Scheme of Valuation
Data Analysis
20IT604/PE2B
B.Tech.,(Semester- VI)

Date	:	May, 2025	Semester End Examination	:	Regular
Duration	:	3 hrs.	Maximum Marks	:	70

1. **(14X1 = 14 Marks)**

(a) **What is the significance of Data Analysis?** CO1 L2
Answer : Data analysis serves as the foundation for informed decision-making.

(b) **Compare Diagnostic data analysis with Descriptive data analysis** CO1 L3
Answer :

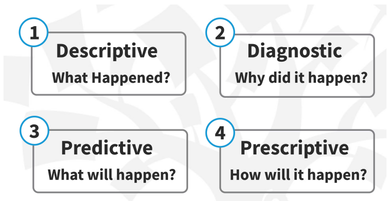

Figure 1: Types of Data Analysis

(c) **How do you access the quality of data?** CO1 L2
Answer : Data quality depends on the intended use of the data. There are many factors comprising data quality, including accuracy, completeness, consistency, timeliness, believability, and interpretability.

(d) **Name the data reduction methods** CO1 L1
Answer : Data reduction techniques can be applied to obtain a reduced representation of the data set that is much smaller in volume, yet closely maintains the integrity of the original data. Data reduction strategies include dimensionality reduction, numerosity reduction, and data compression.

(e) **How to sort the rows of a DataFrame by column values?** CO2 L2
Answer : Pandas provides the method sort_values() to order the DataFrame rows by column values. It has two key parameters: ‘by:’ the column or the list of columns to use for sorting. ‘ascending:’ controls the sort direction. It’s set to True by default (low to high).

(f) **How to split the DataFrame into groups based on certain criteria and then find mean of each group?** CO2 L2
Answer : The DataFrame method ‘groupby()’ can be used to split the DataFrame into groups based on certain criteria. We can then perform aggregate operations on each group.
```
genre_avg_revenue = movies_df.groupby('genre')[['revenue_millions']].mean()
```

(g) **What information does info() method of DataFrame give?** CO2 L2
Answer : Method ‘info()’ provides the essential details about your dataset, such as the number of rows and columns, the number of non-null values, what type of data is in each column, and how much memory your DataFrame is using.

(h) **Briefly describe a Line chart.** CO3 L2
Answer : A line chart is a type of graph used to show how values change over time or across categories. It connects individual data points with straight lines, making it easy to spot trends, patterns, or fluctuations.

(i) **Compare Bar chart with Pie chart.** CO3 L3
Answer :

Feature	**Bar Chart**	**Pie Chart**
Purpose	Compare values across categories	Show parts of a whole
Data Type	Counts and percentages	Percentages or proportions
Readability	Easy to compare exact values	Harder to compare similar-sized slices
Number of Categories	Handles many categories well	Best with 56 categories max
Orientation	Vertical or horizontal bars	Always circular
Precision	Highaligned to scale	Lowneeds labels/legend
Use Case Example	Comparing sales across regions	Showing market share
Visual Impact	Functional and information-dense	Visually appealing for simple data

(j) **Briefly describe a Choropleth chart.** CO3 L2

Answer : A choropleth chart (or map) is a thematic map where areas (like countries, states, or districts) are shaded or colored based on the value of a specific variable.

(k) **Draw a sample dashboard layout.** CO4 L2

Answer :

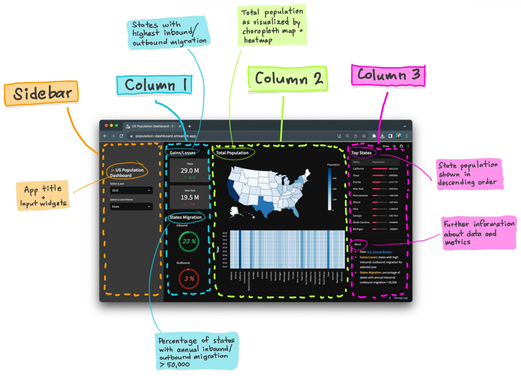
States with highest inbound/ outbound migration
Total population as visualized by choropleth map + heatmap
Sidebar
Column 1
Column 2
Column 3
App title + Input widgets
US Population Dashboard
Gains/Losses
Total Population
Top States
States Migration
State population shown in descending order
Further information about data and metrics
Percentage of states with annual inbound/ outbound migration > 50,000

