Assignment No 3

Aim

To perform data visualization on a selected dataset using R or Python by applying various plotting techniques such as scatter plots, bar plots, box plots, pie charts, and line charts. The goal is to improve the ability to analyze and interpret data effectively using graphical methods.

Objective

The primary objective of this assignment is to:

- Understand the significance and utility of data visualization in data analysis.
- Explore different types of plots to represent data features and relationships visually.
- Learn how to apply these plots using R or Python programming languages.
- Develop analytical thinking by observing trends, patterns, and anomalies through visual insights.

By completing this assignment, learners will gain hands-on experience in interpreting visual data outputs, which is critical in both academic research and real-world data-driven decision-making.

Theoretical

Data visualization is the process of converting raw data into visual formats such as charts and graphs. These formats help simplify complex datasets and reveal hidden insights that are not immediately apparent in raw numerical data. Visualization serves several purposes:

- Understanding data: Plots help to detect distribution, correlation, or outliers.
- Communication: Visuals are more intuitive for conveying findings to others, especially non-technical audiences.
- Exploration: During exploratory data analysis (EDA), visualizations help to form hypotheses about the data.

Key Characteristics of Good Visualizations:

- Accurate representation
- Easy to understand
- Relevant to the context
- Aesthetically clear and uncluttered

Methods and Explanations of Graphical Techniques

Scatter Plot

Description: Plots individual data points on a two-dimensional graph based on two variables. The X and Y axes represent each variable.

Purpose: To detect correlations, clusters, or trends.

Advantages: Clearly shows the relationship (positive, negative, or no correlation).

Applications:

- In machine learning to understand the relationship between features.

- Identifying linear or non-linear trends.

- Detecting outliers in regression models.

Bar Plot

Description: Uses rectangular bars to represent quantities or frequencies of different categories.

Purpose: To compare values across discrete categories.

Advantages: Easy to interpret; good for categorical comparisons.

Applications:

- Comparing product sales, student scores, or gender distribution.

- Showing the frequency of categorical variables (like regions, cities, product types).

Box Plot (Box-and-Whisker Plot)

Description: Visualizes the distribution of numerical data and highlights the median, quartiles, and potential outliers.

Purpose: To detect skewness, spread, and outliers.

Advantages: Summarizes a dataset in five key numbers and easily shows variance and outliers.

Applications:

- In finance, to compare stock returns.

- In quality control to monitor manufacturing process variation.

- Comparing test results of students across different classes.

Pie Chart

Description: Circular chart divided into slices to illustrate numerical proportions.

Purpose: To represent part-to-whole relationships.

Advantages: Visually impactful for showing percentages or proportions.

Applications:

- Showing market share of companies.

- Budget or expenditure distribution.

- Population breakdown by categories (e.g., age groups or regions).

Note: While pie charts are popular, they are less effective than bar charts for comparing many categories.

Line Chart

Description: A series of data points connected by a straight line, usually representing time-based changes.

Purpose: To track changes or trends over intervals.

Advantages: Best suited for continuous data and trend analysis.

Applications:

- Time series analysis in stock market trends.

- Weather data analysis (temperature, rainfall).

- Website traffic trends over months or years.

Applications of Data Visualization

- Business Intelligence: Visual dashboards used for tracking KPIs and sales performance.
- Healthcare: Charts help monitor patient progress and medication effectiveness.
- Finance: Investors use line and candlestick charts to assess trends.
- Education: Institutions use visualizations to analyze student performance.
- Government & Policy: Pie and bar charts are used in reports to show statistical distributions.

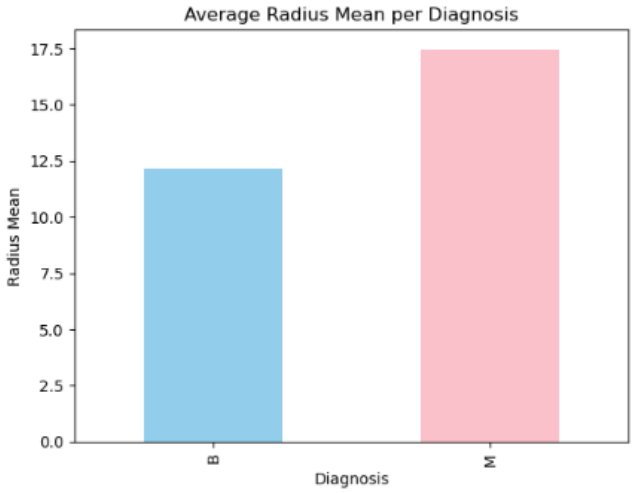
Outputs:

In [1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

In [2]:
```
dataset_path = "C:\\Users\\saksh\\Desktop\\ml_jupyter_lab\\dataset\\breast-cancer - breast-cancer.csv"
df = pd.read_csv(dataset_path)
print("Head of the df: ")
print(df.head())
```

```
Head of the df:
         id diagnosis  radius_mean  texture_mean  perimeter_mean  area_mean  \
0    842302         M        17.99         10.38          122.80     1001.0
1    842517         M        20.57         17.77          132.90     1326.0
2  84300903         M        19.69         21.25          130.00     1203.0
3  84348301         M        11.42         20.38           77.58      386.1
4  84358402         M        20.29         14.34          135.10     1297.0

   smoothness_mean  compactness_mean  concavity_mean  concave points_mean  \
0          0.11840           0.27760          0.3001              0.14710
1          0.08474           0.07864          0.0869              0.07017
2          0.10960           0.15990          0.1974              0.12790
3          0.14250           0.28390          0.2414              0.10520
4          0.10030           0.13280          0.1980              0.10430

   ...  radius_worst  texture_worst  perimeter_worst  area_worst  \
0  ...         25.38          17.33           184.60      2019.0
1  ...         24.99          23.41           158.80      1956.0
2  ...         23.57          25.53           152.50      1709.0
3  ...         14.91          26.50            98.87       567.7
4  ...         22.54          16.67           152.20      1575.0

   smoothness_worst  compactness_worst  concavity_worst  concave points_worst  \
0            0.1622             0.6656           0.7119                0.2654
1            0.1238             0.1866           0.2416                0.1860
2            0.1444             0.4245           0.4504                0.2430
3            0.2098             0.8663           0.6869                0.2575
4            0.1374             0.2050           0.4000                0.1625

   symmetry_worst  fractal_dimension_worst
0          0.4601                  0.11890
1          0.2750                  0.08902
2          0.3613                  0.08758
3          0.6638                  0.17300
4          0.2364                  0.07678

[5 rows x 32 columns]
```

In [12]:
```
# Bar Plot: Average radius mean for each diagnosis
df.groupby("diagnosis")['radius_mean'].mean().plot(kind='bar', color=['skyblue', 'pink'])
plt.title("Average Radius Mean per Diagnosis")
plt.xlabel("Diagnosis")
plt.ylabel("Radius Mean")
plt.show()
```

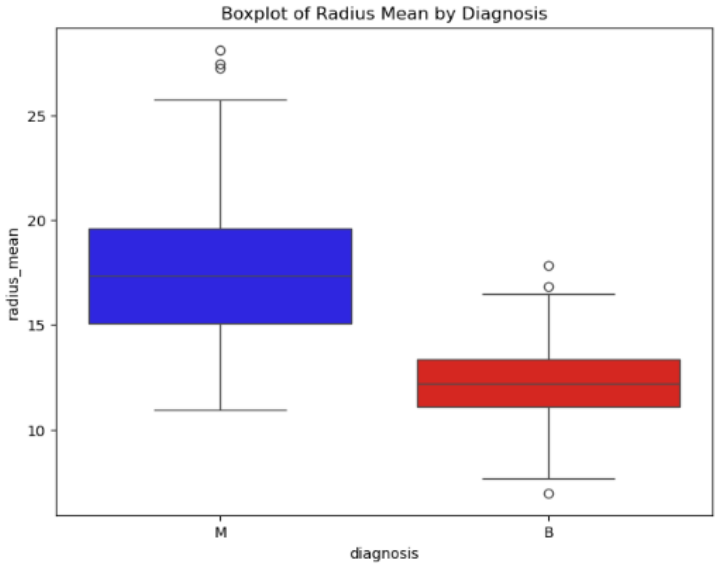


In [14]:
```
# Box Plot: Distribution of radius_mean for each diagnosis
plt.figure(figsize=(8, 6))
sns.boxplot(x="diagnosis", y="radius_mean", data=df, palette=["blue", "red"])
plt.title("Boxplot of Radius Mean by Diagnosis")
plt.show()
```

In [14]:

