Times Series Project

Data Opening Prices of Google Analysis

Report Due on Monday, January 13th, 2025

Students :

Tom Brault

Antoine Jarry

Marceau Le Bot

Alexandre Maghames

Gaël Mallick

Teacher :

Youssef Esstafa

ENSAI – National School for Statistics and Data Analysis

Academic Year 2024 – 2025

Acknowledgements

We would like to express our gratitude to everyone who contributed, both directly and indirectly, to the completion of this report.

Our sincere thanks go to Professor Youssef Esstafa, for his valuable guidance and input throughout the Time Series course at ENSAI.

This report is the result of collaborative effort and thoughtful mentoring. We are grateful to all those who made this experience both enriching and formative.

Abstract

This report is developed as part of the time series course for third-year students at ENSAI, taught by Professor Youssef Esstafa. The project aims to explore and implement various time series modeling techniques studied during the course. We note that all the code for the project is available and can be executed from the repository on `Github` [1].

The central focus of this project is the analysis and forecasting of Alphabet Inc.'s stock prices (GOOGL), using financial data obtained from Yahoo Finance [2]. By leveraging these real-world data, we apply theoretical knowledge to uncover temporal patterns, trends, seasonality, and anomalies.

The methodologies explored include classical statistical models, such as Autoregressive Integrated Moving Average (ARIMA) model and Generalized Autoregressive Conditional Heteroskedasticity (GARCH) model, alongside modern machine learning approaches such as Long Short-Term Memory (LSTM). The purpose of the project is to formalize our understanding of time series modeling, enhance practical implementation skills, and derive actionable insights from financial data.

1. https://github.com/MarceauLB/SeriesTemp.git
2. https://fr.finance.yahoo.com/quote/GOOGL/profile/

Contents

List of Figures

List of Tables

Acronyms

— ACF : Autocorrelation Function
— ADF : Augmented Dickey-Fuller
— AIC : Akaike Information Criterion
— ARIMA : Autoregressive Integrated Moving Average
— BIC : Bayesian Information Criterion
— FARIMA : Fractionally Autoregressive Integrated Moving Average
— GARCH : Generalized Autoregressive Conditional Heteroskedasticity
— KPSS : Kwiatkowski-Phillips-Schmidt-Shin
— LSTM : Long Short-Term Memory
— MAE : Mean Absolute Error
— MA : Moving Average
— MSE : Mean Squared Error
— PACF : Partial Autocorrelation Function
— RMSE : Root Mean Squared Error
— RSI : Relative Strength Index

Introduction

The prediction of a stock's price is a fundamental question in trading. The objective of this financial activity is to buy and sell assets (stocks, cryptocurrencies, commodities, currencies, etc.) on various financial markets to generate profit. In this context, traders speculate on price fluctuations using graphical analysis, indicators such as moving averages or the RSI (Relative Strength Index), or observable trends. However, financial markets, influenced by geopolitical, economic, and social factors, are often unpredictable, which makes forecasting complex.

This issue is all the more relevant today as technological advances, particularly in artificial intelligence, now enable the exploitation of massive volumes of data to better understand market dynamics. Google's stock, as a major player in the technological sphere, constitutes a pertinent subject of study due to its volatility, an important characteristic to understand about a stock.

This study aims to examine, over different time horizons, the prediction of Google's stock price in 2018 using data from previous years. Following a presentation of the collected data, their contents, and their distribution, various models will be implemented, such as ARIMA, GARCH, and LSTM. The ultimate objective of this study will be to determine which of these models provides the most accurate prediction of the stock price. To achieve this, different metrics, presented within this document, will be used to evaluate which prediction aligns most closely, according to each metric, with the stock's actual value.

1 Time series presentation

1.1 Notations

In this subsection, we briefly present some notations that will appear in the following sections.

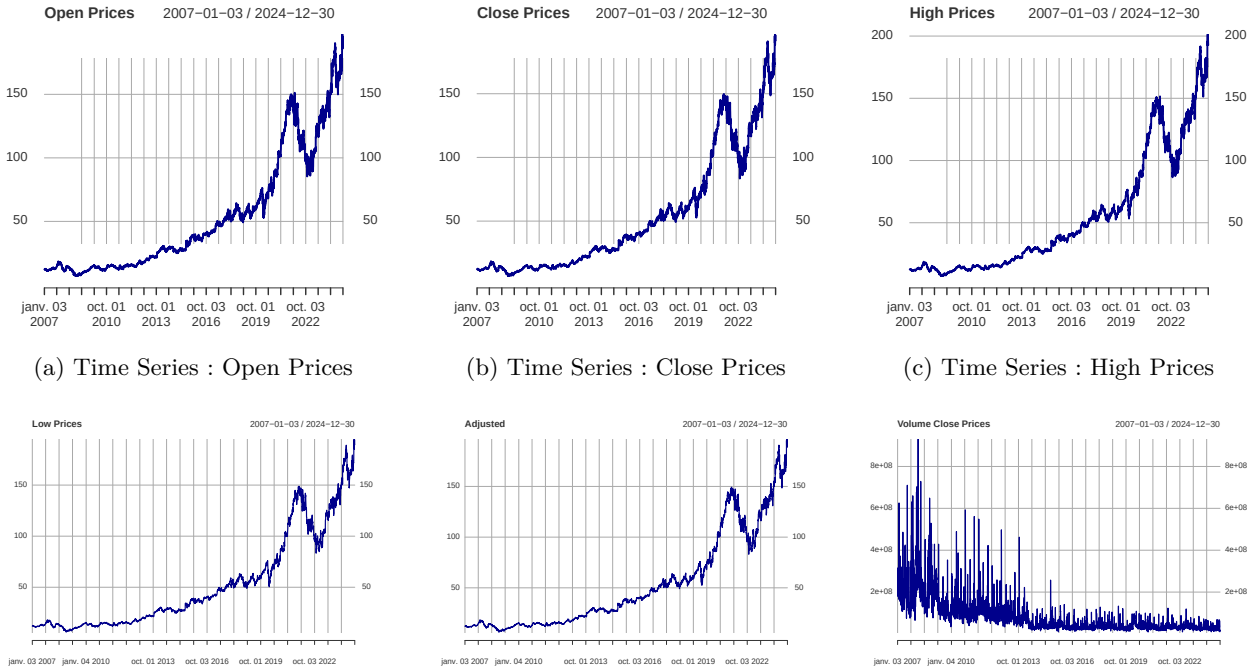
- x_t : The observed value of the time series at time t.
- $\tilde{x}_t = \ln(x_t)$: The logarithmic transformation of the time series at time t.
- $y_t = \Delta\tilde{x}_t = \tilde{x}_t - \tilde{x}_{t-1} = \ln(x_t) - \ln(x_{t-1}) = \ln\left(\frac{x_t}{x_{t-1}}\right)$: the first-order differenced series.

1.2 General presentation

The dataset was loaded [3] into `R` using the `quantmod` (Ryan et al. 2024) package. Data was retrieved for the period from January 1, 2007, to December 31, 2024. This timeframe includes 4.529 samples, with no missing values. These samples are available for each working day during the considered time period [4]. The corresponding dataframe contains the following features :

— **Date** : The date of the stock price observation.
— **Open** : The stock's opening price on that date.
— **High** : The highest price reached during the trading day.
— **Low** : The lowest price reached during the trading day.
— **Close** : The stock's closing price on that date.
— **Adjusted** : The stock's adjusted closing price on that date
— **Volume** : The trading volume (number of shares traded) on that date.