Figure 2: Types of Data Analysis

(l) **How to embed a video in a Streamlit application?** CO4 L1

Answer : `st.video('video.mp4')`

(m) **How to embed a code snippet in a Streamlit application?** CO4 L1

Answer :

```
code = '''def hello_world(input):
                print(input)'''
st.code(code, language='python')
```

(n) **What is the use of Streamlit library in Python?** CO4 L1

Answer : Streamlit is a pure Python frontend development library that lets you create user friendly web applications called Streamlit applications quickly and easily.

UNIT - I

2. (a) **Describe Data Analysis process** (CO1, L2, 7 Marks)

Answer : Data Analysis process comprises of the following steps

i. **Problem Identification:** The first step in the data analysis process is Problem identification. What problem are you trying to solve? In other words, what research question do you want to address with your data analysis?

ii. **Data Collection:** The next step is to collect data. You can do this through various internal and external sources. For example, surveys, questionnaires, focus groups, interviews, etc. The key here is to collect and aggregate the appropriate statistical data.

iii. **Data Cleaning:** The next step is to prepare the data for analysis. That means you must clean or scrub it. This is essential since acquired data can be in different formats. Cleaning ensures you're not dealing with bad data and your results are dependable.

Here are some critical data-cleaning steps:

- Remove white spaces, duplicates, and formatting errors.
- Delete unwanted data points.
- Bring structure to your data.

iv. **Data Analysis:** All types of data analysis fit into the following four categories:

- **Descriptive Analysis:** Descriptive analysis focuses on what happened. It is the starting point for any research before proceeding with deeper explorations. As the first step, it involves breaking down data and summarizing its key characteristics.
- **Diagnostic Analysis:** This analysis focuses on why something has happened. Just as a doctor uses a patients diagnosis to uncover a disease, you can use diagnostic analysis to understand the underlying cause of the problem.
- **Predictive Analysis:** This type of analysis allows you to identify future trends based on historical data. It generally uses the results from the above analysis, machine learning (ML), and artificial intelligence (AI) to forecast future growth.
- **Prescriptive Analysis:** Now you know what to do, you must also understand how youll do it. The prescriptive analysis aims to determine your researchs best course of action.

v. **Data Interpretation:** You dig deeper into your data analysis findings and visualize the data to present insights in meaningful and understandable ways.

(b) **Describe different types of attributes of a data object.** (CO1, L2, 7 Marks)

Answer :

- **Nominal Attributes:**
  - Nominal means relating to names. The values of a nominal attribute are symbols or names of things.
  - Each value represents some kind of category, code, or state, and so nominal attributes are also referred to as categorical.
  - The values do not have any meaningful order.
  - In computer science, the values are also known as enumerations.
  - Mathematical operations on values of nominal attributes are not meaningful.
    hair color: possible values are black, brown, blond, red, auburn, gray, and white.
- **Binary Attributes:**
  - A binary attribute is a nominal attribute with only two categories or states: 0 or 1, where 0 typically means that the attribute is absent, and 1 means that it is present.
  - Binary attributes are referred to as Boolean if the two states correspond to true and false.
  - A binary attribute is symmetric if both of its states are equally valuable and carry the same weight; that is, there is no preference on which outcome should be coded as 0 or 1. For example gender having the states male and female.
  - A binary attribute is asymmetric if the outcomes of the states are not equally important, such as the positive and negative outcomes of a medical test for HIV. By convention, we code the most important outcome, which is usually the rarest one, by 1 (e.g., HIV positive) and the other by 0 (e.g., HIV negative).
- **Ordinal Attributes:**
  - An ordinal attribute is an attribute with possible values that have a meaningful order or ranking among them, but the magnitude between successive values is not known.

