ENHANCING THE PREDICTABILITY OF THE DIVIDEND GROWTH RATE G

Jury:
Supervisor:
Dr. Georges HUBNER
Reader:
Dr. Sophie MOINAS

Master thesis by
Nicolas HABER
To obtain the diploma of
Banking and Asset Management
Academic year 2022/2023

Acknowledgments

I want to express my gratitude to Dr. Georges Hübner, professor of finance at HEC Liège and my supervisor for my master's thesis. His guidance and input were invaluable in shaping and polishing the outcomes of this study. I would also extend my appreciation to Dr. Sophie Moinas, professor of finance at Toulouse School of Management and the reader of my thesis. Her consistent advice and direction were instrumental throughout this process. Lastly, heartfelt thanks go to my dear family members for their unwavering support and boundless confidence, which provided me with the emotional strength I needed.

Contents

Abstract

The price formation of different asset classes in the financial markets is influenced by two fundamental factors: expected future dividends and the rate at which these future values are expected to be discounted. Scholars have carefully researched the consistency of discount rates and dividend growth predictability through time, resulting in a body of work on the subject. Most researchers agreed on the fact that forecasting the dividend growth rate accurately could increase the capacity to estimate discount rates. However, researches on the predictability of dividend growth rates are contradictory. Various studies have taken various methodologies, ranging from simple regressions of dividend growth rates on dividend yield to more complicated vector autoregressive models. Additionally, some researchers have derived expected dividend growth rates from the S&P 500 option market. Most of the established knowledge explores the dividend growth rate in light of the dividend yield and several other economic indicators. However, they ran into a problem known as endogeneity. However, due to a lack of sufficient training data, no previous studies have used artificial intelligence approaches in this domain.

In my research, I used a model established by Ang and Liu (2007), to produce artificial data for the S&P 500. This enabled me to generate two sets of time series data, each having 2000 data points and representing the dividend yield and dividend growth rate. Following that, I fed these datasets into a specific neural network known as a Long Short-Term Memory Recurrent Neural Network (LSTM RNN). This configuration attempted to anticipate the dividend growth rate for the following time step, designated as t+1. The artificial time series I created demonstrated a strong alignment with historical data, confirming their acceptability for inclusion in my research project.

After intensive training and refining of my model, surprising prediction capabilities, notably for the prewar period, emerged. I acquired a remarkable R^2_{OOS} value of roughly 46.52% at a statistically significant 1% level while evaluating its performance using the R^2_{OOS} metric as defined in Goyal and Welch's work from 2008. This success puts my model ahead of many past achievements in this field. In contrast, the model shows limited proficiency in forecasting post-war results, which is consistent with previous research findings. Even though its predictive capacity is limited for this period, my inquiry gives important insights that could potentially set the groundwork for a new strategy, replete with chances for later upgrades and real-world implementation.

Introduction

Context

The financial market is a market in which diverse participants trade equity, bonds, and various investment vehicles. The trade flow is the result of each investor's belief on price formation, which is decided by the future condition and feasibility of investment. Any financial product's pricing is determined by two key factors: the anticipated dividend growth rate and the expected discount rate.

The predictability of dividend growth rates has traditionally been studied in light of the log-linearization of returns given by Campbell Shiller (1988).

$$dp_t = E_t \sum_{j=1}^{infinity} \rho^{j-1} \, . \, \Delta d_{t+j} - E_t \sum_{j=1}^{infinity} \rho^{j-1} \, . \, r_{t+j} = long\ run\ \Delta d\ - long\ run\ r$$

It can be intuitively interpreted that the price formation today is expressed as the long-run expected dividend growth, discounted by the long-run discount rate. High prices today are given by either high dividend expectations or low expected returns or both, and vice versa for low prices.

