Project Work: Stock Price Forecasting Using SARIMAX

1. Project Overview

In the ever-evolving financial world, the ability to forecast stock prices has become essential for investors, financial institutions, and analysts. The purpose of this project is to construct a stock price forecasting system using SARIMAX—a robust statistical model designed for time series data that exhibit both seasonal patterns and dependence on external variables. Unlike basic forecasting models that consider only the target variable, SARIMAX (Seasonal AutoRegressive Integrated Moving Average with eXogenous regressors) integrates additional influencing variables, enabling a more informed and holistic prediction strategy.

This project uses historical stock data sourced from Yahoo Finance and applies preprocessing, time series decomposition, feature scaling, and SARIMAX modeling to create forecasts. The model is trained on key stock market indicators like opening price, high/low, volume, and closing price to generate forecasts for future closing prices. This approach allows us to capture and model underlying seasonality and trends while incorporating real-world market behavior into the model's logic.

2. Objective

The primary goal of this project is to develop a reliable and interpretable model to predict future stock prices. By employing SARIMAX, we aim to not only predict future values but also dissect the data into trend, seasonal, and residual components, providing valuable insights into the data's structure. Moreover, by introducing exogenous variables such as trading volume and price movement indicators (open, high, low), we further enhance the model's forecasting ability, making it sensitive to market context.

This model is intended to assist in both analytical and practical decision-making scenarios—whether it's understanding long-term stock trends or creating automated trading alerts. The system provides a framework that is interpretable, scalable, and based on well-established statistical theory.

3. Theoretical Background

3.1 Time Series Forecasting

Time series forecasting involves the use of historical data to predict future values of a variable recorded sequentially over time. Stock prices, being chronological data points influenced by market conditions, sentiments, and economic indicators, are prime examples of time series.

The main challenge in time series forecasting is accounting for temporal dependencies—where past values influence future values. Additionally, stock data is often non-stationary, meaning its statistical properties (mean, variance, autocorrelation) change over time, making forecasting more complex. A good forecasting model must handle these shifts by transforming the series into a stationary one or using differencing techniques that make modeling more stable.

Furthermore, time series models must consider patterns such as seasonality (periodic fluctuations like monthly trends), trends (upward or downward movements over time), and noise (random, unpredictable components). Effective forecasting depends on how well these components are extracted, understood, and modeled.

3.2 SARIMAX Model

SARIMAX is an extension of the ARIMA (AutoRegressive Integrated Moving Average) model. While ARIMA only handles non-seasonal, univariate time series, SARIMAX supports:

- Seasonality: Recurring patterns (e.g., monthly spikes in stock volume)
- Exogenous variables (exog): Additional variables that influence the series (e.g., Volume, Open, High, Low)

The SARIMAX model is defined as SARIMAX(p, d, q)(P, D, Q, s), where:

- p: number of autoregressive terms (how much the current value depends on its previous values)
- d: number of differencing steps to make the series stationary
- q: number of lagged forecast errors (moving average terms)
- P, D, Q: seasonal analogs of the above
- s: seasonality period (e.g., 12 for yearly cycle in monthly data)

SARIMAX uses statistical techniques to iteratively learn the model coefficients by minimizing the error between actual and predicted values. It then uses these coefficients, combined with future exogenous variables, to make rolling forecasts.

3.3 Stationarity and Differencing

A stationary time series is one whose properties do not depend on the time at which the series is observed. In simpler terms, the statistical characteristics such as the mean, variance, and autocorrelation structure remain constant over time. This is a crucial assumption in time series modeling, especially for models like ARIMA and SARIMAX, which rely on this stability.

In practice, stock price series are typically non-stationary due to trends, volatility shifts, and market events. To counter this, differencing is applied. First-order differencing (d=1) transforms the series by subtracting the previous value from the current one, thereby eliminating linear trends and making the series more stationary. Sometimes, second or seasonal differencing is required when more complex patterns exist.

