Exploratory Data Analysis (EDA)

What is EDA?

Definition:
EDA is the process of summarizing, visualizing, and interpreting data to uncover patterns and relationships. It's like detective work—understanding what your data is trying to tell you.

Why is EDA Important?

- Identify missing or incorrect data.
- Discover relationships between variables.
- Detect outliers and anomalies.
- Validate assumptions before applying statistical or machine learning models.

Steps in EDA

Step 1: Importing Libraries

```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
```

- **Pandas**: Used for data manipulation and analysis.
- **NumPy**: Provides support for numerical operations like mean, median, and standard deviation.
- **Seaborn**: A library for statistical data visualization.
- **Matplotlib**: A library for creating static, interactive, and animated visualizations.

Step 2: Loading the Dataset

```
data = pd.read_csv('data.csv')
```

- Reads a dataset from a CSV file and stores it in the data DataFrame.
- You can replace 'data.csv' with the path to your dataset file.

Step 3: Inspecting the Data

```
data.info()
data.describe()
```

- **data.info()**: Provides a summary of the dataset, including the number of non-null values, data types, and memory usage.
- **data.describe()**: Provides statistical summary for numerical columns, such as mean, median, and standard deviation.

Step 4: Checking for Missing Values

```
data.isnull().sum()
```

- This checks for missing data in each column.
- The .isnull() function identifies missing values, and .sum() gives the count of missing values for each column.

Step 5: Visualizing Data

5.1 Histogram for Distribution

```
sns.histplot(data['ColumnName'])
plt.show()
```

- **Purpose**: To visualize the frequency distribution of a numerical column.
- **sns.histplot()**: Creates a histogram.
- Replace 'ColumnName' with the column you want to analyze.

5.2 Boxplot for Outliers

```
sns.boxplot(x='ColumnName', data=data)
plt.show()
```

- **Purpose**: To detect outliers in the data.
- **sns.boxplot()**: Creates a box plot, showing the median, quartiles, and potential outliers.

5.3 Scatter Plot for Relationships

```
sns.scatterplot(x='Variable1', y='Variable2', data=data)
plt.show()
```

- **Purpose**: To identify the relationship between two numerical variables.
- Replace 'Variable1' and 'Variable2' with the column names.

Step 6: Statistical Analysis

6.1 Correlation Matrix

```
correlation = data.corr()
sns.heatmap(correlation, annot=True, cmap='coolwarm')
plt.show()
```

- **Purpose**: To find the relationships between numerical variables.
- **data.corr()**: Computes the correlation matrix for numerical columns.
- **sns.heatmap()**: Creates a heatmap to visualize correlations.

6.2 Central Tendency Measures

```
mean_value = data['ColumnName'].mean()
median_value = data['ColumnName'].median()
mode_value = data['ColumnName'].mode()[0]
```

- **Purpose**: To compute the mean, median, and mode of a column.
- Replace 'ColumnName' with your desired column.

6.3 Dispersion Measures

```
std_dev = data['ColumnName'].std()
variance = data['ColumnName'].var()
```

- **Purpose**: To calculate the variability in the data.
- **std()**: Standard deviation, indicating data spread.
- **var()**: Variance, a measure of data dispersion.