- Ordinal attributes are useful for registering subjective assessments of qualities that cannot be measured objectively; thus ordinal attributes are often used in surveys for ratings.
- The central tendency of an ordinal attribute can be represented by its mode and its median (the middle value in an ordered sequence), but the mean cannot be defined.

- **Interval-scaled attributes:**
  - Interval-scaled attributes are measured on a scale of equal-size units.
  - The values of interval-scaled attributes have order and can be positive, 0, or negative.
  - Thus, in addition to providing a ranking of values, such attributes allow us to compare and quantify the difference between values. Example: Temperature, Calendar dates
  - Although we can compute the difference between temperature values, we cannot talk of one temperature value as being a multiple of another. Without a true zero, we cannot say, for instance, that 10° C is twice as warm as 5° C. That is, we cannot speak of the values in terms of ratios.
- **Ratio-Scaled Attributes:**
  - A ratio-scaled attribute is a numeric attribute with an inherent zero-point. That is, if a measurement is ratio-scaled, we can speak of a value as being a multiple (or ratio) of another value.
  - In addition, the values are ordered, and we can also compute the difference between values, as well as the mean, median, and mode.
  - Examples: Kelvin (K), count attributes such as years of experience, number of words, attributes to measure weight, height, latitude and longitude and monetary quantities.

OR

3. (a) **Describe the measures of dispersion for given data.** (CO1, L2, 7 Marks)

Answer :

i. **Range:** Let $x_1, x_2, \cdots, x_N$ be a set of N values or observations, for some numeric attribute X. The range of this set of values is the difference between the largest (max()) and smallest (min()) values.

$$range\,(\mathbf{x}) = max(\mathbf{x}) - min(\mathbf{x})$$

ii. **Variance and Standard Deviation:**

- Variance and standard deviation are measures of data dispersion. They indicate how spread out a data distribution is.
- A low standard deviation means that the data observations tend to be very close to the mean, while a high standard deviation indicates that the data are spread out over a large range of values.
- The variance of N observations, $x_1, x_2, \cdots, x_N$, for a numeric attribute X is

$$\sigma^2 = \frac{1}{N}\sum_{i=1}^{N}(x_i - \bar{x})^2 = \left(\frac{1}{N}\sum_{i=1}^{N}x_i^2\right) - \bar{x}^2$$

- The standard deviation, σ, of the observations is the square root of the variance, σ^2.
- σ measures spread about the mean and should be considered only when the mean is chosen as the measure of center.
- $\sigma = 0$ only when there is no spread, that is, when all observations have the same value. Otherwise, $\sigma > 0$.

iii. **Quantiles**

- Quantiles are points taken at regular intervals of a data distribution, dividing it into essentially equalsize consecutive sets. (We say essentially because there may not be data values of X that divide the data into exactly equal-sized subsets. For readability, we will refer to them as equal.)
- The k^{th} q-quantile for a given data distribution is the value x such that at most k/q of the data values are less than x and at most $(q - k)/q$ of the data values are more than x, where k is an integer such that 0 < k < q. There are $q - 1$ q-quantiles.
- The 2-quantile is the data point dividing the lower and upper halves of the data distribution. It corresponds to the median.
- The 4-quantiles are the three data points that split the data distribution into four equal parts; each part represents one-fourth of the data distribution. They are more commonly referred to as 'quartiles'.

- The 100-quantiles are more commonly referred to as ‘percentiles‘; they divide the data distribution into 100 equal-sized consecutive sets.
- The median, quartiles, and percentiles are the most widely used forms of quantiles.