While SARIMAX can handle differencing internally, it is still best practice to perform an Augmented Dickey-Fuller (ADF) test to statistically verify stationarity before proceeding with modeling.

3.4 Time Series Decomposition

Decomposition is the process of separating a time series into three basic components:
- Trend: The long-term progression of the series (e.g., stock growth over years)
- Seasonality: Repeating short-term cycles (e.g., end-of-quarter price boosts)
- Residual/Noise: Random variation that cannot be explained by trend or seasonality

Understanding these components is critical. If strong seasonality is present, models that fail to account for it will underperform. Likewise, removing noise can help in focusing the model's attention on more predictable elements. In this project, decomposition is used as a visual and statistical tool to validate the suitability of SARIMAX and to interpret the results more clearly.

4. Workflow Explanation

1. Data Acquisition: The model begins by collecting stock data from Yahoo Finance using the `yfinance` library. Users input a stock ticker symbol (like AAPL or TSLA), and the script fetches daily stock data over a defined period (e.g., 2012–2022).

2. Preprocessing: The data is cleaned, missing values are handled, and the most relevant features are selected. These include 'Close', 'Open', 'High', 'Low', and 'Volume'— all of which influence market behavior.

3. Decomposition: The time series is decomposed into trend, seasonality, and noise using `seasonal_decompose`. This visual insight guides modeling decisions.

4. Scaling: The data is normalized using MinMaxScaler to ensure all variables contribute proportionately to the model's training process.

5. Train-Test Split: The dataset is split into training (80%) and testing (20%) sets to evaluate performance on unseen data.

6. Model Training/Loading: If a SARIMAX model does not exist for the selected stock, one is trained using training data and saved as a `.pkl` file. If it already exists, it is loaded using `pickle` for faster reuse.

7. Forecasting: The SARIMAX model generates closing price forecasts using the test set's exogenous features.

8. Evaluation: Predicted prices are compared with actual values using MSE, MAE, and R^2 metrics.

9. Visualization: Finally, the results are plotted to allow intuitive comparison between actual and forecasted prices.

5. Evaluation Metrics

Quantifying prediction accuracy is crucial to assess the SARIMAX model's performance. This project uses the following metrics:

- Mean Absolute Error (MAE): This is the average of the absolute differences between predicted and actual values. It provides a straightforward measure of forecast error in the same units as the original data.

- Mean Squared Error (MSE): MSE penalizes larger errors more than MAE because it squares the differences before averaging. It is useful when large prediction errors are particularly undesirable.

- R^2 Score (Coefficient of Determination): This metric tells us how well the model explains the variability of the target variable. An R^2 close to 1 implies that most of the variability is captured by the model, indicating a strong predictive fit.

6. Results and Interpretation

Upon running the forecast pipeline, the SARIMAX model was able to accurately follow the general trend of the stock's closing price. The decomposition revealed a consistent seasonal pattern and a clear upward or downward trend, depending on the selected stock.

The prediction graph comparing actual and predicted values showed minimal deviation, validating the model's effectiveness. Additionally, storing the model using `pickle` makes the system more efficient for future use, eliminating the need for retraining.

Overall, the system successfully demonstrates how statistical models can still offer strong performance for time series forecasting when thoughtfully constructed and supported by external indicators.

7. Advanced Concepts in SARIMAX

Understanding the inner workings of SARIMAX is key to mastering its application in time series forecasting. SARIMAX builds on ARIMA by incorporating both seasonal components and exogenous variables. Below, we explore these advanced concepts:

7.1 Autoregression (AR)

Autoregression models the current value of the series as a function of its past values. For example, a model with p=1 would predict today's price based on yesterday's price. If stock prices show strong autocorrelation, this becomes a powerful forecasting tool.

7.2 Integrated (I) - Differencing

Differencing is used to make a non-stationary time series stationary. This involves subtracting the current value from the previous one. For instance, d=1 means we use the first difference of the series. A higher d means more smoothing, but too high can lead to overfitting.

