https://doi.org/10.53656/math2025-6-2-apd

Science in Education
Научно-методически статии

APPLICATION OF PYTHON IN DATA FORECASTING USING SARIMA MODELS

Vesela Dimitrova
University of National and World Economy, Sofia (Bulgaria)

Abstract. This study aims to build and estimate a SARIMA model to predict the prices of the shares of a company listed on NYSE. With the help of Python, the daily closing prices of the stock for the period from 3.01.2023 to 17.03.2025 were taken. Also, SARIMA models were evaluated and code was compiled to select the most appropriate parameters. Finally, the daily forecasts for one month were calculated. The study found that SARIMA model can be considered reliable for forecasting stock prices and can complement any other analysis method. Future research could also include enhancing the study with external factors and improving the forecasts could be also done by testing of SARIMAX models.

Keywords: Python; forecasting; econometric modelling; SARIMA; financial markets

1. Introduction

Data are all around us, and particularly for stock market operations they typically have the form of time series data – measurements on any variable of interest over time. One of the important analytical tasks, when it comes to such type of data, is to study how a market phenomenon behaves in the time span, are there any trends, and what can be expected for the future of this phenomenon.

This type of data can be collected within many different fields of economic activity where we have an accumulation of data for certain time intervals in finance, manufacturing, international and domestic trade, transport, logistics, tourism, etc. These datasets have in recent years increased both in volume and the speed at which they are accumulated and converted into big data. This intensified data flow makes it difficult to store and process the data with traditional tools and software products. Over the past decade, Python has emerged as the most widely used software for data analysis, particularly for machine learning and big data processing.

SARIMA (Seasonal Autoregressive Integrated Moving Average) models find applications in various areas of economic and financial analysis. A number of indicators such as GDP, economic growth, inflation, unemployment, production and sales of various products in agriculture and industries can be analyzed and forecasted by this method. The models are used in finance to forecast exchange rates, stock and commodity prices or trading volumes. In energy, SARIMA models are used to forecast electricity consumption and demand from various sources. Wide application of such methods can be found in forecasting for various diseases such as COVID-19, cholera, etc., as well as to model patient admissions in hospitals.

Autoregression Integrated Moving Average (ARIMA) and SARIMA models are traditionally introduced to students in the framework of academic courses such as Time Series Analysis, Machine Learning, and Big Data Analytics. Their application is further reinforced through training with specialized software packages, including Stata, R, and Python. Students and PhD researchers can employ these models as methodological tools for forecasting a wide range of dynamic indicators such as prices, sales, production, energy consumption, population and other variables across multiple areas of application.

Hereafter, we will discuss and illustrate the application of this statistical method and the respective libraries in Python that can be used to model and forecast data presented in time series. The article aims to conduct analysis and predict the price of a stock traded on the New York Stock Exchange (NYSE) based on time series using Python. To illustrate the capabilities of Python we deploy its libraries to construct SARIMA models, estimate them, and forecast a series of values for a selected stock market variable. Such models capture both short-term and long-term internal dependencies in the time series in order to provide a more accurate forecasted value.

2. Literature review

ARIMA and SARIMA models are among the leading forecasting tools for modelling stochastic dynamics. With the development of Python software libraries and tools, such type of analyses has gained even more popularity among financial and currency market analysts.

In (Kolková et al., 2021) the authors considered a range of statistical methods and models for Deep Learning to present the possibilities of forecasting in business practice. The methods used by them are Seasonal naïve, TBATS, Facebook Prophet, and SARIMA. The applied methods were experimented with the help of Python. The prediction made is of the daily sales of an e-commerce company. Benayad & Halimi (2022) model and forecast the monthly unemployment rates in Algeria. The method used is SARIMA deploying algorithms of Eviews and Python.

In (Albeladi et al., 2024) the authors evaluated the effectiveness of specialized SARIMA models for forecasting the stock market in the Persian Gulf region. Libraries in Python were also used to apply the models. It was shown that the evaluation results of the SARIMA models performed well in forecasting, especially when the data contained seasonality.

Forecasting the exchange rate is one of the important tasks facing the analysts of the currency markets. In the book (Nokeri, 2021) devotes a whole section on forecasting based on the ARIMA and SARIMA models. The examples have been developed on the basis of exchange rates and other financial variables using the Python libraries. On the basis of weekly values, a 90-day forecast of the USA/INR exchange rate was also obtained by (Tamizharasi et al., 2024) who obtained precise forecasts by a SARIMA model. It has been found that the SARIMA model can have good abilities for forecasting the exchange rate and momentum of the currency market. Several chapters in (Peixero, 2021) are dedicated to predicting time series using Python where ARMA, ARIMA, SARIMA, SARIMAX and other models are considered.

In (Gikungu et al., 2015) forecasts inflation in Kenya by constructing two main models, ARIMA and SARIMA. According to them, SARIMA models are suitable in situations where time series show seasonality, with fluctuations that recur by approximately the same periodicity. This makes SARIMA suitable for studies relating to quarterly data. A comparative study shows that SARIMA outperforms both ARIMA and machine learning approaches in forecasting agricultural commodity prices, enhancing SARIMA's flexibility under different market conditions (Ariyanti & Yusnitasari, 2023).