Figure 1 below presents the time series for each available feature in the dataset.



(a) Time Series : Open Prices (b) Time Series : Close Prices (c) Time Series : High Prices

(d) Time Series : Low Prices (e) Time Series : Adjusted Prices (f) Time Series : Volume

FIGURE 1 – Time Series for Each Available Feature

3. `getSymbols("GOOGL", src = "yahoo", from = "2007-01-01", to = "2024-12-31")`
4. On average, there were 251.6 working days per year for this 18-year timeframe.

As we can see, except the volume time series, the other time series (open, close, high, low, adjusted) exhibit similar patterns and trends. For the remainder of the report, we will focus only on the opening prices time series for models construction and predictions. This choice is made for simplicity and clarity, but it is important to note that all subsequent analyses could have been conducted on any of the other five series.

In the following of this report, we will focus on the time series data from January 1, 2007, to December 31, 2018. This timeframe allows us to exclude the years after 2018, which include periods of instability, notably the COVID-19 crisis, from our study, model construction, and predictions. For being more precise, we will further divide our dataset into two parts : a training period from January 1, 2007, to December 31, 2017, and a testing period for the year 2018 (January 1, 2018, to December 31, 2018). The 2018 year data will be used to evaluate the performance and quality of the different developed models.

The following figure (2) presents the time series data for the specified period under consideration.

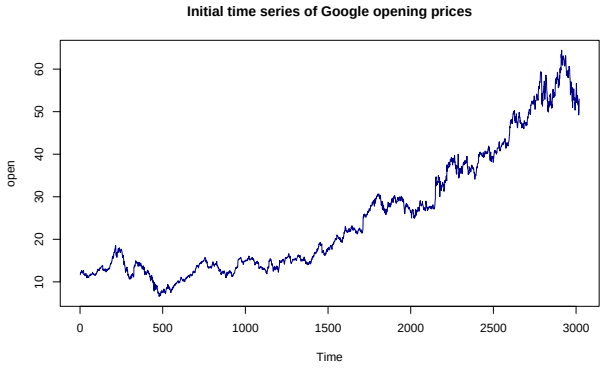


FIGURE 2 – Time Series of Google opening prices (from January 1, 2007 to December 31, 2018)

Based on this plot, we can give two important statements :

— First of all, the time series of opening prices looks wigglier over time, suggesting an increase in volatility during the considered timeframe.

— Moreover, it seems clear that the time series has a global increasing trend. Therefore, it looks necessary to make the series stationary.

Those observations will be the foundations of the next analyses.

1.3 Smoothing of the Time Series

In the following sections of this report, we will consider the log-transformed time series, denoted as $\ln(x_t)$. Taking the logarithm of the series helps stabilize its variance and reduces the impact of large fluctuations.

This transformation also allows underlying patterns in the data to become clearer. Formally, the transformation can be written as :

$$\forall t \in [\![t_{\min}, t_{\max}]\!], \tilde{x}_t = \ln(x_t)$$

Figure 3 to the right illustrates the application of the logarithmic transformation to the original time series, applied to both the training and testing datasets (from January 1, 2007, to December 31, 2018).

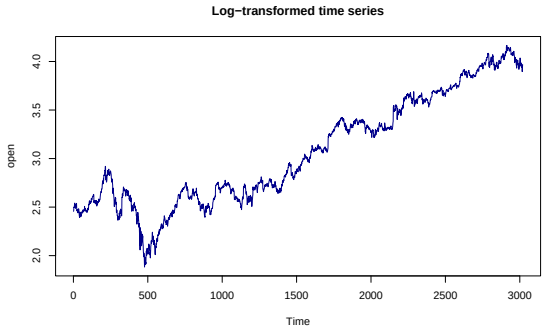
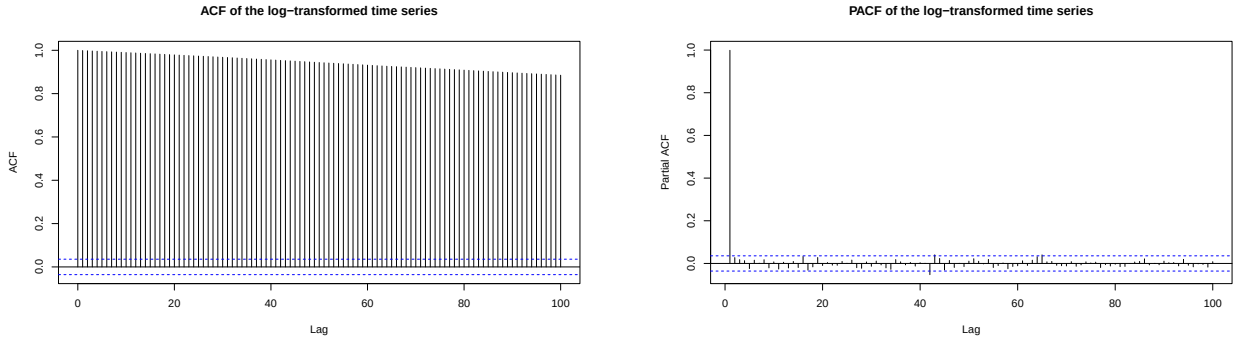


FIGURE 3 – Logarithmic transformation of the time series.

1.4 Time Series Stationarity

A preliminary examination of the time series suggests that the series is not stationary.

Indeed, figure 4 presents both the Autocorrelation Function (ACF) (figure 4a) and the Partial Autocorrelation Function (PACF) (figure 4b) for the log-transformed time series. The ACF is decreasing very slowly and exhibits a long memory effect, which is a typical characteristic of non-stationary time series.



(a) ACF Time Series

(b) PACF Time Series

FIGURE 4 – ACF and PACF for Time Series

To formally test the stationarity of the time series, we can apply one of the following tests : the Augmented Dickey-Fuller (ADF) test or the Kwiatkowski-Phillips-Schmidt-Shin (KPSS) test. These tests will allow us to confirm whether the series exhibits stationarity. In this report, we focus on the ADF test.

- **Test Hypothesis :** $(\mathcal{H}_0)$: The time series is not stationary (existence of a unit root), versus $(\mathcal{H}_1)$: The time series is stationary.
- **Statistical Test :** $DF = -2.9083$
- $p-$**value :** The $p-$value $= 0.1938$. Obtained from the test, it indicates the strength of evidence against the alternative hypothesis.
- **Test Conclusion :** Based on the $p-$value, we cannot reject the null hypothesis $(\mathcal{H}_0)$ (with a level of confidence of 95%) and thus conclude that the series is not stationary.

Since the time series is not stationary, we applied a first-order differencing transformation to remove trends and achieve stationarity. Formally, the differenced series is given by :

$$\begin{aligned} \forall t \in [\![t_{\min}+1, t_{\max}]\!], y_t &= \Delta \tilde{x}_t \\ &= \ln(x_t) - \ln(x_{t-1}) \\ &= \ln\left(\frac{x_t}{x_{t-1}}\right) \end{aligned}$$

Figure 5 to the right illustrates the result of this transformation on the time series.