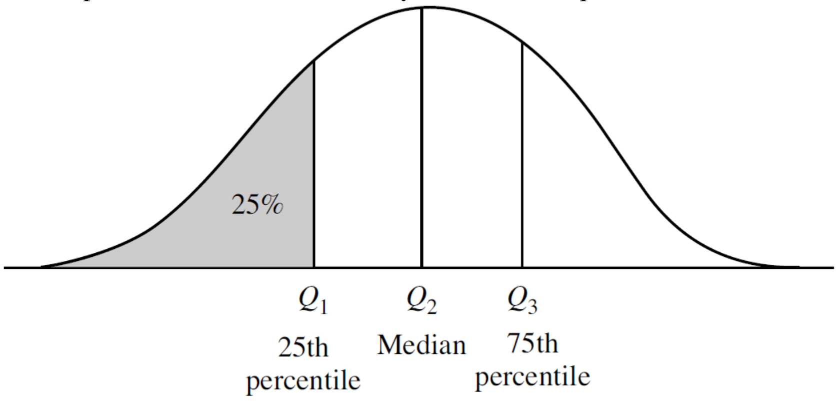

Figure 3: Quantiles, Quartiles and Percentiles

- **Interquartile range:**
  The distance between the first and third quartiles is a simple measure of spread that gives the range covered by the middle half of the data. This distance is called the interquartile range (IQR) and is defined as

$$IQR = Q_3 - Q_1$$

- **Five-number summary:**
  - The five-number summary of a distribution consists of the median (Q2), the quartiles Q1 and Q3, and the smallest and largest individual observations, written in the order of Minimum, Q1, Median, Q3, Maximum.
  - In the symmetric distribution, the median (and other measures of central tendency) splits the data into equal-size halves. This does not occur for skewed distributions. Therefore, it is more informative to also provide the two quartiles Q1 and Q3, along with the median.
  - Because Q1, the median, and Q3 together contain no information about the endpoints (e.g., tails) of the data, a fuller summary of the shape of a distribution can be obtained by providing the lowest and highest data values as well.
  - A common rule of thumb for identifying suspected outliers is to single out values falling at least $1.5 \times IQR$ above the third quartile or below the first quartile.

(b) **Describe Data Transformation process** (CO1, L2, 7 Marks)
Answer : Strategies for data transformation include the following:

- Smoothing, which works to remove noise from the data. Techniques include binning, regression, and clustering.
- Attribute construction (or feature construction), where new attributes are constructed and added from the given set of attributes to help the analysis process.
- Aggregation, where summary or aggregation operations are applied to the data. For example, the daily sales data may be aggregated so as to compute monthly and annual total amounts. This step is typically used in constructing a data cube for data analysis at multiple abstraction levels.
- Normalization, where the attribute data are scaled so as to fall within a smaller range, such as -1.0 to 1.0, or 0.0 to 1.0.
- Discretization, where the raw values of a numeric attribute (e.g., age) are replaced by interval labels (e.g., 010, 1120, etc.) or conceptual labels (e.g., youth, adult, senior). The labels, in turn, can be recursively organized into higher-level concepts, resulting in a concept hierarchy for the numeric attribute.
- Concept hierarchy generation for nominal data, where attributes such as street can be generalized to higher-level concepts, like city or country.

UNIT - II

4. (a) **i) Write python statements to create a ndarray whose shape is (4, 3), to find row sum and to find maximum element in a column ii) Write a python statement to extract every alternative element in the middle column of a ndarray whose shape is (5, 5)** (CO2, L3, 7 Marks)

Answer :

i

```python
import numpy as np
a = np.array([[0, 1, 2], [3, 4, 5],[2, 3, 2], [5, 2, 1]])
print(a.shape)
# (4, 3)
print(a.sum(axis=1))
# array([ 3, 12,  7,  8]) Row sum
print(a.max(axis=0))
# array([5, 4, 5])  Column maximum
```

ii

```python
import numpy as np
a = np.array([[0, 1, 2, 4, 2], [3, 4, 5, 1, 4],[2, 3, 2, 4, 6], [5, 2, 1, 6, 3], [3, 2, 4,
# alternative element in middle column
a[::2, 2]
```

(b) **i) Describe an ndarray. ii) Describe the axis property of a ndarray** (CO2, L2, 7 Marks)

Answer :

- An ndarray (short for N-dimensional array) is the core data structure of the NumPy module, designed for efficient numerical computation and data manipulation.
- It's a homogeneous array: all elements must be of the same data type.
- Supports multiple dimensions: from scalars (0D) to tensors (3D+) for complex datasets.
- Enables fast, memory-efficient operations compared to Python lists
- The easiest way to define a new ndarray is to use the array() function, passing a Python list or tuple containing the elements to be included in it as an argument.

```python
import numpy as np

# 1D array
arr1 = np.array([1, 2, 3])

# 2D array
arr2 = np.array([[1, 2], [3, 4]])

# 3D array
arr3 = np.array([[[1, 2], [3, 4]], [[5, 6], [7, 8]]])

```

- Functions like np.zeros(), np.ones(), np.empty(), or np.arange() to generate special arrays efficiently.