In addition, the adaptability of SARIMA extends beyond single-variable predictions. The increased accuracy of its prediction is also demonstrated when it is integrated with other models. For example, Adineh et al. (2021) highlight the importance of pre-processing in predicting time series using SARIMA, suggesting that combining SARIMA with models such as baseline vector autoregression can lead to improved results.

The SARIMAX application also demonstrates robust forecasting results for various financial and economic indicators. Comparative analysis by (Alqatawna et al. 2023) highlights that the SARIMAX models outperform other approaches, such as ARIMA and LSTM (Long Short-Term Memory Networks), in forecasting the order volume of logistics companies in the UAE, Saudi Arabia and Kuwait, resulting in lower mean absolute percentage errors (MAPE).

Using monthly data, Lee et al. (2024) develop a Seasonal Autoregressive Integrated Moving Average with eXogenous variables (SARIMAX) model for the container capacity of the port of Singapore. The price of West Texas Intermediate (WTI) crude oil and the volume of China's exports, along with the impact of the COVID-19 pandemic, were considered as exogenous variables in the SARIMAX model. These results show that the SARIMAX model – when including WTI prices and the volume of China exports – outperforms linear regression, LASSO regression, Ridge regression, ECM (Error Correction Mechanism) and many others.

Empirical modelling shows that SARIMA models maintain a high level of precision when applied to different financial datasets, with some studies reporting accuracy rates above 80% when forecasting stock prices (Acula & Guzman, 2020). This efficiency leads to its wide application in various financial fields, including commodity pricing and inflation, thereby reinforcing its status as a key analytical tool in financial modelling and forecasting (Wanjuki et al., 2021; Divisekara et al., 2020; Ariyanti & Yusnitasari, 2023).

3. Econometric methodology of SARIMA models

Time series analysis employs various methods that are based on assumptions about the nature and behavior of a given variable. The purpose of time series analysis is to use methods that extract different characteristics

from the data identifying the nature of the trend, seasonality, cyclicality, etc. Predicting future values of a given variable or process is also an important part of this analysis.

Various specific methods can be used to characterize the trend of development of the time series, such as the moving averages method, analytical smoothing, exponential smoothing, etc. A specific issue in the study of time series dynamics is the presence of autocorrelation in the series, or the systematic dependence of the current values of the variable on its previous values (i.e. from previous moments or periods of time). In order to capture a variety of related effects, ARMA models include an Autoregression (AR) component and a Moving Averages (MA) component. The lag of the Autoregressive (AR) process is denoted by "p", which means that the current values of the examined time series are linearly dependent on their past values up to lag "p". The lag of the Moving Average (MA) process is denoted by "q". According to such a stochastic process, the current value of time series is linearly dependent on the current and past values of the error variable up to lag "q".

ARMA models can be applied only to stationary time series. However, many of the time series are non-stationary, containing a trend component. This way, Autoregression Integrated Moving Averages (ARIMA) models assume elimination of the trend by "integration", i.e. differentiation of the initial time series of order "d". The ARIMA (p,d,q) model contains ARIMA(p) autoregression process, with "d" standing for the order of differentiation, and MA(q) is the moving averages process. These models can be applied to non-stationary time series. In other words, ARIMA (p,d,q) represent an ARMA (p,q) model applied not to the original time series variable but to its transformed version obtained after "d" sequential differences.

The autoregression component of the ARIMA model is represented by the parameter "p", which determines the number of lags on the previous time series values. This component captures any autocorrelation in the data, which reflects how internally correlated the series is over time. The parameter "d" eliminates a trend component in the time series and makes it stationary (Mishev & Goev, 2010), i.e. the parameter "d" is equal to the

number of transformations required to make the time series a stationary one. Steady-state is crucial for time series modelling. Moving averages component introduces a relationship between the current value of data and past errors when forecasting, thus helping to capture short-term noise in time series. When modelling time series with ARIMA, calculation algorithms typically need to construct and estimate all meaningful combinations between p, d, and q. Most often, these values are between 0 and 3 for each of these parameters.

SARIMA models are an extension of non-seasonal ARIMA models designed for seasonality data analysis. These models combine AutoRegressive (AR), integrated (I) and Moving Average (MA) models with a seasonal component (s). The model is denoted by SARIMA(p,d,q) (P, D, Q)s, which model has four elements that are not part of the ARIMA model. "P" is a seasonal autoregressive order process AR(P), "D" is a seasonal order integration process, "Q" is a seasonal moving average MA(Q), and "s" is a number of observations over a seasonal period. If data is collected every quarter, then s=4; if it is collected monthly, s=12.

This method finds wide application in predicting time series. The common type of SARIMA (p,d,q)(P,D,Q),s is as follows (with d=0 and D=0):

$$y_t = c + \sum_{n=1}^{p} \alpha_n y_{t-n} + \sum_{n=1}^{q} \gamma_n \varepsilon_{t-n} + \sum_{n=1}^{P} \beta_n y_{t-sn} + \sum_{n=1}^{Q} \delta_n \varepsilon_{t-sn} + \xi_t$$

where:

y_t – the current value of the time series;

$y_{t-n}; y_{t-sn}$ – the preceding values of the time series;

$\alpha_n; \beta_n$ – parameters for the AR components of the model;

$\gamma_n; \delta_n$ – parameters for the MA components of the model;

$\varepsilon_{t-n}; \varepsilon_{t-sn}$ – members of the white noise error at time "t" in the non-seasonal and seasonal components of the model.

