ÉCOLE NATIONALE DE LA STATISTIQUE ET DE L'ANALYSE DE L'INFORMATION

ANNÉE 2024/2025

Advanced Time Series Project

Stock price predictions

Authors :
Mathis Bouillon
Cheryl Kouadio
Marilène Kougoum Moko Mani
Mariyam Ouyassin
Mountaga Wane

Under the supervision of :
Youssef Esstafa

Contents

1 Abstract

This study explores the prediction of CAC40 stock prices using a combination of classical methods and advanced machine learning techniques. Several models were tested, including Ridge regression with exogenous macroeconomic variables and self-constructed technical indicators, which emerged as the most effective with a low RMSE of 64.99. The inclusion of external variables allowed for a robust modeling approach, capturing underlying market trends. Additionally, Long Short-Term Memory (LSTM) networks were tested, showing potential for capturing short-term dependencies, but did not outperform Ridge regression in this context.

Furthermore, an analysis of residuals using a GARCH(1,1) model provided valuable insights into volatility, contributing to a more comprehensive understanding of market uncertainty. This approach highlights the complex and volatile nature of the financial market.

Traditional and widely-used time series models, such as ARIMA and ARDL, however, yielded very unsatisfactory results, reflecting the complexity of the CAC40 time series. These models struggled to capture the intricate patterns and volatility inherent in financial markets, emphasizing the challenge of accurately predicting stock prices.

2 Introduction

Stock price prediction is a significant topic in finance and economics, attracting researchers to develop increasingly accurate predictive models. As mentioned in (Brealey, Myers, and Allen 2014), it not only aids in achieving better investment returns but also contributes to the efficient functioning of financial markets, risk management, and corporate strategy.

Traditional approaches often rely on linear models such as Autoregressive Moving Average (ARMA) and Autoregressive Integrated Moving Average (ARIMA) (Ariyo, Adewumi, and Ayo 2014), which are effective for stationary time series. However, stock prices are inherently characterized by volatility, unpredictability, and non-linear dynamics, limiting the performance of purely linear models. To address these limitations, models such as Autoregressive Conditionally Heteroscedastic (ARCH) (Engle 1993) and Generalized ARCH (GARCH) (Bollerslev 1986) were introduced, focusing on modeling conditional variance. These models remain widely used in finance for capturing and forecasting volatility.

To further enhance predictions, models incorporating explanatory variables, such as Vector AutoRegression (VAR), Vector Error Correction Model (VECM), and Autoregressive Distributed Lag (ARDL), have gained traction. These methods allow the inclusion of relationships between multiple time series and the consideration of lagged effects of predictors, offering a more comprehensive approach to stock price modeling.

In parallel, recent advancements in machine learning have opened new avenues for stock market prediction. Techniques such as Support Vector Machines (SVM), Random Forest (RF), and Gradient Boosting (GB) have shown promise in capturing complex patterns within data (Vijh et al. 2020; Obthong et al. 2020). Additionally, neural network architectures like Artificial Neural Networks (ANN), Convolutional Neural Networks (CNN), Recurrent Neural Networks (RNN), and advanced models such as Long Short-Term Memory (LSTM) networks have demonstrated their ability to model non-linear dependencies and temporal patterns effectively. Since stock prices often exhibit long memory but not consistently, Fractionally Integrated Autoregressive Moving Average (FARIMA) models also remain relevant (Esstafa 2019).

Beyond traditional and machine learning models, the Prophet model has emerged as a robust and flexible tool for time series forecasting. Developed by Facebook, Prophet is particularly adept at handling seasonality, trends, and missing data, making it suitable for financial applications where external shocks and irregular patterns are common (Taylor and Letham 2018). Unlike classical statistical models, Prophet requires minimal parameter tuning and can seamlessly incorporate domain knowledge, such as known holidays or significant events, into its predictions. This makes it a valuable alternative for practitioners seeking quick, reliable insights without deep statistical expertise.

As one can see, accurately predicting stock prices is known to be a very challenging task. The goal of this project is to attempt to predict the stock prices of the CAC 40 index. Since historical data alone often fails to capture all the relevant factors influencing stock prices, we propose integrating additional factors, such as macroeconomic indicators and technical indicators to address this limitation. By using a diverse set of approaches—including ARIMA, FARIMA, GARCH, Prophet, Gradient Boosting, LSTM, and GRU—we aim to tackle this complex but important problem.

This project is organized as follows: first, we will describe the data, specifically the CAC 40, to understand the characteristics and complexity of the stock price data, as well as the explanatory variables used and their sources. In the second time, the methods employed to predict CAC 40 prices will be described. The next section will compare the different models evaluated in order to identify the best approach for predicting CAC 40 prices. Finally, the project will conclude with a summary of the findings.

3 Data

The CAC 40 is a stock market index comprising France's 40 largest listed companies. It is used to measure the performance of the French economy and is an important indicator for investors. In this project, the historical CAC 40 data has been collected from Yahoo Finance. The data contains information about the stock such as High, Low, Open, Close, Adjusted close prices and Volume. Only the day-wise closing price of the stock has been extracted. The dataset does not contain any outliers, i.e. negative prices and no missing values were identified.

We selected a dataset spanning from 2000 to 2024 instead of using all the historical data available to better capture recent market dynamics, which are often more relevant to current predictions. This period includes significant milestones such as the Internet bubble (2000), the financial crisis (2008), the European debt crisis (2011), and the COVID-19 pandemic (2020). These events provide valuable insights for analyzing volatility and recent trends. However, we acknowledge that limiting the dataset to this specific period may introduce a bias, as it excludes the pre-2000 period, which could offer additional context and insights into the market's long-term behavior.

Table 1 shows statistics of the dataset that is used for training and testing.

Table 1: Statistics of the dataset

	Dataset	Training set	Testing set
Time intervals	01/01/2000 to 30/10/2024	01/01/2000 to 31/12/2022	01/01/2023 to 30/10/2024

3.1 Trends and seasonality

3.1.1 Trend analysis of the CAC40

As shown in Figure 1, there is a linear trend in the CAC40 index, although it is not highly pronounced. This trend is confirmed both by the linear regression line and by the statistical analysis of the regression model. Specifically, we performed a regression of the closing prices on time, and the results show that all coefficients are statistically significant, with p-values less than 0.001, indicating a significant linear relationship between time and the index. The Fisher test also confirms the overall significance of the model.

Figure 1: Linear trend of the CAC40, with a linear regression line

However, despite the significance of the coefficients, the model only explains a small portion of the variability in the data, as indicated by the low R^2 value of 0.28. This means that the linear regression, while significant, does not explain much of the variance in the CAC40 closing prices over the studied period.

This limited explanatory power can be partly attributed to the period considered, which starts in the 2000s, a period marked by the burst of the Internet bubble. If we take a longer period, for example starting from 1990, as shown in the figure in Figure 19 in the Appendix, we can see that the regression line becomes much more coherent, and the R-squared increases significantly, reaching 0.6, which indicates a much better explanation of the variance.

Figure 2: Monthly averages of the CAC40

3.1.2 Seasonality analysis of the CAC 40

The seasonal plot of the closing price of the CAC 40 (fig. 3) suggests that the data does not exhibit clear seasonality. While there may be some recurring patterns observed in certain years, such as occasional dips in the middle of the year or rises toward the end, these trends are not consistent across all years. The absolute values vary significantly between years, and the month-to-month behavior does not display a systematic or predictable pattern that would strongly indicate seasonality. This inconsistency suggests that any apparent patterns may be coincidental or influenced by other factors rather than a true seasonal component.

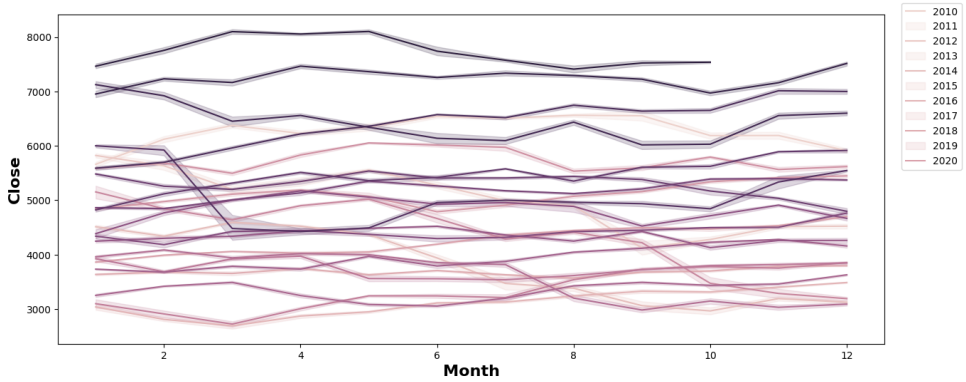


Figure 3: Yearly seasonal trends in the CAC 40 index

3.2 Features engineering

To incorporate explanatory variables into our models, we used macroeconomic variables from the Organisation for Economic Co-operation and Development *OECD* and created new variables from existing information in the dataset. The encoding of each variables are presented in table 11 in annex.

3.2.1 Macroeconomic variables

To collect macroeconomic variables, we explored the OECD Data Explorer. This platform allows users to choose from various themes such as Economy, Finance and Investment, Employment, etc. However, only one category provided forecasts: the "Economic Outlook" section within the "Economy" theme. We selected Economic Outlook No. 116, which is the latest version for the year 2024. This database contains data for 63 countries and offers more than 300 variables. From this extensive dataset, we carefully selected variables in a parsimonious manner, guided by relevant academic articles.

For instance, (Haq and Larsson 2016) investigated the relationship between the US stock market index S&P500 and six macroeconomic indicators, including Personal Spending, Initial Jobless Claims, M1 Money Supply, Building Permits, the Michigan Consumer Sentiment Index, and the ISM Manufacturing Index. Their results demonstrate that all indicators, except for Personal Spending, have a significant long-term impact on stock prices, while in the short term, all indicators except M1 Money Supply are significant, depending on the time period analyzed.

Similarly, (Mathusha 2021) analyzed the effects of macroeconomic variables on stock prices in Sri Lanka. Their findings suggest that, in the long term, stock prices are significantly influenced by the Consumer Price Index (CPI) and the Treasury Bill Rate. In the short term, however, most variables are insignificant, except for the Treasury Bill Rate, which exhibits a negative relationship with stock prices.

Moreover, GDP growth signals economic strength, boosting investor confidence and stock prices, while contractions often lead to declines. Interest rates affect borrowing costs and the attractiveness of stocks versus bonds, with higher rates typically dampening stock performance (Barra 2010; Zhiteng, Ruoyu, et al. 2020). The unemployment rate also plays a critical role; high unemployment reduces consumer spending and corporate earnings, negatively impacting stocks (Camara, Pessarossi, and Philippon 2017).

Considering this information and what was effectively available on the OECD Data Explorer [1], we selected the following variables with a quarterly frequency of observation, for the period from Q1 2000 to Q4 2026:

- Consumer Price Index (CPI) for the United States.
- Central Bank interest rate for the United States.
- Unemployment rate for the United States.
- Current household expenditures, including nonprofit institutions serving households.
- Gross Domestic Product (GDP) in volume, at market prices.
- Long-term interest rate on government bonds.
- Short-term interest rate.
- Gross Domestic Product (GDP) in nominal value, at market prices.
- Harmonized overall inflation rate.

Although the economic data we collected was provided on a quarterly basis, our CAC 40 time series operates on a daily basis. This daily granularity is highly informative, and we did not want to lose valuable insights by downsampling it to match the quarterly frequency of our economic data. Consequently, we chose to interpolate the economic variables to a daily frequency to ensure continuity and alignment with the CAC 40 data.

[1] For example, the variable 'Consumer Price Index' was not available for France during our study period but was available for the USA

The interpolation was performed using a linear method. For each economic variable, we estimated its values on a daily basis by linearly interpolating between the known quarterly values. This method assumes a straight-line relationship between consecutive quarterly data points and fills in the missing daily values accordingly.