Table 2: Attributes of NumPy `ndarray`

Attribute	**Description**
`.shape`	Dimensions of the array as a tuple (e.g., `(3, 4)`)
`.ndim`	Number of array dimensions
`.size`	Total number of elements in the array
`.dtype`	Data type of array elements
`.itemsize`	Size (in bytes) of a single array element
`.nbytes`	Total memory consumed by the array (in bytes)

In NumPy, the axis property of an ndarray refers to the dimension along which operations are performed. Its a fundamental concept for manipulating multi-dimensional arrays efficiently.

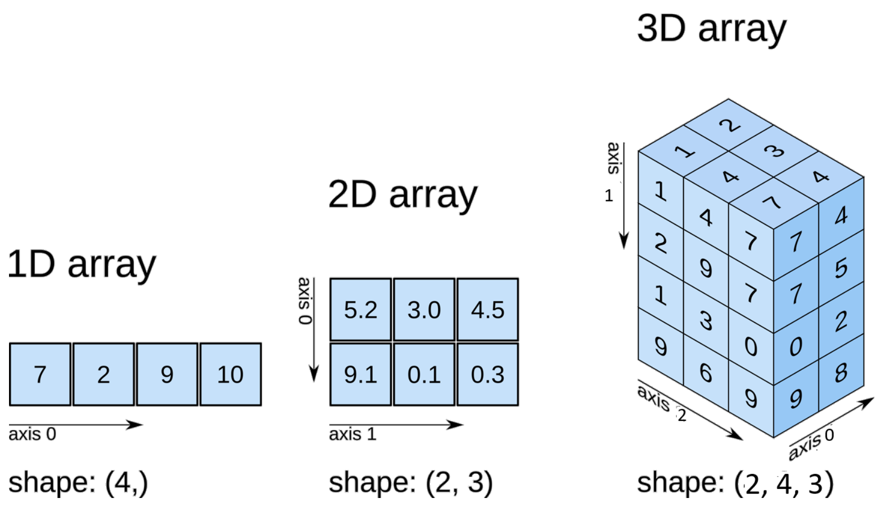

Figure 4: Axis property of an ndarray

OR

5. (a) **In a DataFrame, how to perform the following i) handle duplicate rows ii) handle missing values iii) rename some columns** (CO2, L2, 7 Marks)

Answer :

i. **Handle duplicate rows:**

To identify Duplicate Rows, use duplicated() to flag duplicates:

```python
import pandas as pd

# Sample DataFrame
df = pd.DataFrame({
    'Name': ['Alice', 'Bob', 'Alice', 'Charlie'],
    'Age': [25, 32, 25, 37]
})

# Identify duplicates
duplicates = df.duplicated()
print(duplicates)
```

Returns a Boolean Series: True for duplicates, False for unique rows.
You can specify columns using subset=['col1', 'col2'].
To remove duplicate Rows, use drop_duplicates() to remove them:

```python
# Remove duplicates, keeping the first occurrence
df_cleaned = df.drop_duplicates()

# Remove duplicates, keeping the last occurrence
df_cleaned_last = df.drop_duplicates(keep='last')

# Remove all duplicates (no occurrences kept)
df_no_dupes = df.drop_duplicates(keep=False)

# Use inplace=True to modify the original DataFrame directly.

# This checks for duplicates only in the specified columns.
df.drop_duplicates(subset=['Name', 'Age'], keep='first')
```

To extract just the duplicate rows:

```python
only_dupes = df[df.duplicated(keep=False)]
```

ii. **Handle missing values:**

Identify Missing Values Use isnull() or info() to detect missing values:

```python
df.isnull().sum()
df.info()
```