The seasonal component "s" in SARIMA refers to a recurring pattern in the data (seasonality). Whether the data are daily, monthly, quarterly or measured at any other regular interval, we can have recurring patterns in

these periodicals. Through SARIMA, the periodic pattern is identified and the seasonal component is modelled.

Seasonality can be said to be equal to the seasonal difference in the process of subtracting the data from the time series with a lag. This process helps remove the seasonal component and makes the data stationary, which in turn makes them easier to model.

Through the SARIMAX (Seasonal ARIMA with eXogenous variables) model, an exogenous (external) variable is added to the research time order. On the one hand, the inclusion of such a variable can lead to a more complete study of the changes in the time order, but on the other hand, since predictions are made for these external variables, it can increase the error in the final forecast of the research time order. SARIMAX adds a linear combination of exogenous variables to the SARIMA model. This allows us to model and predict the given time series under the influence of external variables on the future value of the timeline research.

The selection of the best model is based on the smallest value of the AIC or BIC criterion. After the models are built, diagnostic tests should be done. Charts can be constructed for standardized balance, histogram and excess score, normal Q-Q and correlogram.

The goal of data analysis is not just to find the model, but to fit the data in the best possible way by applying a technique that has the best predictive power. At the same time, the performance of a model depends not only on the modelling technique, but also on the characteristics of the data set on which to build the overall analysis. Therefore, data searching, extracting, sorting, cleaning, and its overall preparation are critical to the success of modelling and analysis.

In recent years, Python has emerged as a software application recommended for data analysis and machine learning. Using different libraries in Python allows data analysts to go through different stages of the data analysis and modelling process. When data is collected and has different formats, it is necessary to transform it to be prepared for further analysis. Cleaning, normalizing, and transforming the data is also an important part of preparing it for further analyses. Avoiding missing values, invalid data, or data outside a numerical range should also be considered.

There are a number of libraries in Python that deal with these problems well and relatively easily.

Python has several libraries that facilitate econometric modelling: SciPy, SciKit - learn, StatsModels. Anaconda is the environment that provides data analysis and machine learning for a large part of the used packages and libraries. Various interactive interfaces are included in Anaconda to facilitate programming through Python or R. Particularly suitable for statistical and econometric models programming by Python are Jupyter Lab and Jupyter Notebook. They allow the creation and sharing of documents that contain code, equations, formulas, notes, making them one of the most popular programming tools.

Python utilizes excellent libraries for visualizing through graphical images of the results of the modelling and the diagnostics. Such libraries are Matplotlib, Seaborn, Plotly and Pygal. In the process of constructing, evaluating, forecasting and validating models, the SciKit-Learning library is essential. SciKit-learn is a library part of the SciPy group and is related to the NumPy and SciPy libraries. Many of the methods that belong to machine learning have different characteristics that are determined by the nature of the data and the model to be build. By providing a rich collection of functions and modules, SciKit-Learn provides opportunities to analyze big data, explore complex problems in various fields with very good efficiency and ease.

Python's SciPy library is a set of modules that are designed for data analysis in various fields of science. SciPy builds on NumPy and contains modules for linear algebra, integration, optimization algorithms, etc.

StatsModels is the Python library that provides classes and functions for evaluating many different statistical and econometric models. This library provides a complement to SciPy for statistical calculations, estimates and inference for statistical and econometrics models. The Time Series Analysis module contains classes of models and functions to aid the analysis of data that originate from time series. This module includes descriptive statistics for time series, autocorrelation, partial autocorrelation function, and periodogram, as well as the relevant applications of ARMA for linear stochastic processes associated with these models. Methods for working with

autoregressive and moving average lag polynomials are included. The main models that are included in the Time Series Analysis module are Autoregressive models (AR), univariate Autoregressive Moving Average Models (ARMA), Autoregressive Integrated Moving-Average (ARIMA), etc.

Using Python libraries, code was compiled for the individual stages of applying ARIMA and SARIMA models. The data extraction and analysis is done through the libraries Numpy[1], Pandas[2], Matplotlib[3], SciKit-learn[4] and StatsModels[5].

4. Results of the SARIMA model

For the purposes of this study data on NVIDIA stock in USD measured on a daily basis for the period from 03.01.2023 to 17.03.2025 will be used[6]. NVIDIA is an American company founded in 1993. Its activities are related to the design of graphics processors and the manufacture of single-chip systems for the computer and automotive sectors. The company is publicly traded on the New York Stock Exchange (NYSE).

When working with data that are in time series format it is necessary to comply with the basic requirements for providing compatibility in the time span – the data must be comparable in terms of time range measurement, method of calculation of the indicator, etc. When using Python for data analysis, the first task is to extract the data from the respective database or web page (Figure 1). The data should also be prepared in a tabular form according to the requirements of the analysis methods. Cleaning and transforming the data is also an important part of preparing it for subsequent analysis by Python software: missing values, invalid data, or data outside a numerical range should be avoided.

