EXPERIMENT NO. 4

Title:

Examining the Relationship between Two Variables in the Iris Dataset.

Objective:

To analyze the relationship between two variables in the Iris dataset using scatter plots, correlation coefficients, and regression analysis to identify correlations, dependencies, or causal relationships.

Brief Theory:

Introduction: Understanding the relationship between two variables is crucial in data analysis. It helps in identifying patterns, correlations, and potential causal relationships. This experiment involves visual and statistical techniques to analyze such relationships.

Techniques

- **Scatter Plot:** A graph that uses Cartesian coordinates to display values for two variables. It helps visualize the relationship between the variables.
- **Correlation Coefficient:** A statistical measure that calculates the strength of the relationship between two variables. It ranges from -1 to 1, where:
  - 1 indicates a perfect positive correlation.
  - -1 indicates a perfect negative correlation.
  - 0 indicates no correlation.
- **Regression Analysis:** A statistical process for estimating the relationships among variables. Linear regression is a basic form that models the relationship between a dependent variable and one or more independent variables by fitting a linear equation.

Steps: -

Step 1: Import Necessary Libraries

First, import the libraries required for data manipulation and visualization.

Code: -

```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.datasets import load_iris
from scipy.stats import pearsonr
from sklearn.linear_model
import LinearRegression
```

Step 2: Load the Iris Dataset

Load the Iris dataset using the load_iris function from scikit-learn and convert it into a pandas DataFrame.

Code: -

```
# Load the iris dataset
iris_data = load_iris()
iris = pd.DataFrame(data=iris_data.data, columns=iris_data.feature_names)
iris['species'] = iris_data.target
```

The load_iris() function loads the dataset, which is then converted to a DataFrame with appropriate column names.

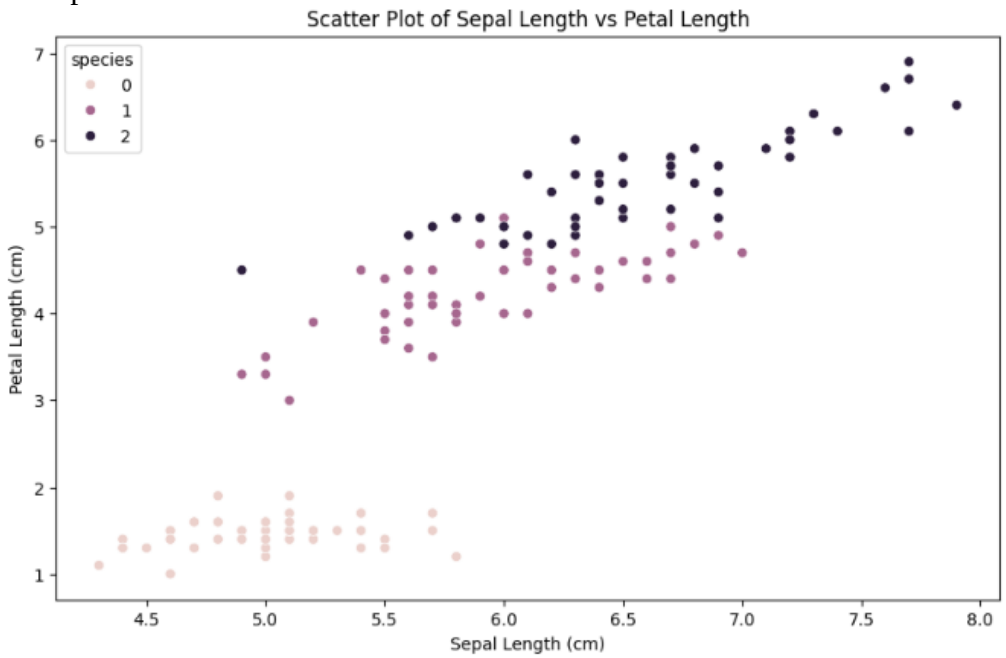
Step 3: Scatter Plot

Create a scatter plot to visualize the relationship between two variables.

Code: -

```
plt.figure(figsize=(10, 6))
sns.scatterplot(x='sepal length (cm)', y='petal length (cm)', hue='species', data=iris)
plt.title('Scatter Plot of Sepal Length vs Petal Length')
plt.xlabel('Sepal Length (cm)')
plt.ylabel('Petal Length (cm)')
plt.show()
```

Output: -



A scatter plot is created to visualize the relationship between sepal length and petal length, with points color-coded by species.

