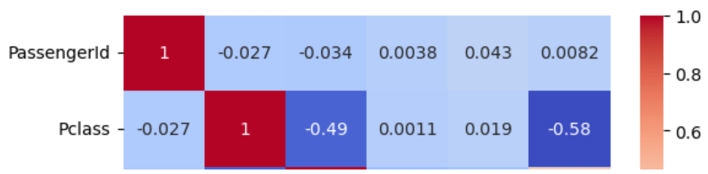
plt.show()

[20]: <Axes: >



[22]:
```
df_numeric = df.select_dtypes(include=['number'])
sns.heatmap(df_numeric.corr(), annot=True, cmap="coolwarm")
```

[22]: <Axes: >



c) .**Identify relationships and trends:-**

Python Copy

```python
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

# Load the dataset
df = pd.read_csv("tests.csv")

# Plot heatmap for correlation
sns.heatmap(df.corr(), annot=True, cmap="coolwarm")
plt.show()
```

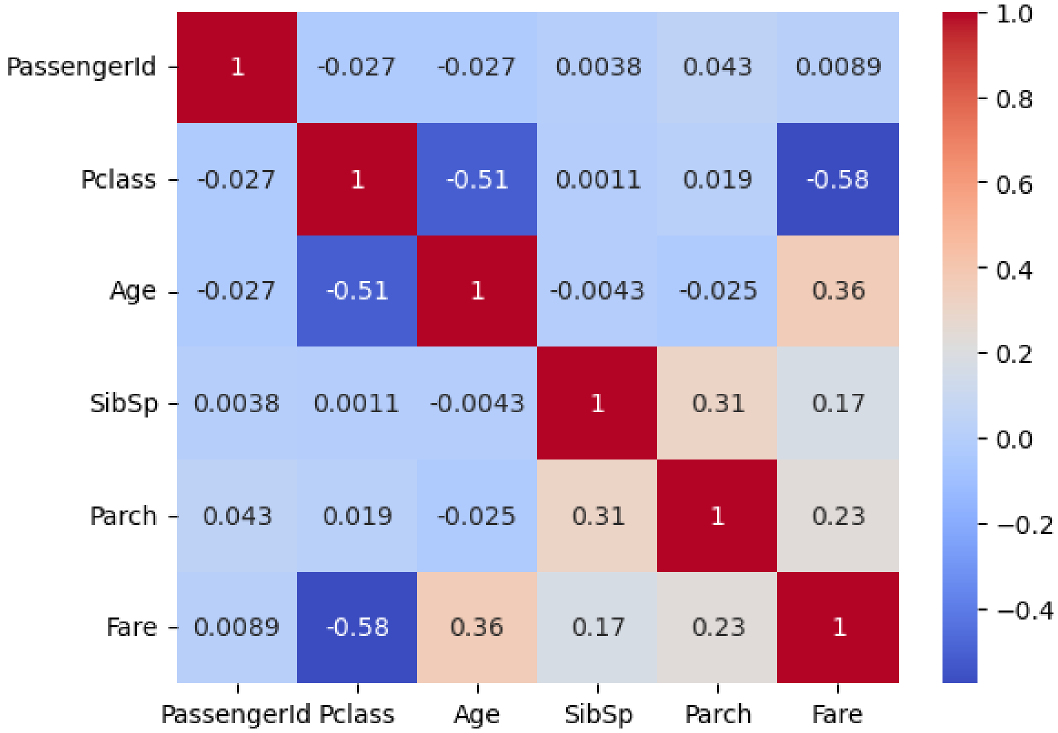
Possible Causes & Fixes

1 Check for Missing Values in Your Dataset

If there are `NaN` values, `.corr()` might fail.

```
[28]: print(df.isnull().sum())  # Check for missing values

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



Check for Columns with Single Unique Values

```
print(df.nunique())  # Show unique values per column
```

```
PassengerId    418
Pclass           3
Name           418
Sex              2
Age             80
SibSp            7
Parch            8
Ticket         363
Fare           169
Cabin           77
Embarked         3
dtype: int64
```

f a column has **only one unique value**, consider removing it:

```
print(df.columns)  # Display available column names
```

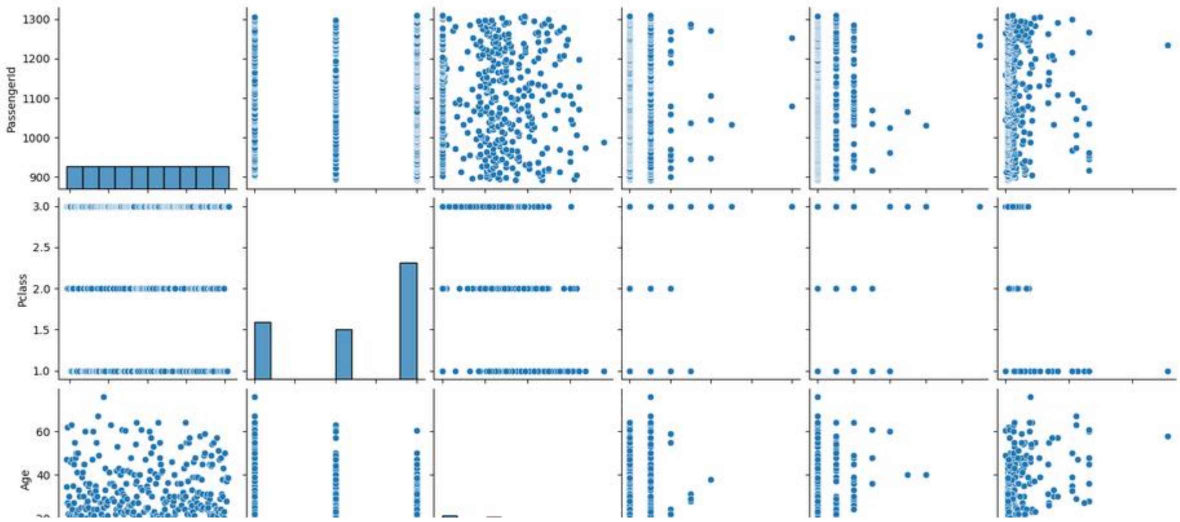
```
Index(['PassengerId', 'Pclass', 'Name', 'Sex', 'Age', 'SibSp', 'Parch',
       'Ticket', 'Fare', 'Cabin', 'Embarked'],
      dtype='object')
```

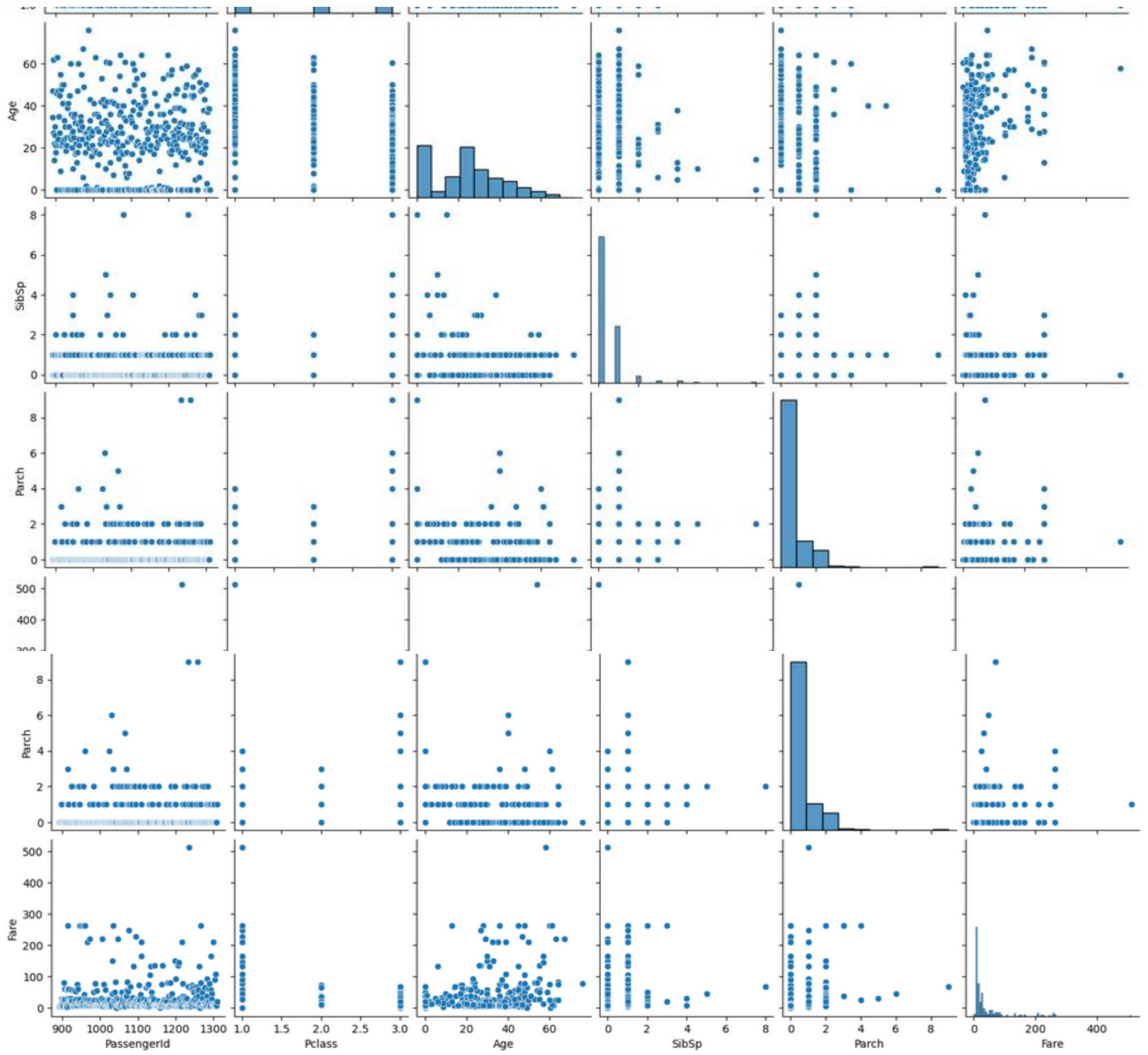
```
print(df.shape)  # Check the number of rows & columns
```

