






# **EXPLORATORY DATA ANALYSIS (EDA) PROJECT**

**SWATHI P|920821106083|ECE**

# Exploratory Data Analysis (EDA) Project

## Objective







The purpose of this project is to carry out Exploratory Data Analysis (EDA) on a dataset to thoroughly understand its structure, derive key statistical measures, handle missing data, and encode categorical variables. The analysis also involves visualizing the data to uncover patterns and insights that will inform further preprocessing or modeling tasks.   

---




## Dataset Overview

**DatasetName:** TitanicDataset

**Source:** Publicly available dataset from [Data Science Dojo Datasets](#)

**Dataset Description:** The Titanic dataset provides information about passengers aboard the Titanic, including demographic details, ticket class, fare, and survival status. It is commonly used for binary classification tasks to predict survival.      

### Key Features:

- **Numerical Columns:** Age, Fare, SibSp (Number of Siblings/Spouses Aboard), Parch (Number of Parents/Children Aboard)
- **Categorical Columns:** Name, Sex, Ticket, Cabin, Embarked, Survived (Target variable)  
  

---

## Function Descriptions

### 1. Load and Inspect the Dataset

This step involves loading the dataset into a Pandas DataFrame and exploring its structure.

- **Libraries Used:** pandas, numpy, matplotlib, seaborn
- **Key Tasks:**
  - Load the dataset using `pandas.read_csv()`.
  - Display the shape of the dataset to understand its size (number of rows and columns).
  - Inspect data types of each column to identify numerical and categorical features.
  - Identify missing values using `isnull().sum()`.

- Summarize the dataset using `df.describe()` to get basic statistical information like mean, standard deviation, and percentiles for numerical columns.



### # 1. Load and Inspect the Dataset

# Load the Titanic dataset (or replace with your dataset)

```
url = "https://raw.githubusercontent.com/datasciencedojo/datasets/master/titanic.csv"
```

```
data = pd.read_csv(url)
```

# Display basic information

```
print("Shape of the dataset:", data.shape)
```

```
print("\nData types:\n", data.dtypes)
```

```
print("\nNumber of missing values:\n", data.isnull().sum())
```

```
print("\nSummary statistics:\n", data.describe())
```


## 2. Basic Statistical Measures

This step calculates measures of spread and identifies outliers in the dataset.

- **Measures of Spread:**

- **Range:** Difference between maximum and minimum values in a column.
- **Variance:** Measure of how data points differ from the mean.
- **Standard Deviation:** Square root of variance, showing data dispersion.
- **Interquartile Range (IQR):** Range between the 75th percentile and 25th percentile values.

- **Outlier Detection:**

- Visualize outliers using box plots for numerical columns.
- Columns like Fare and Age often contain outliers in this dataset. 

### # 2. Basic Statistical Measures

# Compute measures of spread

```
def compute_spread(df, column):
```

```
    spread_metrics = {
```

```
        'Range': df[column].max() - df[column].min(),
```

```

    'Variance': df[column].var(),
    'Standard Deviation': df[column].std(),
    'IQR': df[column].quantile(0.75) - df[column].quantile(0.25)
}
return spread_metrics

numerical_columns = data.select_dtypes(include=['number']).columns
spread_results = {}
for col in numerical_columns:
    spread_results[col] = compute_spread(data, col)

print("\nMeasures of Spread:")
for col, metrics in spread_results.items():
    print(f"{col}: {metrics}")

# Identify outliers using box plots
for col in numerical_columns:
    plt.figure(figsize=(8, 4))
    sns.boxplot(x=data[col])
    plt.title(f"Box plot for {col}")
    plt.show()

```

### 3. Encoding Categorical Data

This step converts categorical data into numerical formats suitable for modeling.

- **Label Encoding:**
  - Converts each unique category into an integer value using LabelEncoder from sklearn.
  - Example: Sex (Male, Female) becomes (0, 1).
- **One-Hot Encoding:**
  - Converts categories into binary columns using pd.get\_dummies().

- Example: Embarked (C, Q, S) becomes Embarked\_C, Embarked\_Q, Embarked\_S with binary indicators (0/1).