As a result, we obtained a continuous dataset with daily frequency for all economic variables. Finally, we merged this interpolated dataset with the CAC 40 time series and removed any observations for which the CAC 40 was not available -such as weekends-. This ensured that the final dataset was complete and aligned with the trading calendar of the CAC 40.

3.2.2 Technical indicators

Since predicting the CAC 40 is a challenging task, we incorporated technical indicators alongside the macroeconomic variables presented. For the prediction purposes, we will use the 1-lag values of these technical indicators.

The technical indicators were constructed following the methods outlined in (Vijh et al. 2020; Patel et al. 2015) :

Table 2: Selected technical indicators and their formulas

Name of indicators	**Formulas**
Stock High minus Low price	$H_t - L_t$
Stock Close minus Open price	$C_t - O_t$
Simple n(here 5, 10, 20)-days Moving Average	$\frac{C_t + C_{t-1} + \cdots + C_{t-n+1}}{n}$
Weighted n(here 5, 10, 20)-days Moving Average	$\frac{n(C_t) + (n-1)(C_{t-1}) + \cdots + 1(C_{t-n+1})}{n(n-1) + \cdots + 1}$
Momentum	$C_t - C_{t-9}$
Stochastic $K\%$	$\frac{C_t - LL_t}{HH_t - LL_t} \times 100$
Stochastic $D\%$	$\frac{\sum_{i=0}^{k} K_i}{10}$
Moving Average Convergence Divergence (MACD)	$EMA(12)_t - EMA(26)_t$
Relative Strength Index (RSI)	$100 - \frac{100}{1 + \sum_{i=0}^{n} UP_{t-i}/n) / \sum_{i=0}^{n} DW_{t-i}/n)}$
Larry William's $R\%$	$\frac{H_t - C_t}{H_t - L_t} \times 100$
CCI (Commodity Channel Index)	$\frac{M_t - SM_t}{0.015 \cdot D_t}$

Lecture note : C_t represent the closing price, O_t the open price, L_t the low price, and H_t the high price at time t. LL_t and HH_t are the lowest low and highest high in the last t days respectively.EMA is the exponential moving average. The midpoint (M_t), simple moving average (SM_t), and deviation (D_t) are calculated as: $M_t = \frac{H_t + L_t + C_t}{3}, SM_t = \frac{\sum_{i=1}^{n} M_{t-i+1}}{n}, D_t = \frac{\sum_{i=1}^{n} |M_{t-i+1} - SM_t|}{n}$. Finally, UP_t represents the upward price change, while DW_t represents the downward price change at time t.

The second two technical indicators are moving averages (MA), which smooth out price data by creating a constantly updated average. This projects uses 5-day, 10-day, and 20-day Simple Moving Average (SMA) and Weighted Moving Average (WMA) for predicting short-term trends. If the price is above the moving average, the trend is considered "up"; otherwise, it is "down" (*Simple Moving Averages Make Trends Stand Out* 2024).

Stochastic oscillators (STC K%, STC D%, Williams R%) indicate trends: if their value at time t exceeds that at $t-1$, the trend is "up" $(+1)$; otherwise, it is "down" (-1).

MACD follows the trend of the stock, i.e. if MACD goes up then stock price also goes up and vice-a-versa. The Relative Strength Index (RSI), ranging from 0 to 100, identifies overbought (> 70, -1) and oversold (< 30, $+1$) levels. For RSI values between 30 and 70, the trend depends on whether RSI at t is higher or lower than at $t-1$.

The Commodity Channel Index (CCI) measures deviations from the average price, identifying strength (high positive values) or weakness (low negative values) and signaling overbought or oversold levels.

Momentum measures the rate of price change, with positive values indicating an "up" trend ($+1$) and negative values signaling a "down" trend (-1).

3.2.3 Stationarity analysis

Stationarity of variables is a crucial property for time series analysis. To assess stationarity, we applied the KPSS test(Kwiatkowski-Phillips-Schmidt-Shin). This test evaluates the null hypothesis that a time series is stationary around a level. A high p-value (> 0.05) suggests stationarity ($I(0)$), while a low p-value indicates the need for differencing ($I(1)$ or higher).

For each variable, we performed the KPSS test both at the level and after first differencing. Based on the results, we classified variables as $I(0)$, $I(1)$, or $I(2)$. In the table in the annex 12, a summary of the stationarity results is presented.

These results informed the selection of variables for further modeling, ensuring that transformations like differencing were applied appropriately to achieve stationarity where required.

4 Models and results

This section provides an overview of the approaches used in this project, distinguishing between univariate models, which rely solely on historical price data, and multivariate models, which incorporate explanatory covariates. It also justifies the methodological choices for the different model families.

For most time series models, it is essential that the series be stationary. Hence, we often use the log return which is close to the price series relative variation (return), and is stationnary (p-value = 0.01). The log return is defined as follows:

$$r_t = \log(\frac{P_t}{P_{t-1}})$$

where P_t is the closing price at time t.

To measure the performance of the models the root mean squared error (RMSE). Since, traditional loss functions such as root mean square forecast error is very sensitive to outliers, we used more robust mean absolute error (MAE).

Finally, for the sake of completeness, the R^2 score will also be displayed with the other performance metrics, particularly for deep learning models. It measures the proportion of the variance of our target variable explained by the model - here the stock price -, with a value of 1 indicating a perfect prediction and 0 a prediction equivalent to the mean.

$$\begin{cases} \text{RMSE} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2}, \\ \text{MAE} = \frac{1}{N}\sum_{i=1}^{N}|y_i - \hat{y}_i|, \\ R^2 = 1 - \frac{\sum_{i=1}^{N}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{N}(y_i - \bar{y})^2}. \end{cases}$$

4.1 Time Series Models Without Covariates

4.1.1 ARIMA model

To begin our analysis, we examined the Autocorrelation Function (ACF) and Partial Autocorrelation Function (PACF) (fig. 6) of the logarithmic returns of the CAC 40. These plots provided valuable insights into the temporal dependencies and potential lags for our ARIMA modeling. As we can see on the graphics below, the empirical autocorrelations are not significant beyond lag 1. This suggests an MA(0) model, which is unusual, as typical models require an order of at least 1.

Regarding the partial autocorrelations, while the overall pattern does not show significant lags, certain individual peaks appear significant at specific orders: lag 3, and then from lags 5 to 7. Based on these observations, we decided to test models with lags up to order 7 to account for these potential dependencies.

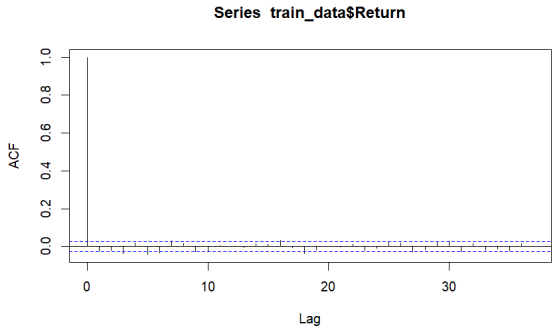


Figure 4: ACF of the log-returns of CAC40.

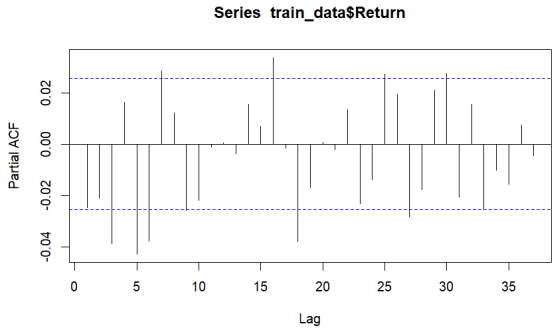


Figure 5: PACF of the log-returns of CAC40.

Figure 6: ACF and PACF of the log-returns of CAC40.

The ARIMA model (*AutoRegressive Integrated Moving Average*) is a time series model defined by three parameters: p (number of autoregressive terms), d (degree of differencing), and q (number of moving average terms). The ARIMA model can be expressed mathematically as:

$$\Phi(B)(1-B)^d Y_t = \Theta(B)\epsilon_t$$

where:

- $\Phi(B) = 1 - \phi_1 B - \phi_2 B^2 - \cdots - \phi_p B^p$ represents the autoregressive part,
- $(1-B)^d$ indicates the differencing operator,
- $\Theta(B) = 1 + \theta_1 B + \theta_2 B^2 + \cdots + \theta_q B^q$ represents the moving average part,
- ϵ_t is white noise, and B is the backshift operator ($BY_t = Y_{t-1}$).

4.1.1.1 Results of ARIMA model

Initially, we applied an ARIMA model to the closing prices of the CAC 40. This approach was justified as the differenced series of closing prices was found to be stationary. While the ARIMA model fit the training data well and aligned closely with the observed prices during the training period, it failed to generalize effectively. Specifically, predictions for the test period (post-2023) remained constant. This limitation can be attributed to the fact that, although ARIMA is known for its robustness and efficiency in short-term forecasting compared to more complex models (Obthong et al. 2020), it struggles with series that exhibit stochastic jumps or

more intricate patterns. Moreover, the forecast results from ARIMA are based solely on past values of the series and previous error terms, making it less capable of predicting the inherent randomness and sudden fluctuations present in such a complex time serie.

Given these limitations, we proceeded to fit an ARMA model directly to the logarithmic returns of the CAC 40. However, despite the stationary nature of the returns, the ARMA model struggled to provide accurate predictions, both during the training and test periods. To optimize the ARMA model, we performed a parameter search using the Akaike Information Criterion (AIC) for p and q values between 1 and 6, selecting only models that passed the Ljung-Box test. The optimal model, minimizing the AIC, was an ARMA(2,5).

The ARMA(2,5) model demonstrated the following performance on the training set:

- **RMSE**: 0.0142
- **MAE**: 0.0098

However, when focusing on specific periods, such as October 2008 during the financial crisis, the performance metrics were significantly worse:

- **RMSE**: 0.0489
- **MAE**: 0.0393

These results highlight the limitations of the ARMA model. As the predictions are averages over many samples, irregularities in the data are smoothed out, resulting in lower MAE values, particularly for logarithmic returns. Additionally, during periods of high volatility, such as crises, the model's performance deteriorates sharply, even on the training set (figure 20. Furthermore, ARMA predictions exhibit delays and reduced intensity, which are inherent to the model structure. On the test set, the ARMA predictions were even poorer, becoming constant and failing to capture any meaningful patterns.

Finally, to convert the predicted logarithmic returns back to the original scale of closing prices, the cumulative product of the predicted returns was multiplied by the last observed closing price from the training set. This allowed the predictions to be placed on the same scale as the original closing prices.

As we can see in the graph below, the ARMA model fails to capture meaningful patterns in the test set, and the predictions become constant and low over time, which is observed here by a small linear augmentation of the closing prices:

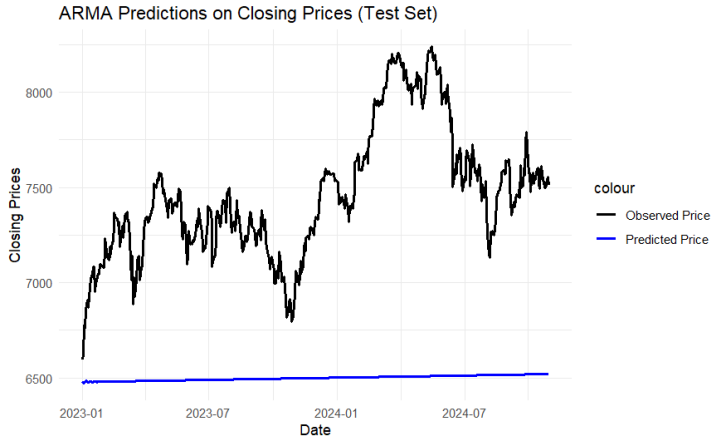


Figure 7

4.1.2 FARIMA Model

The **FARIMA** (Fractionally Integrated AutoRegressive Moving Average) model extends the ARIMA model by allowing fractional differentiation. It is particularly suited for modeling time series that exhibit long memory and stationarity after fractional differencing.
The nature of memory in financial time series, such as stock market returns, has been extensively studied in the literature. While studies like Campbell, Lo, and MacKinlay (1997) (Campbell et al. 1998) suggest that financial returns generally exhibit **short memory** behavior, others, such as Lo (1991) (Lo 1991), highlight the possibility of **long memory** in certain markets. The occurrence of long memory is inconsistent and remains a topic of debate, as its detection often depends on the dataset and methodology. To explore this, we examine whether our dataset demonstrates **short memory** or **long memory** characteristics.