Step 4: Calculate Correlation Coefficient

Calculate the Pearson correlation coefficient between the two variables.

Code: -

```
# Calculate the Pearson correlation coefficient
corr, _ = pearsonr(iris['sepal length (cm)'], iris['petal length (cm)'])
print(f'Pearson correlation coefficient: {corr}')
```

Output: -

```
Pearson correlation coefficient: 0.8717537758865832
```

pearsonr() calculates the Pearson correlation coefficient, which quantifies the strength and direction of the linear relationship between sepal length and petal length.

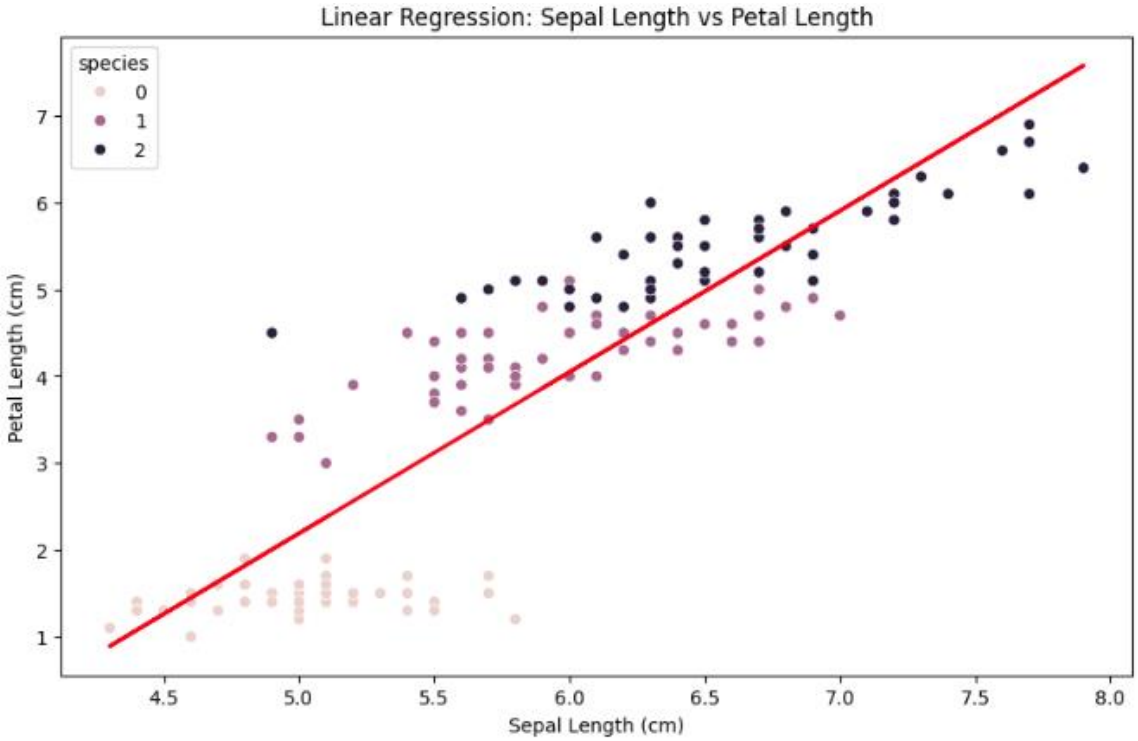
Step 5: Regression Analysis

Perform linear regression analysis to model the relationship between the two variables.

Code: -

```
# Linear Regression
X = iris[['sepal length (cm)']]
y = iris['petal length (cm)']

# Create a linear regression model
reg_model = LinearRegression()
reg_model.fit(X, y)

# Predict values
y_pred = reg_model.predict(X)

# Plot regression line
plt.figure(figsize=(10, 6))
sns.scatterplot(x='sepal length (cm)', y='petal length (cm)', hue='species', data=iris)
plt.plot(iris['sepal length (cm)'], y_pred, color='red', linewidth=2)
plt.title('Linear Regression: Sepal Length vs Petal Length')
plt.xlabel('Sepal Length (cm)')
plt.ylabel('Petal Length (cm)')
plt.show()

# Print regression coefficients
print(f'Intercept: {reg_model.intercept_}')
print(f'Coefficient: {reg_model.coef_[0]}')
```

Output: -




```
Intercept: -7.101443369602459
Coefficient: 1.8584329782548417
```

A linear regression model is created using sepal length as the independent variable and petal length as the dependent variable.