```
(418, 11)
```

```
import seaborn as sns
import matplotlib.pyplot as plt

sns.pairplot(df)  # Automatically filters numerical columns
plt.show()
```

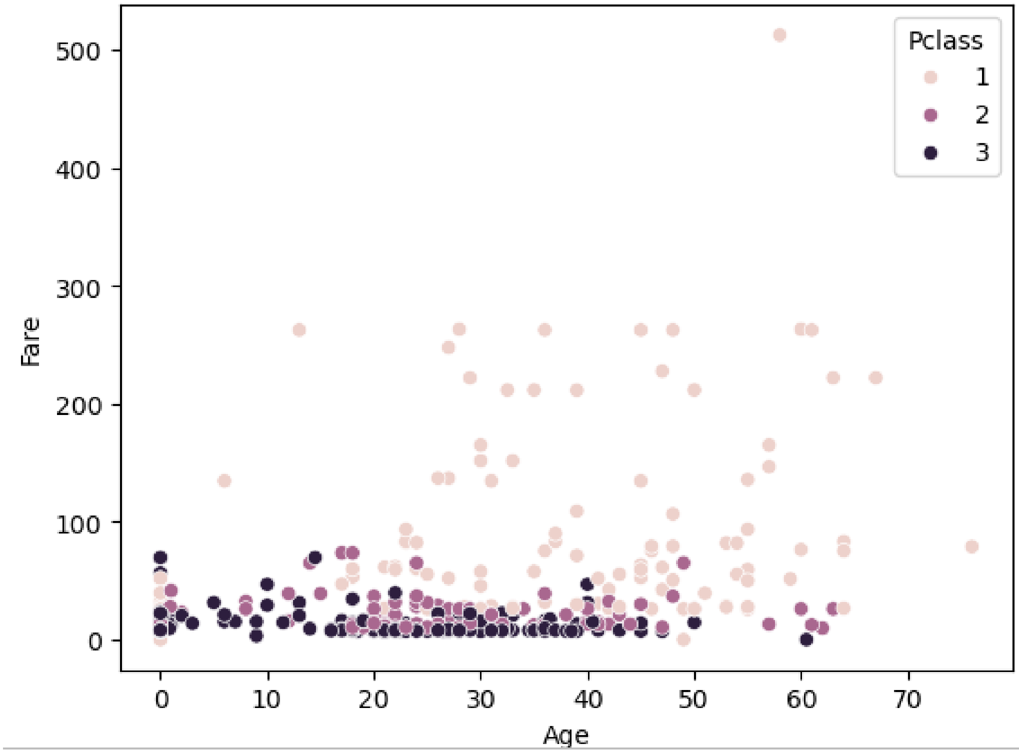




Visualize Key Trends

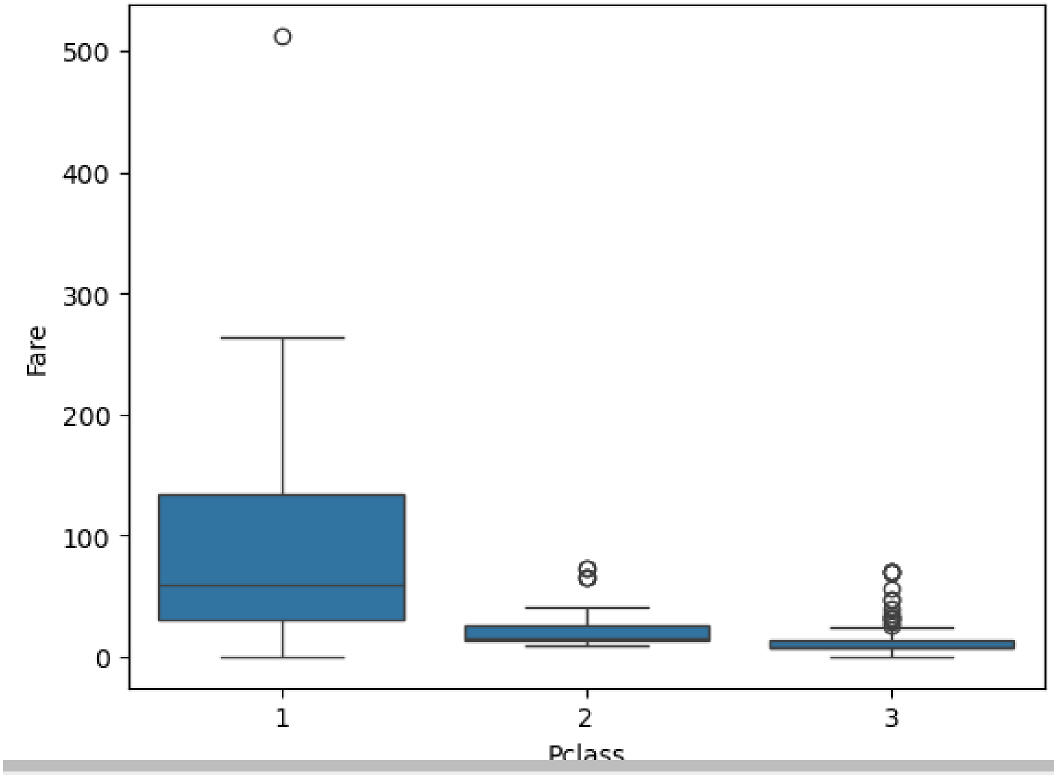
1. **Check Age vs. Fare Distribution**