The data are loaded within the Python environment (Figure 2). Typically, plotting the chart from time series data provides visual insights if there is a trend and periodic patterns (Figure 3). By looking at the closing prices of Figure 9, we can easily observe a systematic upward motion of the mean level with possible periodicity over the chosen period of time. This justifies intuitively that it makes sense to apply the SARIMA model to data that shows both non-stationarity and seasonality.

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import statsmodels.api as sm
from statsmodels.tsa.statespace.sarimax import SARIMAX
from statsmodels.tsa.stattools import adfuller
from statsmodels.graphics.tsaplots import plot_acf,
plot_pacf
from sklearn.metrics import mean_absolute_error,
mean_squared_error
import pmdarima as pm
%matplotlib inline
from matplotlib.pylab import rcParams
import warnings
warnings.filterwarnings('ignore')
from statsmodels.tsa.stattools import adfuller
```

Figure 1. Loading Python libraries (Source: The author)

```
from pandas import ExcelWriter
from pandas import ExcelFile
nv_data = pd.read_csv("D:/Data/nvidia_data.csv", sep=";")
nv_data["Date"] = pd.to_datetime(nv_data['Date'])
```

Figure 2. Reading the data (Source: The author)

```
plt.figure(figsize=(10, 5))
plt.plot(nv_data1['Price'], linewidth=3,c='green')
plt.title('Price of NVIDIA Stock')
plt.xlabel('Date')
plt.ylabel('Price')
plt.show()
```

Figure 3. Data graph (Source: The author)

Another way to identify seasonality is to use time series decomposition. It divides the dynamic line into three main components: trend-cyclical component, seasonal component, and error component (residuals). A trend is the long-term change in the mean level (increases or decreases) over time. A seasonal component is a recurring fluctuation that occurs over the period of time. Residuals capture any irregularity that cannot be explained by the trend or the seasonal component.

Using the seasonal decomposition in the Statsmodels library of Python, we can decompose NVIDIA's stock price time series (Figure 4).

```
import pandas as pd
import matplotlib.pyplot as plt
from statsmodels.tsa.seasonal import seasonal_decompose
from pylab import rcParams
rcParams['figure.figsize'] = 6, 5
decompose = seasonal_decompose(nv_data1['Price'],
model='additive', period=22)
decompose.plot()
plt.show()
```

Figure 4. Seasonal decompose (Source: The author)

The first chart presents the observed data on the closing price of NVIDIA's shares (Figure 10), the second chart depicts the trend component which shows that the stock price is systematically increasing over the time period of study. The third chart presents the seasonal component, where we can clearly see a repeating pattern over time, and the last chart depicts the residuals which are variations in the data that cannot be explained by the trend-cyclical and seasonal components. The seasonal component graph shows that the use of the SARIMA model is appropriate for the analysis in this case.

The next step is to check for the stationarity of the time series of NVIDIA's stock prices. As we mentioned above regarding the existence of a systematic trend component, the examined time series is expected to be non-stationary. We verify this using the ADF test (Figure 5).

```
def check_stationarity(timeseries):
    result = adfuller(timeseries, autolag='AIC')
    p_value = result[1]
    print(f'ADF Statistic: {result[0]}')
    print(f'p-value: {p_value}')
    print('Stationary' if p_value < 0.05 else 'Non-Stationary')
check_stationarity(nv_data1['Price'])
plot_acf(nv_data1['Price'])
plot_pacf(nv_data1['Price'])
plt.show()
```

Figure 5. Perform the Dickey-Fuller test (Source: The author)

Since ADF statistics of -1.0557 is not large enough (in absolute value) and the p-value (0.73) is greater than 0.05, we cannot reject the null hypothesis and therefore the time series for the price of NVIDIA is non-stationary. Therefore, we need to apply transformation to make it

stationary. Typically, differencing is used to remove the effect of the stochastic trend and stabilize the series mean level (Nokeri, 2021).

To identify the parameters of the SARIMA (p, d, q) model, it is recommended to follow the Box-Jenkins methodology (Box et al. 1976) and build the autocorrelation function (ACF) and the partial autocorrelation function (PACF). The next step is to draw the ACF and see if there is an autocorrelation and if the coefficients suddenly become irrelevant after some delay (Mishev & Goev, 2010).