The model is fitted, and predictions are made.

The regression line is plotted on the scatter plot to visualize the linear relationship.

The intercept and coefficient of the regression line are printed, representing the linear equation $y = mx + c$

Conclusion:

In this experiment, we examined the relationship between sepal length and petal length in the Iris dataset. The scatter plot visualized the relationship, the Pearson correlation coefficient quantified the strength and direction of the relationship, and the linear regression analysis modeled the relationship with a linear equation. The high correlation coefficient indicated a strong positive linear relationship between the two variables.

Practice Questions:

1. Create a boxplot comparing the distribution of sepal length across the three species in the Iris dataset. Are there significant differences in the median values? What do you observe about the variability within each species?
2. Use the scatterplot of petal length versus petal width to identify any potential outliers in the data for each species. Are there any extreme data points that don't follow the general pattern? How might outliers affect your analysis and conclusions?
3. Write a Python program that calculates and prints the Pearson correlation coefficient for all pairs of numerical variables (sepal length, sepal width, petal length, petal width) in the Iris dataset. Display the correlation matrix in a readable format.
4. Write a Python program to compute and display the covariance matrix for the numerical features of the Iris dataset. What does the covariance tell you about the relationship between the variables?

Expected Oral Questions

1. Why is a scatterplot useful for examining the relationship between two variables?
2. What is the difference between correlation and covariance?
3. How do you interpret the Pearson correlation coefficient? What does a positive or negative value indicate?
4. What is the purpose of fitting a linear regression model to two variables?
5. How do you identify outliers in this experiment? What impact might outliers have on the analysis?
6. What are the main advantages of using a bar chart to represent categorical data?
7. How would understanding the relationship between variables in a dataset like this be useful in a real-world application?

FAQs in Interviews

Q: How do you interpret the Pearson correlation coefficient?

A: The Pearson correlation coefficient measures the linear relationship between two continuous variables.

- A value close to 1 indicates a strong positive correlation (as one variable increases, the other does too).
- A value close to -1 indicates a strong negative correlation (as one variable increases, the other decreases).
- A value around 0 suggests no linear correlation.

- It's important to note that Pearson correlation only captures linear relationships, so non-linear relationships may have a low Pearson correlation even if they are strongly related.

Q: What is the purpose of fitting a linear regression model?

A: A linear regression model is used to predict the value of one variable based on the value of another variable, assuming a linear relationship between them. In the Iris dataset, for example, you might use petal length to predict petal width. The model generates a regression line that minimizes the difference between predicted and actual values, and provides insights into how one variable changes in response to another.

Q: What is the advantage of using a scatterplot to examine the relationship between two variables?

A: A scatterplot provides a visual representation of the relationship between two numerical variables. It helps you quickly identify patterns, such as positive or negative correlation, clusters, trends, or outliers. Scatterplots also allow you to spot non-linear relationships or interactions that might not be obvious from statistical summaries alone.

Q: What are some key techniques for visualizing data when examining relationships between variables?

A: Important data points can be highlighted using markers, annotations, or different colors for specific sections of the line. You can also add trend lines or emphasize peaks and troughs with callouts.

Q: How can you interpret the shape of a histogram?

A:

- **Scatterplots:** To visualize relationships between two continuous variables.
- **Pairplots:** To examine pairwise relationships between multiple variables simultaneously.
- **Boxplots:** To compare the distribution of a variable across different categories.
- **Heatmaps:** To show correlations between multiple variables.
- **Line plots:** To observe trends over time or another continuous variable.
- **Regression plots:** To fit a line through data points to model linear relationships.

EXPERIMENT NO. 5

Title:

Time Series Analysis on Sample Data Points.

Objective:

To analyze and visualize data points collected at specific time intervals using time series plots, and to understand trends, seasonality, and other patterns in the data.

Brief Theory:

Introduction: Time series analysis involves understanding data points collected or recorded at specific time intervals. This type of analysis is used in various fields such as economics, finance, environmental studies, and more. Key components of time series data include trends, seasonality, and cyclic patterns.