4.1.2.1 Results of FARIMA : empirical analysis of memory properties

To determine whether financial returns exhibit short memory or long memory, we conducted the following analyses:

1. **Autocorrelation Function (ACF)**: We plotted the ACF of the returns to assess the persistence of autocorrelations over time. The ACF showed a **rapid decay** in autocorrelations, which is characteristic of short memory 6.

2. **Hurst Exponent (H)**: The Hurst exponent measures the memory properties of a time series:

   - $H = 0.5$: No memory (e.g., random walk).
   - $H > 0.5$: Long memory, indicating persistent influence of past values.
   - $H < 0.5$: Short memory, with past values' influence diminishing quickly.

   We estimated H using the `hurstexp` function in R (Weron 2002), which employs:

   - **Simple R/S**: Basic rescaled range analysis, potentially biased for small samples.
   - **Corrected R/S**: Reduces bias in small datasets.
   - **Empirical Hurst**: Estimates H based on variance or range scaling.
   - **Corrected Empirical Hurst**: Bias-corrected version of the Empirical Hurst method.
   - **Theoretical Hurst**: Computes H using fractal or self-similarity assumptions.

   The estimated Hurst exponents are:

Method	**Hurst Exponent (H)**
Simple R/S	0.4988
Corrected R/S	0.5198
Empirical Hurst	0.5343
Corrected Empirical Hurst	0.5042
Theoretical Hurst	0.5286

Table 3: Hurst Exponent Estimates for the Time Series.

   The estimates are close to 0.5, with some methods suggesting slight long memory ($H > 0.5$) and others indicating no memory ($H \approx 0.5$). Overall, the series exhibits **weak or no significant long memory**, consistent with ACF findings.

In conclusion, based on the ACF and Hurst exponent, the financial returns of the CAC40 exhibit **short memory** behavior.As a result, a FARIMA model is unnecessary, as modelling a FARIMA in a case where memory is short could add unnecessary complexity without offering any significant advantage.

Since the previous models were unable to capture the overall trend of our data due to their complex nature, we will focus in the next section on modeling another key aspect of the return series: **volatility**.

4.1.3 GARCH model

The GARCH model is a conditional variance model widely used in finance to model the volatility of financial assets and forecast future volatility. It is important to note that the GARCH model does not forecast the actual data or returns; rather, it focuses on predicting the conditional variance (volatility) of the data. This distinction is crucial, as the GARCH model provides insights into the variability or risk associated with financial assets over time, without directly modeling the future values of the stock prices or returns themselves.

It captures key features observed in financial time series, such as the dependence within the series and the presence of volatility clustering. In the CAC 40 series, the price variation data typically exhibits low autocorrelation, making it resemble white noise. However, when examining the squared returns, the time series generally shows strong autocorrelation, evidenced by the presence of volatility clusters. This suggests that the returns represent a heteroscedastic process, making the GARCH model particularly relevant for the scope of our study.

Model Specification

The conditional variance model specifies the parametric form of the conditional variance process σ_t^2. The innovation distribution corresponds to the distribution of the i.i.d process ϵ_t. The GARCH(p,q) model is defined as follows:

$$X_t = \mu + \epsilon_t,$$

with $\epsilon_t = \sigma_t \eta_t$, where η_t is the standardized residuals, an i.i.d process with mean 0 and variance 1.

The conditional variance σ_t^2 is given by:

$$\sigma_t^2 = \omega + \sum_{i=1}^{p} \alpha_i \epsilon_{t-i}^2 + \sum_{j=1}^{q} \beta_j \sigma_{t-j}^2,$$

where ω is a constant, α_i and β_j are the parameters of the model estimated by maximum likelihood, and p and q are the orders of the model.

GARCH model coefficients are fit using the Maximum Likelihood Approach. The estimation process used to obtain the likelihood estimates for ω, β, and α is based on recursively iterating through the log-likelihood function modeling the GARCH coefficient estimates. The log-likelihood function we aim to maximize is defined as (Cryer 2008):

$$L(\omega, \alpha, \beta) = -\frac{n}{2}\log(2\pi) - \frac{1}{2}\sum_{i=1}^{n}\left\{\log\left(\sigma_t^2|t-2\right) + \frac{r_t^2}{\sigma_t^2|t-1}\right\}$$

4.1.3.1 Results of GARCH

Motivated by comments in (Franke, Härdle, and Hafner 2004) suggesting that, in practical applications, GARCH models with smaller orders often sufficiently describe the data and that

in most cases GARCH(1,1) is adequate, we considered four different combinations of p=0, 1 and q=1, 2 for each period to train the GARCH model, assuming that the standardized residuals follow a normal distribution.

Using the AIC criterion to select the best model, we concluded that GARCH(1,1) is indeed the best model, with an AIC of -77,383.98. Although the GARCH(1,2) model has an AIC with a relative gap of only -0.05%, we prefer GARCH(1,1) due to its superior balance between performance and simplicity.

p	q	AIC	Relative gap (in %)
0	1	-31249.09	-59.64
0	2	-53731.83	-30.60
1	1	-77421.86	-0.00
1	2	-77383.98	-0.05

For the performance of this model, we analyze the Ljung-Box test for the standardized residuals and the squared standardized residuals. This test is used to detect autocorrelation in the residuals, which helps determine whether the GARCH(1,1) model has adequately captured the time-dependent structure in the data. The results show that the standardized residuals are not autocorrelated.

As a goodness of fit measure of departure from normality, the Jarque Bera test statistic was applied to the residuals. The test returned a pvalue inferior to $2,2 \times 10^{-16}$, leading us to reject the null hypothesis of normality for the distribution of the residuals. This implies that the data to be fitted is not normally distributed.

To account for this, other distributions were assessed: the Student-t distribution (Std), for the skewed Student-t distribution (Sstd), the generalized error distribution (Ged), the skewed generalized error distribution (Sged)

Each of these distributions was fitted to the GARCH(1,1) model, and the performance was evaluated using metrics such as RMSE, MSE, and HMAE. The results are presented in the following table:

Model	RMSE	MSE	HMAE
Norm GARCH(1,1)	0.0060	0.0051	5.1457
Std GARCH(1,1)	0.0060	0.0050	5.0086
Sstd GARCH(1,1)	0.0059	0.0049	4.9721
Ged GARCH(1,1)	0.0060	0.0050	5.0548
Sged GARCH(1,1)	0.0060	0.0050	5.0111

To compute these metrics, since the GARCH model predicts only the volatility, we rely on a proxy variable for volatility. When intraday data is not available to calculate proxies such as Realized Volatility or Realized High-Low Volatility, the squared return is generally considered the second-best proxy for variance, as noted in (De Vilder and Visser 2007).

As we can see, the best model is the GARCH(1,1) model with the skewed Student-t distribution (Sstd), as it achieves the lowest RMSE (0.0059), MSE (0.0049), and HMAE (4.9721). This indicates that the skewed Student-t distribution provides the best fit for the data among the tested models. Moreover, when assessing the autocorrelation of the standardized residuals and squared standardized residuals from this model using the Ljung-Box test, we observe that there is no significant autocorrelation present up to lag 10.

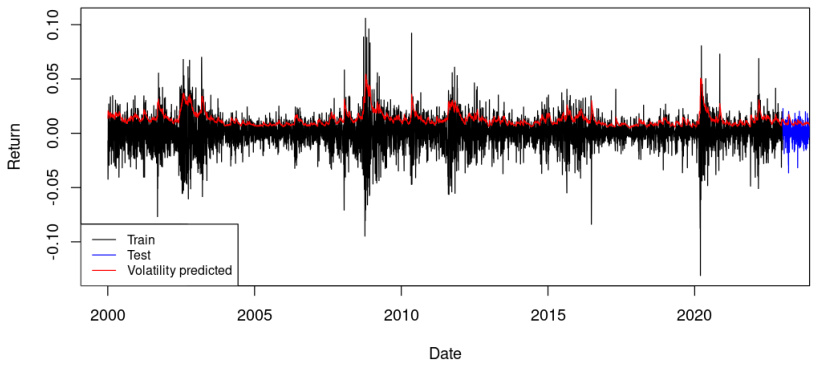


Figure 8: Volatility forecast of log-returns using GARCH(1,1)

Standard GARCH models assume that positive and negative error terms have symmetric effects on the volatility. In other words, good and bad news have the same impact on the volatility in these models. In practice, however, this assumption is frequently violated, particularly in the case of stock returns, where the volatility tends to increase more significantly after bad news than after good news. This phenomenon is known as the Leverage Effect, which was first introduced by (Black 1976).

To address this limitation, alternative models such as the Exponential GARCH (EGARCH) (Nelson 1991) and Threshold GARCH (TGARCH) (He, Teräsvirta, and Malmsten 2002) have been developed. Hence we implemented these models using the skewed Student-t distribution and check the performance of these models as seen in the following table :

Model	RMSE	MAE	HMAE
Sstd EGARCH(1,1)	≈ 0.0000	0.0051	4.9362
Sstd TGARCH(1,1)	≈ 0.0000	0.0050	4.9370

Finally, the best model compared to the GARCH(1,1) model with the skewed Student-t distribution is the TGARCH model. However, since the difference in performance between the TGARCH(1,1) and the GARCH(1,1) models is negligible, we prefer the GARCH(1,1) model due to its simplicity and lower computational complexity.

This choice ensures a balance between performance and practicality, making the GARCH(1,1) model more suitable for applications requiring computational efficiency while maintaining accuracy. The conditional variance σ_t^2 from the GARCH (1,1) selected with skewed student-t distribution SSTD((skew=0.90, shape = 7.38).

is modeled as:

$$\sigma_t^2 = \omega + \alpha_1 \epsilon_{t-1}^2 + \beta_1 \sigma_{t-1}^2$$

where:

- $\omega \approx 0.000$ (constant term),
- $\alpha_1 = 0.097$ (impact of past squared residuals on current variance, i.e., the ARCH effect),
- $\beta_1 = 0.896$ (impact of past variance on current variance, i.e., the GARCH effect).

4.2 Time Series Models Incorporating Covariates

This section highlights multivariate models that incorporate macroeconomic and technical explanatory variables in order to enhance forecasting.

4.2.1 ARMAX Model

The ARMAX(**AutoRegressive Moving Average with eXogenous inputs**) model is an extension of ARMA, incorporating external variables X_t:

$$Y_t = \Phi(B)Y_{t-1} + \Theta(B)\epsilon_t + \Gamma(B)X_t$$

where $\Gamma(B)$ is the polynomial capturing the effects of exogenous variables.

4.2.1.1 Results of ARMAX

For the ARMAX model, only stationary variables ($I(0)$) could be directly included. Non-stationary variables ($I(1)$) were differenced before integration into the model. Given the large number of potential explanatory variables, we limited our search to combinations of up to four variables to identify the best-performing model. The selection criterion was based on minimizing the RMSE and MAE on the training set. We also chose to keep the ARMA(2,5) model and add only the exogenous variables to perform our ARMAX model, as testing different combinations of variables already presented a high time complexity.

We first searched for the best combinations among the $I(1)$ variables by minimizing the RMSE on the training set, limiting the combinations to a maximum of four variables. Among the best combinations found, we then added up to four $I(0)$ variables to further improve the model.