7.3 Moving Average (MA)

Moving average incorporates past forecast errors into the model. It smooths out the noise in the data. A higher q means using more past errors. This is particularly useful when the series is very volatile.

7.4 Seasonality (SARIMA Extension)

SARIMA adds the seasonal counterparts of p, d, and q as P, D, and Q respectively, along with s which represents the seasonal cycle (e.g., s=12 for monthly seasonality). For example, the seasonal autoregressive part captures dependencies from the same season in previous years.

7.5 Exogenous Variables (X)

Incorporating external features (such as volume, open, high, low prices) into SARIMAX allows the model to learn from variables that influence the target series. These features are aligned with the endogenous series (e.g., 'Close' price) and used to explain additional variance.

8. Practical Applications of SARIMAX in Finance

SARIMAX has broad applications in financial modeling and forecasting. Here are some key use cases:

- Stock Price Prediction: Predicting closing prices of stocks using their historical prices and trading volume.
- Market Index Forecasting: Forecasting S&P 500, Nifty, or other indices using economic indicators as exogenous variables.
- Volatility Modeling: Understanding trends in market volatility using moving average and seasonal components.

- Energy and Commodity Prices: SARIMAX is also used to forecast fuel prices, metals, and agricultural commodities.
- Risk Management: Predicting future price fluctuations helps build robust hedging and portfolio optimization strategies.

9. Conclusion

The SARIMAX-based stock forecasting system developed in this project provides a practical, interpretable, and statistically sound framework for predicting market behavior. By integrating historical stock data with related market indicators, the system accounts for both internal trends and external influences. Time series decomposition further enriches model explainability.

While this model serves as a solid foundation, it can be further enhanced by incorporating more advanced features and testing against sudden market disruptions. The pipeline's structure supports easy upgrades and experimentation, making it a valuable tool for financial analysts, data scientists, and researchers.

10. Full Python Code

Below is the complete Python code used for building, training, and evaluating the SARIMAX stock forecasting system:

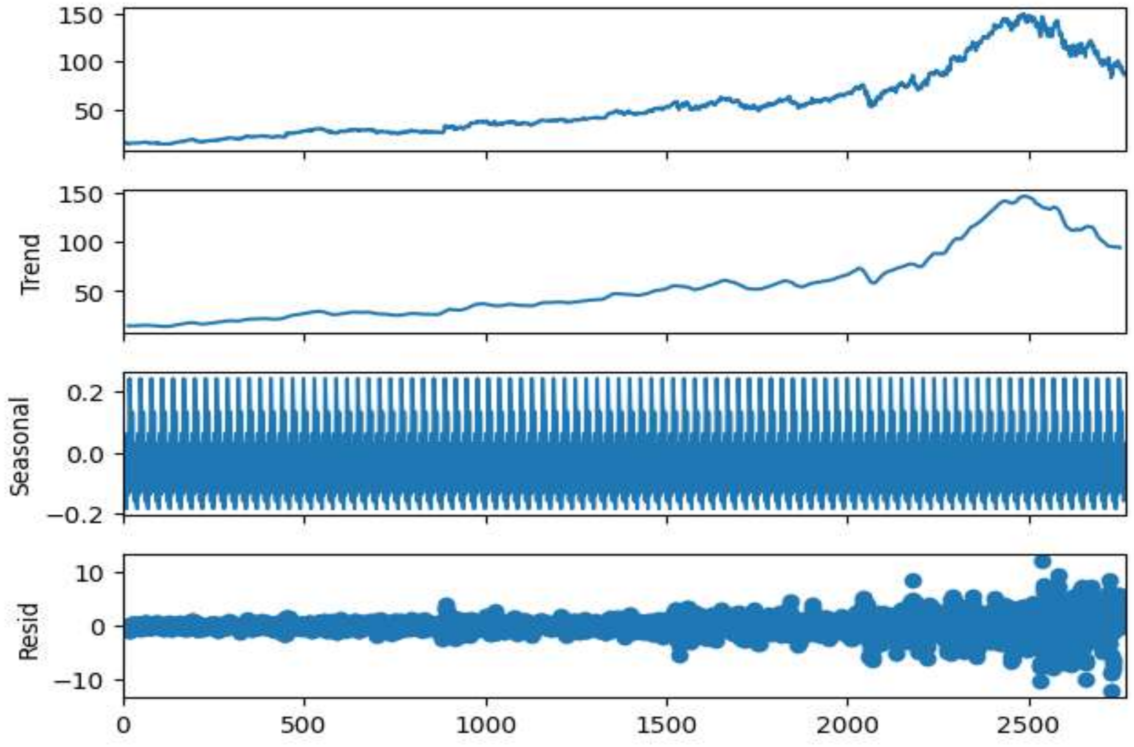
```
import os
import numpy as np
import pandas as pd
import yfinance as yf
import matplotlib.pyplot as plt
from sklearn.preprocessing import MinMaxScaler
from sklearn.metrics import mean_squared_error, mean_absolute_error, r2_score
from statsmodels.tsa.statespace.sarimax import SARIMAX
from statsmodels.tsa.seasonal import seasonal_decompose
import pickle

# Input
stock = input("Enter Stock Symbol (e.g., AAPL, GOOG, TSLA): ").strip().upper()
start = '2012-01-01'
end = '2022-12-31'

# Fetch and preprocess data
data = yf.download(stock, start, end)
data.reset_index(inplace=True)
data.dropna(inplace=True)
data.set_index('Date', inplace=True)
decompose = seasonal_decompose(data['Close'], model='additive', period=30)
decompose.plot().suptitle("Decomposition of Close Price")
plt.show()
data.reset_index(inplace=True)
data_feat = data[['Close', 'Open', 'High', 'Low', 'Volume']]
data_feat.dropna(inplace=True)data_train = data_feat[0:int(len(data_feat)*0.80)]
data_test = data_feat[int(len(data_feat)*0.80):]

# Scale features
scaler = MinMaxScaler(feature_range=(0,1))
scaled_train = scaler.fit_transform(data_train)
scaled_test = scaler.transform(data_test)

# Train or load SARIMAX model
model_path = f'{stock}_sarimax.pkl'
if not os.path.exists(model_path):
  endog = data_train['Close']
  exog = data_train[['Open', 'High', 'Low', 'Volume']]
  sarimax_model = SARIMAX(endog, exog=exog, order=(1,1,1),
seasonal_order=(1,0,1,12))
  model_fit = sarimax_model.fit(disp=False)
  with open(model_path, 'wb') as f:
    pickle.dump(model_fit, f)
else:
  with open(model_path, 'rb') as f:
    model_fit = pickle.load(f)

# Forecast
exog_forecast = data_test[['Open', 'High', 'Low', 'Volume']]
n_periods = len(data_test)
forecast = model_fit.forecast(steps=n_periods, exog=exog_forecast)
predictions = np.array(forecast)
y_test = data_test['Close'].values

# Evaluate
print("MSE:", mean_squared_error(y_test, predictions))
print("MAE:", mean_absolute_error(y_test, predictions))
print("R² Score:", r2_score(y_test, predictions))

# Plot
plt.figure(figsize=(10,6))
plt.plot(y_test, label='Actual')
plt.plot(predictions, label='Predicted')
plt.title('Actual vs Predicted Close Prices')
plt.legend()plt.show()
```