```
SARIMAX_model = pm.auto_arima(time_series[['Price']],
                exogenous=[time_series.index.weekday],
                                start_p=1, start_q=1,
                                test='adf',
                                max_p=3, max_q=3, m=22,
                                start_P=0, seasonal=True,
                                d=None, D=1)
best_order = SARIMAX_model.order
best_seasonal_order = SARIMAX_model.seasonal_order
```

Figure 6. Selection of the most suitable parameters of the SARIMA model (Source: The author)

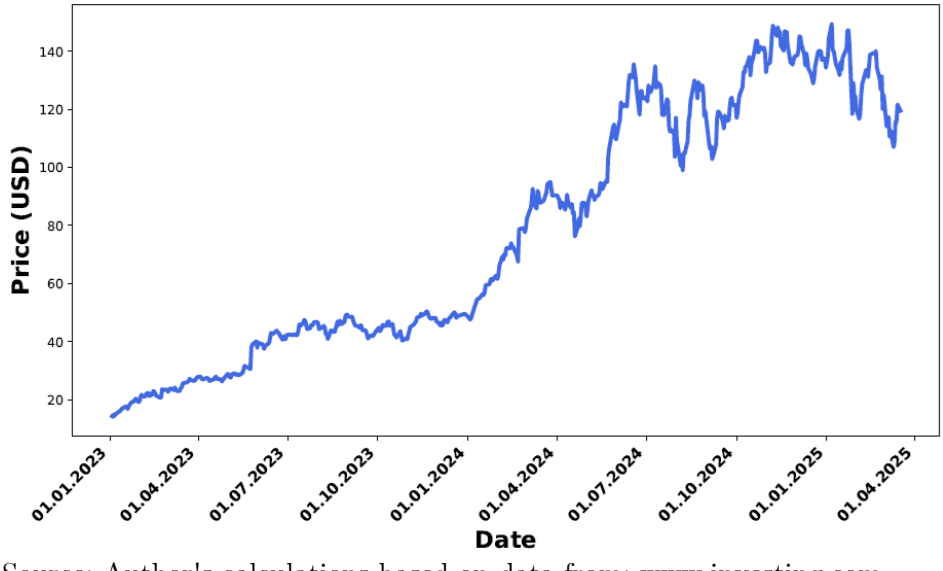
```
p, d, q = 3, 0, 0
P, D, Q, s = 0, 1, 1, 22
model = SARIMAX(nv_data1['Price'], order=(p, d, q),
                                seasonal_order=(P, D, Q, s))
results = model.fit()
print(results.summary())
forecast = results.forecast(steps=22)
print(forecast)
results.plot_diagnostics(figsize=(15, 12))
plt.show()
```

Figure 7. Data evaluation and forecasting with the SARIMA model (Source: The author)

Most peaks in the ACF diagram are statistically significant (Figure 11): the coefficients capturing the lagged effects are outside the 95 percent boundary. The PACF graph shows that there is a significant peak in lag 1 and lag 2. Therefore, lags can explain any higher-order autocorrelation. There is a very high positive partial autocorrelation up to lag 2 after which it is close to 0.

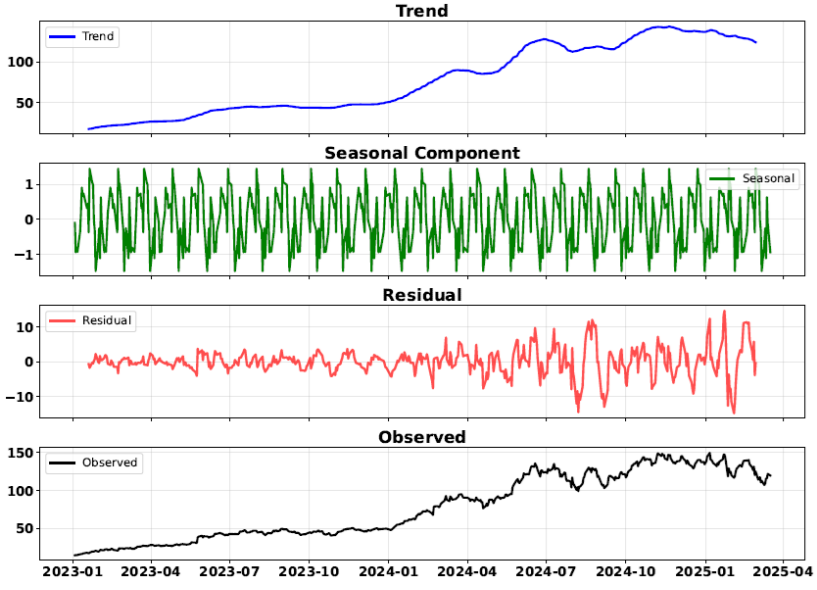
```
forecast_periods = 20
forecast = results.get_forecast(steps=forecast_periods)
forecast_mean = forecast.predicted_mean
forecast_ci = forecast.conf_int()
plt.figure(figsize=(6, 4))
plt.plot(forecast_mean.index, forecast_mean,
                label='Forecast', color='red')
plt.fill_between(forecast_ci.index, forecast_ci.iloc[:, 0],
            forecast_ci.iloc[:, 1], color='pink', alpha=0.3)
plt.title("Price Forecast")
plt.xlabel("Date")
plt.ylabel("Price")
plt.legend()
plt.show()
```

Figure 8. Graphical representation of forecast data with the SARIMA model (Source: The author)



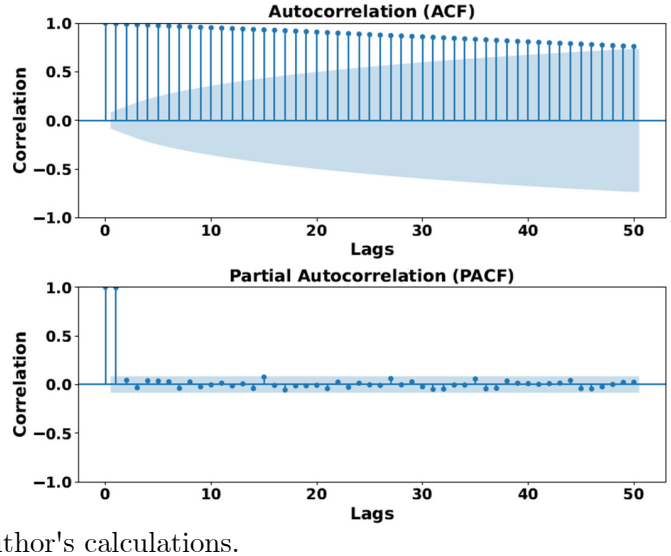
Source: Author's calculations based on data from: www.investing.com

Figure 9. NVIDIA USD price dynamics for the period from 03.01.2023 to 17.03.2025



Source: Author's calculations.

Figure 10. A breakdown of NVIDIA's share price data



Source: Author's calculations.

Figure 11. The Autocorrelation function (ACF) and the Partial autocorrelation function (PACF) of NVIDIA price

Different combinations of parameters (p, q, P, Q) of the SARIMA models have been tested. For the empirical study, a code was compiled for automated model selection based on the parameter constraints of the ARIMA models (Figure 6). For the "s" parameter, the values of 5 (five working days in a week), 20 and 22 (working days in a month) were set. After testing the code, the best model is SARIMA (3,0,0)(0,1,1)22.

The Akaike Information Criterion (AIC) was used to evaluate and compare the configuration of the parameters of the models. The model with the lowest AIC value is considered to have the best performance. The graph of residuals (Figure 13) over time shows a non-constant dispersion, which is an inconsistency compared to the expected white noise. The second graph shows the distribution of residuals, which is relatively close to the normal distribution. Diagram Q-Q leads us to the same conclusion, as it shows a line that is relatively straight, which means that the distribution of residuals resembles the normal distribution. Looking at the correlogram at the bottom right, we see that the coefficient appears to be significant at lag 1, and all other coefficients are insignificant (around 0) just like white noise.