The best combination of variables selected was: **CPIH_YTYPCT, UNR_us, SMA_5, Close_minus_Open, Momentum, Stochastic_K, CCI.**

We then ran a model with this combination and obtained the following performance metrics on the training and test sets (table 4):

Metrics	Training Set	Test Set
RMSE	862.4758	775.2753
MAE	659.5934	699.6763

Table 4: Performance metrics for the training and test datasets.

Thus, integrating the exogenous variables helped improve the RMSE by approximately 300 points (compared to the ARMA model). However, this improvement was still insufficient to make the model adequately predict on the test sample. Notably, the predictions follow the general trend of the CAC 40 prices, like downfalls for example, but, as shown in the graph 9, the prices exhibit a downward trend in the long term, which is not desirable for our case.

This can be explained by the fact that the model remains fundamentally an ARMA model (or at least its underlying structure), and as such, it accumulates its errors over time. Since ARMA models rely heavily on past values and errors, this accumulation of errors leads to an increasing discrepancy between predicted and observed values as time progresses.

4.2.2 ARDL model

The ARDL (Autoregressive Distributed Lag) model is an econometric model particularly suitable when the variables under study are integrated of different orders, typically I(0) and I(1). Unlike other models that require all variables to be of order I(1), the ARDL model allows for the analysis of dynamic relationships in both the short and long term between variables, even when some are integrated at the level (I(0)) and others in the first difference (I(1)).

Mathematically, an ARDL model can be expressed as follows:

$$Y_t = \alpha + \sum_{i=1}^{p} \beta_i Y_{t-i} + \sum_{j=0}^{q} \gamma_j X_{t-j} + \epsilon_t$$

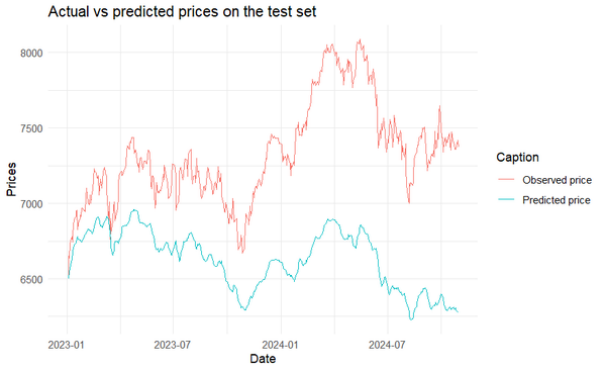


Figure 9

where:

- Y_t is the dependent variable at time t,
- X_t is the explanatory variable,
- p and q are the orders of lags for Y and X,
- α is the intercept, and ϵ_t is the stochastic error term.

This model is frequently used in situations where the relationship between variables may change over time, and the aim is to test not only the short-term effects but also the long-term effects of the explanatory variables. In our case, the ARDL model is particularly relevant to analyze the impact of various economic and financial variables on returns of the CAC 40 index, accounting for both immediate and delayed effects.

However, the application of this model relies on several assumptions. First, it is necessary that the variables are cointegrated, meaning that they share a long-term relationship. Furthermore, the variables must be of type I(0) or I(1), which was verified using the KPSS test, as described in previous section. Another important criterion is that the explanatory variables should not be strongly multicollinear. Lastly, it is necessary that the target variable, i.e., the logarithmic returns, does not Granger cause the exogenous variables (Stock and Watson 2020) -which is called the exogeneity assumption .

The Granger causality test is used to examine whether one variable can be used to predict another within the ARDL model. In other words, it helps determine whether the explanatory variable has an effect on the dependent variable by analyzing whether the lags of one help predict the future values of the other.

The hypotheses of the test are as follows:

- Null Hypothesis: The explanatory variable does not Granger cause the target variable (absence of causality).
- Alternative Hypothesis: The explanatory variable Granger causes the target variable (presence of causality).

4.2.2.1 Results of ARDL

The Granger causality test was performed on all the explanatory variables to determine if there exists a causal relationship between the logarithmic returns and the other variables. The results of these tests are summarized in the table in annex 13, where each explanatory variable is tested for its causality both on the returns and in the reverse direction.

We observe that the only variable that is not Granger-caused by our dependent variable Return is the one concerning inflation *CPIH_ YTYPCT*. This implies that it is the only variable we can consider when building our ARDL model. Consequently, we can immediately notice that the model will be limited in its scope.

Following this analysis, we applied the ARDL model using the `auto_ardl` function, considering only *CPIH_ YTYPCT* in the model. After testing various lags (with a maximum of 10 lags), the optimal lag order that minimized the AIC was found to be 3 lags for both *Return* and *CPIH_ YTYPCT*. The results of the ARDL estimation are summarized as follows 5:

Table 5: ARDL Estimation Results

Variable	Estimate	Standard Error	t-value	p-value
(Intercept)	0.0004465	0.0003161	1.412	0.15786
L(Return, 1)	-0.0288242	0.0130307	-2.212	0.02700 *
L(Return, 2)	-0.0241636	0.0130234	-1.855	0.06359 .
L(Return, 3)	-0.0398000	0.0130335	-3.054	0.00227 **
CPIH_YTYPCT	0.0088849	0.0287519	0.309	0.75732
L(CPIH_YTYPCT, 1)	-0.0262937	0.0483087	-0.544	0.58627
L(CPIH_YTYPCT, 2)	0.0937777	0.0482931	1.942	0.05220 .
L(CPIH_YTYPCT, 3)	-0.0766398	0.0287733	-2.664	0.00775 **

Based on this table, we observed that the coefficients for `L(Return, 1)`, `L(Return, 3)`, and `L(CPIH_YTYPCT, 3)` are significant at the 5% level (`p-value` < 0.05). However, the `CPIH_YTYPCT` variable itself and its lags do not exhibit significant effects on the return series, as indicated by the `p-value` of 0.75732 for `CPIH_YTYPCT` and the values for its lags.

The Durbin-Watson statistic, which tests for autocorrelation in the residuals, is 1.9986. This value is close to 2, suggesting that there is no significant autocorrelation in the residuals. The corresponding `p-value` of 0.4721 from the test confirms this, indicating that the residuals are not autocorrelated. Upon further examination, the residuals of the ARDL model appear to follow a normal distribution, but there is a notable level of volatility. This suggests that while the model is reasonable in its assumptions, it does not fully capture all of the variance in the returns series. This indicates that there might be additional variables or factors influencing the returns that were not included in the model -which is understandable considering that we only add one variable to our model, and that the coefficients are not significant for most of them-.

Next, we conducted the Bounds F-test for cointegration. The null hypothesis is that there is no cointegration, while the alternative hypothesis is that there is a possible cointegration relationship between the variables. The results are as follows:

```
F-statistic = 745.03, p-value = 1e-06
```

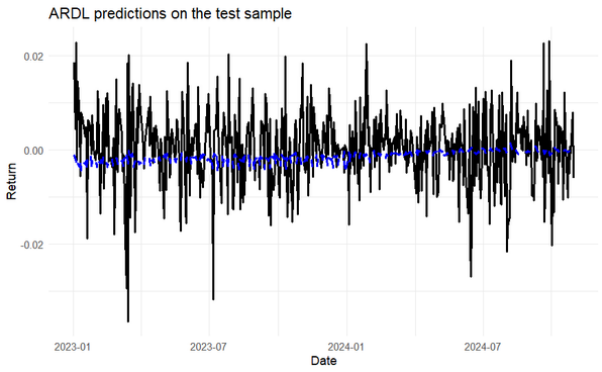
The low `p-value` suggests the possibility of cointegration between the variables, confirming that there is a long-term relationship between the logarithmic returns of the CAC 40 index and the inflation variable (`CPIH_YTYPCT`).

After estimating the ARDL model, we proceeded to make predictions using the model. Forecasting with ARDL models is straightforward because these models can be converted into regular linear regression models, which allow us to predict future values based on the estimated coefficients.

In our case, the predictions were made for the logarithmic returns of the CAC 40 index. However, as we can see from the forecast plot (Figure 10), the model fails to capture the volatility of the return series effectively. This is evident as the predicted returns appear to be overly smooth and do not reflect the sharp fluctuations typically observed in financial time series.

Moreover, the predicted returns are negative for the majority of the forecast period, which contrasts with the observed growth in the CAC 40 index during the same period. This suggests

Figure 10: Predicted logarithmic returns of the CAC 40 index using the ARDL model



that the model's predictions do not align well with the actual data, particularly with regard to the direction of the market. The forecasted negative returns lead to a decaying trend in the predictions, while we would expect a growing trend for the CAC 40 index over this period.

This issue is likely due to the model's inability to adequately capture the volatility and structural dynamics of the market. Financial time series often exhibit volatility clustering and other patterns that demand more sophisticated modeling approaches. We will then explore a more flexible and adaptable model to address these challenges in the following section.

4.2.3 Prophet model

Prophet is a forecasting model developed by Taylor and Letham in 2017, specifically designed to predict time series data with non-linear trends, strong seasonal effects, and irregular changes in the data. Stock markets are generally influenced by various external factors, such as economic events, government policies, and seasonal business trends, which typically result in non-linear and irregular fluctuations in asset prices and, consequently, in the value of stock market indices. Prophet is capable of modeling these dynamics flexibly, taking into account exceptional events that can impact stock prices. This makes it a relevant candidate for the scope of our study.

4.2.3.1 Model Specification

Prophet is a time series forecasting methodology based on an additive model, whose specification is as follows:

$$y(t) = g(t) + s(t) + h(t) + \epsilon_t \tag{1}$$

- $g(t)$: Models the long-term trend, which may be nonlinear.Prophet offers two types of trend models:
  - *Saturated Growth Model*: Suitable for data with growth limits.

$$g(t) = \frac{C(t)}{1 + \exp(-(k + a(t)^T\delta)(t - (m + a(t)^T\gamma)))} \tag{2}$$

    where $C(t)$ is the carrying capacity, k is the growth rate, and m is an offset parameter. The term $a(t)^T\delta$ allows for adjustments at changepoints.
  - *Piecewise Linear Model*: Suitable for data with abrupt trend changes.

$$g(t) = (k + a(t)^T\delta)t + (m + a(t)^T\gamma) \tag{3}$$

    Here, k is the baseline growth rate, and δ represents the rate adjustments at changepoints.

The vector $a(t)$is defined as follows:

$$a_j(t) = \begin{cases} 1, & \text{if } t \geq s_j, \\ 0, & \text{otherwise.} \end{cases}$$

Here, s_j represents the time at which the j^{th} changepoint occurs. The vector $a(t)$ has a dimension of S, where S is the number of changepoints, and it indicates which changepoints are active at time t.

- $s(t)$: Models periodic changes due to seasonal factors, such as weekly or yearly effects, using a Fourier series representation:

$$s(t) = \sum_{n=1}^{N} \left[a_n \cos\left(\frac{2\pi nt}{P}\right) + b_n \sin\left(\frac{2\pi nt}{P}\right) \right] \tag{4}$$

  where P is the period, and N determines the complexity of the seasonal pattern.

- $h(t)$: Captures the impact of holidays or major events on the time series:

$$h(t) = Z(t)\kappa \tag{5}$$

  where $Z(t)$ is an indicator matrix for holidays, and κ represents their corresponding impacts.

- ϵ_t: Represents the irreducible error term for idiosyncratic variations not accounted for by the model.

This formulation is therefore closer to a fitting exercise, which is intrinsically different from the structure of the majority of time series models that explicitly take into account the temporal dependency structure of the data. Nevertheless, it offers a number of practical advantages, such as flexibility in specifying trends and seasonality; observations don't have to be regularly spaced, and there's no need to interpolate missing values. Additionally it allows for the inclusion of additional regressors to account for external factors influencing the time series. These regressors can be continuous or binary variables that represent external signals such as macroeconomic indicators. Prophet models these regressors additively, as follows:

$$y(t) = g(t) + s(t) + h(t) + \sum_{k=1}^{K} \beta_k x_k(t) + \epsilon_t$$

Here, $x_k(t)$ represents the value of the k^{th} regressor at time t, and β_k is its corresponding coefficient, which is estimated during model fitting.

