Sairam Konar D15C 27 2024-2025

Aim: Perform following data visualization and exploration on your selected dataset.

- Create bar graph, contingency table using any 2 features.
- Plot Scatter plot, box plot, Heatmap using seaborn.
- Create histogram and normalized Histogram.
- Describe what this graph and table indicates.
- Handle outlier using box plot and Inter quartile range.

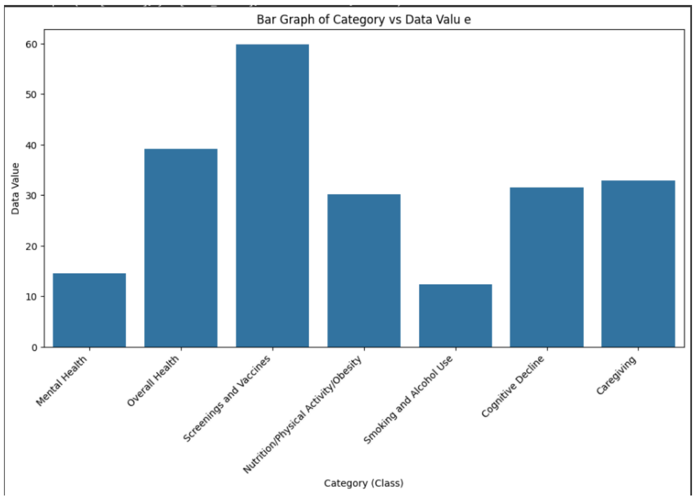
Steps:

1) **Create bar graph, contingency table using any 2 features.**

Seaborn and Matplotlib are used together to create an effective bar graph. Matplotlib sets the figure size, labels, and titles to ensure clarity in the visualization. Seaborn's barplot function is used to plot categorical data, where the x-axis represents different categories (Class), and the y-axis represents the corresponding data values. The estimator='mean' argument allows automatic aggregation, providing the average value for each category. This helps in identifying trends and comparisons across different classes efficiently. The rotation of x-axis labels enhances readability.

Code:

```
import matplotlib.pyplot as plt
import seaborn as sns
plt.figure(figsize=(12,6))
sns.barplot(x=df["Class"], y=df["Data_Value"], estimator='mean', ci=None)
plt.xticks(rotation=45, ha='right')
plt.xlabel("Category (Class)")
plt.ylabel("Data Value")
plt.title("Bar Graph of Category vs Data Valu e")
plt.show()
```