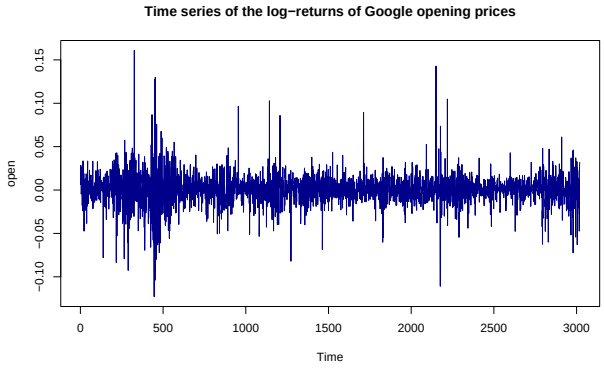
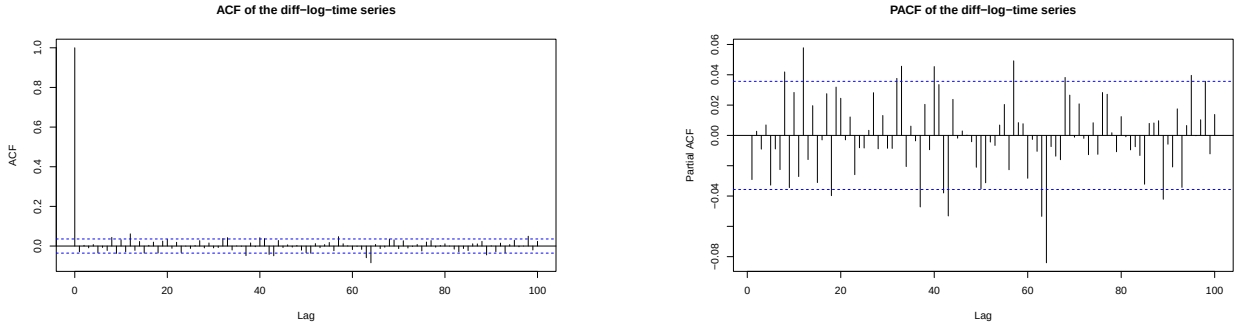


FIGURE 5 – First-order differenced time series.

We regenerated two plots of the ACF (figure 6a) and PACF (figure 6b) for the differenced log-time series to ensure that the ACF decreases exponentially. The results seem satisfactory for the ACF, suggesting that the time series is now stationary. By applying the Augmented Dickey-Fuller test to the differenced time series, we obtained a test statistic value of $DF = -14.211$, with a corresponding $p-$value of 0.01. Given that the

$p-$value is less than the typical significance level of 0.05, we can reject the null hypothesis and conclude that the time series is stationary.



(a) ACF Time Series

(b) PACF Time Series

FIGURE 6 – ACF and PACF for Time Series

Figure 6 provides valuable insights for constructing models in the subsequent section. With the time series now prepared for analysis, we can split the data into two subsets : a training dataset and a testing dataset, both based on the log-differenced time series. The training dataset will be used to develop and train the models, while the testing dataset will assess their performance. The data will be divided as follows :

- `Training`: January 1, 2007, to December 31, 2017
- `Testing`: January 1, 2018, to December 31, 2018

1.5 Some relevant features

Before training the different models, a relevant step is the creation of various indicators that may be useful in predicting stock prices. Therefore, we will define several indicators based on the available data.

1.5.1 Moving Average (MA)

The first indicator created is the moving average. Here is how this indicator is defined :

Definition 1.1 (Moving Average (MA))

A moving average (MA) is defined by :

$$MA_t(n) = \frac{1}{n}\sum_{i=1}^{n-1} P_{t-i}$$

where :

— P_{t-i} is the opening price at period $t-i$,

— n is the chosen period for the moving average.

This indicator is widely used to smooth data and identify trends in stock prices. By averaging prices over a specific period, it reduces daily fluctuations and highlights long-term trends. A price above the moving average may indicate an upward trend, while a price below may signal a downward trend. Comparing moving averages over different periods can also help identify potential buy or sell opportunities.

1.5.2 Relative Strength Index (RSI)

Another indicator, slightly more complex, is the Relative Strength Index (RSI).

Definition 1.2 (Relative Strength Index (RSI))

The Relative Strength Index (RSI) is defined, for a fixed period n, by :

$$RSI = 100 - \frac{100}{1 + RS}$$

where :

— RS is the ratio of the average gains to the average losses over a specified period n.

This indicator is used to predict a potential decline or increase in the stock price. Indeed, the higher the RSI, the more it may indicate an overbought condition, anticipating a possible decrease in the stock price. Conversely, when it is too low, it could signal a potential increase in the stock price. This indicator ranges between 0 and 100, and a value around 50 does not specifically allow for any assumptions to be made.

1.5.3 Bollinger Bands

The last indicators are the Bollinger Bands.

Definition 1.3 (Bollinger Bands)

The Bollinger Bands are defined by :

$$\text{Upper Band} = MA_t(n) + k \times \sigma(n) \quad \text{and} \quad \text{Lower Band} = MA_t(n) - k \times \sigma(n)$$

where :

— $MA_t(n)$ is a moving average,

— k is a hyperparameter,

— $\sigma(n)$ is the standard deviation of prices over the same period n.

When the price touches or exceeds the upper Bollinger Band, it suggests that the asset may be overbought, potentially signaling a forthcoming price decline. Conversely, if the price touches or drops below the lower band, it indicates the asset might be oversold, which could lead to a potential price increase. The space between the upper and lower bands reflects the asset's volatility. A wider gap suggests higher volatility, while a small gap indicates lower volatility, providing valuable insights into the market's overall movement.

1.5.4 Visualizing the indicators

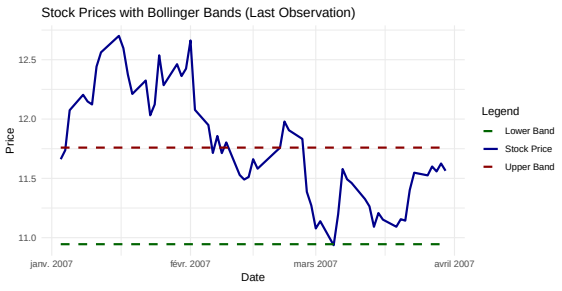


FIGURE 7 – Bollinger Bands and Stock Price Time Series from January to April 2007

The figure 7 allows us to visualize the previously defined Bollinger bands. These were calculated over a 14-day period and correspond to the bands for the last observation, on March 30, 2007. Thus, if the stock price approaches either of the two bands in early April, it may indicate a potential upward or downward trend. Additionally, the moving average of this series over the last 14 days is 11.35, while the RSI indicator is 53.7, which suggests that we are not in an overbought or oversold situation.

Now that all the data has been collected and processed, it is possible to proceed with training several time series models on this data.

2 Construction of different models

In this section 2, we will focus on the construction of different models.

2.1 Autoregressive Integrated Moving Average (ARIMA)

2.1.1 Model presentation

Initially, this study will focus on ARIMA models. The ARIMA model, which stands for AutoRegressive Integrated Moving Average, is a widely used statistical method for time series forecasting. This model is especially effective for univariate time series data that exhibits patterns and trends, as long as the data is stationary or can be transformed to become stationary, which applies in this case.

Definition 2.1 ($ARIMA(p,d,q)$ process)

An autoregressive integrated moving average process (X_t) of orders p, d, and q, denoted by $ARIMA(p,d,q)$, is defined by :

$$\left(I - \sum_{i=1}^{p} \alpha_i B^i\right)(I-B)^d X_t = \mu + \varepsilon_t + \sum_{j=1}^{q} \theta_j B^j \varepsilon_t$$

where :

- $\left(I - \sum_{i=1}^{p} \alpha_i B^i\right)$ corresponds to the AutoRegressive (AR) part,
- $(I-B)^d$ corresponds to the Integrated (I) part,
- μ corresponds to the mean value of the process,
- (ε_t) corresponds to a white noise process,
- $\sum_{j=1}^{q} \theta_j B^j$ corresponds to the Moving Average (MA) part.