Divergent perspectives are found in the scientific literature, revealing a wide range of opinions among academics on the predictability of dividend growth rates. While some studies claim that features like the dividend price ratio can be used to predict dividend growth (as demonstrated by Fama and French in 1988, Lewellen in 2004, and Cochrane in 2008a), others argue that such predictability can be influenced by statistical biases, making it difficult to use (as demonstrated by Stambaugh in 1999 and Goyal and Welch in 2008). Cochrane (2008) notes, “Excess return forecastability is not a comforting result. Our lives would be so much easier if we could trace price movements back to visible news about dividends or cash flows...But that is where the data have forced us, and they still do so.” He proceeds to point out the implication of this rather striking result: “If all market price-dividend ratio variation comes from varying expected returns and none from varying expected growth in dividends or earnings, much of the rest of finance still needs to be rewritten.”

According to Golez (2014), the documented improvement in forecasting returns as a result of better dividend growth (DG) forecasts translates into a gain of 0.32 in terms of the Sharpe ratio for a mean-variance investor. The big gain in the Sharpe ratio incentivized me to establish a model for DG prediction. My thesis is set in the context of two elements: the fact that different models have failed to have predictive power over dividend growth, and the increase in computing power and neural network advanced architectures.

The foundation of neural networks came with the model proposed by Warren McCulloch and Walter Pitts in 1943. They established the simple binary threshold neuron, which will be the pillar for future developments. Following that, Frank Rosenblatt established the most fundamental forms of a neural network, known as perceptron in 1957.

Research on neural networks was confronted by big drawbacks, and disregarded for 11 years from 1969 to 1980, which is known as the "Neural Network Winter". It was caused by the absence of a training mechanism and computational power.

In 1986 a groundbreaking paper was published by David Rumelhart, Geoffrey Hinton, and Ronald Williams. They proposed an efficient backpropagation algorithm that allowed the training and weight updating, based on errors of multi-layer neural networks. This paper gave life back to neural network research and boosted the interest in neural networks during late 1990, which sparked many innovative ideas and the exploration of different complex architectures and faster training algorithms.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey Hinton won the ImageNet competition in 2012. That marked an important milestone in deep learning development and proved the important role of deep learning in image recognition tasks, which caused a surge in investment in deep learning research and usage. Remarkable advancement was witnessed after their success, notably, the development of recurrent neural network (RNN), long short-term memory (LSTM), generative adversarial network (GAN), and many more.

Artificial intelligence's arrival changed the landscape in many businesses, from self-driving vehicles, to face recognition and the financial industry like robot advisory, risk management, and algorithmic trading. The invasive conquest of AI will likely to continue due the increasing computational power, more efficient architectures, innovative optimization techniques, and much more.

Main objective

Considering the factors that I've outlined earlier, my plan involves crafting, refining, and evaluating an LSTM RNN configuration using TensorFlow. This endeavor aims to forecast forthcoming dividend growth rates at time step t+1. Its non-parametric nature sets this model apart, enabling it to capture intricate interplays within input sequences that traditional mathematical models fail to capture.

To embark on this journey, I'll leverage the training, validation, and testing phases by artificially generated data using the generative model proposed by Ang and Liu (2007). Their model simulates time sequences encompassing dividend growth rates and dividend yields (further elaboration is available in the data section). Post-training, the subsequent phase involves an out-of-sample assessment. This entails putting the structure to the test using real-world data from the S&P 500 spanning the years 1872 to 2022. The gauge of success here hinges on the concept of predictability power, gauged by the R^2_{OOS} metric introduced by Goyal and Welch (2008). This metric essentially compares the predicted outcomes against a historical mean benchmark, furnishing insights into the neural network's efficiency and robustness.

Research motivations

Forecasting returns and dividend growth rates have a substantial impact on a variety of facets within the financial services sector including asset allocation, risk management, and stock research, to name a few.

My motivation for pursuing this research originates from the significant contribution I could offer to the financial industry. This is especially notable given the ongoing heated debate over dividend growth rate predictability. Notably, the use of an LSTM RNN to estimate dividend growth rate has never been explored

before, owing to a lack of data. In this work, I hope to question and push the boundaries of existing literature, which I consider a great privilege.

Contributions

I envision that my work will bring valuable contributions to both academia and industry practitioners, inspiring them through my unconventional approach. The distinctive aspect of this work lies in its ability to overcome the challenge of limited annual financial data, and its pioneering utilization of an LSTM. An unexplored avenue thus far.