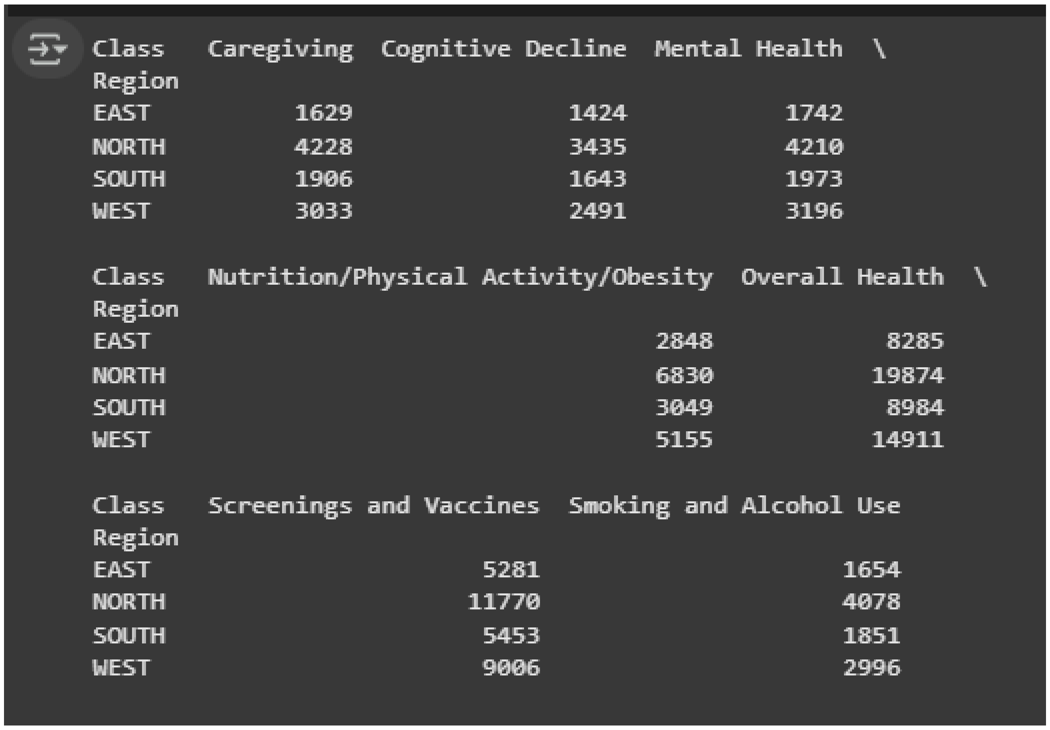


The bar graph compares the average Data Value across categories, highlighting dominant classes. Taller bars, such as Overall Health and Screenings & Vaccines, indicate higher averages, while categories like Mental Health & Smoking and Alcohol Use show more variability. It effectively reveals categorical trends and relative impact in the dataset.

The contingency table is visualized using Seaborn's heatmap function, which highlights patterns in data distribution with color gradients. The table represents the relationship between two categorical variables, Region and Class. The color intensity helps in quickly identifying regions with higher or lower values for different categories. Matplotlib is used to define the figure settings and ensure the visualization is clear. This approach allows for an intuitive understanding of data distribution, making it easier to identify trends and variations between different regions and classes.

Code:

```
import pandas as pd
contingency_table = pd.crosstab(df['Region'], df['Class'])
print(contingency_table)
```



```
Class   Caregiving  Cognitive Decline  Mental Health  \
Region
EAST          1629               1424           1742
NORTH         4228               3435           4210
SOUTH         1906               1643           1973
WEST          3033               2491           3196

Class   Nutrition/Physical Activity/Obesity  Overall Health  \
Region
EAST                                   2848            8285
NORTH                                  6830           19874
SOUTH                                  3049            8984
WEST                                   5155           14911

Class   Screenings and Vaccines  Smoking and Alcohol Use
Region
EAST                       5281                     1654
NORTH                     11770                     4078
SOUTH                      5453                     1851
WEST                       9006                     2996
```

The contingency table displays category frequencies across regions, helping identify regional variations. The heatmap enhances interpretation using color gradients, where darker shades indicate higher counts. This visualization simplifies pattern detection, revealing regional dominance in certain categories and making data comparison easier by highlighting imbalances and trends across different regions.

2) Plot Scatter plot, box plot, Heatmap using seaborn.

In this part, we have plotted Heatmap and seaborn as they are relevant to our dataset.