In our case, we showed that one difference was necessary to make the series stationary. Thus $d = 1$.

2.1.2 Statistical tools

In this section, we will introduce various statistical tools that will be used in the rest of the study.

First of all, the Box-Pierce test is a statistical method used to evaluate the hypothesis of independence of residuals in a time series. More specifically, it checks whether the autocorrelations of the residuals up to a certain lag, denoted by h, are zero, which is a key assumption in time series models such as ARIMA.

Definition 2.2 (Box-Pierce Test)

- **Test Hypothesis :** $(\mathcal{H}_0)$: The autocorrelations of residuals are zero for all lags up to h, implying that the residuals are independent, versus $(\mathcal{H}_1)$: at least one of the autocorrelations up to h is non-zero.
- **Statistical Test :** The test statistic Q is calculated as follows :

$$Q = n \sum_{k=1}^{h} \hat{\rho}_k^2$$

where :

- n is the sample size,
- h is the number of lags considerer,
- $\hat{\rho}_k$ is the estimated autocorrelation at lag k.

Under $(\mathcal{H}_0)$, Q follows a chi-squared distribution with h degrees of freedom $(\chi^2(h))$.

Thus, in the remainder of this study, all models rejecting the null hypothesis of the Box-Pierce test will not

be considered [5]. Next, to select the different models, we will use two criteria commonly employed in statistics. The first of these criteria is the Bayesian Information Criterion (BIC), which is based on the maximized likelihood of the model with a penalty for its complexity.

Definition 2.3 (Bayesian Information Criterion (BIC))

The BIC is defined as :

$$BIC = -2\ln(\hat{L}) + k \times \ln(n)$$

where :

— $\hat{L}$ is the maximized likelihood of the model,

— k is the number of estimated parameters in the model,

— n is the sample size.

The first term of BIC criterion rewards models that fit the data well and the second term penalizes models with more parameters, discouraging overfitting. So, a model with a lower BIC value is preferred, as it represents a better trade-off between accuracy and simplicity.

Then, the other criterion that will be used to select the various models is the Akaike Information Criterion (AIC). The AIC focuses more on goodness-of-fit and tends to favor more complex models, while the BIC imposes a stronger penalty on complexity, especially for a large sample size.

Definition 2.4 (Akaike Information Criterion (AIC))

The AIC is defined as :

$$AIC = -2\ln(\hat{L}) + 2k$$

where :

— $\hat{L}$ is the maximized likelihood of the model,

— k is the number of estimated parameters in the model.

2.1.3 Presentation of results

Different ARIMA(p,1,q) models were trained using the training dataset, with BIC and AIC criteria used to identify the best fit. A grid of values for p and q was established, and several models were trained for each pair. Only models that pass the Box-Pierce test were considered ; others were assigned an AIC/BIC value of `Inf` [6]. We set $p_{\max} = 6$ based on significant peaks in the PACF plot (figure 4b) and $q_{\max} = 4$ based on the ACF plot (figure 4a).

After training the different models, table 1 summarizes all the calculated BIC criteria.

p \ q	0	1	2	3	4
0	Inf	Inf	Inf	Inf	Inf
1	Inf	Inf	Inf	**−14361.50**	−14353.81
2	Inf	Inf	−14358.99	−14353.09	−14345.74
3	Inf	−14361.40	−14353.12	−14345.44	−14346.94
4	Inf	−14353.67	−14345.58	−14342.87	−14342.77
5	−14343.53	−14335.62	−14337.81	−14338.98	−14326.75
6	−14335.70	−14327.83	−14330.90	−14324.97	−14318.40

TABLE 1 – BIC for ARIMA models

Similarly, table 2 summarizes all the calculated AIC criteria.

5. Note that other similar tests could have been used, such as the Ljung-Box test which will be studied further for a complete analysis of residuals.

6. infty

p \ q	0	1	2	3	4
0	Inf	Inf	Inf	Inf	Inf
1	Inf	Inf	Inf	−14397.05	−14395.29
2	Inf	Inf	−14394.54	−14394.57	−14393.15
3	Inf	−14361.40	−14394.60	−14392.85	−14400.27
4	Inf	−14395.15	−14392.99	−14396.20	**−14402.02**
5	−14385.02	−14383.02	−14391.14	−14398.23	−14391.93
6	−14383.10	−14381.16	−14390.16	−14390.15	−14389.51

TABLE 2 – AIC for ARIMA models

Based on these tables, the objective is to determine the model that best fits the data. To achieve this, we examine the (p, q) pair that minimizes the BIC and AIC criteria, if possible.

From our analysis, we observe that the BIC criterion suggests that the best model to fit the series among all the generated models is an $ARIMA(1, 1, 3)$ model. On the other hand, the AIC criterion recommends an $ARIMA(4, 1, 4)$ model. In such cases, the model selection is guided by the principle of sparsity, that is, choosing the simplest model that adequately represents the data. Therefore, as it estimated less parameters, the selected model is the $ARIMA(1, 1, 3)$ [7].

For this model, the parameters estimated are presented in the following table 3.

	ar1	**ma1**	**ma2**	**ma3**	**intercept**
Estimate	−0.9331	0.9239	−0.0130	−0.0312	6×10^{-4}
s.e.	0.0308	0.0361	0.0259	0.0198	3×10^{-4}

TABLE 3 – Parameter Estimates with Standard Errors

2.1.4 Residual analysis

We end up checking the correctness of the ARIMA model by conducting an additional residual analysis. We will use a new test called the Ljung-Box test, which is used, similarly to the Box-Pierce test, to evaluate the hypothesis of independence of residuals in a time series. We first introduce this new statistical test below.

Definition 2.5 (Ljung-Box Test)

— **Test Hypothesis :** $(\mathcal{H}_0)$: The autocorrelations of residuals are zero for all lags up to h, implying that the residuals are independent, versus $(\mathcal{H}_1)$: at least one of the autocorrelations up to h is non-zero.

— **Statistical Test :** The test statistic Q is calculated as follows :

$$Q = n(n+2)\sum_{k=1}^{h}\frac{\hat{\rho}_k^2}{nk}$$

where :

— n is the sample size,

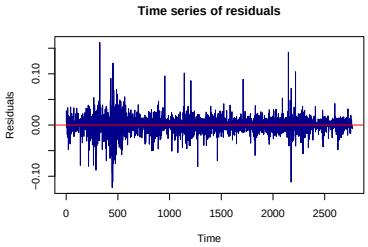
— h is the number of lags considerer,

— $\hat{\rho}_k$ is the estimated autocorrelation at lag k.

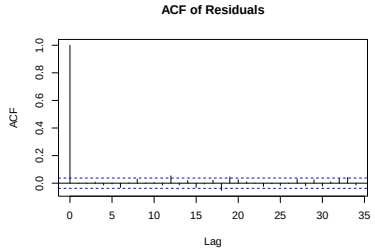
Under $(\mathcal{H}_0)$, Q follows a chi-squared distribution with h degrees of freedom $(\chi^2(h))$.

7. As additional information, we note that the `auto.arima` function from `R` suggests an $ARIMA(3, 1, 2)$ for this presented time series. We doesn't consider this informations for construction any model as this function can sometimes generated bad results.