Drop Missing Values Remove rows or columns with missing data:

```
# Drop rows with any missing values
df.dropna()

# Drop columns with any missing values
df.dropna(axis=1)
```

You can also specify a threshold:

```
df.dropna(thresh=3)  # Keep rows with at least 3 non-NA values
```

Fill Missing Values Fill missing values with a specific value or strategy:

```
# Fill with a constant
df.fillna(0)

# Fill with mean, median, or mode
df['column_name'].fillna(df['column_name'].mean(), inplace=True)
df['column_name'].fillna(df['column_name'].median(), inplace=True)
df['column_name'].fillna(df['column_name'].mode()[0], inplace=True)
```

Forward or Backward Fill Use adjacent values to fill missing data:

```
df.fillna(method='ffill')  # Forward fill
df.fillna(method='bfill')  # Backward fill
```

Interpolation Estimate missing values using interpolation: df.interpolate()

iii. **Rename some columns:**
To rename columns in a Pandas DataFrame, you can use the .rename() method or directly assign to the .columns attribute. Here are both approaches:

Method 1: Using .rename() with a dictionary

```
import pandas as pd

# Sample DataFrame
df = pd.DataFrame({
    'A': [1, 2],
    'B': [3, 4]
})

# Rename columns
df = df.rename(columns={'A': 'Alpha', 'B': 'Beta'})
```

Method 2: Assigning to .columns. This method replaces all column names, so make sure the list matches the number of columns.

```
df.columns = ['Alpha', 'Beta']
```

(b) **In a DataFrame, how to perform the following i) summary of columns ii) extract rows of a DataFrame iii) extract rows of a DataFrame that satisfies a condition** (CO2, L2, 7 Marks)

Answer :

i. **Summary of columns of a DataFrame**

```
# To calculate basic statistical summary like count, mean, std, min, I quartile, II quar

df.describe()

# To calculate summary of all columns (Including Non-Numeric)
df.describe(include='all')

```

ii. **Extract rows of a DataFrame** To extract a set of rows from a Pandas DataFrame, you can use several methods

```

df[10:20]  # Rows from position 10 to 19

df.loc[['label1', 'label5', 'label10']]  # Rows with labels 'label1', 'label5', 'label10'.

df.loc['label1' : 'label2']  # Rows with index labels from label1 to label2 (inclusive)

df.iloc[10:20]  # Rows at positions 10 to 19

df.iloc[[0, 5, 10]]  # Rows at positions 0, 5 and 10
```

iii. **Extract rows of a DataFrame that satisfy a condition**

```
# by boolean condition

df[df['column_name'] > 100]  # Rows where column value is greater than 100


# Using Query Method

df.query('column_name > 100 and other_column == "value"')
```

UNIT - III

6. (a) **What information does a Scatter plot convey?** (CO3, L2, 7 Marks)

Answer : A scatter plot is like a window into the relationship between two numerical variables. Here's what it helps you see:

- **Correlation between variables:** It visually reveals whether the variables are positively correlated (as one increases, so does the other), negatively correlated (one increases while the other decreases), or have no clear relationship.
- **Trend patterns:** You can often spot linear or nonlinear trends. If the points roughly form a line, that suggests a linear trend.
- **Outliers:** Distant points from the general cluster may indicate outliersthose quirky data points that deviate significantly from the norm.
- **Clusters and groupings:** You might notice natural groupings of points, which can hint at subcategories or segments within your data.
- **Data distribution:** It shows how the data is spreaddense in certain regions, sparse in othersoffering insight into the variability.