This master's research thesis paves the way for future investigations in this field, whether through the exploration of alternative generative models for data, or the embrace of different machine-learning techniques. I hope it encourages traditional practitioners who have yet to adopt ANN technology, whether due to skepticism or time constraints, to reconsider and appreciate the advantages it offers.

Lastly, a prewar R^2_{OOS} value of 46.52% confirms the validity of the model introduced by Ang and Liu (2007), which intricately captures the interplay among dividend growth rate, dividend yield, return drift, and return diffusion in an endogenous manner.

Structure of the thesis

This paragraph outlines the organization of my thesis, offering readers a roadmap for navigation. To begin, I present a comprehensive literature review and delve into prior research. This serves as a foundational step to understand the principles and essential considerations of the subject.

Moving on, I delve into the key theoretical concepts central to this work. I'll explain the Cox Ingersoll and Ross model, followed by a detailed breakdown of the artificial data generation model I've employed. Lastly, I'll provide an in-depth exploration of LSTM RNN.

Transitioning to the subsequent section, I delve into the data aspect. This involves a meticulous breakdown of the data, followed by an examination of the artificial data's compatibility with real-world data.

Literature review

In this part of my research thesis, I will present all the major studies on the predictability of the dividend growth rate. My focus entails a dual perspective, as both dividend growth and returns represent two interrelated facets (price formation), always studied in conjunction. This study stands at the heart of the financial markets and its dynamics, which produced a very wide and diversified literature on this matter. I based my study on the most recent papers which will be presented in chronological order.

Lettau and Ludvigson (2005) conducted a study on a consumption present value model, examining returns and dividend growth rates in relation to a constructed variable called "cay," representing the log consumption-wealth ratio. Their findings indicate that the constructed variable “cay” holds greater significance than the dividend yield in explaining variations in post-war US stock market returns. This conclusion aligns with previous research by Nelson and Kim (1993), Ang and Bekaert (2007), and Goyal and Welch (2003), which also emphasized the limitations of using the dividend price ratio to predict the dynamics between returns and dividends due to their positive covariance, which obscures the predictability of the dividend price ratio. They demonstrated that the fluctuations in returns and dividend growth are more prominent than what can be explained by the dividend yield, and the covariation between these variables influences the predictive power of the dividend yield, but not necessarily of the log consumption-wealth ratio.

After only one year, Lettau and Nieuwerburgh (2008) discovered that unpredictability in return and dividend growth can be linked to instability in model parameters across different subsamples. Using CSRP data from 1927 to 2004, they performed a 30-year rolling window technique to regress dividend growth and returns on the lagged dividend yield. They specifically regressed the difference in the lagged dividend yield at time t and the mean dividend yield, on the difference in the dividend growth at time t+1 and the average growth rate. Their study yielded parameters that were both statistically insignificant and unstable. The R-square values were never higher than 16%, indicating a limited explanatory power, and they found no predictability for either returns or dividend growth rates over the 1990s.

Subsequently, in the year 2010, Binsbergen and Koijen introduced an opposing viewpoint in contrast to the prevailing body of literature that contended that the dividend yield possesses limited predictive ability when it comes to stock returns and dividend growth. They showcased that employing ordinary least squares (OLS) regression for gauging the interplay among dividend growth, returns, and lagged dividend yield leads to a bias arising from an error-in-variable. The error-in-variable (EIV) problem, specifically impacts the dividend growth coefficient and contradicts the third Gauss-Markov assumption, which assumes no conditional expected shocks. This in turn, suggests that the dividend yield is endogenous. At the heart of the matter lies the interconnection between returns and dividend growth, characterized by opposing fluctuations in relation to the dividend yield. Essentially, both returns and dividend growth exert an impact on the dividend yield, yet the degree of influence for each remains a prominent subject of contention within the scholarly discourse. Within their research, Binsbergen and Koijen crafted a comprehensive closed-form present value model, drawing inspiration from Gabaix (2007). This model encompasses an AR(1) process that accounts for expected returns and expected dividend growth. Additionally, they departed from the linear assumption inherent in the AR model, forging a path toward a linear present value model for the dividend yield. This model portrays the dividend yield as an affine function, intricately intertwined with expected returns and dividend growth. This model allowed them to

break down stock returns into components such as expected return shock, persistent and transient components of the dividend growth shock, and an unexpected dividend growth shock. Through this filtering process, they achieved promising results, with an R-square of 18% for stock returns and 16% for dividend growth.