The estimation process in Prophet revolves around efficient fitting of the additive components $g(t)$, $s(t)$, and $h(t)$. These are estimated simultaneously using a **maximum a posteriori (MAP)** approach.

4.2.3.2 Results of Prophet

To fully leverage the advantages offered by the Prophet model, particularly its ability to detect trend breaks and its flexibility with respect to the non-stationarity of time series, we have chosen to apply our modeling directly to the stock index.

For our modeling with Prophet, we included both annual and weekly seasonality, as well as special events such as presidential and legislative elections, and public holidays. The weekly seasonality, known in the literature as the **"week effect"**, captures regular variations observed according to the market's opening days. The inclusion of public holidays is justified by their

impact on returns, as illustrated by the **"holiday effect"** (Lakonishok & Smidt, 1988). Finally, since the CAC40 reflects the performance of French companies, the inclusion of elections as special events is relevant, as political changes directly influence the economic climate and investor expectations.

To provide additional, non-redundant information to the model, we have selected exogenous variables that are strongly correlated with the stock index (with a correlation of at least 0.6), but not correlated with each other.This step is particularly important, as models like Prophet handle multicolinearity poorly. Specifically, we chose `UNR_us`, `CPI_us`, `GDP`, and `WMA_20`.

The model's performance on the test sample is presented below (fig 11) :

Figure 11: Actual values vs predictions on test sample : prophet model

Compared to the models presented so far, Prophet yields relatively satisfactory results. The relatively low RMSE suggest that the forecasts are quite close to the actual values, with a moderate mean absolute deviation (RMSE = 148.51). The R^2 of 0.81 indicates that the model explains 81% of the price variance, reflecting a good ability to capture trends and fluctuations. These results (table 6) suggest that Prophet, with the integration of exogenous variables and special events, provides reasonably reliable short-term forecasts for the CAC40.

Table 6: Model performance on the test and training samples

Metric	**Test Sample**	**Train Sample**
RMSE	148.50	115.76
MAE	118.36	83.02
R^2	0.81	0.98

To complement the analysis of the results, examining the model's components helps to better understand the influence of each factor on the forecasts and validate the relevance of the modeling.

Figure 12: prophet model component's analysis

The component graph (fig. 12) highlights changes in the trend's slope at different points in time, demonstrating that the Prophet model is capable of detecting structural breaks. Additionally, the weekly seasonality reveals a "weekly effect," with a downward trend between Sunday and Monday, and an upward trend between Saturday and Sunday, confirming the influence of the weekly cycle on market fluctuations. Moreover, exogenous variables play a key role in capturing short-term variations. By analyzing the individual contributions of these variables, we've observed that GDP and the 20-day smoothed moving average before the observation are the main contributors. GDP reflects broader economic trends, while the smoothed moving average captures immediate market fluctuations, thus enhancing the accuracy of short-term forecasts.

The flexibility of the Prophet model, which allowed us to achieve a significant gain in predictive performance, now makes it relevant to explore recent advancements in machine and deep learning, which open up new perspectives for financial market forecasting. These models offer a more sophisticated and adaptable approach to forecasting by leveraging additional external variables that can enhance our predictions.

4.3 Machine and deep learning models

Machine learning and deep learning models have become essential tools for uncovering patterns, relationships, and making predictions from data, especially in complex domains like financial forecasting. In this study, we aim to predict the daily closing price of the CAC40 index by leveraging these advanced methods. Machine learning techniques, such as Ridge Regression, offer simplicity and interpretability, serving as strong baselines for comparison. On the other hand, deep learning models exploit interconnected layers of neurons to automatically extract relevant features and capture complex, non-linear relationships between variables. These models are particularly powerful for financial data, as they do not require the assumption of stationarity and can adapt to intricate market dynamics. By combining the strengths of machine learning and deep learning, this section explores their potential for delivering accurate and reliable forecasts of the CAC40 index.

We explored several models during this study; however, only those that demonstrated strong performance have been selected for presentation in this report. For further details and to review the other tested models, please refer to the annex.

4.3.1 Ridge regression

Ridge regression is a linear regression model that incorporates L_2-regularization to prevent overfitting by penalizing large coefficients. By adding a penalty term proportional to the square of the coefficients' magnitude, it balances the trade-off between model complexity and prediction accuracy, making it particularly effective when dealing with multicollinearity or high-dimensional data.

The Ridge regression model can be expressed as:

$$\hat{\beta} = \arg\min_{\beta} \left(\sum_{i=1}^{n} (y_i - X_i\beta)^2 + \lambda \sum_{j=1}^{p} \beta_j^2 \right),$$

where X_i represents the predictor variables, y_i is the target variable, λ is the regularization parameter controlling the penalty strength.

Choosing the optimal λ is crucial for balancing bias and variance, and this is typically achieved through cross-validation. Ridge regression's simplicity and interpretability make it a valuable baseline for evaluating the performance of more complex models.

4.3.1.1 Results of the ridge regression

By applying Ridge Regression to the entire training dataset, we obtain the results summarized in Table 7. At first glance, observing Figure 13, which shows the actual vs. predicted closing prices of the CAC 40 index, one might suspect an overfitting effect in the model. However, as demonstrated in Table 7, the model does not exhibit any signs of overfitting. The performance metrics on both the training and test sets are consistent, indicating that the regularization parameter (λ=0.01) effectively prevents overfitting by achieving a balance between model complexity and generalization.

The high R^2 values on both sets suggest that the model captures the underlying patterns in the data well, while the relatively low RMSE, and MAE values confirm its accuracy in predicting the target variable. The small discrepancy between the training and test set errors is expected and reflects natural variability in the data rather than model bias.

Table 7: Performance metrics for train and test sets with optimal regularization ($\alpha = 0.01$)

Metric	**Train**	**Test**
R^2	0.99	0.96
RMSE	62.60	64.99
MAE	44.67	49.14

These results demonstrate that Ridge Regression, when appropriately regularized, can serve as a robust baseline for forecasting tasks, providing both interpretability and reliable predictions. Future work could focus on comparing Ridge Regression with more complex non-linear models to explore potential improvements in capturing subtle relationships within the data.

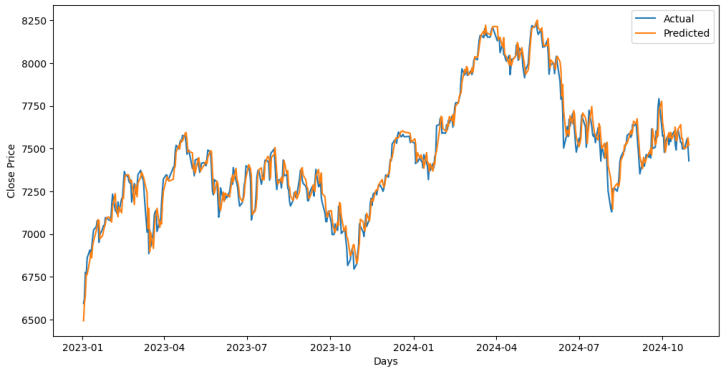


Figure 13: Actual vs predicted closing price of CAC 40 in the test dataset using ridge regression model

Furthermore, we can extract the SHAP values of the variables to understand their contribution in predicting the CAC 40 closing price index. By using SHAP (SHapley Additive exPlanations), we gain insights into the importance of each feature and how it influences the model's predictions.

The idea behind SHAP feature importance is straightforward: features with large absolute Shapley values are considered important. To determine global feature importance, we calculate the average of the absolute Shapley values for each feature across the entire dataset. The figure below (14) illustrates the SHAP feature importance for the trained Ridge Regression model, with features sorted in decreasing order of importance. As observed, the most important features include the weighted moving averages (WMA) and simple moving averages (SMA) for 5-day and 20-day periods.

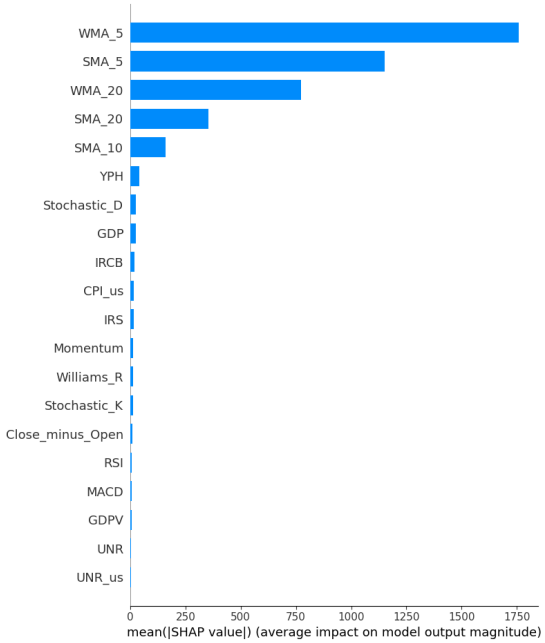


Figure 14: SHAP features importance for trained ridge regression

In addition to feature important, we use the summary plot in annexe 21 to understand the effects of each variable: each point represents a Shapley value for a feature and a specific instance, with the y-axis indicating the feature and the x-axis showing the Shapley value. The color reflects the feature value, ranging from low (blue) to high (red), and the features are sorted by their global importance.

The analysis highlights that short-term and medium-term moving averages, particularly WMA_5, SMA_5, and WMA_20, are the most influential features in predicting the CAC 40 closing price. WMA_5 exhibits a complex relationship with the target variable, while SMA_20 shows that higher values positively contribute to predictions and lower values negatively contribute. Less significant features, such as YPH and Stochastic_D, play a minor role in the model's predictions. This provides valuable insights into how the model interprets and leverages the input features.

Ridge Regression demonstrated strong performance and effectively highlighted the importance of key features such as moving averages. However, its linear nature may limit its capacity to capture intricate relationships and temporal dependencies within the data. To address these limitations, we explore deep learning approaches, specifically the Long Short-Term Memory (LSTM) model.

4.3.2 LSTM (Long-Short Term Memory)

LSTM (Long-Short Term Memory) model is a variant of RNN's which particularity is to be able to capture both short and long-term dependencies and leveraging the sequential structure of time series data. Each LSTM cell incorporates memory mechanisms through three main gates: the input gate, the forgetting gate and the output gate, as we can see in fig. 22. These gates enable the model to retain relevant information in a sequential context.

In concrete terms, to make the predictions, sliding windows on the data will be used: each window contain a certain number of time steps h, which is an indicator of how deep, from a fixed time, we want to delve in the data to make the prediction. The data will be normalized to ensure that all variables are on the same scale, helping LSTM to converge more efficiently (Luca and Oleksandr 2016).

Moreover, given that the performance of neural networks is highly dependent on the values of their hyperparameters, we will test several combinations of these in order to achieve the best possible results.

4.3.2.1 Results of LSTM with historical prices only

This first implementation was based solely on the temporal evolution of the Close prices of the CAC40.

Layer (type)	Output Shape
lstm (LSTM)	(None, 60, 50)
dropout (Dropout)	(None, 60, 50)
lstm_1 (LSTM)	(None, 50)
dropout_1 (Dropout)	(None, 50)
dense (Dense)	(None, 25)
dense_1 (Dense)	(None, 1)

Figure 15: A model used for time analysis

On table 15, we can see the structure of a model used for temporal analysis. Here, the length of the time sequence we described previously is fixed at 60, which means that, to predict a value at time t, the model is going to use the close prices from t-60 to t-1. The first layer of the network is a LSTM layer for which an output size is specified; (60,50) in our example: this means that for each of the 60 timesteps, the layer returns a vector of dimension 50 which represents the informations taken from the data. The larger the output size, the more information we collect. However, one must be aware that having a too large output size can lead to overfitting.

This LSTM layer is followed by a Dropout one whose role is to prevent overfitting of the model. Indeed, it randomly deactivates a given percentage of neurons during training in order to improve generalization.