Techniques

- **Time Series Plot:** A line plot where the x-axis represents time and the y-axis represents the variable being measured.
- **Decomposition:** The process of breaking down a time series into its constituent components: trend, seasonality, and residuals.
- **Rolling Statistics:** Calculations like mean or standard deviation applied over a moving window to understand local changes over time.
- **Autocorrelation:** The correlation of a time series with a lagged version of itself to understand repeated patterns over time.

Steps: -

Step 1: Import Necessary Libraries

First, import the libraries required for data manipulation and visualization.

Code: -

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from statsmodels.tsa.seasonal import seasonal_decompose
from statsmodels.graphics.tsaplots import plot_acf, plot_pacf
```

Step 2: Create Sample Time Series Data

Generate sample time series data points.

Code: -

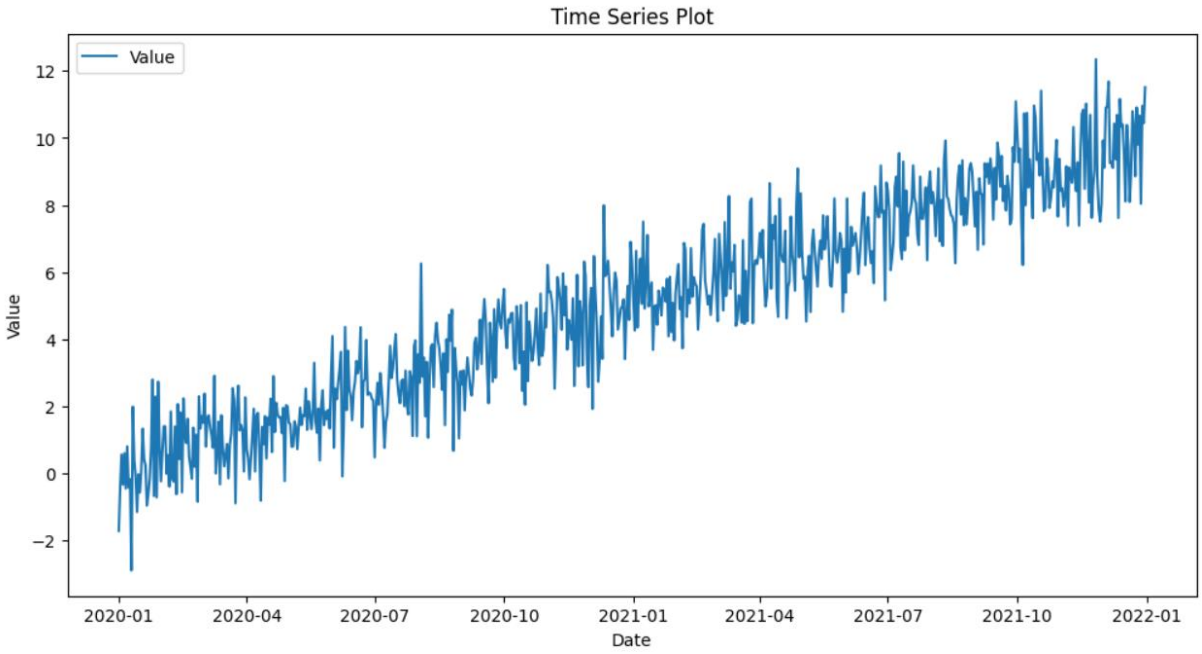
```
# Create a date range
date_rng = pd.date_range(start='2020-01-01', end='2021-12-31', freq='D')

# Create a sample time series data
data = pd.DataFrame(date_rng, columns=['date'])
data['value'] = np.random.randn(len(date_rng)) + np.linspace(0, 10, len(date_rng))

# Set the date column as the index
data.set_index('date', inplace=True)
```

Step 3: Time Series Plot

Create a time series plot to visualize the data.

Code: -