Over time, Chen, L. (2009) undertook a study centered around the predictability of stock returns and dividend growth (DG). This research approach involved employing a regression of the dividend yield within a sliding 20-year timeframe. The study underscored the pivotal significance of variable construction within this context. Within the existing body of research, two conflicting perspectives emerge: In one school of thought, dividends are presumed to be reinvested at the prevailing market rate (as seen in works by Cochrane (1992, 2001, 2008)). On the contrary, another perspective dismisses reinvestment and instead aggregates the dividends received over the year (as evidenced by studies like Campbell and Shiller (1988, 1998), Goyal and Welch (2003), and Ang and Bekaert (2007)).

Chen, L. (2009) illustrated that adopting the reinvestment approach significantly undermines the ability to forecast dividend growth. The dividend growth from reinvestment becomes tainted by returns characterized by a higher standard deviation, displaying a correlation of 0.65 with the returns. In contrast, the method of simple summation demonstrates a correlation of 0.18. Moreover, the research unveiled a shift in the predictability of dividend growth when employing the non-reinvestment strategy. From 1891 to 1956, the regression of DG on the DP ratio produced promising results, with an average t-statistic of -4.39 and an R-square of 0.49. However, these coefficients gradually diminished to zero from the 1960s onwards.

In 2010, a paper was published by Lacerda and Santa-Clara, utilized the log-linearization of return technique first introduced by Campbell and Shiller in 1988. The study focused on an AR(1) process for the expected return, and forecasted the expected dividend growth by its ten-year historical average, which represents a complete business cycle. This choice was bolstered by earlier findings in research by Bansal and Yaron (2004), Lettau and Ludvigson (2005), and Binsbergen and Koijen (2010), which demonstrated the presence of a persistent unobservable component in past dividend growth. To achieve a more accurate estimation of the dividend growth rate, two regression attempts were made. In the first attempt, Lacerda and Santa-Clara conducted a regression of dividend growth against the lagged dividend yield. In the second attempt, they performed the regression using ten lags of the dividend growth rate. Interestingly, the R-square value for the first regression was only 3%, whereas it significantly increased to 42% for the second regression.

A completely distinct method was brought forth by Golez in 2014. He employed a present value model derived from the log-linearization of returns, initially established by Campbell and Shiller (1988). His argument was centered on the inadequacy of the dividend yield as a predictor of expected stock return. This inadequacy primarily stems from its vulnerability to fluctuations in expected dividend growth. To estimate the expected return, he considered an AR(1) model for both expected dividend growth and expected returns, similar to Lacerda and Santa-Clara (2010). Rather than adopting a multivariate regression, Golez (2014) put forth an alternative version of the dividend yield using a two-stage least squares regression. This method involved filtering out the expected dividend growth's variation from the DP ratio. He contended that historical dividend growth cannot be a reliable predictor of the future due to the time-varying nature of expected dividend growth, which continually updates with new information. In the process of gauging anticipated dividend growth, Golez (2014) made use of the implied dividend

growth derived from the option market of the S&P 500 index. He deemed this selection beneficial given the high liquidity of the index market, enabling it to efficiently integrate new information concerning the implied dividend growth. By employing non-arbitrage principles, cost of carry for index futures, and put-call parity, he derived the implied dividend yield and subsequently adjusted the DP ratio.

The adjustment made to the DP ratio unveiled a notable increase in the variability of the expected return, as evidenced by regression analysis, compared to the uncorrected scenario. Moreover, the implied dividend growth demonstrated a strong explanatory power of 18.42% in predicting future dividend growth, with highly significant t-statistics ranging between 3.25 and 4.76.

Zhu et al. (2018) employed the present value identity and considered AR(1) models for both expected dividend growth (DG) and stock returns. The objective was to isolate the variations in the dividend-price (DP) ratio caused by DG fluctuations. To achieve this, they needed to estimate the expected DG accurately. For this purpose, they utilized a comprehensive set of economic factors, including the payout ratio, net equity expansion, inflation, and book-to-market ratio. As a result, they obtained a reliable estimation of the expected DG.