(b) **How to draw subplots using Matplotlib library?** (CO3, L2, 7 Marks)

Answer :

```python
import matplotlib.pyplot as plt
import numpy as np

# Sample data
x = np.linspace(0, 10, 100)
y1 = np.sin(x)
y2 = np.cos(x)

# Create 1 row, 2 columns of subplots
fig, (ax1, ax2) = plt.subplots(1, 2, figsize=(10, 4))

# Plot on each subplot
ax1.plot(x, y1)
ax1.set_title("Sine Wave")

ax2.plot(x, y2)
ax2.set_title("Cosine Wave")

# Adjust layout
plt.tight_layout()
plt.show()
```

OR

7. (a) **What information does a Box plot convey?** (CO3, L2, 7 Marks)

Answer :

A box plot, also known as a box-and-whisker plot, is a compact yet powerful way to visualize the distribution, central tendency, and spread of data. It's especially handy for spotting outliers and comparing distributions across different groups.

i. **Median (Q2):** The central line inside the box shows the median (50th percentile) of the data.
ii. **Interquartile Range (IQR):** The box spans from the first quartile (Q1 25th percentile) to the third quartile (Q3 75th percentile), covering the middle 50
iii. **Whiskers:** These lines extend from the box to the smallest and largest values within 1.5ŒIQR from Q1 and Q3, respectively. They show the "normal" spread of the data.
iv. **Outliers:** Any data points outside the whiskers are plotted as individual dots. These are values significantly higher or lower than the rest of the data.
v. **Skewness:** If the median line is closer to the top or bottom of the box, or if one whisker is noticeably longer, that suggests skewness in the data.

(b) **What information does a Histogram convey?** (CO3, L2, 7 Marks)

Answer :

A histogram shows how the data is distributed across different value ranges. The information that it conveys is

i. **Frequency distribution:** It tells you how many data points fall within specific intervals or bins. Taller bars mean more values in that range.

ii. **Shape of the distribution:** You can quickly tell if the data is:
   - Normal (bell-shaped)
   - Skewed (to the left or right)
   - Uniform (roughly equal across ranges)
   - Bimodal (two peaks)

iii. **Central tendency & spread:** While it doesn't mark the mean or median directly, you can visually estimate where most data clusters.

iv. **Outliers or gaps:** Sparse bins or isolated bars can hint at outliers or data groupings that stand apart.

UNIT - IV

8. (a) **In a Streamlit application, how to configure the following i) page layout ii) columns iii) tabs** (CO4, L2, 7 Marks)
Answer :

To configure the page layout in a Streamlit application, you use the st.set_page_config() function and it must be the very first Streamlit command in your script. This function lets you control how your app appears in the browser.

```python
import streamlit as st

st.set_page_config(
    page_title = "My Streamlit App", # Sets the title shown in the browser tab
    page_icon = ":bar_chart:", # Sets the favicon (can be emoji, image URL, or Material ico
    layout = "wide",  # or "centered" (default), "wide" to use full screen width
    initial_sidebar_state = "expanded"  # Controls sidebar visibility: "auto", "expanded",
)
```

In Streamlit, you can configure columns using st.columns()

```python
import streamlit as st

# Create 3 equal-width columns
col1, col2, col3 = st.columns(3)

with col1:
    st.header("Sales")
    st.metric("Total", "\$15K")

with col2:
    st.header("Growth")
    st.metric("Monthly", "12\%")

with col3:
    st.header("Orders")
    st.metric("Count", "320")
```

Tabbed layout in Streamlit app can be configured using the st.tabs() function, to organize dashboard content into separate views.

```python
import streamlit as st

# Create tabs
tab1, tab2, tab3 = st.tabs(["Chart View", "Data Table", "Report"])

# Populate each tab with content
with tab1:
    st.write("Here you can place your chart, e.g. Matplotlib or Plotly")
    st.line_chart({"data": [1, 5, 2, 6]})

with tab2:
    st.write("Raw data table or DataFrame goes here")
    st.dataframe({"A": [1, 2], "B": [3, 4]})

with tab3:
    st.write("This could be a summary, report or formatted LaTeX output")
    st.latex(r"\alpha^2 + \beta^2 = \gamma^2")
```

(b) **In a Streamlit application, describe the following i) Expander ii) Sidebar iii) Metric** (CO4, L2, 7 Marks)
Answer : In Streamlit, an Expander is like a collapsible container that lets you neatly organize optional or detailed content without overwhelming the main layout. Keeps the UI clean while still providing access to deeper content. Useful for sidebar filters, help documentation, code snippets, etc. Can be nested inside tabs, columns, or even other expanders for layered layouts.

```python
import streamlit as st

with st.expander("Click to see details"):
    st.write("Here are some extra insights or hidden content.")
    st.dataframe({"Details": ["Insight 1", "Insight 2"]})
```