We repeat the same scheme by putting another LSTM layer, this time without the temporal sequences dimension, followed also by another Dropout layer. This process, consisting of reducing the dimension of an output, by summation, maximum or another process, is called pooling in Deep Learning. We finish by 2 Dense layers. They link each neuron in a layer to all neurons in the next layer, enabling a complete combination of the information extracted by the previous layers. They are essential for synthesizing and transforming learned features into final outputs. Here, the output of the last layer corresponds to the stock price prediction.

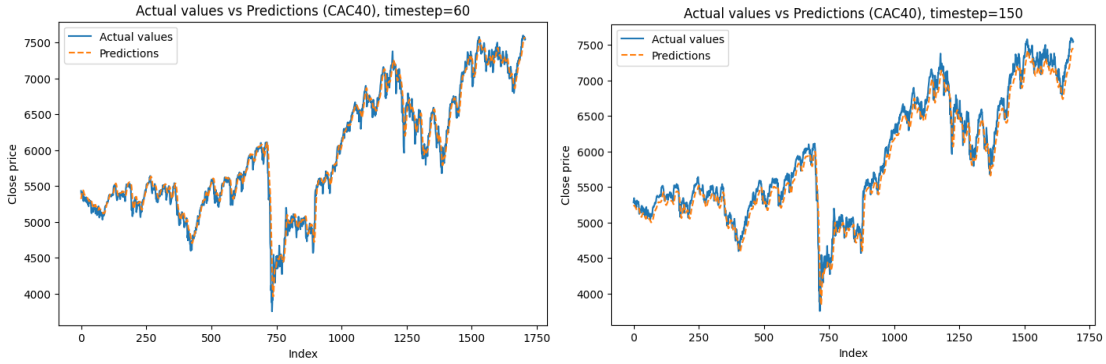


Figure 16: Comparison Predictions / Actual values

The above graphs in fig. 16 show the comparison between the performances of 2 models on the test set. The difference between them is the choosen size for the sliding window: on the left we have $h = 60$ and on the right, $h = 500$. The model seems to fit better with $h = 60$, which is confirmed by computing and comparing the MSE and the MAE ($MAE60 = 85.83, RMSE60 = 120.54$ vs $MAE150 = 127.11, RMSE150 = 153.19$). This result can be considered as an indication that the important relationships are found principally in short-term timesteps, thus short-term memory models are more efficient than long-term memory ones. This observation is consistent with our earlier findings, where we did not detect significant long-term relationships when attempting to implement a FARIMA model. This suggests that, in the context of predicting the CAC40's daily closing price, the market dynamics are primarily influenced by short-term factors.

The results provided by this model are quite satisfactory; however, they are inferior to those obtained with our Ridge regression. We are therefore curious to see how these results would change if we added additional information to the data.

4.3.2.2 Results of LSTM using explanatory variables

Now, the aim is to use the explanatory variables to implement an LSTM model for stock price prediction. We want this model to combine the added data with the time evolution and see if it can improve the results.

We make a first model using all available variables and indicators that we described earlier in this paper.

To find out which model fits the best, many changes were made in the hyperparameters to get better results. In particular, we've tried to optimize learning by changing the learning rate. It controls the speed at which a model adjusts its weights during training, directly influencing convergence to a minimum of the loss function. A poorly chosen learning rate can prevent the model from converging when it's too high, or make training unnecessarily slow when it's too low.

Another hyperparameter that is crucial if we want better results is the number of epochs

of training which determines how many times the training data set is used to adjust the model weights. Too few epochs can lead to undertraining, where the model doesn't have enough time to learn the relationships in the data, while too many epochs can lead to overtraining. Thus, it is essential to find the right balance to a obtain a model that generalizes well. To have an idea of the ideal number of epochs of training to choose, we can plot the loss function on the training set and the one on test or validation set following the number of epochs of training. Generally, the loss on the training set never stops decreasing and. For the test set, the loss decreases at the beginning, and at some point, stays more or less stable or starts increasing. The number of epochs at which this change is noticed can be used as the adequate one for the training (cf fig 23).

Many other hyperparameters can be optimized: the output size of the layers, the number of layers and their type (LSTM, Dense, Pooling, etc.), the Dropout percentage, etc. But optimizing these hyperparameters automatically can take a lot of time, especially if we work on a big dataset, which is our case. Thus, some examples of changes on hyperparameters and the performances made in each case is presented in the following table 8. Previously, many executions with different values of the hyperparameters have been made so that we have an idea of the best combination:

Table 8: Performances of the models when changing output size of LSTM layers

Model	RMSE	MAE	R^2
Layer1(50), Layer2(50)	648.84	582.83	0.35
Layer1(100), Layer2(90)	286.35	243.53	0.87
Layer1(300), Layer2(150)	309.59	278.19	0.85
Layer1(400), Layer2(200)	266.33	233.15	0.89

All these results have been obtained with 20 epochs of training. Note that, in some cases, we could have better results due to overfitting. For example, for the last combination, overfitting point has been reached quite quickly due to the relative big size of the output, which means that the loss on the test set started to slowly increase. We made another training with only 15 epochs and the fit significantly was better ($RMSE = 196.69$ and $MAE = 159.09$).

Changes have been applied on the output size of the LSTM layers. The other hyperparameters will remain constant so that we focus on the optimization of the output sizes. The other hyperparameters are given in the table below:

Table 9: Values of other hyperparameters.

Number of LSTM layers	**Number of Dense layers**	**Learning rate**	**Lookback (h)**
2	2	10^{-5}	30

To see the comparison between the actual values of the closing prices and the predictions on the test set for the 4th combination (model with the best performances), cf fig 24. Subsequently, another deep learning architecture, GRU (see annex), will be tested and improve on the good results already obtained with LSTM. However, the results are still slightly poorer than with Ridge regression.

Our results show that adding macroeconomic variables to the LSTM model did not improve performance compared to the version without covariates. Although these variables are used in the Ridge model, they do not seem to provide additional benefits for the LSTM model. This could be due to the difficulty of the LSTM model in capturing relevant relationships or an overload of information that may disrupt learning from time series data. The simpler Ridge model appears to be better suited for this task.

5 Models comparison

In our analysis, we explored several models to predict the closing prices of the CAC40, considering both classical time series models and approaches based on deep learning. The performance of the models was evaluated using the RMSE, MAE, and R2 metrics to determine the most effective approach for this prediction task. While we compared the predictive performance of the models within each family, it is important to make an overall comparison to identify which model is best suited to our data. The results show that Ridge Regression is the most performant model, with an RMSE of 64.99 and an MAE of 49.14, significantly outperforming the other models.

Table 10: Performance Comparison of Models

Metric	**GBM (cf annex)**	**LSTM**	**GRU (cf annex)**	**ARMAX**	**ARDL**	**Prophet**	**Ridge Regression**
RMSE	868.91	266.33	200.13	775.28	3345	148.50	64.99
MAE	808.95	233.15	171.28	699.68	3144	118.36	49.14
R^2	-5.64	0.89	0.94	-4.55	-97.36	0.81	0.96

Since ridge regression is the best-performing model among those tested, it is worthwhile to investigate the presence of GARCH effects in its residuals.

5.1 GARCH model on the residuals of the best model : Ridge regression

Upon examining the plot of the residuals in 17, the data appears to exhibit volatility clustering. Additionally, the autocorrelation of the squared residuals (cf fig. 26 in annex) provides evidence of dependence, indicating that the residuals are not homoscedastic. This suggests that the assumption of constant variance is violated. Therefore, applying a GARCH model could be beneficial to capture the variance dynamics effectively.

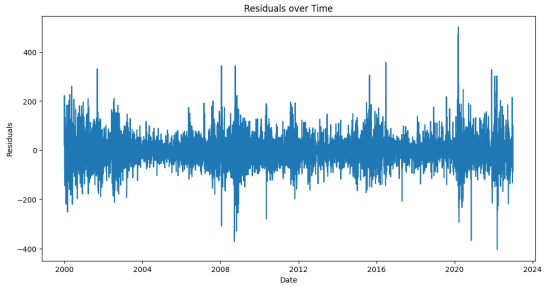


Figure 17: Residuals of ridge regression on the training dataset

By modeling both the mean and variance dynamics, a GARCH model enhances predictive accuracy and provides more reliable estimates of uncertainty, especially in volatile and uncertain environments.

Based on the literature, we fit a GARCH(1,1) model to the residuals from the training dataset to capture their time-varying volatility.

To evaluate the performance of this model, we test the assumptions of GARCH, including the normality of the standardized residuals and the absence of autocorrelation in the squared residuals. The Portmanteau test, applied up to the 11th lag, shows no significant autocorrelation, indicating that the GARCH model successfully captures the variance dynamics in the residuals. This confirms the adequacy of the fitted GARCH(1,1) model in modeling the volatility of the data.

However, when testing the normality assumption using the Jarque-Bera test, the hypothesis of normality is strongly rejected (p-value $\leq 2,2 \times 10^{-16}$). As a result, we investigate the appropriate distribution for the standardized residuals. Upon calculating the kurtosis (4.43) and skewness (-0.33) of the residuals, these values may suggest that a Student's t-distribution might be a better fit.

To confirm this, we fit a Student's t-distribution to the standardized residuals and evaluate the fit using the Kolmogorov-Smirnov (KS) test. The resulting pp-value of 0.9 indicates that the Student's t-distribution is a good match for the data.

Using a GARCH(1,1) model with a Student's t-distribution for the standardized residuals, we observe a slight improvement over the GARCH(1,1) model with a normal distribution, as evidenced by lower AIC and BIC values. Furthermore, there remains no autocorrelation in the standardized residuals up to the 11th lag. This indicates that the GARCH(1,1) model with a Student's t-distribution provides a better fit for the volatility of the Ridge Regression model.

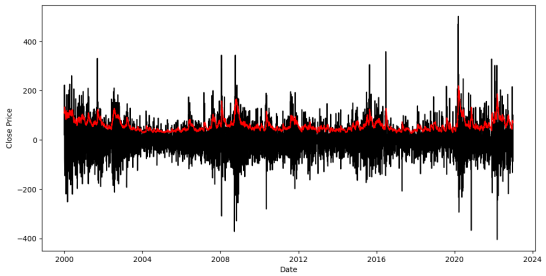


Figure 18: Prediction of GARCH(1,1) on ridge regression' residuals in the test dataset

The best model fitted is then the GARCH(1,1) as following :

$$\sigma_t^2 = 41.76 + 0.08\epsilon_{t-1}^2 + 0.91\sigma_{t-1}^2,$$

The residuals ϵ_t follow a standardized Student's t-distribution, characterized by a degree of freedom parameter ($\nu = 7.93$).

6 Conclusion and perspectives

This study demonstrates that predicting CAC40 stock prices, by combining traditional methods with advanced machine learning and deep learning techniques, is both a complex and captivating challenge. While the results are promising, they reveal the inherent complexity of the financial market, where each fluctuation is influenced by a multitude of interconnected factors.

Various models were tested to predict the closing prices of the CAC40, particularly Ridge regression, which proved to be the most effective. This model provided reliable predictions with low error (RMSE = 64.99) while effectively capturing the underlying trends in the data, thus demonstrating its robustness in the face of the volatility and unpredictability of the financial markets.

The integration of exogenous variables and special events through the Prophet model also yielded interesting results, highlighting the importance of including external factors to improve the accuracy of forecasts.

The use of LSTM models highlighted their potential in forecasting financial time series, particularly regarding short-term dependencies. Although these models showed satisfactory results, they did not outperform Ridge regression in this specific context. This suggests that, in certain situations, simpler models may be more effective, especially when available data is limited or unforeseen events disrupt complex models.

Nonetheless, it would be a missed opportunity not to further explore modern techniques, which continue to evolve. Advances in machine learning and deep learning offer exciting prospects for further enhancing predictive performance.