Dep. Variable:	Price	No. Observations:	552
Model:	SARIMAX(3, 0, 0)x(0, 1, [1], 22)	Log Likelihood	-1361.438
Date:	Wed, 16 Apr 2025	AIC	2732.875
Time:	10:24:38	BIC	2754.240
Sample:	0	HQIC	2741.238
	- 552		
Covariance Type:	opg		

	coef	std err	z	P>|z|	[0.025	0.975]
ar.L1	0.9168	0.031	30.039	0.000	0.857	0.977
ar.L2	0.1747	0.050	3.524	0.000	0.078	0.272
ar.L3	-0.0942	0.035	-2.716	0.007	-0.162	-0.026
ma.S.L22	-0.9022	0.027	-33.987	0.000	-0.954	-0.850
sigma2	9.2747	0.265	35.022	0.000	8.756	9.794

Ljung-Box (L1) (Q):	0.02	Jarque-Bera (JB):	2757.15
Prob(Q):	0.88	Prob(JB):	0.00
Heteroskedasticity (H):	16.12	Skew:	-1.24
Prob(H) (two-sided):	0.00	Kurtosis:	13.90

Source: Author's calculations.

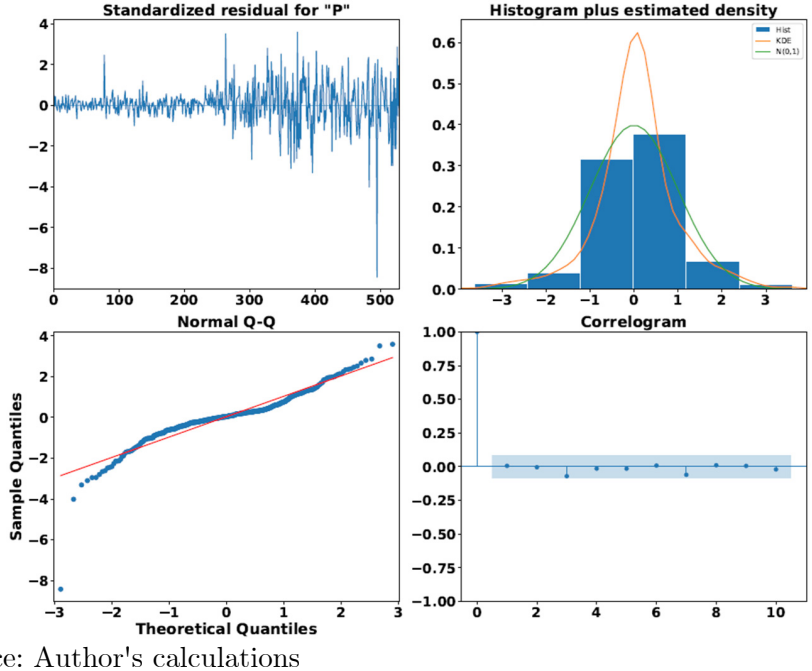
Figure 12. Results from NVIDIA's SARIMA pricing model

Using the estimated SARIMA (3,0,0) (0,1,1)22 specification for the price model (Figure 7 and 12), future NVIDIA's closing price values can be predicted. Figure 8 and 14 and Table 1 shows future closing price values after March 17, 2025.

Table 1. The future and actual values of NVIDIA's closing stock prices

Data	Predicted value	Actual value[6]
18.3.2025	120.63	119.53
19.3.2025	120.69	115.43
20.3.2025	121.75	117.52
21.3.2025	122.26	118.53
24.3.2025	123.40	117.70
25.3.2025	123.00	121.41
26.3.2025	123.14	120.69

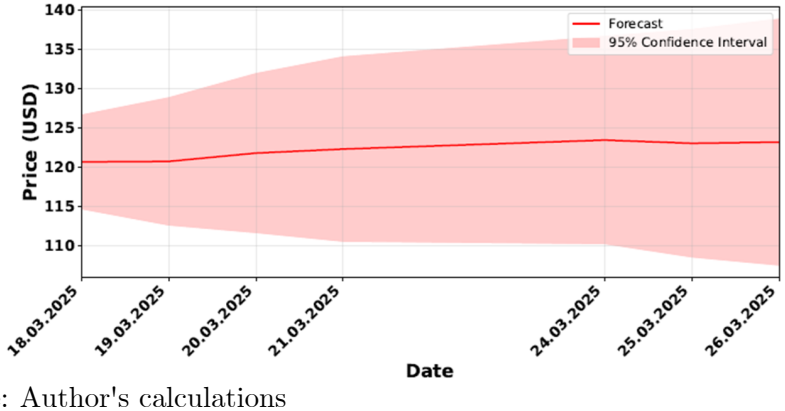
Source: Author's calculations



Source: Author's calculations

Figure 13. Diagnostic Tests for SARIMA(3,0,0) (0,1,1) 22

The analyzed SARIMA model shows relatively accurate predictions, with slightly larger deviations during some periods.



Source: Author's calculations

Figure 14. Estimated closing price values of NVIDIA's common stock (3,0,0) (0,1,1)22 model

5. Discussion

In this study, we proposed a SARIMA model to predict the prices of NVIDIA shares traded on the NYSE. Using Python, the data for NVIDIA were extracted and loaded, and different libraries were then used. To determine the order of the model, we first analysed the dynamics of the data from the time series of the stock prices. We decomposed the dynamics of the stock to determine whether the timeline contained a trend and a seasonal component, and checked for stationarity. We then identified the most appropriate values for the order of the autoregression and seasonal components of the SARIMA model. Through a series of experiments involving different combinations of parameters and the creation of code in Python, we chose their best combination. In this article, we have shown that predicting the time series of the closing prices of NVIDIA shares on the NYSE can be successfully accomplished using the SARIMA models. Our results showed that the SARIMA model specification (3,0,0) (0,1,1)22 can be considered reliable for predicting the closing prices of NVIDIA shares. Using Python to forecast data on stock prices in financial markets, exchange rates and other financial indicators proves to be effective and acceptable. However, when using big data on these indicators, there is a delay in getting

results because of some specific or unintended consequences that affect the SARIMA models.