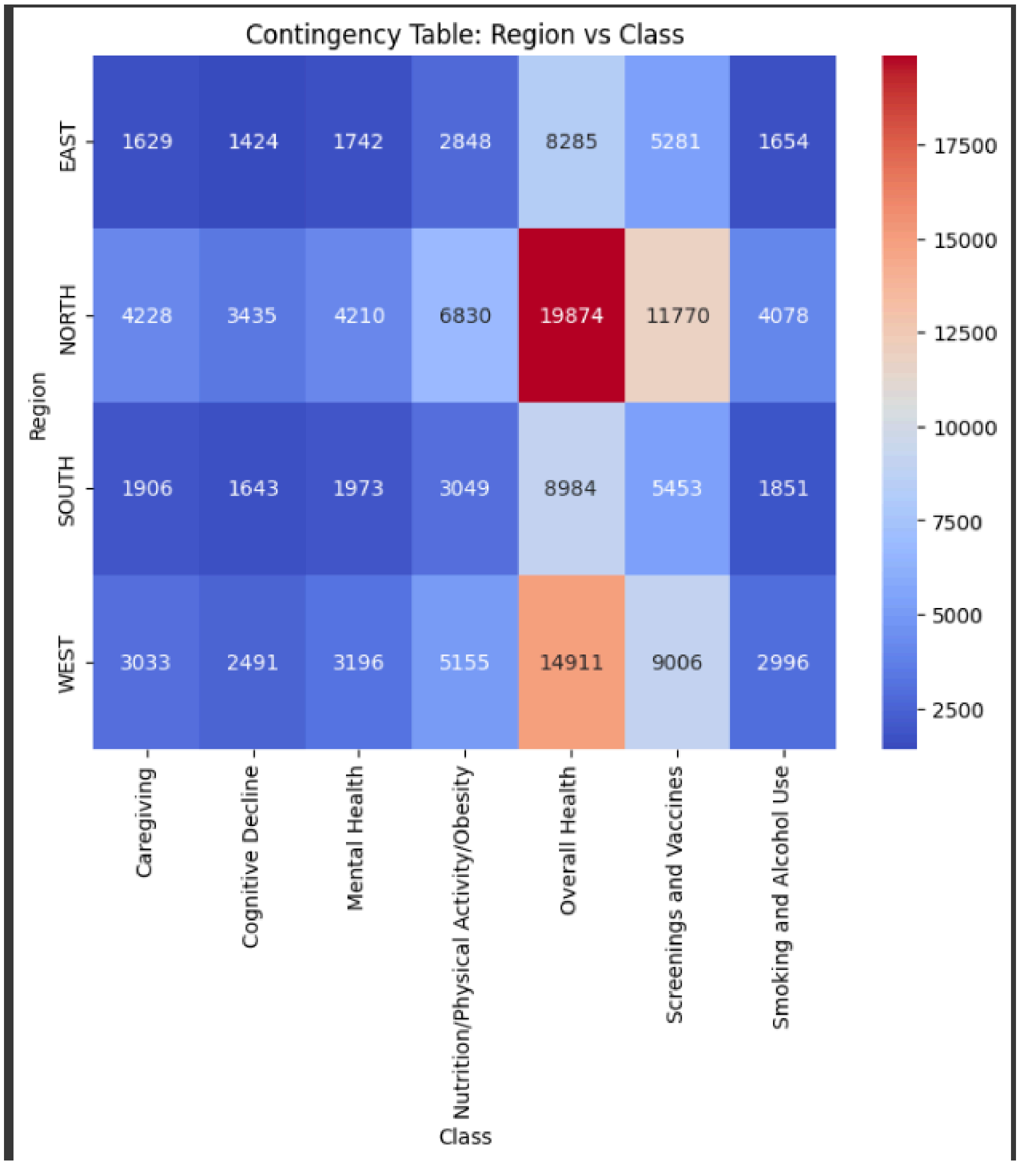
A heatmap is used to visualize the contingency table, which represents the relationship between two categorical variables, Region and Class. Seaborn’s heatmap function is applied to display data distribution using a color gradient (cmap="coolwarm"), where darker shades indicate higher values. The annot=True argument ensures that numerical values are displayed in each cell for clarity. Matplotlib is used to set figure size and labels, enhancing readability. This visualization helps in quickly identifying trends and correlations between categories.

Code:

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

plt.figure(figsize=(8,6))
sns.heatmap(contingency_table, annot=True, cmap="coolwarm", fmt='d')
plt.title("Contingency Table: Region vs Class")
plt.xlabel("Class")
plt.ylabel("Region")
plt.show()
```

Output:

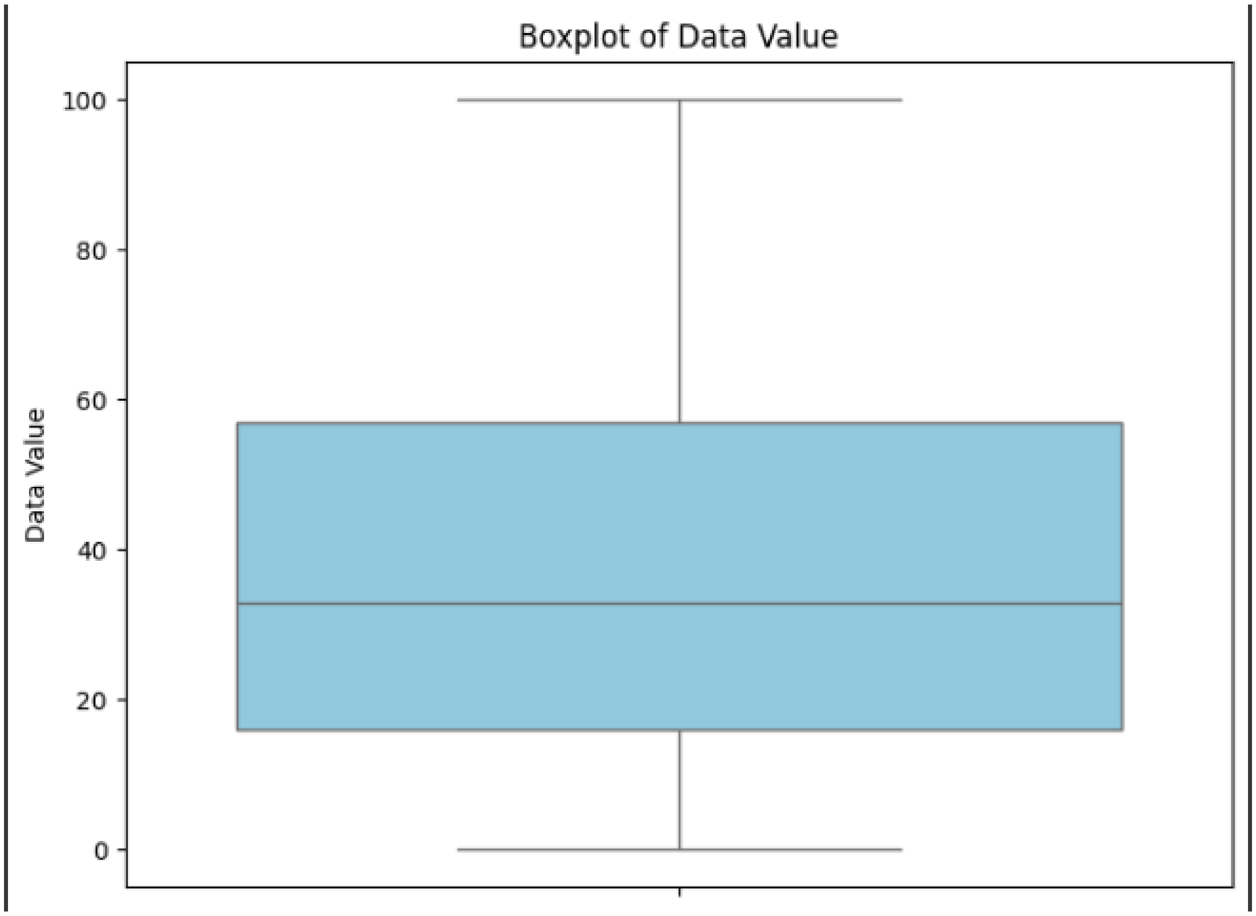


Region	Caregiving	Cognitive Decline	Mental Health	Nutrition/Physical Activity/Obesity	Overall Health	Screenings and Vaccines	Smoking and Alcohol Use
EAST	1629	1424	1742	2848	8285	5281	1654
NORTH	4228	3435	4210	6830	19874	11770	4078
SOUTH	1906	1643	1973	3049	8984	5453	1851
WEST	3033	2491	3196	5155	14911	9006	2996

A box plot is used to summarize the distribution of the Data_Value column, helping to identify outliers, the median, and the interquartile range. Seaborn's boxplot function plots the Data_Value on the y-axis with a skyblue color for better visibility. The box represents the middle 50% of the data, while the whiskers indicate the overall spread. Matplotlib ensures proper figure formatting with clear labels and titles. This visualization is useful for detecting anomalies and understanding data spread efficiently.

Code:
```
import matplotlib.pyplot as plt
import seaborn as sns
plt.figure(figsize=(8,6))
sns.boxplot(y=df["Data_Value"], color="skyblue")
plt.ylabel("Data Value")
plt.title("Boxplot of Data Value")
plt.show()
```

Output:



The boxplot visually summarizes the distribution of Data Value, showing its spread and central tendency. The box represents the interquartile range (middle 50% of data), with the median slightly below center, indicating right skewness. The wider IQR and longer upper whisker suggest high variability and possible extreme values affecting the distribution.

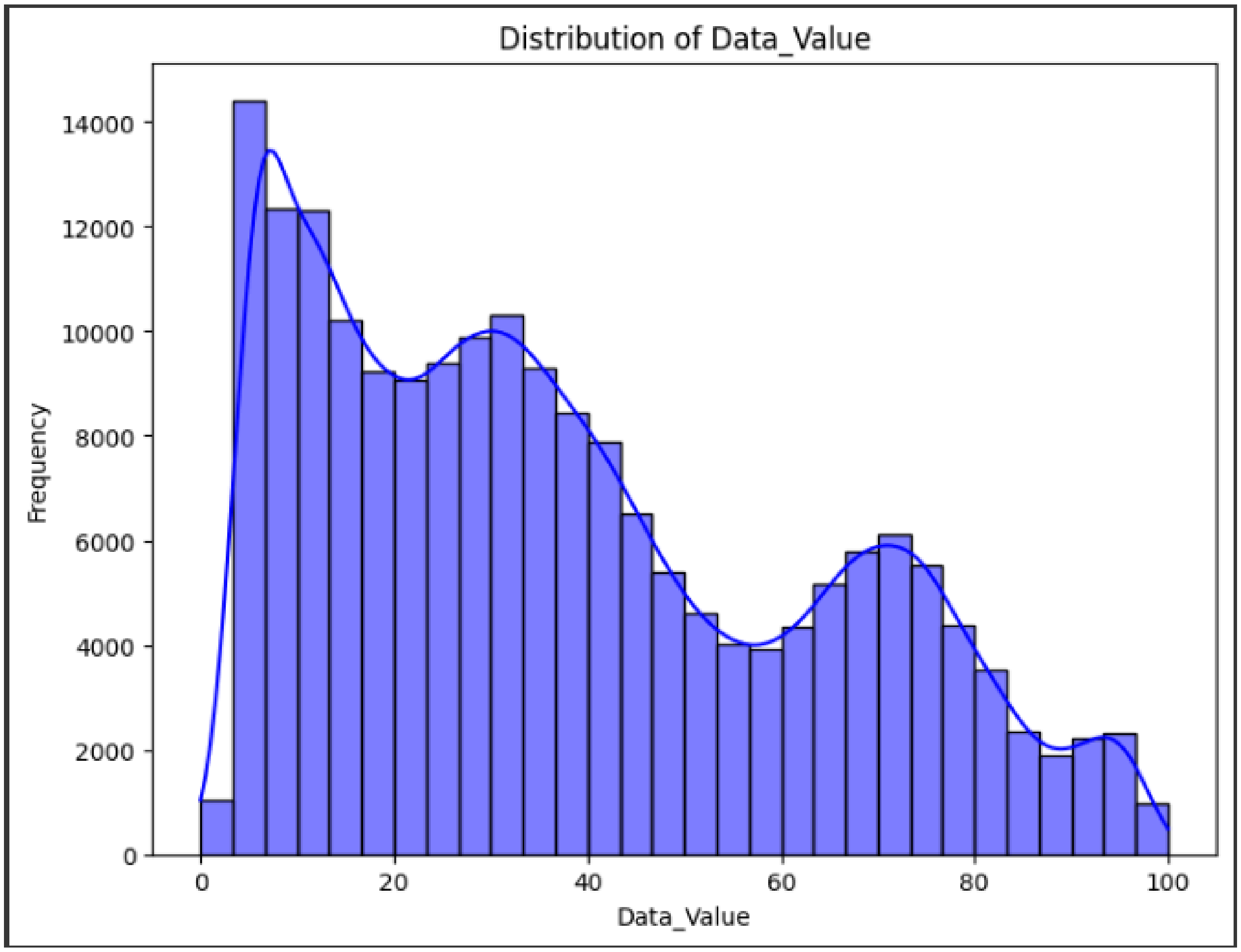
3) Create histogram and normalized Histogram

A histogram is used to visualize the distribution of Data_Value, showing how frequently different values occur. The sns.histplot function from Seaborn is used with bins=30 to divide data into 30 intervals, providing a detailed view of the distribution. The kde=True parameter adds a Kernel Density Estimate (KDE) curve for a smooth representation. The color is set to blue for better visibility. This helps in understanding data skewness, spread, and common value ranges.

Code:

```
import seaborn as sns
import matplotlib.pyplot as plt
plt.figure(figsize=(8,6))
sns.histplot(df["Data_Value"], bins=30, kde=True, color="blue")  # KDE for smooth curve
plt.title("Distribution of Data_Value")
plt.xlabel("Data_Value")
plt.ylabel("Frequency")
plt.show()
```

Output:



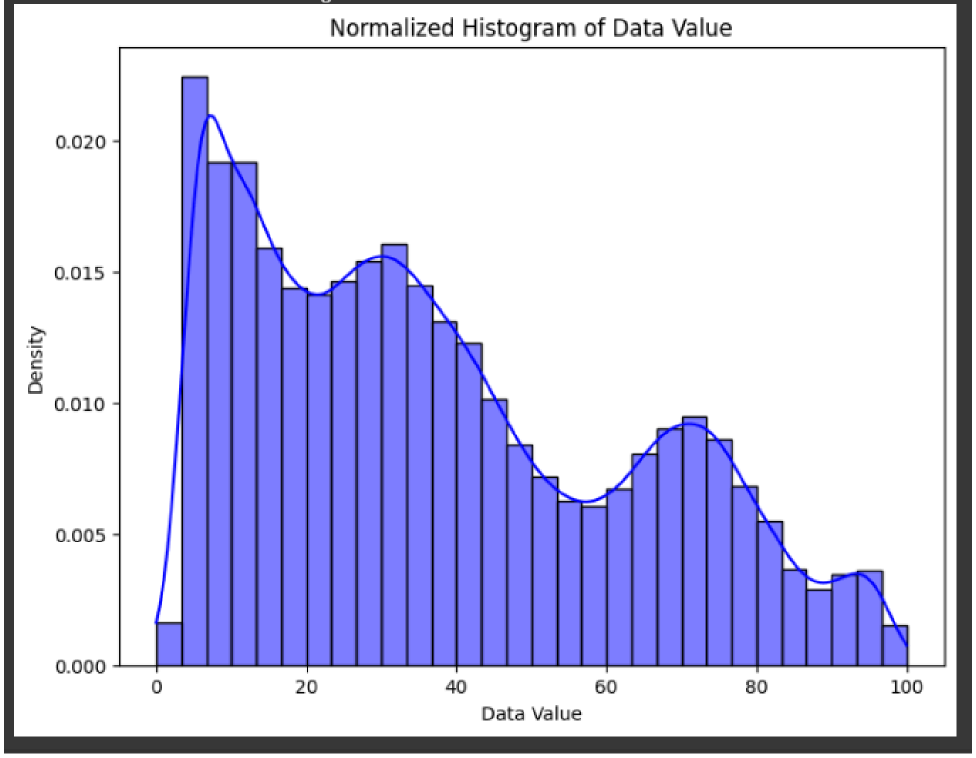
A normalized histogram represents data in terms of density instead of raw frequency, ensuring that the total area under the bars equals 1. The stat='density' parameter in sns.histplot normalizes the counts. The bin width is calculated, and the total area of the histogram is verified to confirm normalization. This type of histogram is useful when comparing distributions across different datasets or when absolute frequencies are not relevant.

Code:

```
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
plt.figure(figsize=(8,6))
hist_data = sns.histplot(df["Data_Value"], bins=30, kde=True, stat='density', color="blue")
heights = [patch.get_height() for patch in hist_data.patches]
bin_width = hist_data.patches[1].get_x() - hist_data.patches[0].get_x()
area = sum(heights) * bin_width
print("Total Area of Normalized Histogram:", area)
plt.xlabel("Data Value")
plt.ylabel("Density")
plt.title("Normalized Histogram of Data Value")
plt.show()
```

Output:

```
Total Area of Normalized Histogram: 1.0
```



The histogram shows the frequency of Data_Value occurrences, while the normalized histogram represents probability density. Multiple peaks suggest a multimodal distribution, indicating distinct groupings. The KDE curve highlights trends, and skewness hints at extreme values. The normalized histogram aids comparison by removing sample size effects, enhancing distribution analysis.

4) **Describe what this graph and table indicates.**

- **Bar graph:**

The bar graph compares the average Data Value across categories, highlighting dominant classes. Taller bars, such as Overall Health and Screenings & Vaccines, indicate higher averages, while categories like Mental Health & Smoking and Alcohol Use show more variability. It effectively reveals categorical trends and relative impact in the dataset.

- **Contingency table and Heatmap**

The contingency table displays category frequencies across regions, helping identify regional variations. The heatmap enhances interpretation using color gradients, where darker shades indicate higher counts. This visualization simplifies pattern detection, revealing regional dominance in certain categories and making data comparison easier by highlighting imbalances and trends across different regions.

- **Box Plot**

The boxplot visually summarizes the distribution of Data Value, showing its spread and central tendency. The box represents the interquartile range (middle 50% of data), with the median slightly below center, indicating right skewness. The wider IQR and longer upper whisker suggest high variability and possible extreme values affecting the distribution.

- **Histogram and Normalised Histogram**

The histogram shows the frequency of Data_Value occurrences, while the normalized histogram represents probability density. Multiple peaks suggest a multimodal distribution, indicating distinct groupings. The KDE curve highlights trends, and skewness hints at extreme values. The normalized histogram aids comparison by removing sample size effects, enhancing distribution analysis.

5) **Handle outlier using box plot and Inter quartile range.**

Outliers in a dataset can significantly impact statistical analysis, making it essential to identify and handle them effectively. A boxplot is a useful visualization tool for detecting outliers, as it highlights the spread of data, median, and interquartile range (IQR). The IQR method is a robust technique for outlier detection, where values falling outside **Q1 - 1.5 * IQR** and **Q3 + 1.5 * IQR** are considered potential outliers. By computing Q1 (25th percentile) and Q3 (75th percentile), we determine the IQR and set boundaries to identify extreme values. Outliers can distort measures of central tendency and variability, potentially leading to misleading insights. Once detected, these values can be analyzed to determine if they result from data entry errors, anomalies, or natural variability. Proper handling, such as transformation, capping, or removal, depends on the dataset's context. The combination of a boxplot and IQR analysis ensures a balanced approach to managing outliers.

Code:

Box-plot:

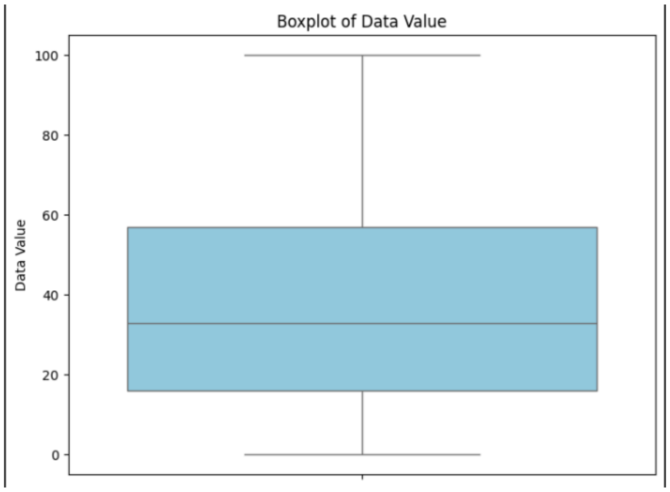
```
import matplotlib.pyplot as plt
import seaborn as sns
plt.figure(figsize=(8,6))
sns.boxplot(y=df["Data_Value"], color="skyblue")
plt.ylabel("Data Value")
plt.title("Boxplot of Data Value")
plt.show()
```

IQR:

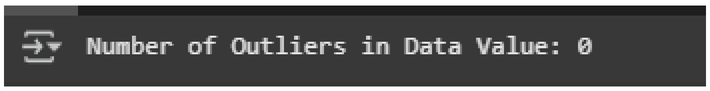
```
Q1 = df['Data_Value'].quantile(0.25)
Q3 = df['Data_Value'].quantile(0.75)
IQR = Q3 - Q1
lower_bound = Q1 - 1.5 * IQR
upper_bound = Q3 + 1.5 * IQR
outliers = df[(df['Data_Value'] < lower_bound) | (df['Data_Value'] > upper_bound)]
print("Number of Outliers in Data Value:", len(outliers))
```

Outputs:

Box Plot:



IQR:



```
Number of Outliers in Data Value: 0
```

Conclusion: This experiment provided valuable insights into the dataset through various visualizations. The bar graph highlighted variations in Data_Value across different categories, helping identify dominant and underrepresented classes. The contingency table and heatmap effectively showcased regional differences, revealing areas with higher concentrations in specific categories.The histogram and KDE curve indicated a multimodal distribution, suggesting the presence of distinct subgroups within the data. The box plot and IQR method identified several outliers, with extreme values that could potentially skew statistical analysis. Detecting and managing these outliers is crucial to maintaining data accuracy and ensuring reliable conclusions. Overall, this study reinforced the importance of exploratory data analysis (EDA) in understanding data distribution, detecting patterns, and handling anomalies. These visualizations provide a clearer perspective on data trends, aiding in better decision-making and more precise interpretations of the dataset.