- **Comparison:**

- Label encoding is simpler but may introduce ordinal relationships between categories.
- One-hot encoding increases dimensionality but avoids ordinal implications.



### # 3. Encoding Categorical Data

#### # Identify categorical columns

```
categorical_columns = data.select_dtypes(include=['object']).columns
print("\nCategorical Columns:", categorical_columns.tolist())
```

#### # Label Encoding

```
label_encoder = LabelEncoder()
data_encoded = data.copy()
for col in categorical_columns:
    data_encoded[col] = label_encoder.fit_transform(data_encoded[col].astype(str))

print("\nData after Label Encoding:\n", data_encoded.head())
```

#### # One-Hot Encoding

```
data_onehot = pd.get_dummies(data, columns=categorical_columns)
print("\nData after One-Hot Encoding:\n", data_onehot.head())
```




### 4. Data Visualization

Visualizations are key for understanding distributions and relationships in the data.

- **Histograms:**

- Display frequency distributions of numerical columns.
- Example: Age shows a right-skewed distribution with most passengers aged 20-40.

- **Dist Plots:**

- Combine histograms with KDE (Kernel Density Estimation) to show data spread.
- Example: Fare exhibits a long tail due to outliers.
- **Box Plots:**
  - Highlight potential outliers and visualize data spread.
  - Example: Fare has significant outliers at higher values.   

#### # 4. Data Visualization

##### # Histogram for numerical columns

for col in numerical\_columns:

```
plt.figure(figsize=(8, 4))
plt.hist(data[col].dropna(), bins=30, edgecolor='k', alpha=0.7)
plt.title(f"Histogram of {col}")
plt.xlabel(col)
plt.ylabel("Frequency")
plt.show()
```

##### # Dist plot for spread visualization

for col in numerical\_columns:




```
plt.figure(figsize=(8, 4))
sns.histplot(data[col], kde=True, bins=30)
plt.title(f"Dist plot of {col}")
plt.xlabel(col)
plt.ylabel("Density")
plt.show()
```

---




### Key Insights

### Data Inspection




- **Shape of Dataset:** 891 rows and 12 columns.
- **Missing Values:**
  - Age and Embarked have moderate missing values.

- Cabin has a high percentage of missing data, which may require exclusion.
- **Data Types:**
  - Numerical: 5 columns
  - Categorical: 7 columns   

### Statistical Measures

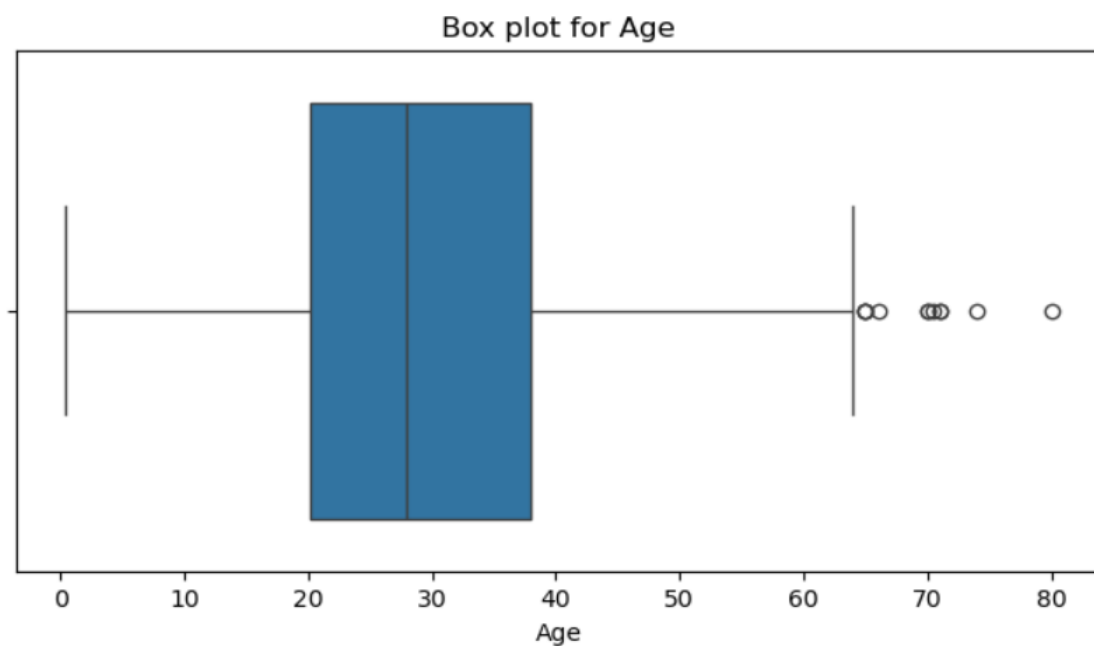
- **Outliers Identified:**
  - Fare: Significant outliers in high-ticket prices.
  - Age: A few outliers present but less prominent.
- **Spread Metrics:**
  - Fare has a range of 512.33 and an IQR of 23.09, indicating a skewed distribution.
  - Age has a smaller IQR but contains gaps due to missing values.   