An interesting avenue to explore would be the use of ensemble learning models, as suggested by Brownlee (2020), where several machine learning models would be combined to leverage their respective strengths. This approach would help better manage the specific errors of each model while maximizing the accuracy of predictions for CAC40 stock prices. A combination of classical models, such as Ridge regression, and short-term neural networks like LSTM or GRU (see annex) could thus optimize the consideration of short-term dependencies while benefiting from the advantages of simpler models to capture linear trends. Another quick change that can be made to improve the results would be to integrate an attention mechanism in the deep learning models (Q. Liu, Hu, and H. Liu 2024).

Working on this topic has been a rewarding experience, allowing us to explore various techniques at the crossroads of several fields, ranging from classical methods to advanced approaches in machine learning and deep learning. This diversity provided us with a fresh perspective on the application of these tools in financial forecasting and has been very educational throughout the study.

Annex

7 Annex

Table 11: List of variables used and their descriptions.

Variable	Description
Macroeconomic Indicators	
CPI_us	Consumer Price Index for the United States
IRCB_us	Central Bank interest rate for the United States
UNR_us	Unemployment rate for the United States
YPH	Current household expenditures, including nonprofit institutions serving households
GDPV	Gross Domestic Product in volume, at market prices
IRL	Long-term interest rate on government bonds
IRS	Short-term interest rate
GDP	Gross Domestic Product in nominal value, at market prices
CPIH_YTYPCT	Harmonized overall inflation rate
Technical Indicators	
High_minus_Low	Stock High minus Low price
Close_minus_Open	Stock Close minus Open price
SMA_5, SMA_10, SMA_20	Simple Moving Average for 5, 10, or 20 days
WMA_5, WMA_10, WMA_20	Weighted Moving Average for 5, 10, or 20 days
Momentum	Momentum indicator
Stochastic_K	Stochastic %K indicator
Stochastic_D	Stochastic %D indicator
MACD	Moving Average Convergence Divergence
RSI	Relative Strength Index
Williams_R	Larry Williams' %R indicator
CCI	Commodity Channel Index

Figure 19: Linear trend for data starting in 1990

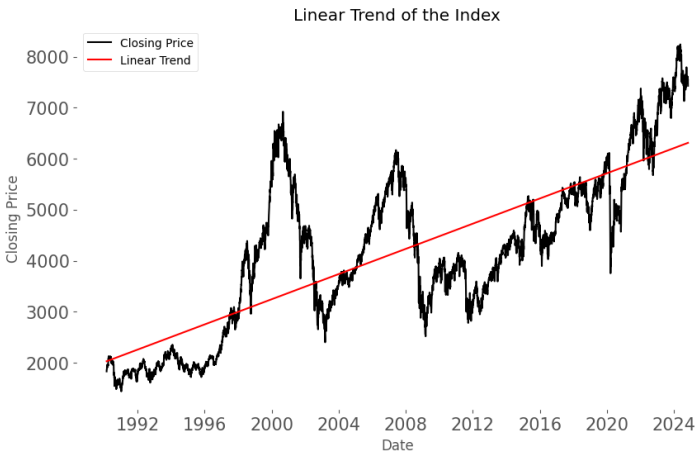


Figure 20: Predicted vs observed returns on the 2005-2009 period

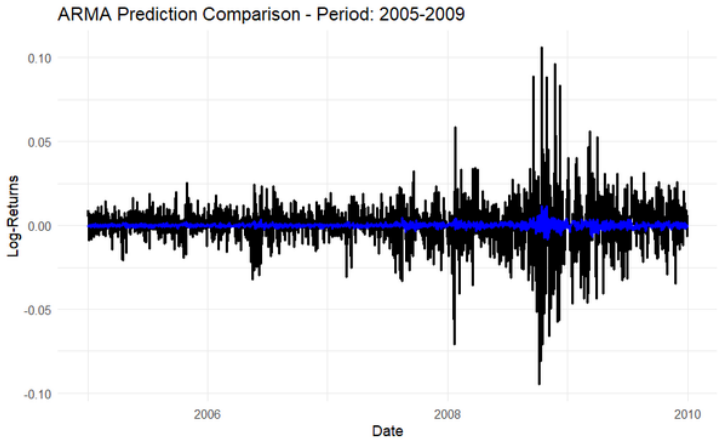


Table 12: Stationarity of variables based on the KPSS test.

Variable	**Stationarity (KPSS)**
GDPV	I(1)
IRL	I(2)
IRS	I(2)
GDP	I(2)
CPIH_YTYPCT	I(1)
IRCB	I(2)
UNR	I(2)
YPH	I(2)
UNR_us	I(1)
IRCB_us	I(2)
CPI_us	I(2)
Close	I(1)
Return	I(0)
High_minus_Low	I(1)
Close_minus_Open	I(0)
SMA_5	I(1)
SMA_10	I(1)
SMA_20	I(1)
WMA_5	I(1)
WMA_10	I(1)
WMA_20	I(1)
Momentum	I(0)
RSI	I(1)
Williams_R	I(1)
Stochastic_K	I(0)
Stochastic_D	I(0)
CCI	I(0)
MACD	I(1)

Table 13: Granger Causality Test Results

Variable	Causality Test	p-value
GDPV	GDPV Granger-cause Return	0.1991
	Return Granger-cause GDPV	0.0074
CPIH_YTYPCT	CPIH_YTYPCT Granger-cause Return	0.5588
	Return Granger-cause CPIH_YTYPCT	0.6766
UNR_us	UNR_us Granger-cause Return	0.6806
	Return Granger-cause UNR_us	0.0032
Close	Close Granger-cause Return	0.1272
	Return Granger-cause Close	0.05766
High_minus_Low	High_minus_Low Granger-cause Return	0.4537
	Return Granger-cause High_minus_Low	$< 2.2e-16$
Close_minus_Open	Close_minus_Open Granger-cause Return	0.7847
	Return Granger-cause Close_minus_Open	$< 2.2e-16$
SMA_5	SMA_5 Granger-cause Return	0.1585
	Return Granger-cause SMA_5	$< 2.2e-16$
SMA_10	SMA_10 Granger-cause Return	0.1851
	Return Granger-cause SMA_10	$< 2.2e-16$
SMA_20	SMA_20 Granger-cause Return	0.2012
	Return Granger-cause SMA_20	$< 2.2e-16$
WMA_5	WMA_5 Granger-cause Return	0.1484
	Return Granger-cause WMA_5	$< 2.2e-16$
WMA_10	WMA_10 Granger-cause Return	0.1694
	Return Granger-cause WMA_10	$< 2.2e-16$
WMA_20	WMA_20 Granger-cause Return	0.1894
	Return Granger-cause WMA_20	$< 2.2e-16$
Momentum	Momentum Granger-cause Return	0.0141
	Return Granger-cause Momentum	$< 2.2e-16$
RSI	RSI Granger-cause Return	0.02953
	Return Granger-cause RSI	$< 2.2e-16$
Williams_R	Williams_R Granger-cause Return	0.02857
	Return Granger-cause Williams_R	$< 2.2e-16$
Stochastic_K	Stochastic_K Granger-cause Return	0.04938
	Return Granger-cause Stochastic_K	$< 2.2e-16$
Stochastic_D	Stochastic_D Granger-cause Return	0.09381
	Return Granger-cause Stochastic_D	$< 2.2e-16$

Figure 21: SHAP features importance and effects for trained ridge regression

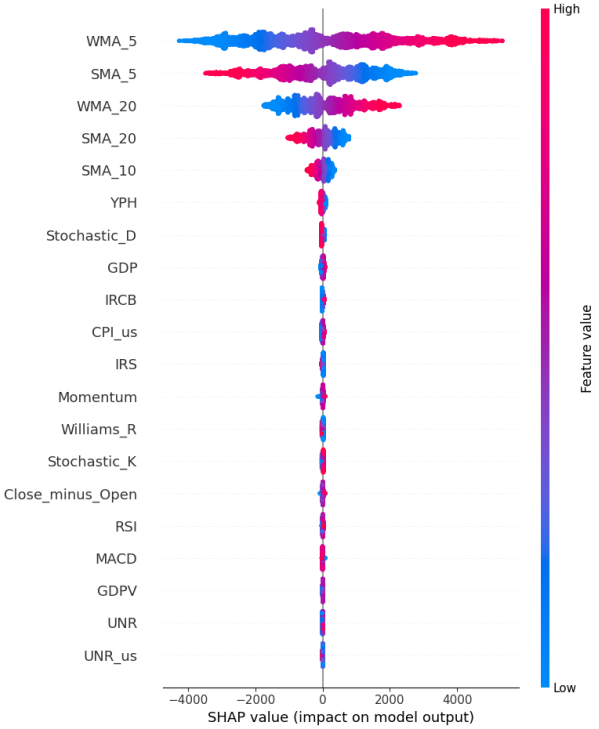


Figure 22: LSTM network illustration

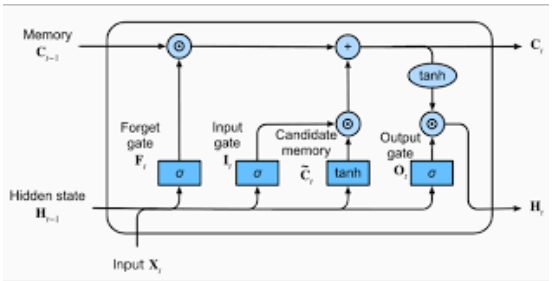


Figure 23: Example of a loss plot

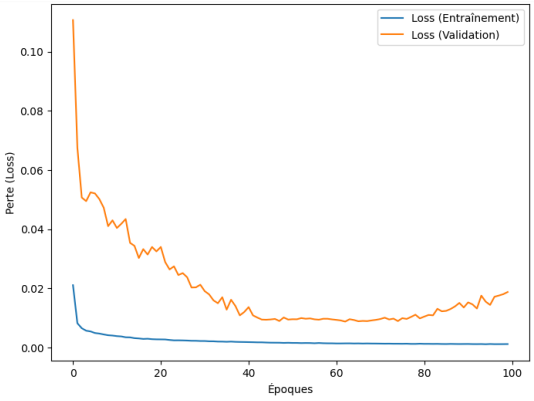


Figure 24: Prediction vs Actual values comparison, combination 4 LSTM

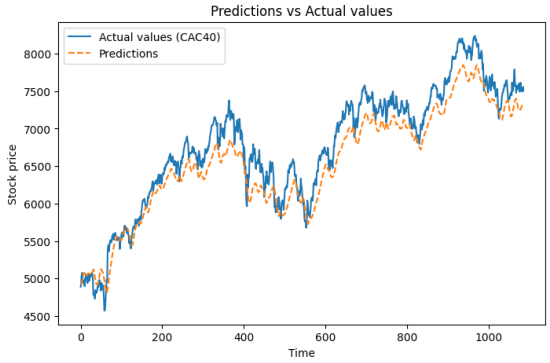
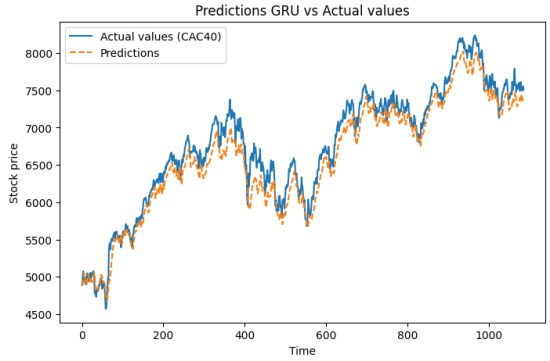


Figure 25: Prediction vs Actual values comparison, combination 4 GRU



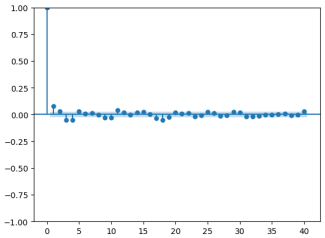


Figure 26: ACF des résidus du modèle ridge regression sur le training dataset

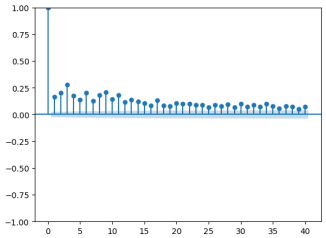


Figure 27: ACF des résidus du modèle ridge regression sur le training dataset

To go further

1. Gradient Boosting

In this study, we applied GBM to predict the daily **Close** price of the CAC40 index. We used several explanatory variables to build the model. To enhance performance, we implemented a variable selection process to identify the most relevant predictors, ensuring the model's robustness.

a) Variable Selection Process

To select the most informative predictors, we applied the following criteria:

- **Strong correlation with the target**: Variables with an absolute correlation greater than 0.4 with **Close** were retained.
- **Low inter-variable correlation**: Among variables with high pairwise correlation (> 0.7), we kept the one most strongly correlated with **Close** and removed the others to avoid multicollinearity.

