**Data Warehousing and Data Mining.**

**Name : Antuley Aman Siraj.**

**Roll No. : 23CO25.**

**Class : TE-CO**

**Batch : 01**

**Experiment - 05**

**Aim:** Implementation of Data Discretization & Visualization.

**Theory:**

Data discretization is the process of converting continuous data into discrete buckets by grouping it. Discretization is also known for easy maintainability of the data. Training a model with discrete data becomes faster and more effective than when attempting the same with continuous data. Although continuous-valued data contains more information, huge amounts of data can slow the model down. Here, discretization can help us strike a balance between both. Some famous methods of data discretization are binning and using a histogram. Although data discretization is useful, we need to effectively pick the range of each bucket, which is a challenge.

The main challenge in discretization is to choose the number of intervals or bins and how to decide on their width. Here we make use of a function called pandas.cut(). This function is useful to achieve the bucketing and sorting of segmented data. In today’s world, a lot of data is being generated on a daily basis. And sometimes to analyze this data for certain trends, patterns may become difficult if the data is in its raw format. To overcome this data visualization comes into play. Data visualization provides a good, organized pictorial representation of the data which makes it easier to understand, observe, analyze. In this tutorial, we will discuss how to visualize data using Python.

Python provides various libraries that come with different features for visualizing data. All these libraries come with different features and can support various types of graphs. Here, we discuss four such libraries i.e Matplotlib, Seaborn, Bokeh, Plotly.

**Database Used:**

Tips Database Tips database is the record of the tip given by the customers in a restaurant for two and a half months in the early 1990s. It contains 5 columns such as week, day, no. of people, bill, tip.

**Program:**

**import pandas as pd**

**from IPython.display import display**

**# Reading the database**

**data = pd.read\_csv("/content/tips.csv")**

**print("\n")**

**# Printing the data**

**display(data.head(10))**

**print("\n")**

**Output :**

****

**Scatter Plot:**

Scatter plots are used to observe relationships between variables and uses dots to represent the relationship between them. The scatter() method in the matplotlib library is used to draw a scatter plot.

**Program :**

**import pandas as pd**

**import matplotlib.pyplot as plt**

**# Reading the database**

**data = pd.read\_csv("/content/tips.csv")**

**# Scatter Plot with total\_bill against tip**

**plt.scatter(data['total\_bill'], data['tip'])**

**# Adding the title**

**plt.title("Scatter Plot")**

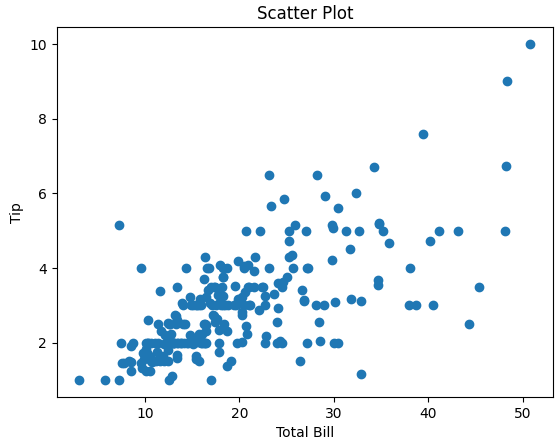
**# Setting the x and y labels**

**plt.xlabel('Total Bill')**

**plt.ylabel('Tip')**

**# Show plot**

**plt.show()**

****

**Bar Chart:**

A bar plot or bar chart is a graph that represents the category of data with rectangular bars with lengths and heights that is proportional to the values which they represent. It can be created using the bar() method.

**Program:**

**import pandas as pd**

**import matplotlib.pyplot as plt**

**# Read the CSV**

**data = pd.read\_csv("tips.csv")**

**# Check the columns in your dataset**

**print(data.columns)**

**# Use the correct column name for the bar plot**

**plt.bar(data['total\_bill'], data['tip'])**

**# Title and labels**

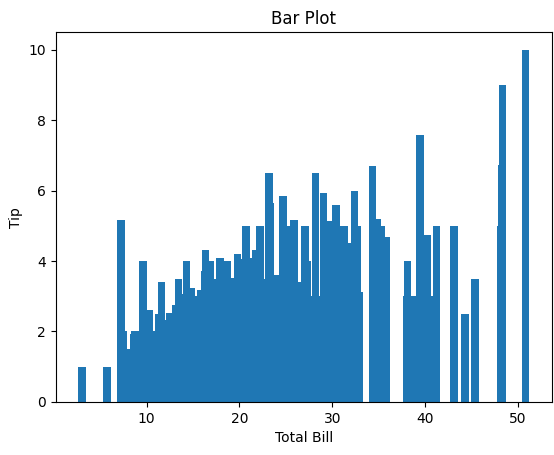
**plt.title("Bar Plot")**

**plt.xlabel('Total Bill')**

**plt.ylabel('Tip')**

**# Show plot**

**plt.show()**

****

**Conclusion:**

In this experiment, I explored organizing and visualizing data using Python. Data discretization, achieved through 'pandas.cut()', helps simplify analysis by grouping continuous data into distinct categories. I applied this to the 'Tips' restaurant database. For visualization, we used two methods: a scatter plot, showcasing relationships between variables like bill and tip for each day, and a bar chart, representing categorical data comparisons. These visualizations offer a clear view of patterns and trends in the data, facilitating better insights and decision-making. Python's libraries provide powerful tools for creating informative visual representations, enhancing data analysis.