Experiment No. 5

Aim- Implementation of ARIMA model inR Programming.

Objective- To Understand use of Auto-Regression Integrated Moving Average Time Series Model

Theory-

In R programming , data analysis and visualization is so easy to learn the behavior of the data. Moreover, the R language is used mostly in the data science field after Python. Time series analysis is a type of analysis of data used to check the behavior of data over a period of time. The data is collected over time sequentially by the ts() function along with some parameters. It helps in analyzing the pattern of the data over a graph. There are many techniques used to forecast the time series object over the plot graph but the ARIMA model is the most widely used approach out of them.

Time Series Forecasting

Time series forecasting is a process of predicting future values with the help of some statistical tools and methods used on a data set with historical data. Some of the applications of time series forecasting are:

- Predicting stock prices
- Forecast weather
- Forecast the sales of a product

ARIMA model

ARIMA stands for AutoRegressive Integrated Moving Average and is specified by three order parameters: (p, d, q).

- AR(p) Autoregression: A regression model that utilizes the dependent relationship between a current observation and observations over a previous period.An auto regressive (AR(p)) component refers to the use of past values in the regression equation forthe time series.
- I(d) Integration: Uses differencing of observations (subtracting an observation from

observation at the previous time step) in order to make the time series stationary. Differencing involves the subtraction of the current values of a series with its previous values d number of times.

1. Load the data set after installing the package forecast.

2. The Steps of Pre-processing are done, which creates a separate time-series or timestamp.

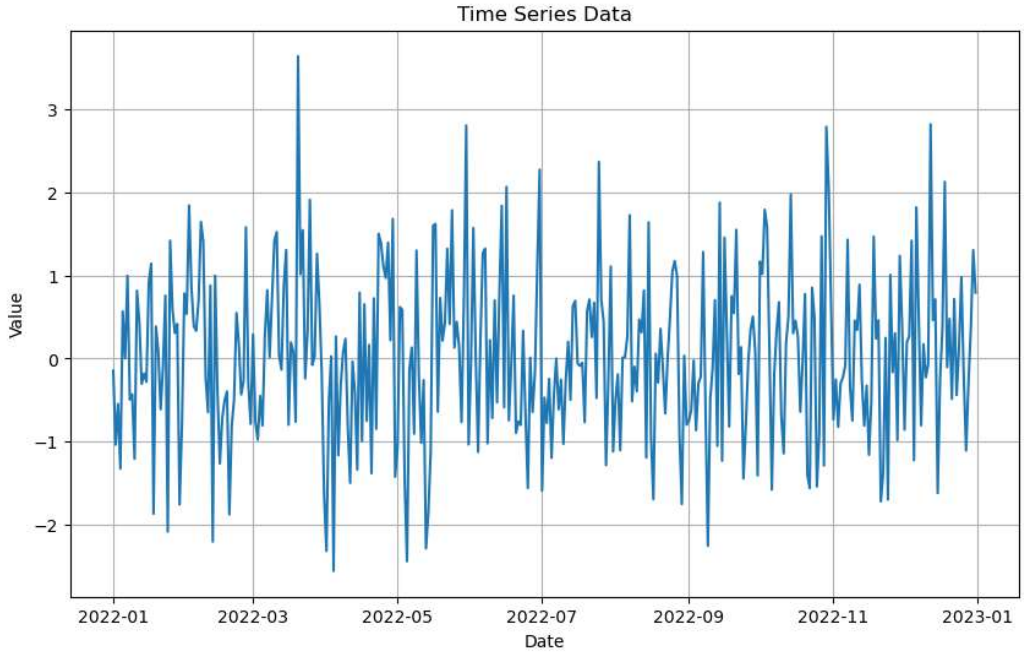
3. Making Time-series stationary and check the required transformations.
4. The difference value 'd' will be performed.
5. The core important step in ARIMA is plotting ACF and PACF.
6. Determine the two parameters p and q from the plots.
7. The previously created value fits the Aroma model and predicts the future values. The Fitting Process is also named as Box-Jenkins Method.

8. Doing Validation.auto. Arima() function is used for automatic prediction and ARIMA Models. This function uses unit root tests, minimization of the AIC and MLE to obtain an ARIMA model.
To make the series stationary, we need to differentiate a previous value from the current value.