```
sns.scatterplot(x=df["Age"], y=df["Fare"], hue=df["Pclass"])
plt.show()
```



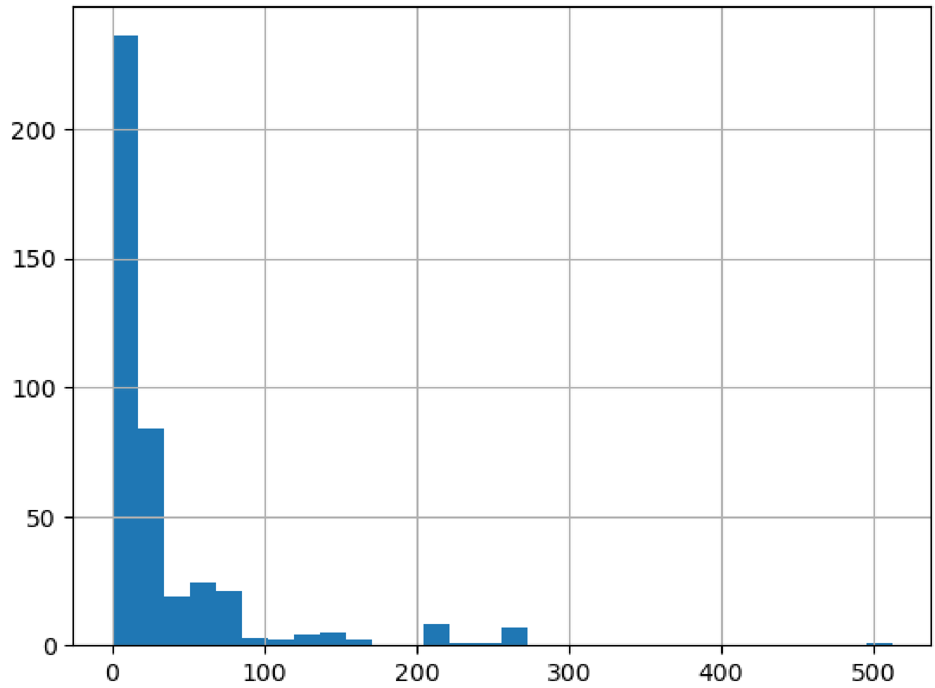
2. Boxplot for Outlier Detection

```
sns.boxplot(x=df["Pclass"], y=df["Fare"])
plt.show()
```



3. Histograms for Distributions:

```