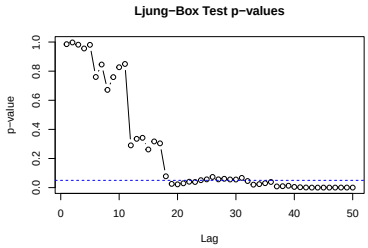
Then, an analysis of the residuals corresponding to the best model we have found is represented below (figure 8). Our aim is to recover the charasteristics of a white noise process : a null expected value and an absence of autocorrelations. The figure 8a displays the evolution of residuals over time. The mean value seems to be around 0, which is what we expected. The figure 8b represents the ACF of the residuals. No peak is significant (with a 95 % confidence level) so there does not seem to be any autocorrelation. Finally, the figure 8c exhibits the Ljung-Box test $p-$values depending on the lag value. As expected, until lag 30, those values lie over the significancy level of 0.05 which means that we do not reject ($\mathcal{H}_0$) for such lags and accept the hypothesis of the absence of autocorrelations for residual values. Notwithstanding, we note that $p-$values lie under the significancy level for lags (strictly) larger than 31. So, some significant autocorrelations could still be significant when considering a larger scale. Thus, the model does not look perfectly adapted to the log-returns time series.



(a) Time series of residuals
Note : The red line indicates the mean value of the time series of residuals.



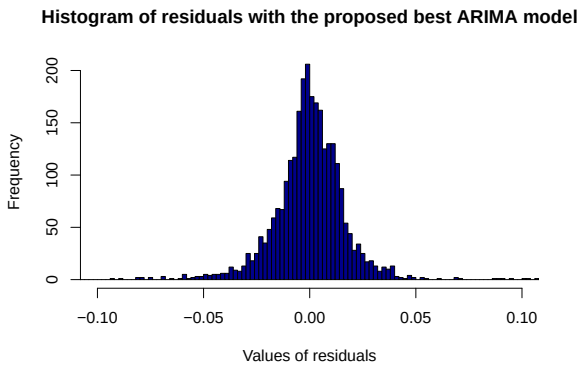
(b) ACF of Residuals



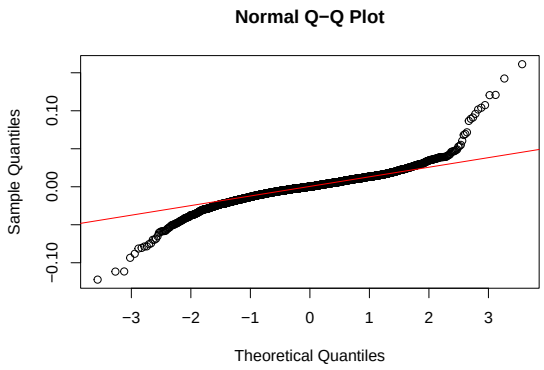
(c) $p-$values for Ljung-Box statistic
Note : The blue line indicates the significancy level for the Ljung-Box test (0.05)

FIGURE 8 – Analysis of residuals using the Ljung-Box test

Then, we check whether the residuals appear to follow a normal distribution. In such a case, the process (ε_t) would correspond to a Gaussian white noise process. We present the results in figure 9. In order to check that, we present in figure 9a, an histogram, as well as a quantile-quantile plot in figure 9b. Although the hypothesis of a Gaussian distribution seems fair, we note that the distribution is a bit skew on the right, indicating potential outliers.



(a) Histogram of the residuals



(b) QQ-plot of the residuals

FIGURE 9 – Predictions models

This analysis of residuals suggests that the proposed ARIMA model is quite acceptable, as the residuals exhibit most of the characteristics of a Gaussian white noise. Still, some autocorrelations can be noticeable (for large values of lag) and the empirical distribution does not have the exact same trend as a Gaussian one. Therefore, we can use this model but need to remain careful when using it to predict future values.

2.1.5 $FARIMA$ process

An extension of the ARIMA process, with fractional differencing, called Fractionally Autoregressive Integrated Moving Average (FARIMA) is presented in appendix 1 but does not look effective, suggesting that the ARIMA model is better with our data.

2.2 Generalized Autoregressive Conditional Heteroskedasticity (GARCH)

2.2.1 Model presentation

The Generalized Autoregressive Conditional Heteroskedasticity (GARCH) model is used to model the evolution of volatility in a time series. In other words, the goal of such a model is to study how the variability of future values depends on past information. Unlike classical models such as ARMA, which aim to explain the relationships between current and past values of the series, GARCH models focus on the volatility of the data. Thus, instead of focusing on the levels of the series, this model examines how the uncertainty of values changes over time based on past events.

Definition 2.6 ($GARCH(p,q)$ process)

A process $(X_t)_{t\in\mathcal{X}}$ is said to be a GARCH(p,q) process if there exists $w > 0$ and $\alpha_1, \ldots, \alpha_p, \beta_1, \ldots, \beta_q \in \mathbb{R}_+$ such that, for all $t \in \mathbb{Z}$,

$$\begin{cases} X_t = \sigma_t \eta_t \\ \sigma_t^2 = w + \sum_{i=1}^{p} \alpha_i X_{t-i}^2 + \sum_{j=1}^{q} \beta_j \sigma_{t-j}^2 \end{cases}$$

where $(\eta_t)_{t\in\mathbb{Z}}$ is a sequence of independent, centered, and standardized random variables.

Definition 2.7 (t-value)

The t-value, defined as the ratio of a parameter estimate to its standard error, measures how many standard errors the estimate is away from zero :

$$t = \frac{\text{Estimate}}{\text{Standard Error}}.$$

A larger absolute t-value indicates stronger evidence against the null hypothesis, suggesting statistical significance when paired with a small p-value.

2.2.2 Model selection and estimation

To identify the best GARCH specification for modeling Google's daily opening price log-returns, we conducted an extensive search over different combinations of orders (p, q) and error distributions. Specifically, we tested :

— Orders $(p, q) \in \{(1, 1), (1, 2), (2, 1), (2, 2)\}$,

— Error distributions : Gaussian (`norm`), Student-t (`std`), and Generalized Error Distribution (`ged`).

To identify the best model, we focused on minimizing the Bayesian Information Criterion (BIC), as outlined in the subsection on the ARIMA model.

From this analysis, a GARCH(1,2) model with a Student-t distribution for residuals emerged as the best-fitting specification.

Tables 4 and 5 summarize the parameter estimates for the GARCH model and the model's goodness of fit respectively.

The results indicated significant parameter estimates for both short-term shock effects ($\alpha_1 = 0.1058, p < 0.001$) and long-term persistence ($\beta_2 = 0.7198, p < 0.001$), highlighting strong volatility clustering in Google's stock returns.

Parameter	Estimate	Std. Error	$t-$value	$p-$value
μ	0.1104	0.0239	4.62	< 0.001
$AR(1)$	0.0324	0.0191	1.69	0.0903
ω	0.0617	0.0273	2.26	0.0237
α_1	0.1058	0.0282	3.75	< 0.001
β_1	0.1551	0.1511	1.03	0.3046
β_2	0.7198	0.1461	4.93	< 0.001
Shape	4.434	0.3505	12.65	< 0.001

TABLE 4 – Summary of Garch(1,2) Model Estimation Results

The weighted Ljung-Box test on standardized residuals with a lag of 5 showed no significant autocorrelation ($p = 0.6156$), suggesting that the model adequately captures the dependency structure in the data.

