**UNIT – II: Data Visualization**

**Visualization and Graphing: Visualizing Categorical Distributions, Visualizing Numerical Distributions, Overlaid Graphs, plots, and summary statistics of Exploratory Data Analysis (EDA). Exploring Univariate Data - Histograms -Stem-and-Leaf Quantile Based Plots - Continuous Distributions -Quantile Plots- QQ Plot- Box Plots.**

**1. Introduction to Visualization in EDA**

Exploratory Data Analysis (EDA) is a crucial step in data analysis, where visualization techniques help to uncover patterns, detect anomalies, and extract insights from data. Visualization is divided into:

* Categorical Distributions
* Numerical Distributions
* Overlaid Graphs and Plots
* Summary Statistics

**Effective visualization techniques help:**

* Identify data trends and outliers.
* Understand the underlying data distribution.
* Make data-driven decisions.

**2. Visualizing Categorical Distributions**

**Categorical data is often represented using:**

* Bar Charts: Display frequency counts of categories. Suitable for both nominal and ordinal data.
* Pie Charts: Useful for showing proportions. Avoid using this for datasets with many categories.
* Count Plots: Ideal for representing categorical counts with better scalability than pie charts.

**Example Explanation - Bar Chart**

Suppose we have a dataset containing customer feedback ratings categorized as "Excellent," "Good," "Average," and "Poor." A bar chart can visually demonstrate which category received the most responses.

**Example Code:**

import seaborn as sns

import matplotlib.pyplot as plt

sns.countplot(x='Rating', data=df)

plt.title('Customer Feedback Ratings')

plt.xlabel('Rating Categories')

plt.ylabel('Count')

plt.show()

**3. Visualizing Numerical Distributions**

**a) Histograms**

* Displays the frequency distribution of numerical data.
* Ideal for understanding data spread and skewness.
* Select an appropriate number of bins to balance detail and clarity.

**Example Explanation - Histogram**

Suppose we have a dataset of students' exam scores. A histogram helps reveal score distributions, such as a bell curve indicating normal distribution or skewness.

**Example Code:**

plt.hist(df['Exam\_Scores'], bins=15, color='skyblue')

plt.title('Distribution of Exam Scores')

plt.xlabel('Scores')

plt.ylabel('Frequency')

plt.grid(True)

plt.show()

**b) Stem-and-Leaf Plots**

* Splits data values into "stems" (leading digits) and "leaves" (trailing digits).
* Effective for small datasets.

**Example (Manual Approach):**

7 | 0 1 4 5

8 | 0 3 6 8

9 | 1 2 5 9

In this example, scores like 70, 71, and 74 are represented in the '7' stem, while 80, 83, and 86 are in the '8' stem.

**c) Box Plots**

* Represents minimum, first quartile (Q1), median, third quartile (Q3), and maximum.
* Ideal for spotting outliers and data symmetry.

**Example Explanation - Box Plot**

For visualizing income levels across different groups, a box plot efficiently shows data distribution, highlighting median values and identifying outliers.

**Example Code:**

sns.boxplot(x='Income\_Group', y='Income', data=df)

plt.title('Income Distribution Across Groups')

plt.show()

**d) Quantile-Based Plots**

* Displays data distribution based on percentile values.
* Useful for checking data spread and skewness.

**Example Explanation - QQ Plot Suppose you want to confirm if students' scores follow a normal distribution. A QQ plot will compare the sample data’s quantiles to a theoretical normal distribution.**

**Example Code:**

import scipy.stats as stats

import matplotlib.pyplot as plt

stats.probplot(df['Scores'], dist="norm", plot=plt)

plt.title("QQ Plot for Scores")

plt.show()

**4. Continuous Distributions**

**Continuous data is often visualized using:**

* Density Plots: For smoothed distributions with continuous variables.
* ECDF (Empirical Cumulative Distribution Function): Visualizes cumulative frequency distribution.

**Example Explanation - Density Plot To understand body temperature distribution across individuals, a density plot will reveal peaks and potential outliers smoothly.**

**Example Code:**

sns.kdeplot(df['Body\_Temperature'], shade=True)

plt.title('Body Temperature Density Plot')

plt.xlabel('Temperature (°C)')

plt.ylabel('Density')

plt.show()

**5. Quantile Plots and QQ Plots**

**a) Quantile Plots**

* Visualizes data values versus their theoretical quantiles.
* Helps detect skewness or kurtosis.

**b) QQ (Quantile-Quantile) Plots**

* Compares sample distribution against a theoretical distribution (e.g., normal distribution).
* Useful for detecting normality in data.

**Example Explanation - QQ Plot A QQ plot can be used to compare housing prices against a normal distribution. Deviations from the line indicate non-normal distribution.**

**Example Code:**

import scipy.stats as stats

import matplotlib.pyplot as plt

stats.probplot(df['House\_Prices'], dist="norm", plot=plt)

plt.title('QQ Plot for House Prices')

plt.show()

**6. Overlaid Graphs and Summary Statistics**

**Overlaid Graphs:**

* Combine multiple plots to compare different data distributions.
* Examples include overlaying histograms, density plots, or line plots.

**Example Explanation - Overlaid Graphs To compare weight distributions among men and women, we can overlay two density plots for visual comparison.**

**Example Code:**

sns.kdeplot(df[df['Gender'] == 'Male']['Weight'], color='blue', shade=True)

sns.kdeplot(df[df['Gender'] == 'Female']['Weight'], color='red', shade=True)

plt.legend(['Male', 'Female'])

plt.title('Weight Distribution by Gender')

plt.show()

**Summary Statistics:**

Key metrics to summarize data include:

* Mean: Average value.
* Median: Middle value when data is sorted.
* Mode: Most frequently occurring value.
* Standard Deviation: Measures data spread around the mean.
* Interquartile Range (IQR): Measures data spread between Q1 and Q3.

**Example Code:**

print(df.describe())

**7. Best Practices for Visualization in EDA**

* Always label axes and provide meaningful titles.
* Use appropriate color schemes for better clarity.
* Ensure scalability for large datasets.
* Combine visualizations with summary statistics for deeper insights.
* Avoid overloading graphs with excessive information.
* Use legends effectively to differentiate data.