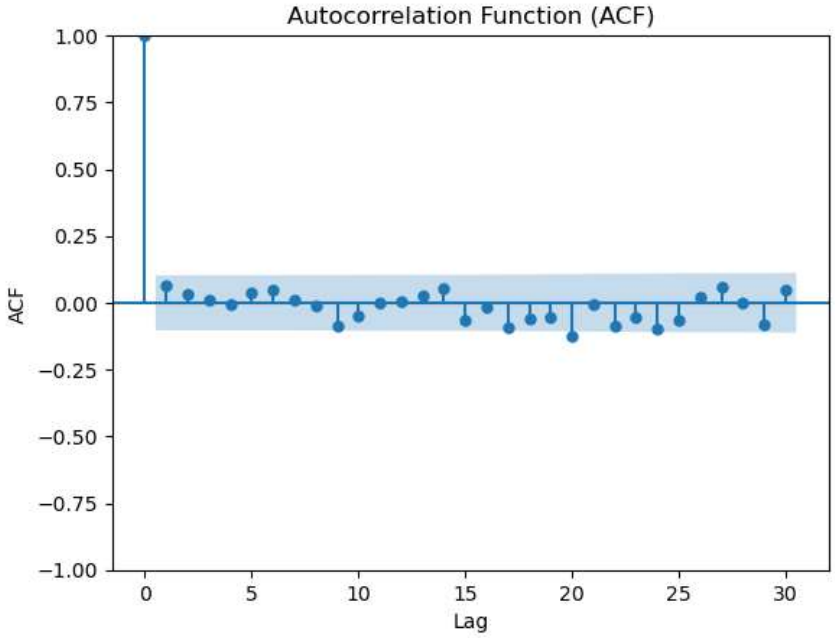
d= pval-cval, if the value is already stationary the d=O.
predict() - Used to predict the model based on the results of the various fitting model used.

Implementation of ARIMA model in R:

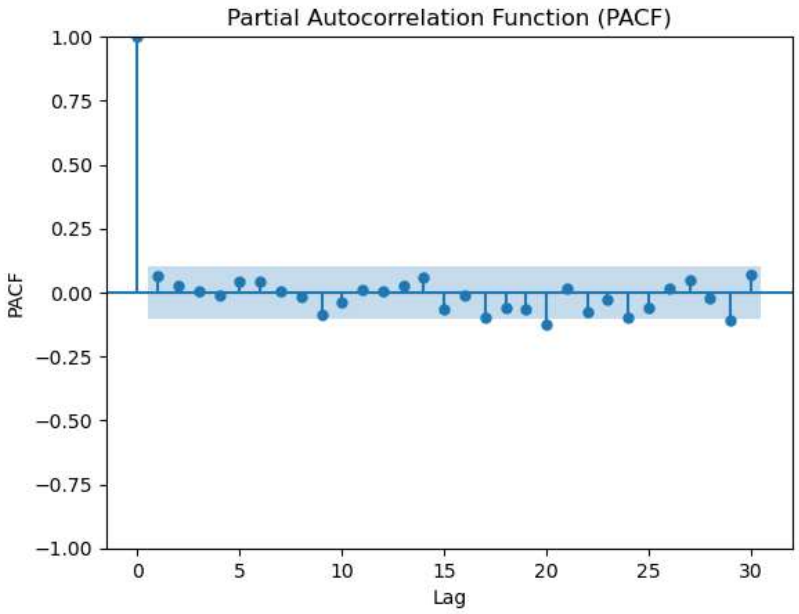
```
# Import necessary libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from statsmodels.tsa.arima.model import ARIMA
from statsmodels.graphics.tsaplots import plot_acf, plot_pacf

# Create a sample time series data
date_range = pd.date_range(start='2022-01-01', end='2022-12-31', freq='D')
data = np.random.randn(len(date_range))  # Random data for demonstration
ts = pd.Series(data, index=date_range)

# Visualize the time series data
plt.figure(figsize=(10, 6))
plt.plot(ts)
plt.title('Time Series Data')
plt.xlabel('Date')
plt.ylabel('Value')
plt.grid(True)
plt.show()
```



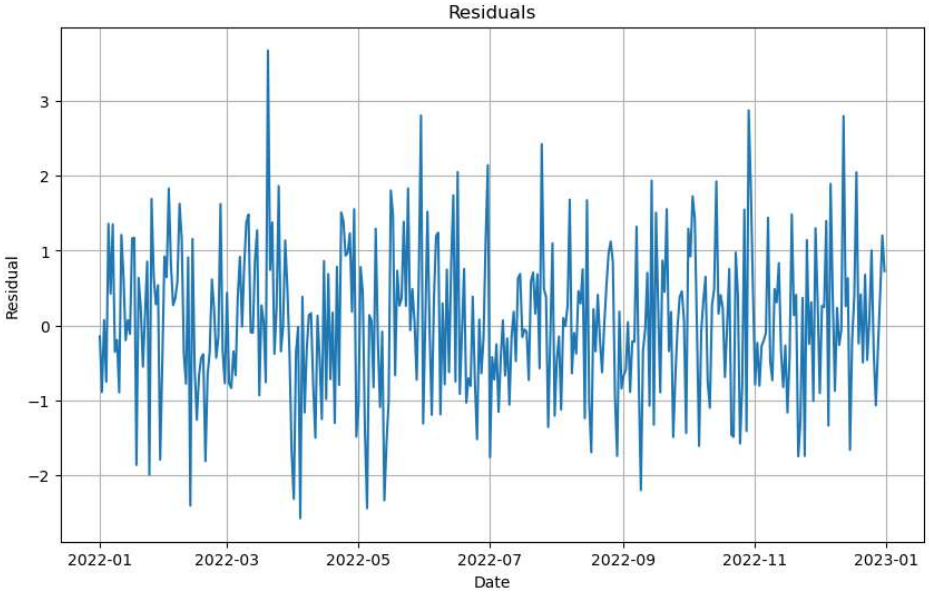

```
# Plot ACF and PACF to determine AR and MA orders
plot_acf(ts, lags=30)
plt.title('Autocorrelation Function (ACF)')
plt.xlabel('Lag')
plt.ylabel('ACF')
plt.show()
```



