Title : Time Series Forecasting

Introduction

This project presents a **Stock Price Forecasting** application using time-series models implemented with **Streamlit**. The application allows users to fetch stock data, analyze trends, check stationarity, and predict future prices using **ARIMA** and **SARIMA** models. The goal is to provide insights into stock price movements and help users make informed decisions.

Approach

The project follows a structured approach to implement stock price forecasting:

1. **Data Collection**: Fetching historical stock data from Yahoo Finance.
2. **Exploratory Data Analysis (EDA)**:
    - Checking stationarity using the Augmented Dickey-Fuller (ADF) test.
    - Visualizing historical stock prices.
    - Plotting autocorrelation to analyze time dependencies.
3. **Model Selection & Training**:
    - **ARIMA (AutoRegressive Integrated Moving Average)**: A standard time-series forecasting model for univariate data.
    - **SARIMA (Seasonal ARIMA)**: An extension of ARIMA that incorporates seasonality in the data.
    - Hyperparameter tuning for optimal model performance.
4. **Forecasting**:
    - Predicting stock prices for different time periods (next day, next week, next month).
    - Visualizing forecasted prices along with historical trends.
5. **Performance Evaluation**:
    - Metrics used: **Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), and Mean Absolute Percentage Error (MAPE)** to assess model accuracy.

Methodology

1. Data Preprocessing

- The application allows users to select a stock symbol, start date, and end date.
- The stock data is retrieved using **Yahoo Finance API**.

2. Data Cleaning:

- **Handling Missing Values:** Any missing values in the dataset are either filled using forward-fill or dropped based on context.
- **Outlier Detection:** The code removes extreme outliers that could impact model performance.
- **Ensuring Data Integrity:** Non-numeric values and duplicates are handled before processing.

3. Stationarity Check

- **ADF Test**: Checks whether the stock price time series is stationary or requires differencing.
- **Autocorrelation Plot**: Helps determine time dependencies in the data.

4. Model Training

ARIMA Model

- The ARIMA model is trained with predefined order parameters `(p,d,q)`, where:
  - `p` (AutoRegressive term) determines the number of lag observations.
  - `d` (Differencing order) ensures stationarity.
  - `q` (Moving Average term) defines the size of the error component.
- Forecasting is done using the **fitted ARIMA model**.

SARIMA Model

- The SARIMA model extends ARIMA by incorporating seasonal components.
- The seasonal order `(P,D,Q,s)` captures seasonal patterns in stock prices.

5. Forecasting & Visualization

- Users select a forecasting period (next day, week, or month).
- The trained model predicts future prices, which are visualized alongside historical data.

6. Performance Evaluation

- The predicted values are compared against actual stock prices (for the latest data points).
- The evaluation metrics used:
  - **MAE (Mean Absolute Error)**
  - **RMSE (Root Mean Squared Error)**
  - **MAPE (Mean Absolute Percentage Error)**

Results & Conclusion

- The application successfully forecasts stock prices with a reasonable level of accuracy.
- Seasonal trends are captured effectively using **SARIMA**, making it a suitable choice for stocks with periodic fluctuations.
- Future improvements could involve incorporating machine learning models like **XGBoost** or **Random Forest** for hybrid forecasting.

Technologies Used

- **Python** (Pandas, NumPy, Matplotlib, Seaborn, Scikit-Learn)
- **Streamlit** (Interactive UI for visualization)
- **Statsmodels** (Time-series modeling with ARIMA and SARIMA)
- **Yahoo Finance API** (Stock data retrieval)

Deployment

- The application can be deployed on **Streamlit Cloud** or **Heroku** for real-time stock forecasting.

This project is a practical implementation of **time-series forecasting** using statistical models, aiming to provide insights into stock market trends.