In a Streamlit application, the Sidebar helps to separate inputs from outputs and keep the layout clean and professionalespecially helpful when you're working on interactive dashboards or data visualizations. Streamlit provides the st.sidebar namespace, which lets you place widgets and content in the sidebar rather than the main page. Almost all Streamlit widgets can be used inside the sidebarbuttons, checkboxes, sliders, radios, text inputs, etc.

```python
import streamlit as st

with st.sidebar:
    # Sidebar content
    st.title("Navigation Panel")
    st.radio("Choose View", ["Home", "Charts", "Analysis"])
    st.slider("Set Threshold", 0, 100, 50)
```

The st.metric(label, value, delta = None, delta_color = 'normal') command displays in large and bold fonts a metric, indicator, or variable in a dashboard style. An optional change of status indicator can be added.

```python
import streamlit as st

st.metric(label="Revenue", value="$25,000", delta="+5%")

# label: A short description of the metric (e.g., "Revenue", "Users")
# value: The current value (e.g., "$25,000", "98%", "1200 users")
# delta: Optional; shows change compared to previous value
# delta_color: Set to 'normal', 'inverse', or 'off' to control indicator color
```

OR

9. **Given marks of students sorted in the ascending order of their roll numbers, write code to develop a dashboard that contains information obtained from the analysis of the students marks. Assume marks in 4 subjects are stored in marks.xlsx file. The first column has roll numbers and the rest of the columns in a row contain marks in different subjects scored by a student out of a maximum mark of 100.** (CO4, L3, 14 Marks)

Answer :

```python
import streamlit as st
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns


# Load the Excel file
df = pd.read_excel("marks.xlsx", engine='openpyxl')

# Set the title of the dashboard
st.title("Student Marks Analysis Dashboard")

# Display the raw data
st.subheader("Raw Data")
st.dataframe(df)

# Summary statistics
st.subheader("Summary Statistics")
st.write(df.describe())

# Subject-wise average marks
st.subheader("Subject-wise Average Marks")
```

```python
average_marks = df.iloc[:, 1:].mean()
st.bar_chart(average_marks)

# Highest and lowest scores in each subject
st.subheader("Highest and Lowest Scores")
highest_scores = df.iloc[:, 1:].max()
lowest_scores = df.iloc[:, 1:].min()
scores_df = pd.DataFrame({'Highest': highest_scores, 'Lowest': lowest_scores})
st.dataframe(scores_df)

# Visualizations: Bar charts and histograms for each subject
st.subheader("Visualizations")

for subject in df.columns[1:]:
    st.write(f"### {subject}")

    # Bar chart of marks
    st.write("Bar Chart")
    fig, ax = plt.subplots()
    ax.bar(df['rollno'], df[subject])
    ax.set_xlabel("Roll Number")
    ax.set_ylabel("Marks")
    ax.set_title(f"Marks Distribution in {subject}")
    plt.xticks(rotation=45)
    st.pyplot(fig)

    # Histogram of marks
    st.write("Histogram")
    fig, ax = plt.subplots()
    ax.hist(df[subject], bins=10, edgecolor='black')
    ax.set_xlabel("Marks")
    ax.set_ylabel("Frequency")
    ax.set_title(f"Histogram of Marks in {subject}")
    st.pyplot(fig)


# Set the style for the plot
sns.set(style="whitegrid")

# Create a box plot for the four subjects
plt.figure(figsize=(10, 6))
sns.boxplot(data=df[['sub1', 'sub2', 'sub3', 'sub4']])
plt.title("Comparison of Student Performance Across Subjects")
plt.xlabel("Subjects")
plt.ylabel("Marks")
plt.tight_layout()

# Save the plot to a file
plt.savefig("boxplot_subject_comparison.png")

# Show the plot
plt.show()
```

Scheme prepared by

(N.Sivaram Prasad, IT Dept.)

Signature of HOD

Signatures of evaluators

SNo	Name of the evaluator	College Name	Signature