```
plt.figure(figsize=(12, 6))
plt.plot(data.index, data['value'], label='Value')
plt.title('Time Series Plot')
plt.xlabel('Date')
plt.ylabel('Value')
plt.legend()
plt.show()
```

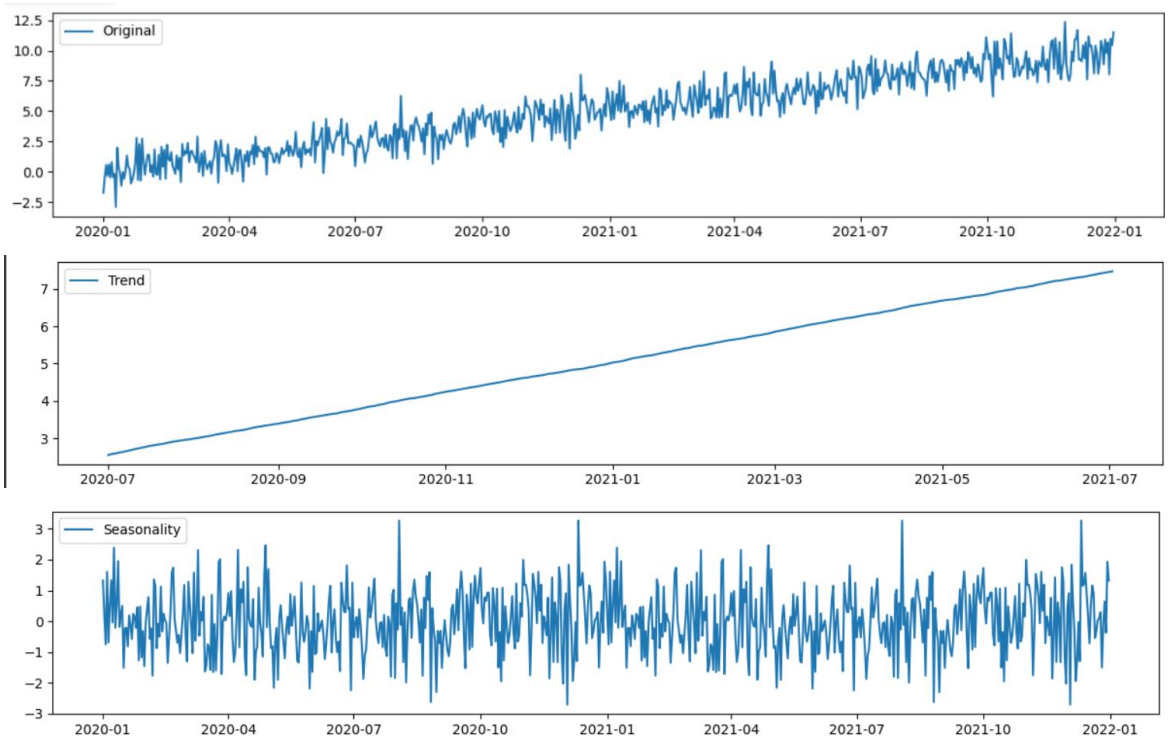


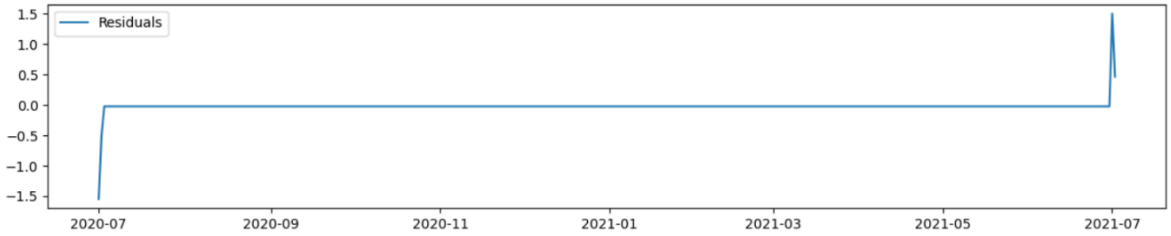
Step 4: Decompose the Time Series

Decompose the time series into trend, seasonality, and residuals.

Code: -

```
decomposition = seasonal_decompose(data['value'], model='additive', period=365)
trend = decomposition.trend
seasonal = decomposition.seasonal
residual = decomposition.resid