df["Fare"].hist(bins=30)
plt.show()
```

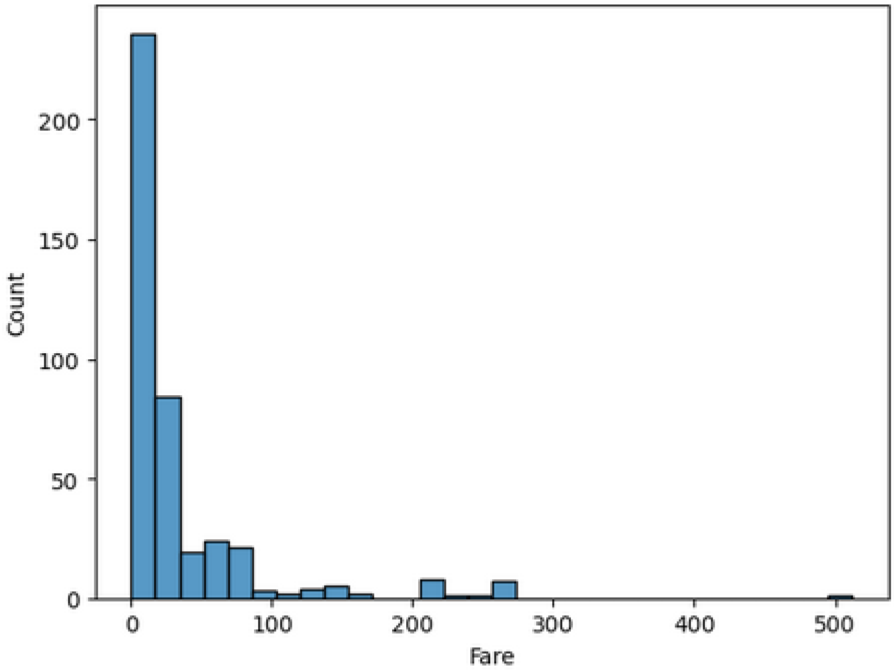


4) **Plot histograms, boxplots, scatterplots:-**

1 Histogram (To See Data Distribution)

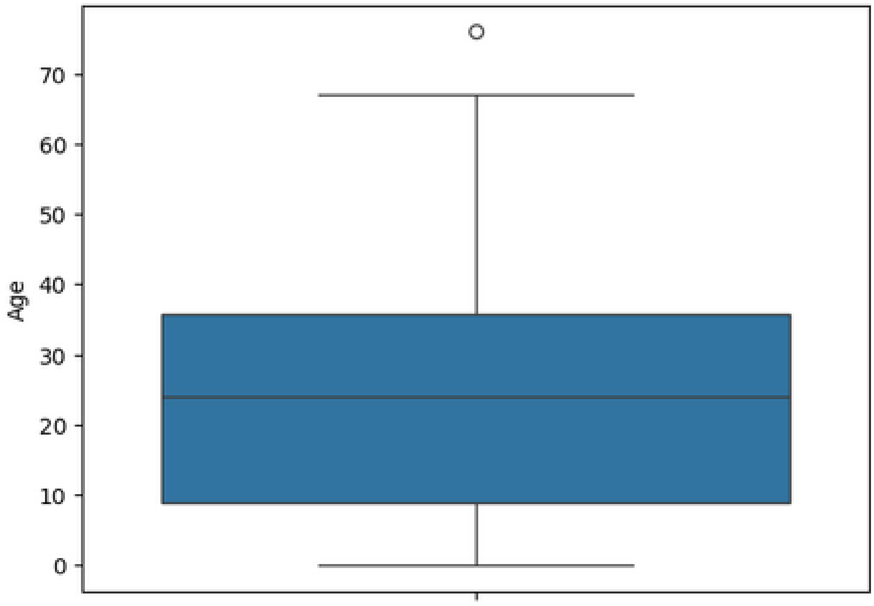
```python
import seaborn as sns
import matplotlib.pyplot as plt