The SARIMA model captures seasonality and trends in past data values to predict future values. This model helps in making informed decisions related to currency trading, financial instruments and international business operations (Tamizharasi et al., 2024; Benayad & Halimi, 2022; Ariyanti & Yusnitasari, 2023). ARIMA and SARIMA models can be successfully used to forecast a range of economic processes, such as unemployment (Wanjuki et al., 2021), inflation (Gikungu et al., 2015), interest rates, food prices, crude oil (Ariyanti & Yusnitasari, 2023), etc.

The accuracy of forecasts based on ARIMA models is limited and depends on several factors, including the quality of the data, the stability of the phenomena and markets being studied (currency, financial or commodity). The difference between the forecasted and actual values can increase significantly. This may be due to the researchers' judgment on what time span to include when constructing the models. As shown in this study and in others, the findings confirm that SARIMA models, although they do not consider external factors, still provide relatively accurate forecasts and the models can be used as a tool for short-term decision-making (Divisekara, et al., 2020; Lee & Bang, 2024).

The research in this article is important because it demonstrates the capabilities of Python and the econometric technique of ARIMA models for estimating and predicting future values of certain phenomena and processes. The presented algorithm and code can be applied to different datasets, serving both for training and for forecasting purposes by other researchers.

6. Conclusions

Predicting business phenomena is still one of the biggest challenges for empirical analyses. On the one hand, predicting business processes based on time series data becomes relatively difficult, as their dynamics are affected by various factors. Some of the factors are unpredictable in terms of their origin and nature. For example, very often, in addition to economic factors, political factors can also have an impact and then the accuracy of the forecasts may decline. However, forecasting methods have become increasingly precise in recent years. The use of Python as a programming

language by data analysts has expanded when looking for open-source software to analyse big data, such as the data from stock and currency markets.

The effectiveness of the models studied may vary depending on the specific tasks. It is recommended to build and experiment different models before assessing which of them performs best in forecasting. From researching and forecasting the closing price of NVIDIA's shares, it can be concluded that the model has its unique strengths. The forecasts are close to the actual values for the forecast period, but still there is room for improvement. Future research could also include supplementing the study with the use of external factors to assist in the forecasting of NVIDIA's stock price. Improving forecasts could be achieved by including exogenous factors and testing SARIMAX models. In addition, consideration could be given to the delayed effects of these factors.

Finally, this article aims to showcase the methodology and practical application of ARIMA models implemented in Python, with the key intention of supporting educators who teach students and PhD researchers conducting empirical academic research. Furthermore, the approach to deploying such models can be effectively utilized in computer science training for high school students, providing a valuable framework for forecasting time series data.

NOTES

1. https://numpy.org/doc/stable/
2. https://pandas.pydata.org/docs/
3. https://matplotlib.org/stable/index.html
4. https://scikit-learn.org/stable/
5. https://www.statsmodels.org/stable/index.html