plt.figure(figsize=(12, 10))
plt.subplot(411)
plt.plot(data['value'], label='Original')
plt.legend(loc='upper left')
plt.subplot(412)
plt.plot(trend, label='Trend')
plt.legend(loc='upper left')
plt.subplot(413)
plt.plot(seasonal, label='Seasonality')
plt.legend(loc='upper left')
plt.subplot(414)
plt.plot(residual, label='Residuals')
plt.legend(loc='upper left')
plt.tight_layout()
plt.show()
```



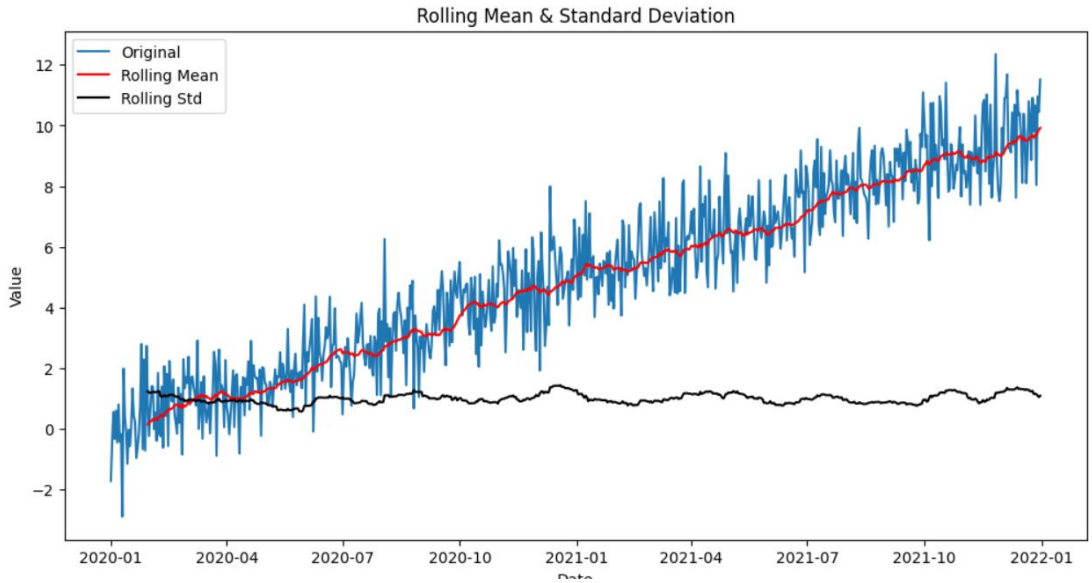


Step 5: Rolling Statistics

Calculate and plot rolling mean and standard deviation.

Code: -

```
rolling_mean = data['value'].rolling(window=30).mean()
rolling_std = data['value'].rolling(window=30).std()