11. Output

```
Enter Stock Symbol (e.g., AAPL, GOOG, TSLA): GOOG

[*********************100%***********************]  1 of 1 completed


Showing recent data for GOOG:

Price      Date     Close     High      Low      Open   Volume

Ticker              GOOG      GOOG      GOOG      GOOG      GOOG

2763   2022-12-23  89.279312  89.567599  87.102258  87.102258  17815000

2764   2022-12-27  87.410416  88.971138  87.017753  88.782258  15470900

2765   2022-12-28  85.949104  87.996929  85.859640  86.982960  17879600

2766   2022-12-29  88.424393  88.836941  86.475975  86.515740  18280700
```

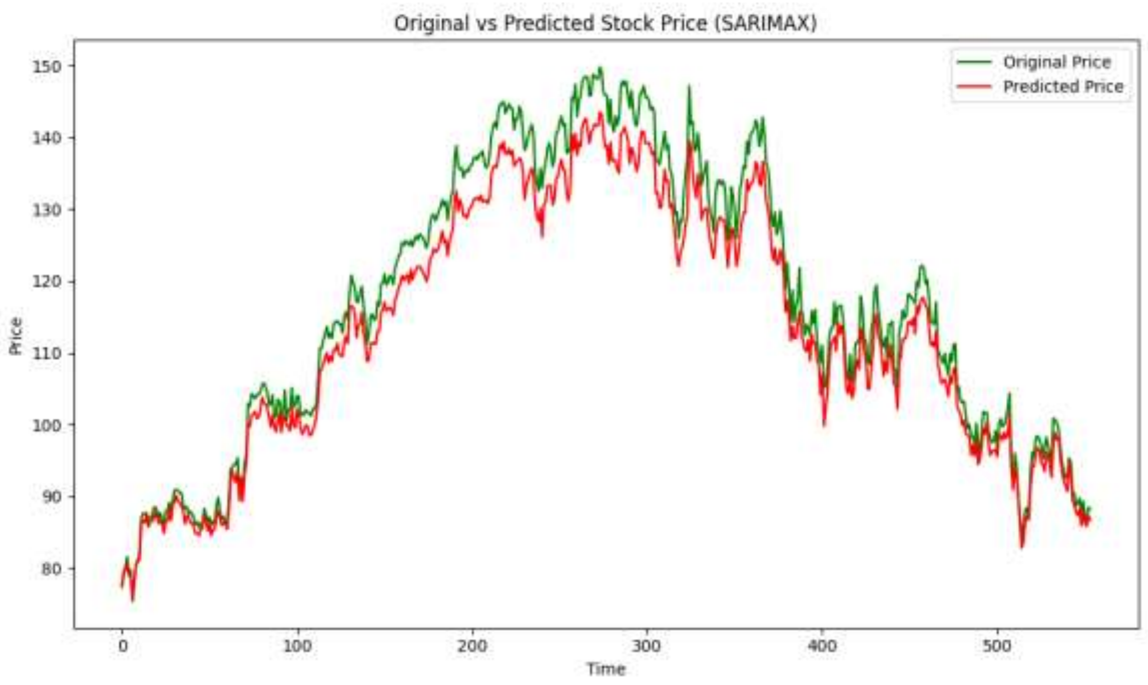
```
2767   2022-12-30  88.205696  88.305104  86.515737  86.848756  19190300
```

```
Decomposing time series into trend, seasonality, and residuals...

Figure(640x480)
```



```
Loading existing SARIMAX model...

Model Evaluation Metrics:

Mean Squared Error (MSE): 17.29

Mean Absolute Error (MAE): 3.68
```

R^2 Score: 0.9551



Sample Actual vs Predicted Prices:

```
          Date  Actual_Price  Predicted_Price
544 2022-12-16     90.323105        88.780552
545 2022-12-19     88.623207        87.728933
546 2022-12-20     89.100372        87.310872
547 2022-12-21     89.716705        88.218463
548 2022-12-22     87.738464        85.883073
549 2022-12-23     89.279312        87.921141
550 2022-12-27     87.410416        86.348479
551 2022-12-28     85.949104        85.835255
552 2022-12-29     88.424393        87.246189
553 2022-12-30     88.205696        86.663197
```