6. https://www.investing.com/equities/nvidia-corp-historical-data

REFERENCES

Acula, D., & Guzman, T. (2020). Application of enhanced hidden Markov model in stock price prediction. *Journal of Modeling and Simulation of Materials*, *3*(1), 70 – 78. https://doi.org/10.21467/jmsm.3.1

Adineh, A., Narimani, Z., & Satapathy, S. (2021). Importance of data preprocessing in time series prediction using SARIMA: a case study. *International Journal of Knowledge-Based and Intelligent Engineering Systems*, *24*(4), 331 – 342. https://doi.org/10.3233/kes-200065

Albeladi, K., Zafar, B., & Mueen, A. (2024). A novel deep-learning based approach for time series forecasting using SARIMA, Neural Prophet and Fb Prophet. *Journal of Management & Technology*, *24*, 72 – 86. https://doi.org/10.20397/2177-6652/2024.v24iSpecial.2818

Alqatawna, A., Abu-Salih, B., Obeid, N., & Almiani, M. (2023). Incorporating time-series forecasting techniques to predict logistics companies' staffing needs and order volume. *Computation*, *11*(7). https://doi.org/10.3390/computation11070141

Ariyanti, V., & Yusnitasari, T. (2023). Comparison of ARIMA and SARIMA for forecasting crude oil prices. *Jurnal Resti (Rekayasa Sistem Dan Teknologi Informasi)*, *7*(2), 405 – 413. https://doi.org/10.29207/resti.v7i2.4895

Benayad, W., & Halimi, W. (2022). Forecasting Algeria's unemployment rates using SARIMA model in Python programming during 2001-2021. *Forum for Economic Studies and Research Journal*, *01*, 586 – 600.

Box, G., Jenkins, G., & Reinsel, G. (1976). *Time Series Analysis: Forecasting and Control*. New York: John Wiley and Sons Inc.

Divisekara, R., Jayasinghe, G., & Kumari, K. (2020). Forecasting the red lentils commodity market price using SARIMA models. *SN Business & Economics*, *1*(1), 56 – 63. https://doi.org/10.1007/s43546-020-00020-x

Gikungu, W., Waititu, A., & Kihoro, J. (2015). Forecasting inflation rate in Kenya using SARIMA model. *American Journal of Theoretical and Applied Statistics*, *4*(1), 15 – 18. https://doi.org/10.11648/j.ajtas.20150401.13

Kolková, A., & Navrátil, M. (2021). Demand Forecasting in Python: Deep Learning Model Based on LSTM Architecture versus Statistical Models, *Acta Polytechnica Hungarica*, *1*, №8, 123 – 141. https://acta.uni-obuda.hu/Kolkova_Navratil_115.pdf

Lee, G., & Bang, J. (2024). Forecasting container throughput of Singapore port considering various exogenous variables based on SARIMAX

models. *Forecasting*, *6*(3), 748 – 760.
https://doi.org/10.3390/forecast6030038

Mishev, G., & Goev, V. (2010). Statistical Analysis of Time Series. Sofia: *Avangard Prima*.

Nokeri, T. (2021). Implementing Machine Learning for Finance. New York *Apress*.

Peixero, M. (2021). Time Series Forecasting in Python. New York. *Manning Publications*.

Tamizharasi D., Bung, P., & Jahnavi, M. (2024). Predicting exchange rate between the us dollar (USD) and Indian rupee (INR): an empirical analysis using SARIMA model. *International Journal of Research in Finance and Management*, *7*(1), 56 – 63.
https://doi.org/10.33545/26175754.2024.v7.i1a.283

Wanjuki, T., Wagala, A., & Muriithi, D. (2021). Forecasting commodity price index of food and beverages in Kenya using Seasonal Autoregressive Integrated Moving Average (SARIMA) models. *European Journal of Mathematics and Statistics*, *2*(6), 50 – 63.
https://doi.org/10.24018/ejmath.2021.2.6.80

✉ **Dr. Vesela Dimitrova, Chief Assist. Prof.**
ORCID iD: 0009-0003-3004-6819
University of National and World Economy,
Sofia, Bulgaria
E-mail: vesela.dimitrova@unwe.bg