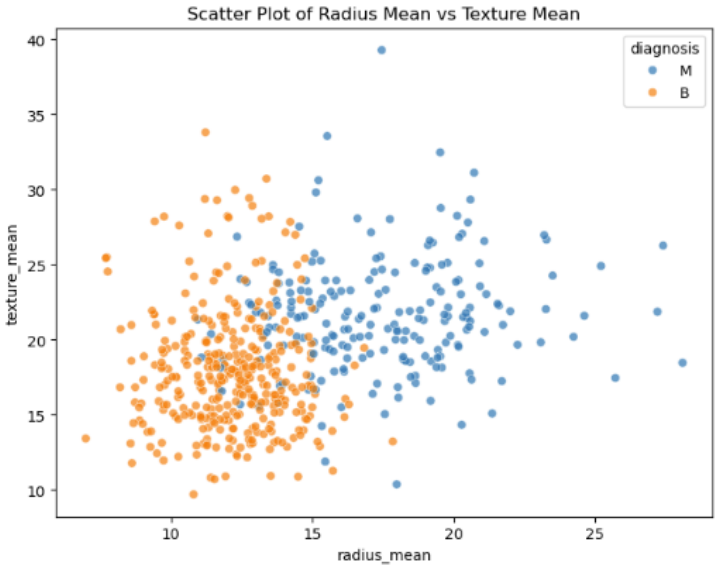
```
# Box Plot: Distribution of radius_mean for each diagnosis
plt.figure(figsize=(8, 6))
sns.boxplot(x="diagnosis", y="radius_mean", data=df, palette=["blue", "red"])
plt.title("Boxplot of Radius Mean by Diagnosis")
plt.show()
```

```
C:\Users\saksh\AppData\Local\Temp\ipykernel_16336\216248278.py:3: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the same effect.

  sns.boxplot(x="diagnosis", y="radius_mean", data=df, palette=["blue", "red"])
```



In [16]:

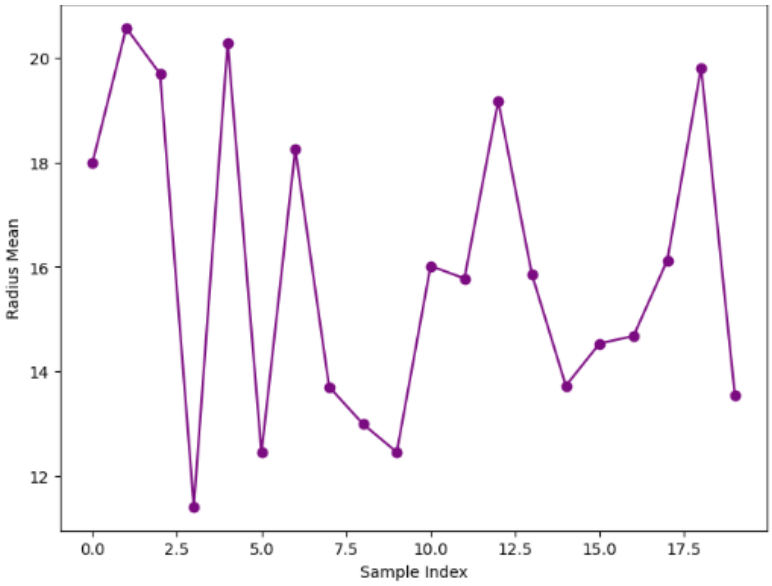
```
# Scatter Plot: radius_mean vs texture_mean, colored by diagnosis
plt.figure(figsize=(8, 6))
sns.scatterplot(x='radius_mean', y='texture_mean', hue='diagnosis', data=df, alpha=0.7)
plt.title("Scatter Plot of Radius Mean vs Texture Mean")
plt.show()
```



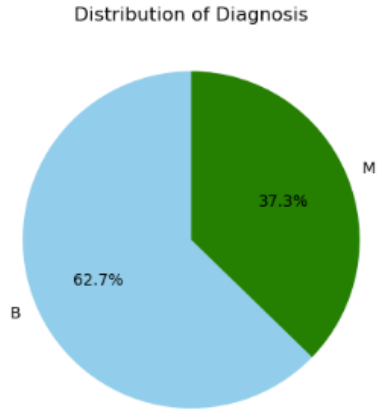
In [18]:

```
# Line Chart: Variation of radius_mean for first 20 samples
plt.figure(figsize=(8, 6))
df['radius_mean'][:20].plot(kind='line', marker='o', linestyle='-', color='purple')
plt.title("Line Chart of Radius Mean (First 20 Samples)")
plt.xlabel("Sample Index")
plt.ylabel("Radius Mean")
plt.show()
```

Line Chart of Radius Mean (First 20 Samples)



In [22]:
```
# Pie Chart: Distribution of diagnosis values
df['diagnosis'].value_counts().plot(kind='pie', autopct='%1.1f%%', colors=['skyblue', 'green'], startangle=90)
plt.title("Distribution of Diagnosis")
plt.ylabel("")
plt.show()
```



Conclusion

We have successfully visualized data using a variety of graphical techniques in R/Python, including scatter plots, bar plots, box plots, pie charts, and line charts. These visualizations helped us gain deeper insight into the structure, patterns, relationships, and distributions present in the dataset.