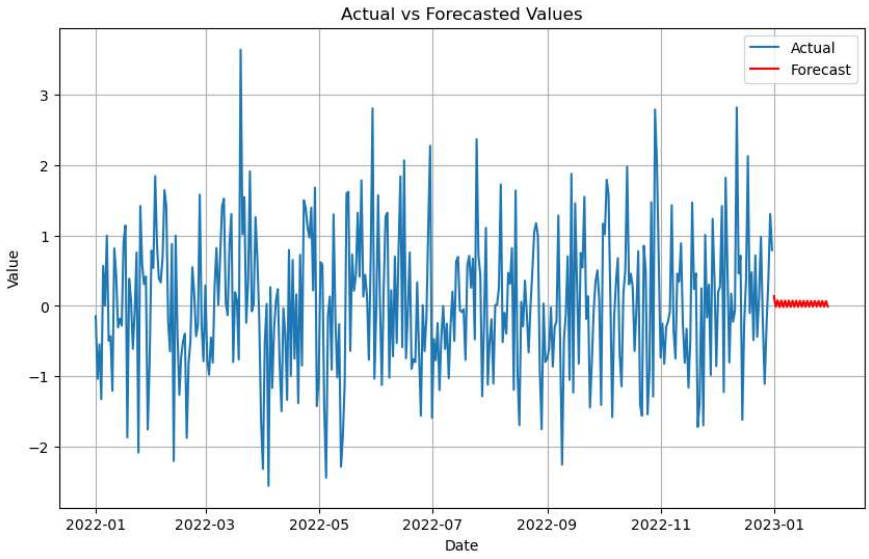
```
plot_pacf(ts, lags=30)
plt.title('Partial Autocorrelation Function (PACF)')
plt.xlabel('Lag')
plt.ylabel('PACF')
plt.show()
```




```
# Fit ARIMA model
order = (2, 1, 2)  # Example order (p, d, q)
model = ARIMA(ts, order=order)
result = model.fit()
# Print model summary
print(result.summary())

# Plot residuals
plt.figure(figsize=(10, 6))
plt.plot(result.resid)
plt.title('Residuals')
plt.xlabel('Date')
plt.ylabel('Residual')
plt.grid(True)
plt.show()
```




```
# Forecast future values
forecast_steps = 30  # Example forecast steps
forecast = result.forecast(steps=forecast_steps)
# Plot forecasted values
plt.figure(figsize=(10, 6))
plt.plot(ts, label='Actual')
plt.plot(forecast, label='Forecast', color='red')
plt.title('Actual vs Forecasted Values')
plt.xlabel('Date')
plt.ylabel('Value')
plt.legend()
plt.grid(True)
plt.show()
```



Conclusion:-

1. The difference between the actual value of the time series and the forecasted value called-

The difference between the actual value of a time series and the forecasted value is typically referred to as the "forecast error" or "residual." These terms essentially mean the same thing: they represent the discrepancy between what was predicted by the forecasting model and what actually occurred. Forecast errors are crucial for assessing the accuracy and reliability of forecasting models. By analyzing these errors, analysts can evaluate the performance of their models, identify any patterns or biases in the forecasts, and potentially improve the forecasting process.

2. Use of ARIMA ()-

The `ARIMA()` function, which stands for AutoRegressive Integrated Moving Average, is a powerful tool in time series analysis for modeling and forecasting. It's a versatile method capable of capturing a wide range of temporal patterns, including trends, seasonality, and autocorrelation.

3. Use of forecast()-

The `forecast()` function in R is used to generate forecasts from time series models, including ARIMA models. It takes as input a fitted time series model and produces forecasts for future time points. With optional arguments, you can specify the number of periods to forecast, confidence intervals, and other forecasting options. The function returns a forecast object that can be plotted, summarized, or further analyzed. `forecast()` is a powerful tool for predicting future values and assessing the uncertainty associated with those predictions in time series analysis.