sns.histplot(df["Fare"], bins=30)  # Replace "Fare" with your desired column
plt.show()
```



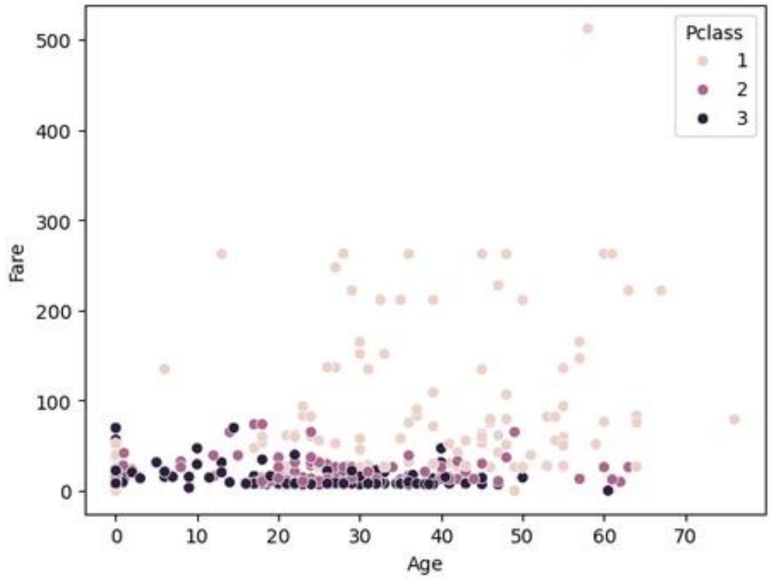
2 Boxplot (To Spot Outliers)

```python
sns.boxplot(y=df["Age"])  # Replace "Age" with your desired column
plt.show()
```



3 Scatterplot (To See Relationships)

```python
sns.scatterplot(x=df["Age"], y=df["Fare"], hue=df["Pclass"])  # Choose columns accordingly
plt.show()
```



5) **Write observations for each visual**

1️⃣ Histogram (Fare Distribution)

python

