Task 1: Data Cleaning & Preprocessing Objective:

Learn how to clean and prepare raw data for ML.

Tools: Python, Pandas, NumPy, Matplotlib/Seaborn

Hints/Mini Guide:
1.Import the dataset and explore basic info (nu ls, data types).
2.Handle missing values using mean/median/imputation.
3.Convert categorical features into numerical using encoding.
4.Normalize/standardize the numerical features.
5.Visualize outliers using boxplots and remove them. Dataset:

You can use any dataset relevant to the task, e.g., Titanic Dataset link to download: click here to download dataset

CODE:

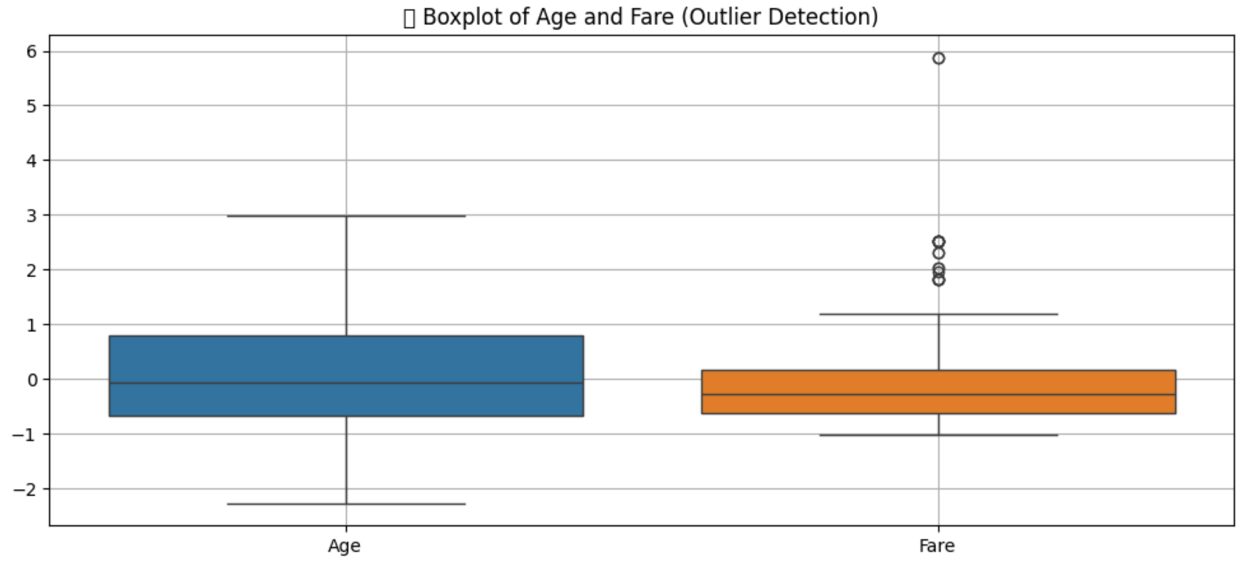
```
# -----------------------------------------
# Titanic Data Preprocessing - All Steps
# -----------------------------------------

# Step 1: Import libraries
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.preprocessing import LabelEncoder, StandardScaler

# Step 2: Upload the CSV file
from google.colab import files
uploaded = files.upload()

# Step 3: Load dataset
df = pd.read_csv("/content/Titanic-Dataset.csv")

# Step 4: Explore basic info
print("🔍 First 5 rows:\n", df.head())
print("\n📊 Dataset Info:")
print(df.info())
print("\n📈 Summary Statistics:\n", df.describe())

# Step 5: Handle missing values
print("\n❗ Missing values before handling:\n", df.isnull().sum())

# Fill numerical 'Age' with median
df['Age'].fillna(df['Age'].median(), inplace=True)

# Fill categorical 'Embarked' with mode
df['Embarked'].fillna(df['Embarked'].mode()[0], inplace=True)

# Drop rows with any remaining missing values (if any)
df.dropna(inplace=True)

print("\n✅ Missing values after handling:\n", df.isnull().sum())

# Step 6: Encode categorical variables
le = LabelEncoder()
df['Sex'] = le.fit_transform(df['Sex'])           # male=1, female=0
df['Embarked'] = le.fit_transform(df['Embarked'])  # example: S=2, C=0, Q=1

print("\n🔢 Encoded 'Sex' and 'Embarked':\n", df[['Sex', 'Embarked']].head())

# Step 7: Standardize numerical features
scaler = StandardScaler()
df[['Age', 'Fare']] = scaler.fit_transform(df[['Age', 'Fare']])

print("\n📐 Standardized 'Age' and 'Fare':\n", df[['Age', 'Fare']].head())

# Step 8: Visualize outliers using boxplots
plt.figure(figsize=(12, 5))
sns.boxplot(data=df[['Age', 'Fare']])
plt.title("📦 Boxplot of Age and Fare (Outlier Detection)")
plt.grid()
plt.show()

# Step 9: Remove outliers using IQR
for col in ['Age', 'Fare']:
    Q1 = df[col].quantile(0.25)
    Q3 = df[col].quantile(0.75)
    IQR = Q3 - Q1
    lower = Q1 - 1.5 * IQR
    upper = Q3 + 1.5 * IQR
    df = df[(df[col] >= lower) & (df[col] <= upper)]

print("\n✅ Dataset shape after removing outliers:", df.shape)
```