Log-likelihood	BIC	AIC	HQC	Shibata
-4973.62	3.6137	3.5987	3.6041	3.5987

TABLE 5 – Goodness-of-Fit Measures for the final GARCH Model

2.3 Long Short-Term Memory (LSTM)

Deep Learning has seen significant growth in recent years, especially in time series forecasting. Certain models have proven to be highly efficient for this task, particularly Recurrent Neural Networks (RNNs), which leverage the temporal dimension to predict various types of sequences. Among these models, Long Short-Term Memory (LSTM) networks are the most commonly used, as they address a key issue with traditional RNNs : the vanishing/exploding gradient problem, which makes it difficult for standard RNNs to maintain information over long sequences.

In this context, we will use LSTM models for one type of predictions. But there exists other one that can be experimented. The first approach we used in our project, known as Sequence-to-One, involves using an LSTM model that takes input from n previous days to predict the next day. The second approach, Sequence-to-Sequence, incorporates an Encoder-Decoder architecture to predict m future days from n input days.

Before presenting the results, it's important to note that we will be using a substantial amount of input data. However, our goal is to predict based on the most recent year of data. To achieve this, we need to adjust the train and test datasets accordingly. Specifically, we will shift the datasets to ensure that the test set includes n input days from the previous year, while removing these from the training set. This adjustment will help us better evaluate the performance of the model on real-world, future data.

We implemented our code in Python using PyTorch, as it offers a simpler and more intuitive approach to deep learning compared to R. Additionally, Python benefits from extensive documentation and community support, whereas R has relatively limited resources for deep learning applications.

2.3.1 Methodology

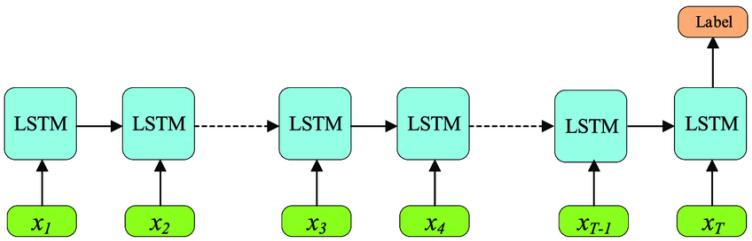


FIGURE 10 – Many to one LSTM

In this project, we trained an LSTM model, as shown in Figure 10. We opted for a sliding window approach, using 100 input days to divide the time series into fixed-length segments, referred to as p-windows. These windows, treated as independent, were randomly selected for training since predicting the next step should not depend on their order.

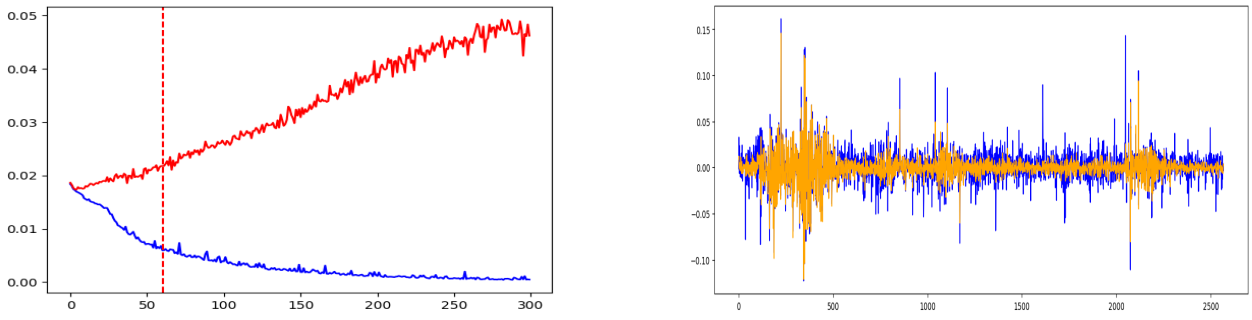
Before training, we performed hyperparameter optimization to determine the best combination of the number of hidden layers and hidden cells. We tested 12 combinations :

— Number of hidden cells : 16, 32, 64, 128

— Number of hidden layers : 1, 2, 3

The combination that resulted in the lowest Mean Squared Error (MSE) was (32 hidden cells, 3 hidden layers).

Next, we tuned the number of epochs. Figure 11a shows the evolution of the loss function for both the training and validation datasets as the number of epochs increases. Based on the figure, 60 epochs were selected as the optimal value, striking a balance between allowing the model enough time to learn and keeping computation time reasonable.



(a) Train loss vs Validation loss

(b) Train dataset predictions

FIGURE 11 – Comparison of loss evolution and predictions on the train dataset.

After training the model with the selected hyperparameters, we made predictions on the test data, consisting of the log returns of opening prices. The test dataset included the last 350 days of the series, ensuring that the model had the 100 input days required to predict the next day.

The model was trained to predict the next day's log return using the true inputs from the test set for each step. While this approach worked well on the training dataset (See Figure 11b, forecasting over multiple days required a recursive method : using the prediction of one step as the input for the next. This method better reflects real-world forecasting but introduces compounding errors, as discussed in section 3.2.1.

The results presented in section 3.2.1 show that multi-day forecasts were poor. This outcome is expected, as the model was specifically trained to predict one day ahead. A better approach for multi-day forecasting would involve training a Sequence-to-Sequence (Seq2Seq) model. Seq2Seq models, illustrated in figure 14, are designed to handle sequences as both inputs and outputs, making them particularly effective for multi-step predictions.

Another alternative worth exploring is the use of encoder-decoder architectures, such as Transformers. These models have shown exceptional performance in Natural Language Processing (NLP) tasks and are being adapted for time series forecasting. However, their suitability for time series depends on the dataset and the specific task requirements.

A key challenge in this project was balancing the need for stationarity with the requirement for interpretability. The model was trained on log returns, as they are stationary and well-suited for time series modeling. However, our goal was to provide predictions in terms of actual opening prices, which are more interpretable for non-specialists.

To convert log return predictions to opening prices, we used the following transformation :

$$P_{t+1} = P_t \cdot e^{R_{t+1}}$$

where P_{t+1} is the predicted opening price, P_t is the current price, and R_{t+1} is the predicted log return.

This transformation introduces an exponential accumulation of errors during recursive predictions over multiple days. This trade-off highlights a critical decision :

— Option 1 : Work with stationary variables (log returns) for a more robust model, requiring additional transformations for interpretability.

— Option 2 : Train directly on non-stationary variables (opening prices) to avoid compounding errors but risk reduced model performance due to non-stationarity.

The choice depends on the primary objective of the study : prioritizing model performance or ensuring interpretability for end users.

3 Discussions

3.1 Used Metrics

In this subsection, we present the metrics used to evaluate the quality of the model's predictions by comparing them with the test data. The selected metrics are : Mean Absolute Error (MAE), Mean Squared Error (MSE), and Root Mean Squared Error (RMSE). These measures allow for an overall evaluation of the model's performance from different perspectives.

The notations used are as follows :

— n : the total number of samples in the test dataset,

— y_i : the actual observed value for sample i,

— $\hat{y}_i$: the predicted value by the model for sample i.

Each metric is defined as follows :

Definition 3.1 (Mean Absolute Error (MAE))

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|$$

MAE measures the average absolute error between the predictions and the actual values.
It is expressed in the same units as the variable of interest and is robust to outliers.

Definition 3.2 (Mean Squared Error (MSE))

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

MSE quantifies the average of the squared errors.
This metric is sensitive to large errors, making it useful for detecting anomalies or significant deviations.

Definition 3.3 (Root Mean Squared Error (RMSE))

$$\text{RMSE} = \sqrt{\text{MSE}} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}$$

RMSE is the square root of the MSE, expressed in the same units as the variable of interest.
It combines the advantages of MSE while providing a more intuitive interpretation.