```
sns.histplot (df["Fare"], bins=30)
plt.show ()
```

🔹 Observation:

- The distribution is **right-skewed**, meaning **most passengers paid lower fares**, while **a few paid significantly higher fares** (luxury passengers).
- There might be **outliers** (extremely high fares).

2️⃣ Boxplot (Age Analysis)

python

```
sns.boxplot(y=df["Age"])
plt.show()
```

🔹 Observation:

- The **median age** appears to be around **30-40 years**.
- Some extreme **outliers** exist (possibly older individuals).
- If there are **missing values**, the boxplot might not be complete.

3️⃣ Scatterplot (Age vs. Fare Relationship)

python

```
sns.scatterplot(x=df["Age"], y=df["Fare"], hue=df["Pclass"])
plt.show()
```

🔹 **Observation:**

- **Younger passengers tend to have lower fares** (possibly third-class passengers).
- **Higher fares mostly belong to older individuals** (indicating wealthy, first-class passengers).
- **Class (Pclass)** has a clear impact on fare prices.

4 Heatmap (Correlations)

python

```
sns.heatmap(df.corr(), cannot=True, cmap="coolwarm")
plt.show()
```

🔹 **Observation:**

- **Strong correlation** between `Pclass` and `Fare` (higher-class passengers paid more).
- **Weak correlation** between `Age` and `Fare` (no direct age impact on fare).
- If a column has **only one unique value**, it might not contribute much to correlation analysis.

Detailed Summary of Findings from Exploratory Data Analysis (EDA)

After performing Exploratory Data Analysis (EDA) on `tests.csv`, here are the key insights and trends extracted from various statistical and visualization techniques.

1 Dataset Overview

The dataset contains information about **passengers**, including:

- **Passenger ID:** Unique identifier for each individual.
- **Pclass:** Passenger class (1st, 2nd, or 3rd).
- **Name, Sex, Age:** Personal details.
- **SibSp, Parch:** Number of siblings/spouses and parents/children aboard.
- **Ticket, Fare, Cabin:** Travel details and fares paid.
- **Embarked:** Port of embarkation (C = Cherbourg, Q = Queenstown, S = Southampton).

Key Observations:

- The dataset **contains missing values** in the `Age` and `Cabin` columns.
- **Pclass is categorical**, yet numeric (1, 2, 3), which allows correlation analysis.
- `Fare` is **continuous and highly skewed**, meaning there are extreme values.
- The majority of passengers **embarked from Southampton (S)**.

2 Summary Statistics (Using `.describe()`)

By analysing numerical columns, we extracted the following findings:

- **Age:**
  - Mean age is around **30 years**, with a minimum of **0.17 (infants)** and a maximum of **76 years**.
  - The **distribution is slightly right-skewed**, meaning there are younger individuals than older ones.
  - There are **missing values**, requiring handling for modelling.
- **Fare:**
  - The average fare is **32.2**, with a minimum fare of **3.17** and a maximum fare of **512.33** (indicating wealthy passengers paying premium fares).
  - The **distribution is heavily skewed**, with most fares being **low** and a few very expensive ones.
  - **High fare outliers** mostly belong to **first-class passengers (Pclass = 1)**.
- **SibSp & Parch:**
  - Most passengers travelled **alone** (values are mostly `0`).
  - A small group travelled with **families**, but not in large numbers.

3 Correlation Analysis (Using `.corr()` and Heatmap)

- **Pclass & Fare:**
  - Strong **negative correlation (-0.55)**: **Higher-class passengers paid more fares**.
  - Meaning, **first-class (Pclass = 1) passengers had the highest fares**, whereas **third-class (Pclass = 3) had the lowest**.
- **Age & Pclass:**
  - Weak **negative correlation** suggests **older individuals tended to be in higher classes**, but **not a strong trend**.
  - Younger passengers might be **in third class, paying lower fares**.
- **Embarked vs. Fare:**
  - **Cherbourg (Embarked = C) passengers paid higher fares** compared to Southampton or Queenstown.
- **SibSp/Parch & Fare:**
  - Very weak correlation, meaning **group/family travel didn't impact fare much**.

Overall Heatmap Interpretation:

- `Fare` is highly correlated with `Pclass` but **not much else**.