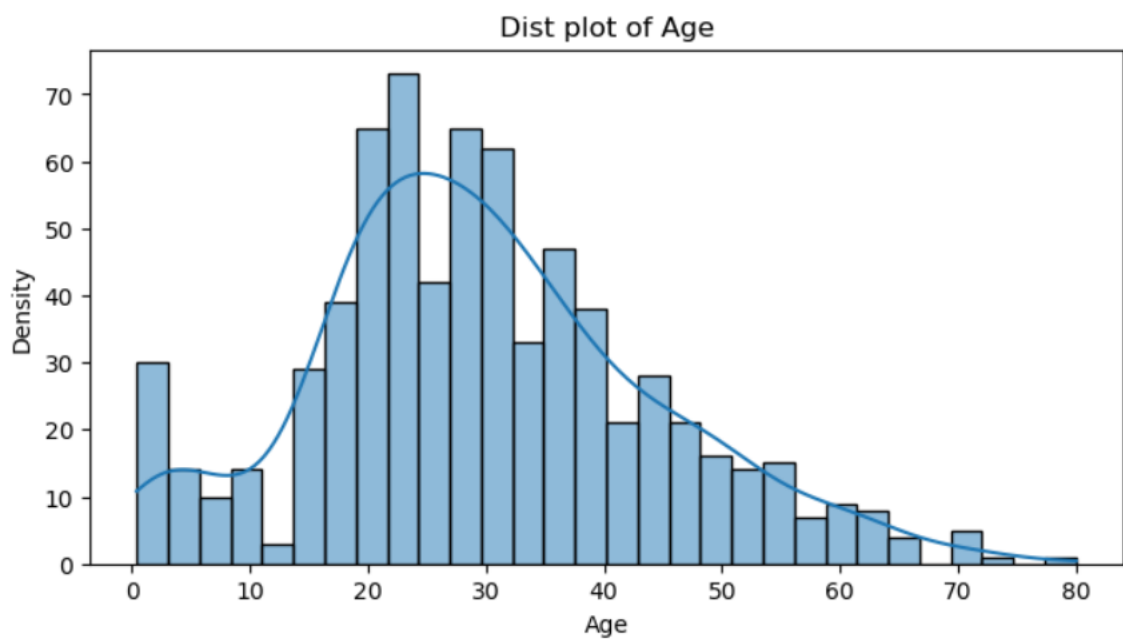
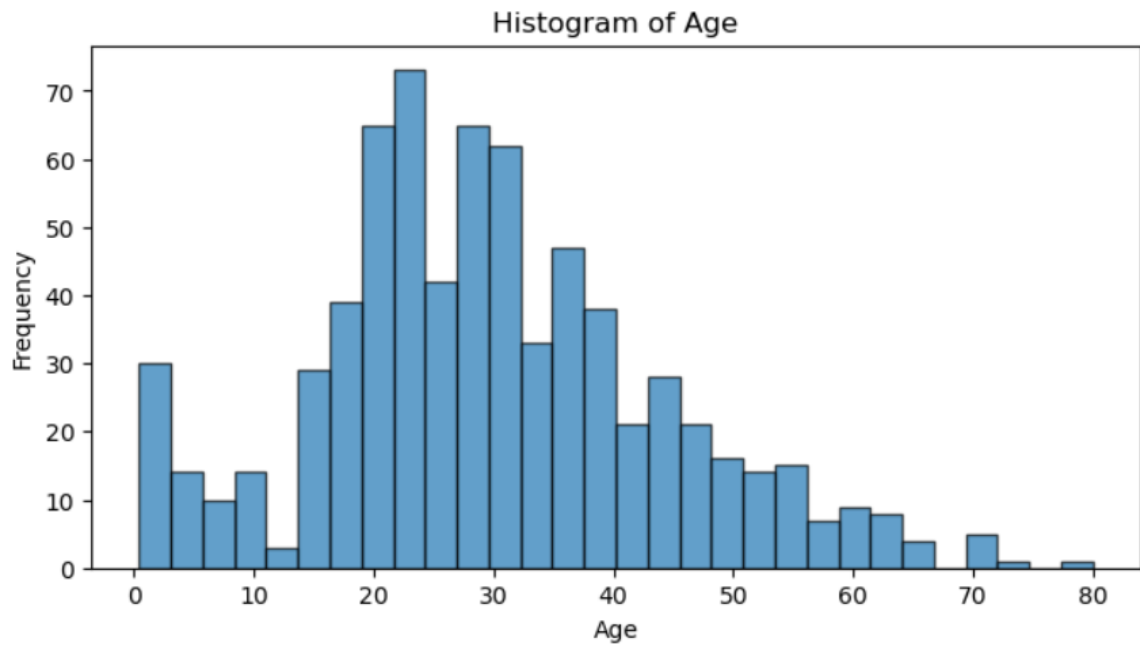
### Encoding Categorical Data