These metrics provide complementary insights into the model's performance. MAE is suited for measuring the overall average error, while MSE and RMSE place more emphasis on larger errors due to their quadratic nature.

3.2 Model comparisons

3.2.1 Model performances

Metrics \ Models	*Predicted days*	**ARIMA(1,1,3)**	**GARCH(1,2)**	**LSTM**
MAE	*5 days*	1.57	1.52	5.31
	22 days	3.78	3.47	15.74
	250 days	3.30	5.52	45.94
MSE	*5 days*	3.33	3.07	35.74
	22 days	17.30	14.46	310.22
	250 days	16.44	55.15	2273.84
RMSE	*5 days*	1.83	1.75	5.94
	22 days	4.16	3.80	17.61
	250 days	4.05	7.43	45.94

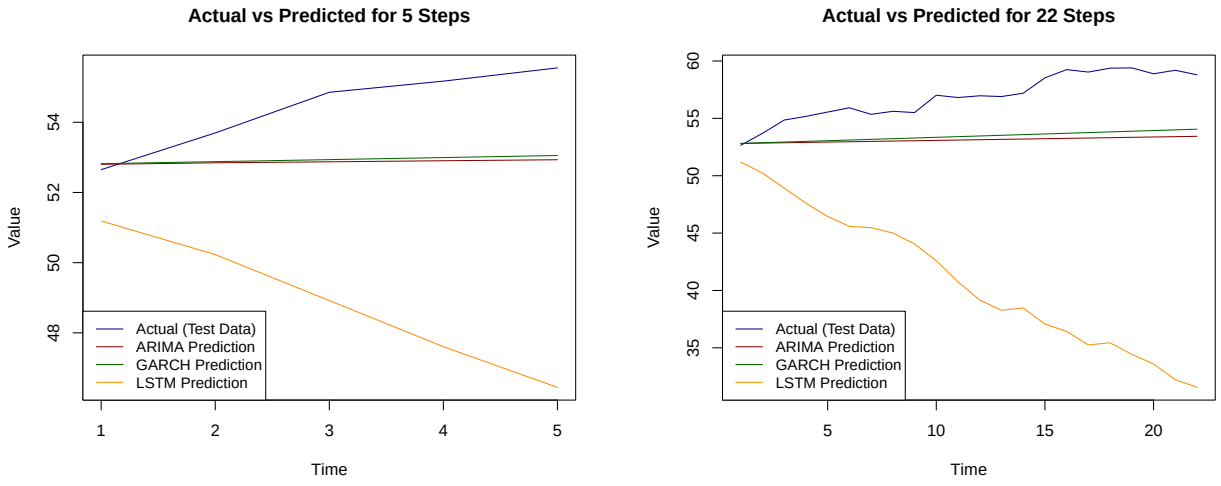
TABLE 6 – Comparison of model performance for different forecasting horizons (MAE, MSE and RMSE)

This table 6 summarizes the comparison of model performance across different forecasting horizons (5 days, 22 days, and 250 days) using the threes key metrics presented previously : Mean Absolute Error (MAE), Mean Squared Error (MSE), and Root Mean Squared Error (RMSE).

3.3 Model Selection and Investment advices

Based on the results in Table 6, we can draw several conclusions regarding model performance and its implications for investment decision-making.

3.3.1 Short-Term Forecasting (5-22 days)



(a) Prediction along 5 days

(b) Prediction along 22 days

FIGURE 12 – Predictions models

— **GARCH(1,2) :** This model consistently shows lower MAE, MSE, and RMSE than ARIMA(1,1,3) and LSTM for both the 5-day and 22-day horizons, indicating that it better captures short-term fluctuations.

— **ARIMA(1,1,3) :** ARIMA ranks second, demonstrating higher errors compared to GARCH(1,2), but still performs better than LSTM.

— **LSTM :** The LSTM model exhibits relatively larger errors, suggesting that in its current configuration, it may not be the most suitable choice for short-term forecasts.

3.3.2 Long-Term Forecasting (250 days)

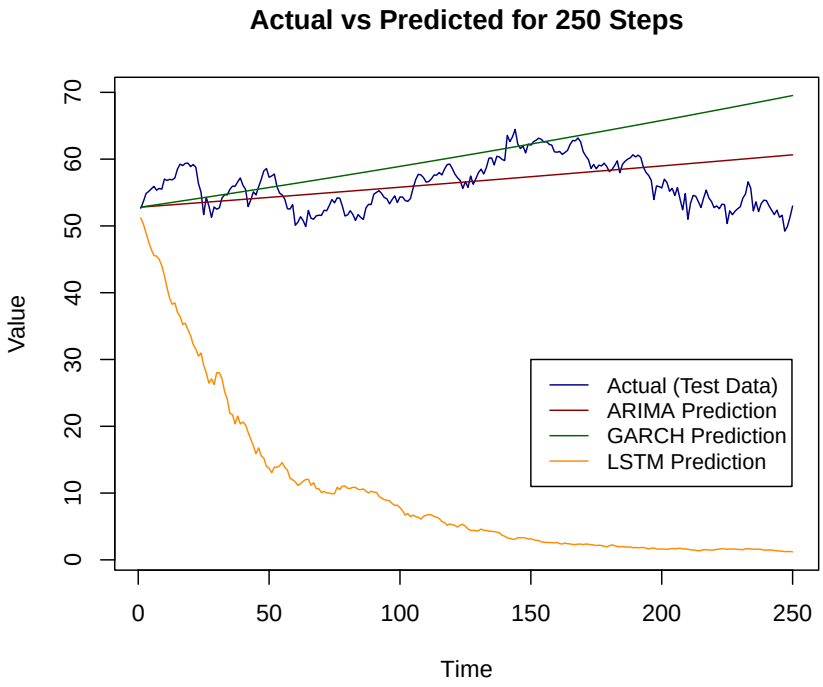


FIGURE 13 – Predictions along 250 days for 3 different models

— **ARIMA(1,1,3) :** This model outperforms both GARCH(1,2) and LSTM, indicating that for longer forecast windows, the simpler linear structure of ARIMA might generalize better than GARCH's volatility-focused approach.

— **GARCH(1,2) :** While effective for short-term horizons, GARCH does not perform as well in long-term forecasting scenarios.

— **LSTM :** The LSTM model shows significantly higher error metrics, implying that additional tuning or more extensive training data might be required for neural-network methods to be effective at this horizon.

3.3.3 Investment Decision Support

— **Short-Term Trading Strategies :** GARCH's superior short-term performance can be valuable for active traders or hedge strategies that rely on accurate near-future volatility estimates.

— **Long-Term Portfolio Allocation :** ARIMA's better performance over 250 days suggests it may be more suitable for strategic asset allocation or long-horizon investment planning, where stable, longer-range trends are of greater importance.

— **Further Model Development :** Given LSTM's underperformance, an investor or analyst might explore more sophisticated deep-learning architectures or hyperparameter tuning if a neural-network approach is desired.

Conclusion

This project has demonstrated a comprehensive approach to time series modeling for Google's stock prices, focusing on ARIMA, GARCH, and LSTM methodologies. The data—spanning January 1, 2007, to December 31, 2018—were obtained from Yahoo Finance, log-transformed, and differenced to address non-stationarity. Rigorous diagnostic checks, including the Augmented Dickey-Fuller and Ljung-Box tests, were applied to ensure that the selected models captured the underlying dynamics of the time series as accurately as possible.