OUTPUT:

```
Titanic-Dataset.csv(text/csv) - 61194 bytes, last modified: 6/23/2025 - 100% done
Saving Titanic-Dataset.csv to Titanic-Dataset (1).csv
🔍 First 5 rows:
   PassengerId  Survived  Pclass  \
0            1         0       3
1            2         1       1
2            3         1       3
3            4         1       1
4            5         0       3

                                                Name     Sex   Age  SibSp  \
0                            Braund, Mr. Owen Harris    male  22.0      1
1  Cumings, Mrs. John Bradley (Florence Briggs Th...  female  38.0      1
2                             Heikkinen, Miss. Laina  female  26.0      0
3       Futrelle, Mrs. Jacques Heath (Lily May Peel)  female  35.0      1
4                           Allen, Mr. William Henry    male  35.0      0

   Parch            Ticket     Fare Cabin Embarked
0      0         A/5 21171   7.2500   NaN        S
1      0          PC 17599  71.2833   C85        C
2      0  STON/O2. 3101282   7.9250   NaN        S
3      0            113803  53.1000  C123        S
4      0            373450   8.0500   NaN        S

📊 Dataset Info:
<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
None

📈 Summary Statistics:
       PassengerId    Survived      Pclass         Age       SibSp  \
count   891.000000  891.000000  891.000000  714.000000  891.000000   
mean    446.000000    0.383838    2.308642   29.699118    0.523008   
std     257.353842    0.486592    0.836071   14.526497    1.102743   
min       1.000000    0.000000    1.000000    0.420000    0.000000   
25%     223.500000    0.000000    2.000000   20.125000    0.000000   
50%     446.000000    0.000000    3.000000   28.000000    0.000000   
75%     668.500000    1.000000    3.000000   38.000000    1.000000   
max     891.000000    1.000000    3.000000   80.000000    8.000000   

            Parch        Fare  
count  891.000000  891.000000  
mean     0.381594   32.204208  
std      0.806057   49.693429  
min      0.000000    0.000000  
25%      0.000000    7.910400  
50%      0.000000   14.454200  
75%      0.000000   31.000000  
max      6.000000  512.329200  

❗ Missing values before handling:
 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

✅ Missing values after handling:
 PassengerId    0
Survived       0
Pclass         0
Name           0
Sex            0
Age            0
SibSp          0
Parch          0
Ticket         0
Fare           0
Cabin          0
Embarked       0
dtype: int64

🔢 Encoded 'Sex' and 'Embarked':
    Sex  Embarked
1     0         0
3     0         2
6     1         2
10    0         2
11    0         2

📐 Standardized 'Age' and 'Fare':
         Age      Fare
1   0.192508 -0.065466
3  -0.006645 -0.310494
6   1.254659 -0.327170
10 -2.064562 -0.800999
11  1.520196 -0.668266
/tmp/ipython-input-3-3493694269.py:29: FutureWarning: A value is trying to be set on a copy of a DataFrame
or Series through chained assignment using an inplace method.
The behavior will change in pandas 3.0. This inplace method will never work because the intermediate object
on which we are setting values always behaves as a copy.

For example, when doing 'df[col].method(value, inplace=True)', try using 'df.method({col: value},
inplace=True)' or df[col] = df[col].method(value) instead, to perform the operation inplace on the original
object.


  df['Age'].fillna(df['Age'].median(), inplace=True)
```

```
/tmp/ipython-input-3-3493694269.py:32: FutureWarning: A value is trying to be set on a copy of a DataFrame
or Series through chained assignment using an inplace method.
The behavior will change in pandas 3.0. This inplace method will never work because the intermediate object
on which we are setting values always behaves as a copy.

For example, when doing 'df[col].method(value, inplace=True)', try using 'df.method({col: value},
inplace=True)' or df[col] = df[col].method(value) instead, to perform the operation inplace on the original
object.


  df['Embarked'].fillna(df['Embarked'].mode()[0], inplace=True)
/usr/local/lib/python3.11/dist-packages/IPython/core/pylabtools.py:151: UserWarning: Glyph 128230
(\N{PACKAGE}) missing from font(s) DejaVu Sans.
  fig.canvas.print_figure(bytes_io, **kw)
```



```
✅ Dataset shape after removing outliers: (187, 12)
```