After this process, the following variables were selected:

- **UNR_us**: The unemployment rate in the United States.
- **CPI_us**: The Consumer Price Index (CPI) in the United States, measuring inflation.
- **UNR**: The unemployment rate in France.

We experimented with different correlation thresholds (e.g., 0.2 for target correlation and 0.9 for inter-variable correlation) to include additional variables. However, these adjustments did not significantly improve the model's RMSE. Thus, the final model with **UNR_us**, **CPI_us**, and **UNR** was retained.

b) Model Training and Results

We trained the Gradient Boosting Model (GBM) using the `gbm` package in R. To minimize the Root Mean Squared Error (RMSE), we tested various hyperparameter combinations and selected the optimal configuration as follows: the number of trees (`n.trees`) was set to 1000, the maximum depth of each tree (`interaction.depth`) was 5, the shrinkage rate (`shrinkage`) was 0.01, and the minimum number of observations in a node (`n.minobsinnode`) was 15.

The relative contributions of the variables, expressed as percentages, indicate how much each variable influences the model's predictions. These percentages are calculated based on the improvement in the model's performance (e.g., reduction in error) when each variable is used. The results are as follows:

- **UNR_us**: 57.42% (highest impact on predictions). The U.S. unemployment rate (**UNR_us**) has the strongest influence on the CAC40 **Close** price. This highlights the importance of U.S. economic conditions for the French stock market, as changes in U.S. employment can affect global investor sentiment and capital flows.
- **CPI_us**: 29.18% (significant influence). The U.S. Consumer Price Index (**CPI_us**), which measures inflation, also plays a key role. Inflation in the U.S. can impact global interest rates and currency values, indirectly affecting the CAC40 index. This shows that U.S. inflation trends are relevant for predicting French stock market movements.

- **UNR**: 13.34% (notable but smaller contribution). The French unemployment rate (**UNR**) has a smaller but still meaningful impact. While domestic economic conditions matter, their influence is less pronounced compared to U.S. indicators, suggesting that global factors dominate in driving the CAC40 index.

c) Model Performance on Test Data

The performance of the Gradient Boosting Model (GBM) on the test set is summarized in Table 14.

Table 14: Performance Metrics of the Gradient Boosting Model (GBM) on the Test Set

Metric	Value
Root Mean Squared Error (RMSE)	868.91
Mean Absolute Error (MAE)	808.95
Harmonic Mean Absolute Error (HMAE)	0.11
R-squared (R^2)	-5.64

The GBM demonstrated poor performance on the test set, as reflected by an RMSE of **868.91** and an MAE of **808.95**, which indicate large discrepancies between predicted and actual values. The HMAE of **0.11** shows that, on average, predictions deviate by 11% from the true values. Additionally, the negative R^2 value of -5.64 underscores the model's failure to generalize, performing worse than a simple mean-based prediction.

These metrics confirm the model's inability to capture the underlying patterns in the data. Figure 28 further illustrates this issue, showing significant deviations between true values and predictions. The scatter plot highlights the model's poor fit, as the predictions deviate substantially from the ideal diagonal line, suggesting unreliable estimates for the CAC40 closing prices.

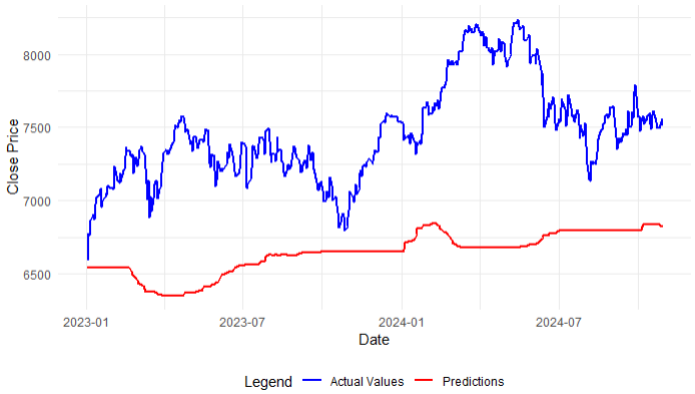


Figure 28: True Values vs Predictions of the GBM on the Test Set.

2. GRU (Gated Recurrent Unit)

Once we worked on LSTM and saw the great results we could get from it, and after trying many combinations of hyperparameters to optimize our results, we thought about doing the same work with GRU. Similarly to LSTM, they are designed to process time series by capturing short and long term dependencies. Like LSTMs, they use gate mechanisms to regulate the flow of information, but with a simpler architecture, which reduces their computational cost. In particular, they combine the functions of LSTM's input and oblivion gates into a single "update gate", and use a "reset gate" to control the storage of past information. In a financial

context, such as predicting stock prices, GRUs can be particularly useful for efficiently processing large quantities of sequential data, while limiting the risk of overlearning and also reducing computation time, thanks to their simpler architecture.

Model	RMSE	MAE	R2
Layer1(50), Layer2(50)	305.22	241.99	0.86
Layer1(100), Layer2(90)	281.85	250.86	0.88
Layer1(300), Layer2(150)	234.41	208.72	0.92
Layer1(400), Layer2(200)	200.13	171.28	0.94

Table 15: Performances of the models when changing output size of GRU layers

The table 15 shows the results given by the implementation and evaluation of GRUs models on the test set. The training is performed using the macroeconomic data described earlier in this paper. After seeing the results we got by changing them, we finally used the same hyperparameters as for LSTM, except for the number of epochs which was 15, and the learning rate 10^{-04}. Indeed, the learning phase took less time than it did with LSTM, as expected.

The results, regardless of the metric we use to compare them, are better with GRU than with LSTM. Plus, GRU requires less parameters to perform than LSTM (Rahmadeyan and Mustakim 2024). For example, for the last model in the GRU table 15, the number of parameters of the first GRU layer is 514,800 while it is 684,800 for the first LSTM layer of the best model here 8. To visualize the comparison between the actual values of the closing prices and the predictions on the test set for the 4th combination (model with the best performances), cf fig 25.

8 Bibliography

Ariyo, Adebiyi A, Adewumi O Adewumi, and Charles K Ayo (2014). "Stock price prediction using the ARIMA model". In: *2014 UKSim-AMSS 16th international conference on computer modelling and simulation*. IEEE, pp. 106–112.

Barra, MSCI (2010). "Is there a link between GDP growth and equity returns?" In: *Research Bulletin*.

Black, Fischer (1976). "Studies of stock market volatility changes". In: *Proceedings of the American Statistical Association, Business & Economic Statistics Section, 1976*.

Bollerslev, Tim (1986). "Generalized autoregressive conditional heteroskedasticity". In: *Journal of Econometrics* 31.3, pp. 307–327. ISSN: 0304-4076. DOI: https://doi.org/10.1016/0304-4076(86)90063-1. URL: https://www.sciencedirect.com/science/article/pii/0304407686900631.

Brealey, Richard A, Stewart C Myers, and Franklin Allen (2014). *Principles of corporate finance*. McGraw-hill.

Camara, Boubacar, Pierre Pessarossi, and Thomas Philippon (2017). "Backtesting european stress tests". In.

Campbell, John Y et al. (1998). "The econometrics of financial markets". In: *Macroeconomic Dynamics* 2.4, pp. 559–562.

Cryer, Jonathan D (2008). *Time series analysis*. Springer.

De Vilder, Robin and Marcel P Visser (2007). "Proxies for daily volatility". In.

Engle, Robert F (1993). "Statistical models for financial volatility". In: *Financial Analysts Journal* 49.1, pp. 72–78.

Esstafa, Youssef (2019). "Modèles de séries temporelles à mémoire longue avec innovations dépendantes". PhD thesis. Bourgogne Franche-Comté.

Franke, Jürgen, Wolfgang Karl Härdle, and Christian M Hafner (2004). *Statistics of financial markets*. Vol. 2. Springer.

Haq, Sebastian and Rasmus Larsson (2016). "The dynamics of stock market returns and macroeconomic indicators: An ARDL approach with cointegration". MA thesis. KTH Industrial Engineering and Management.

He, Changli, Timo Teräsvirta, and Hans Malmsten (2002). "Moment structure of a family of first-order exponential GARCH models". In: *Econometric Theory* 18.4, pp. 868–885.

Liu, Qingyang, Yanrong Hu, and Hongjiu Liu (2024). "Enhanced stock price prediction with optimized ensemble modeling using multi-source heterogeneous data: Integrating LSTM attention mechanism and multidimensional gray model". In: *Journal of Industrial Information Integration* 42. URL: https://www.sciencedirect.com/science/article/pii/S2452414X24001547.

Lo, Andrew W (1991). "Long-term memory in stock market prices". In: *Econometrica: Journal of the Econometric Society*, pp. 1279–1313.

Luca, DI PERSIO and Honchar Oleksandr (2016). "Artificial Neural Networks Approach to the Forecast of Stock Market Price Movements". In: *INTERNATIONAL JOURNAL OF ECONOMICS AND MANAGEMENT SYSTEMS*. URL: http://iaras.org/iaras/journals/caijems/artificial-neural-networks-approach-to-the-forecast-of-stock-market-price-movements.

Mathusha, S. (June 2021). "The Impact of Macroeconomic Variables on Stock Price: An ARDL Approach to Sri Lanka". In: *Peradeniya Management Review* 3.1.

Nelson, Daniel B (1991). "Conditional heteroskedasticity in asset returns: A new approach". In: *Econometrica: Journal of the econometric society*, pp. 347–370.

Obthong, Mehtabhorn et al. (2020). "A survey on machine learning for stock price prediction: Algorithms and techniques". In.

Patel, Jigar et al. (2015). "Predicting stock and stock price index movement using trend deterministic data preparation and machine learning techniques". In: *Expert systems with applications* 42.1, pp. 259–268.

Rahmadeyan, Akhas and Mustakim (2024). "Long Short-Term Memory and Gated Recurrent Unit for Stock Price Prediction". In: *Procedia Computer Science* 234. URL: https://www.sciencedirect.com/science/article/pii/S1877050924003533.

Simple Moving Averages Make Trends Stand Out (2024). en. URL: https://www.investopedia.com/articles/technical/052201.asp (visited on 12/30/2024).

Stock, James H and Mark W Watson (2020). *Introduction to econometrics*. Pearson.

Taylor, Sean J. and Benjamin Letham (2018). "Forecasting at Scale". In: *Facebook*. Facebook, Menlo Park, California, United States. URL: https://www.facebook.com.

Vijh, Mehar et al. (2020). "Stock Closing Price Prediction using Machine Learning Techniques". In: *Procedia Computer Science* 167, pp. 599–606. ISSN: 1877-0509. DOI: https://doi.org/10.1016/j.procs.2020.03.326. URL: https://www.sciencedirect.com/science/article/pii/S1877050920307924.

Weron, Rafał (2002). "Estimating long-range dependence: finite sample properties and confidence intervals". In: *Physica A: Statistical Mechanics and its Applications* 312.1-2, pp. 285–299.

Zhiteng, Wang, Wang Ruoyu, et al. (2020). "The Impact of Long-and Short-Term Interest Rate Adjustments on Stock Prices". In: *Frontiers in Educational Research* 3.15.

9 Iconography

List of Figures

List of Tables