From a volatility perspective, GARCH(1,2) provided the best short-term forecasts (5-22 days), underscoring its ability to capture autoregressive effects in the conditional variance. ARIMA(1,1,3), on the other hand, exhibited superior performance over the 250-day horizon, indicating that simpler linear models can outperform volatility-focused models when the prediction window is extended. LSTM, while theoretically promising, underperformed in the current configuration, suggesting a need for further hyperparameter tuning, additional training data, or architectural modifications to fully leverage deep learning's potential.

These findings underscore two key points. First, model selection should be closely aligned with investment horizons. Traders aiming for short-term profit may benefit most from GARCH-based forecasts due to their capacity to model volatility clustering, while investors with longer-term objectives can utilize ARIMA models to capture broader trends. Second, although each model displayed strengths under particular conditions, none alone achieved sufficient robustness for real-world trading or investment decisions ; an ensemble-based framework, which combines the predictions of multiple models, may yield more reliable forecasts in practice.

Overall, by examining and contrasting three modeling paradigms, this highlights the importance of tailoring the time series approach to a specific forecasting horizon and objective. The moderate predictive success across models reinforces the inherent complexity of financial data, where market sentiments, macroeconomic indicators, and unexpected events can disrupt established patterns—motivating future research into advanced machine learning architectures, hybrid models, or alternative data sources.

To advance further, a new class of models has emerged that combines classical approaches like GARCH and ARIMA with deep learning techniques such as LSTM. These models, known as hybrid models, have demonstrated their effectiveness Wang et al. 2013. Classical models excel at capturing trends and seasonality but are less effective at handling noise. Deep learning models, on the other hand, are well-suited for managing such irregularities and can complement classical models to enhance predictions. For future work, it would be worthwhile to experiment with these hybrid models and compare their performance against non-hybrid approaches, such as those utilized in our study.

References

Maechler, Martin, Chris Fraley, Friedrich Leisch, Valderio Reisen, Artur Lemonte et Rob Hyndman. 2024. *fracdiff : Fractionally Differenced ARIMA aka ARFIMA(P,d,q) Models.* Version 1.5-3. Maximum likelihood estimation of the parameters of a fractionally differenced ARIMA(p,d,q) model ; including inference and basic methods. Some alternative algorithms to estimate "H". R package version 1.5-3. https://doi.org/10.32614/CRAN.package.fracdiff. https://github.com/mmaechler/fracdiff.

Ryan, Jeffrey A., Joshua M. Ulrich, Ethan B. Smith, Wouter Thielen, Paul Teetor et Steve Bronder. 2024. *quantmod : Quantitative Financial Modelling Framework.* https://github.com/joshuaulrich/quantmod. Version 0.4.26. R package version 0.4.26. https://doi.org/10.32614/CRAN.package.quantmod. https://www.quantmod.com/.

Wang, L., H. Zou, J. Su, L. Li et S. Chaudhry. 2013. « An ARIMA-ANN hybrid model for time series forecasting ». *Strategic and Resourced Economics* 27 (5) : 1-15. https://doi.org/10.1002/sres.2179.

Appendix

1 Fractionally Autoregressive Integrated Moving Average (FARIMA)

1.1 Model presentation

In section 2.1, we presented the AutoRegressive Integrated Moving Average (ARIMA) process. With such a model, the differencing parameter d could only take integers as values. Below, we introduce an extension of the ARIMA model called Fractionally Autoregressive Integrated Moving Average (FARIMA) which can, as its name suggests, take fractional values as well.

Definition 1.1 ($FARIMA(p, d, q)$ process)

A fractionally autoregressive integrated moving average process (X_t) of orders p, $d \in]-0.5, 0.5[$, and q, denoted by $FARIMA(p, d, q)$, is defined if the process (X_t) is **stationary** and satisfies :

$$\left(I - \sum_{i=1}^{p} \alpha_i B^i\right) (I - B)^d X_t = \mu + \varepsilon_t + \sum_{j=1}^{q} \theta_j B^j \varepsilon_t$$

where :

- $\left(I - \sum_{i=1}^{p} \alpha_i B^i\right)$ corresponds to the AutoRegressive (AR) part,
- $(I - B)^d = \sum_{k=0}^{\infty} \gamma_k(d) B^j$, where $\gamma_0(d) = 1$ and $\forall k \geq 1, \gamma_k(d) = \frac{(-1)^k}{k!} \prod_{l=0}^{k-1} (d - k)$, corresponds to the Fractionally Integrated (I) part,
- μ corresponds to the mean value of the process,
- (ε_t) corresponds to a white noise process,
- $\sum_{j=1}^{q} \theta_j B^j$ corresponds to the Moving Average (MA) part.

We note that this process is known as a **long-memory process**. This means that such a method can be useful when a stationary time series exhibits important autocorrelations.

1.2 Presentation of results

Since this method requires a stationary time series, it is necessary to consider the time series of log-returns and not the logarithmic transformation of the initial time series. Based on figure 6a, as the ACF decays quickly, the time series does not seem to have important autocorrelations which suggests that a FARIMA model is not convenient for our data. But we will still try to identify what seem to be the best parameters for a FARIMA model.

In R, FARIMA processes can be used to model time series with the `fracdiff` package (Maechler et al. 2024). As we did before with the ARIMA model, by considering a sparse model with the lowest AIC and BIC, we found out that once again that the best parameters were $p = 1$ and $q = 3$. The results are presented in the following table 7

Parameter	Estimate	Standard Error (s.e.)	95% Confidence Interval
d	0.002	0.028	$[-0.053; 0.057]$

TABLE 7 – Estimation of the fractional differencing parameter

As the value 0 lies in the confidence interval, this suggests that the parameter d is not significant. Therefore, we can consider the value $d = 0$, which corresponds to a simple ARMA(1,3) model for the differenced time series. This means that the FARIMA model is not appropriate in our case.

2 SeqtoSeq model structure

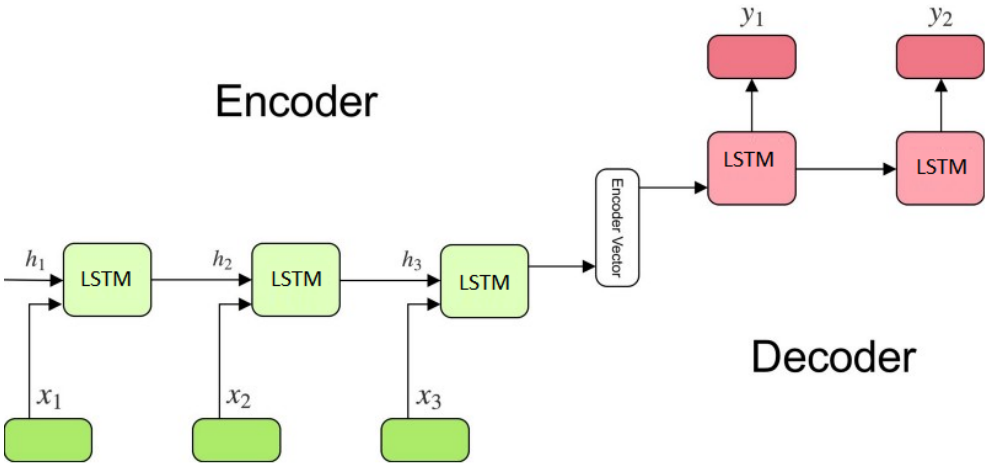


FIGURE 14 – SeqtoSeq LSTM