plt.figure(figsize=(12, 6))
plt.plot(data['value'], label='Original')
plt.plot(rolling_mean, color='red', label='Rolling Mean')
plt.plot(rolling_std, color='black', label='Rolling Std')
plt.title('Rolling Mean & Standard Deviation')
plt.xlabel('Date')
plt.ylabel('Value')
plt.legend()
plt.show()
```

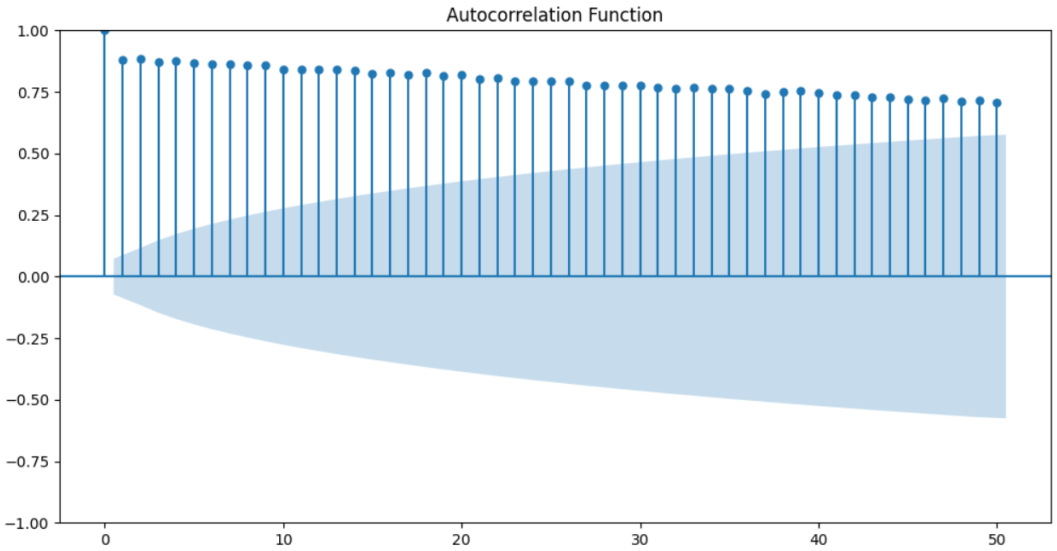


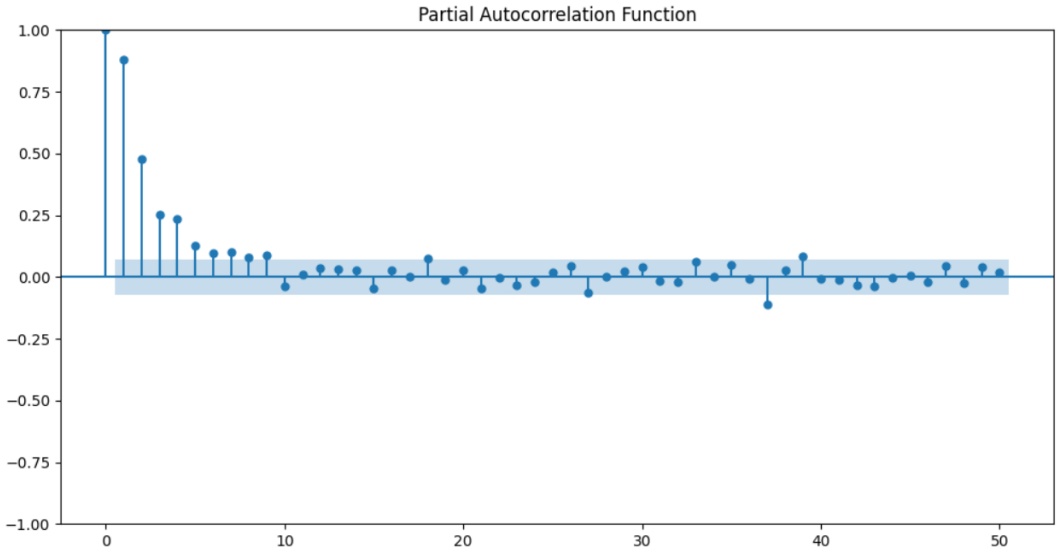
Step 6: Autocorrelation

Plot the autocorrelation and partial autocorrelation functions.

Code: -

```
plt.figure(figsize=(12, 6))
plot_acf(data['value'], lags=50, ax=plt.gca())
plt.title('Autocorrelation Function')
plt.show()
plt.figure(figsize=(12, 6))
plot_pacf(data['value'], lags=50, ax=plt.gca())
plt.title('Partial Autocorrelation Function')
plt.show()
```





Conclusion:

In this experiment, we analyzed time series data points collected at specific intervals. We visualized the data using a time series plot, decomposed the series into trend, seasonality, and residuals, calculated rolling statistics to observe local changes, and examined autocorrelation to understand repeated patterns. These techniques are essential for understanding the underlying structure and dynamics of time series data.

Practice Questions:

1. Using the given sample time series data, plot the data points on a line graph. Identify and describe any visible trends (e.g., upward, downward, or cyclical patterns) over time. What does the trend suggest about the underlying process?
2. Perform an autocorrelation analysis on the time series data. Create a correlogram (plot of autocorrelation function). Identify the significant lags in the data and explain their meaning. How do these lags impact the interpretation of the time series?
3. Write a Python program using **Matplotlib** or **Plotly** to plot the provided time series data points. The x-axis should represent time, and the y-axis should represent the data values. Customize the graph with labels, a title, and gridlines. *(Hint: Use* `plt.plot()` *in Matplotlib or* `px.line()` *in Plotly.).*
4. Using the **statsmodels** library, write a program to perform an **additive decomposition** of a time series into trend, seasonal, and residual components. Display each component as a separate plot. *(Hint: Use* `seasonal_decompose()` *from* `statsmodels.tsa.seasonal`*.)*

Expected Oral Questions

1. What is a time series and how does it differ from other types of data?
2. Can you explain the components of a time series, such as **trend**, **seasonality**, and **residual**? How do these components affect the analysis?
3. What is **autocorrelation** in the context of time series data?
4. What is the difference between **additive** and **multiplicative** decomposition in time series analysis? How do you decide which one to use for a particular dataset?
5. Can you provide a real-world example where **time series analysis** is commonly used?
6. How can time series forecasting be applied in fields like **finance**, **weather forecasting**, or **sales predictions**?