Relationship Between Two Variables in Bivariate Analysis

Bivariate Analysis focuses on the relationship between two variables. It is used to identify patterns, associations, or dependencies between them. The relationship can be explored using **visualizations** or **statistical measures**.

Types of Relationships

1. **Linear Relationship**: The two variables are linearly dependent (e.g., height and weight).
2. **Non-Linear Relationship**: The relationship between variables does not follow a straight line.
3. **No Relationship**: Variables are independent and show no correlation.

Common Tools for Bivariate Analysis

1. **Scatter Plot**:
   - Visualizes the relationship between two variables.
   - Helps identify the direction, form, and strength of the relationship.
2. **Correlation Coefficient**:
   - Measures the strength and direction of a linear relationship.
   - Values range between -1 (perfect negative) and +1 (perfect positive).
   - Calculated using numpy or pandas.
3. **Regression Line**:
   - A straight line that best fits the data.
   - Indicates how one variable can predict another.

Example: Scatter Plot and Correlation

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from scipy.stats import pearsonr

# Sample dataset
data = {
   'Study_Hours': [1, 2, 3, 4, 5, 6, 7, 8, 9, 10],
   'Scores': [35, 50, 55, 65, 70, 75, 78, 85, 88, 95]
}

# Create a DataFrame
df = pd.DataFrame(data)

# Calculate Correlation Coefficient
correlation, p_value = pearsonr(df['Study_Hours'], df['Scores'])
```

```
# Create Scatter Plot
plt.figure(figsize=(8, 6))
plt.scatter(df['Study_Hours'], df['Scores'], color='blue', label='Data Points')
plt.title(f"Scatter Plot of Study Hours vs. Scores\nCorrelation: {correlation:.2f}")
plt.xlabel('Study Hours')
plt.ylabel('Scores')
plt.axhline(y=df['Scores'].mean(), color='red', linestyle='--', label='Mean Line (Scores)')
plt.axvline(x=df['Study_Hours'].mean(), color='green', linestyle='--', label='Mean Line (Hours)')
plt.legend()
plt.grid()
plt.show()
```

Output

1. **Correlation**:
   - Example output: Correlation: 0.98, indicating a strong positive relationship.
2. **Scatter Plot**:
   - Data points are clustered along an upward trend line.
   - Visual confirmation of the positive correlation.

1. **Dataset**:
   - The dataset includes two variables: Study_Hours (independent) and Scores (dependent).
2. **Correlation Coefficient**:
   - The **Pearson correlation coefficient** is calculated using pearsonr.
   - Value close to 1 indicates a strong positive linear relationship.
3. **Scatter Plot**:
   - Visualizes the distribution of points.
   - Shows a clear increasing trend, indicating a positive relationship.
4. **Mean Lines**:
   - Horizontal and vertical lines represent the mean of Scores and Study_Hours for reference.

Example: Adding a Regression Line

```
from sklearn.linear_model import LinearRegression
import numpy as np

# Reshape data for Linear Regression
X = df['Study_Hours'].values.reshape(-1, 1)
y = df['Scores'].values

# Fit the model
model = LinearRegression()
model.fit(X, y)
```

```
# Predict values
y_pred = model.predict(X)

# Plot Scatter and Regression Line
plt.figure(figsize=(8, 6))
plt.scatter(df['Study_Hours'], df['Scores'], color='blue', label='Data Points')
plt.plot(df['Study_Hours'], y_pred, color='orange', label='Regression Line', linewidth=2)
plt.title("Scatter Plot with Regression Line")
plt.xlabel("Study Hours")
plt.ylabel("Scores")
plt.legend()
plt.grid()
plt.show()
```

1. **Linear Regression**:
   - A regression model is fitted using LinearRegression from sklearn.
   - The independent variable (Study_Hours) is reshaped to match the input format for the model.
2. **Regression Line**:
   - The model predicts Scores based on Study_Hours.
   - A straight line representing the best fit is plotted.
3. **Plot**:
   - The regression line visually complements the scatter plot, confirming the linear trend.

Advanced Metrics for Relationship Analysis

1. **R-Squared (Coefficient of Determination)**:
   - Measures the proportion of variance in the dependent variable explained by the independent variable.

```
r_squared = model.score(X, y)
print(f"R-Squared Value: {r_squared:.2f}")
```

2. **P-Value**:
   - Tests the statistical significance of the correlation.
   - Example: p_value < 0.05 indicates a significant relationship.