- **Label Encoding:** Simplified conversion but unsuitable for complex categories.
- **One-Hot Encoding:** Comprehensive but increases dimensionality, especially for features like Embarked.   

### Visualization Insights

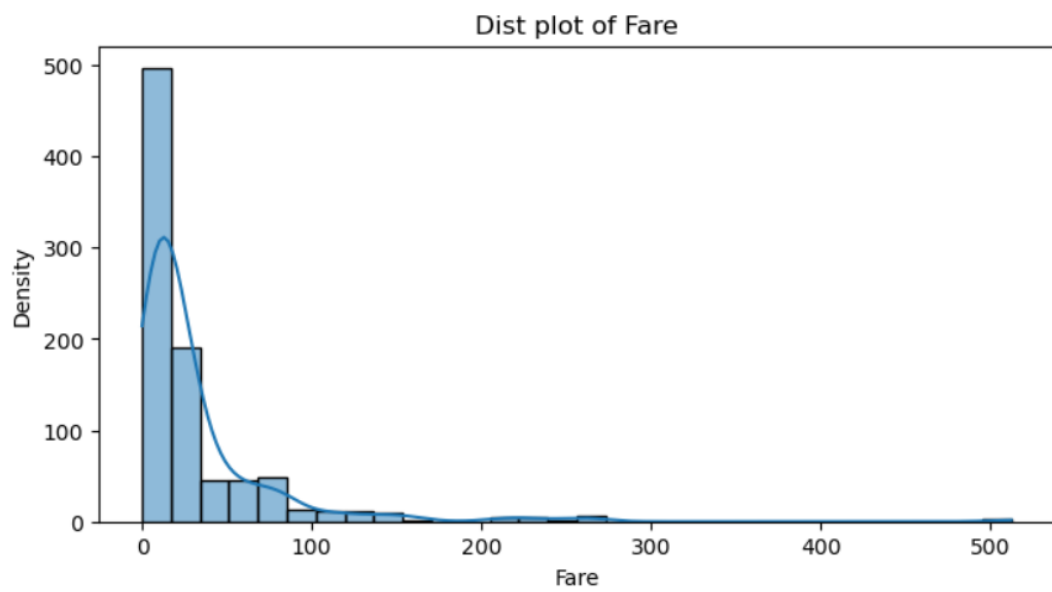
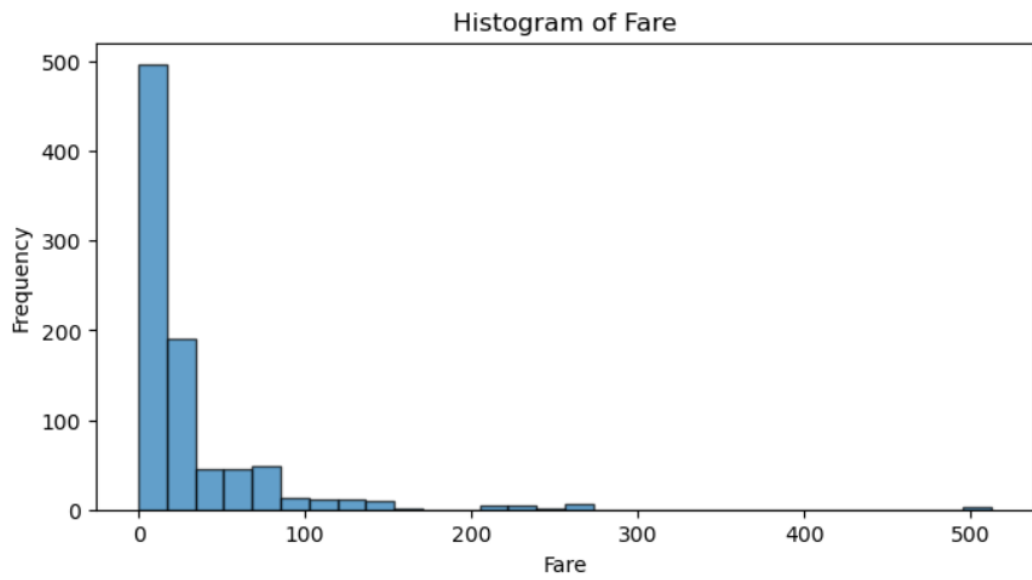
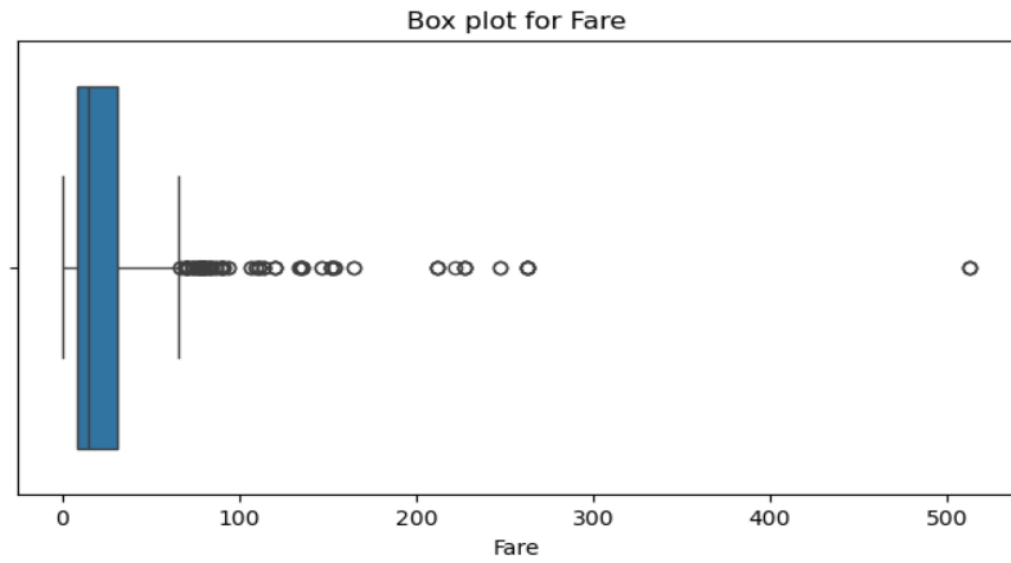
- **Age:** Majority of passengers are young adults aged 20-40.








- **Fare:** Most passengers paid under \$50, but a small subset paid extremely high fares.







- **Embarked:** Most passengers boarded from port S.   

---

### Summary of Findings

1. The dataset contains missing values, particularly in Cabin, which may need exclusion or imputation.
  2. Outliers in Fare and Age require preprocessing steps like capping or transformation.
  3. Categorical encoding methods must be chosen based on the intended use case (e.g., modeling requirements).
  4. Visualizations highlight key trends and support decision-making for feature engineering and cleaning.  
-