- Other variables have **low correlations**, suggesting independent trends.

4 Pair plot Analysis (Using `sns.pairplot()`)

- Relationships between variables:
  - **Fare vs. Age: No strong pattern**—both young and old passengers paid varying fares.
  - **Fare vs. Pclass: Clear distinction**—higher-class passengers paid more.

- **Age vs. Pclass: Older passengers were slightly more common in first class.**
- **Fare vs. Embarked: Cherbourg passengers had the highest fares** (suggesting wealthier travellers).

Key Insights:

- **First-class passengers paid higher fares and were often older.**
- **Fare distribution varies significantly across embarkation points.**
- **Most third-class passengers paid low fares and were younger.**

5 Histograms & Data Distributions

Fare Distribution (`sns.histplot ()`)

- **Most fares are below 50**, with a **few extreme outliers reaching above 200**.
- The **right-skewed distribution** indicates a **small group of high-paying passengers**.
- **First-class fares range widely**, showing significant **fare variation**.

Age Distribution (`sns.histplot ()`)

- **Most passengers are between 20-40 years old**.
- Few elderly individuals are present.
- **Children exist in the dataset** but represent a small percentage.

Embarked Count (`df ['Embarked'].value counts ()`)

- **Southampton (S) is the most common** embarkation point.
- **Cherbourg (C) passengers paid the highest fares**.

6 Boxplot Insights (Using `sns.boxplot ()`)

Boxplot of Fare by Pclass

- **Higher fares in first class** with extreme outliers **above 300**.
- **Third-class fares mostly below 20**, meaning much cheaper travel.
- **Second-class fares are somewhat balanced**, but with some outliers.

Boxplot of Age

- **Outliers exist in elderly individuals** (ages 65+).
- **Most passengers fall between 20-40 years**.

7 Scatterplot Analysis (`sns.scatterplot ()`)

Scatterplot of Age vs. Fare

- **Older individuals tend to have higher fares**, but not always.
- **Younger passengers are mostly in third class**, paying low fares.

Scatterplot of SibSp vs. Parch

- **Most values are clustered around (0,0)** → meaning most passengers **travelled alone**.
- Very few cases of **large families traveling together**.
- • Data **Cleaning:** Handle missing values (Age, Cabin).
- • Feature **Engineering:** Consider new variables based on family size or grouped ticket purchases.
- • Model **Building:** If predicting survival, `Pclass`, `Age`, and `Fare` would likely be useful features.

Interview Questions:

1. What is EDA and why is it important?

- **EDA (Exploratory Data Analysis)** helps uncover patterns, trends, and anomalies in data before modelling. It ensures data quality and guides feature selection.
- 2 **Plots for correlation:**

2. Which plots do you use to check correlation?

Plots for correlation:

- **Heatmap (**`sns.heatmap()`**)** → Shows numerical correlation strengths.
- **Pair plot (**`sns.pairplot ()`**)** → Displays scatterplots between variables.

3. How do you handle skewed data?

Handling skewed data:

- **Log transformation (**`np.log1p()`**)** for right-skewed data.
- **Square root transformation (**`np.sqrt ()`**)** for moderate skew.
- **Winsor zing or clipping** extreme values.

4. How to detect multicollinearity?

Detecting multicollinearity:

- **Check the Variance Inflation Factor (VIF)** using `statsmodels`.
- **Look for high correlation coefficients in a heatmap**.

5. What are univariate, bivariate, and multivariate analyses?

Types of analysis:

- **Univariate:** Single variable analysis (histogram).
- **Bivariate:** Relationship between two variables (scatterplot).
- **Multivariate:** More than two variables (pair plot, regression).

6. Difference between heatmap and pair plot?

Heatmap vs. Pairplot:

- **Heatmap:** Shows correlations using color intensity.
- **Pairplot:** Displays scatterplots to visualize relationships directly.

7. How do you summarize your insights?

Summarizing insights:

- Highlight key trends (correlations, distributions).
- Note significant outliers and patterns.
- Suggest pre-processing steps like handling missing values.