Their analysis demonstrated that the lagged DG, payout ratio, and stock variance were superior to using the historical mean in estimating DG. They evaluated their estimation using an out-of-sample test and obtained a positive R-square value (R-square is a metric described by Goyal and Welch (2008)). In the subsequent section, the authors showed an enhanced returns predictability when incorporating the newly expected dividend growth.

Trends and oppositions in the literature

In the dividend growth predictability literature, we can see many oppositions and similarities on the ability of the dividend yield to uncover the dividend growth dynamics, and on whether the DG rate is iid. In this section, I will show opposition and similarities to have an informed overview of prior practices, which will give me the ability to have a qualitative common sense that will guide my quantitative research.

It has become a stylized fact that dividend yield is not able to predict dividend growth. That is demonstrated by Lettau and Nieuwerburgh (2008), by attributing the instability in the regression parameters as the cause of weak predictability. Furthermore, Lettau and Ludvigson (2005) demonstrated that the dividend growth rate is more predictable when a log wealth consumption ratio "cay" is used, which is unaffected by the covariation of returns and dividend growth rate.

Acknowledging the statistical issues that come along with the dividend yield factor, many researchers have tried to circumvent these challenges by introducing a corrected dividend yield ratio. The created ratio will be filtered from the fluctuations related to the expected dividend growth. It has become a trend to use the present value identity of Campbell and Shiller (1988) to separate dividend yield variation caused by time-varying expected dividend growth from other factors. For instance, Lacerda Santa-Clara (2010) introduced a corrected dividend yield by estimating the expected dividend growth rate through a retrospective consideration of the previous ten lags. Also, Golez (2014) adopted a similar approach by extracting the implied dividend growth rate from the option market. In a similar vein, Zhu et al. (2018) also employed an extensive array of economic variables to estimate the expected dividend growth rate.

Another divergence in the literature is reported by Chen, L. (2009). The dividend growth should be calculated with an annual frequency to avoid seasonality, which will introduce a disagreement on the yearly dividend calculation. Chen, L. (2009) showed that reinvested dividends in the markets can blur the predictability of the dividend growth rate, supported by (e.g., Campbell and Shiller (1988, 1998), Goyal and Welch (2003), and Ang and Bekaert (2007)). Their viewpoint asserts that dividends should solely be aggregated over the year, disregarding any reinvestment. Contrariwise, (e.g., Cochrane (1992, 2001, 2008)) consider reinvestment of the dividends at the prevailing rate which is more in line with investors' practices.

Several studies found no disparity in predictability between pre and post war periods, while others such as Chen, L. (2009) and Chen, Da, and Priestley (2012) demonstrated that the practice of smoothing dividends and changes in policies after the war had masked the predictability of dividend growth.

In light of these trends and oppositions, I'm able to determine my variable construction methodology and many assumptions that will shape my research.

Literature review summary

To produce quality research work, a good summary of the existing literature should be developed. It gives the writer and the reader a clear and strong base to delve deeper into the subject. In this paragraph, I will present the most important takeaways that influenced my research.

- With a linear predictive model, the dividend yield proves inadequate in predicting both the dividend growth rate and the returns. The dividend yield suffers from an endogeneity problem, which causes biases and unreliable estimates. As a result, a predictive model like an LSTM RNN can come as a solution, since it can take into consideration complex relationships that are not captured by simple linear models.
- A good predictability of the dividend growth rate can enhance the predictive power of the dividend yield over the returns. Constructing a corrected dividend yield that takes into account the predicted dividend growth rate filters the influence of any variation of the latter.

- Two opposing trends are presented regarding the reinvestment strategy. The first argues that the dividend growth rate data should be reinvested at the market rate, which is defended by the fact that investors will reinvest the dividends. Conversely, many researchers consider taking the pure dividend rate by simply summing the dividends received during a year. They argued that the reinvested dividends are contaminated by the returns, which degrade the information held by the pure dividends.
- The reversal in the predictability of the dividend growth rate should be taken into consideration. Certain scholars demonstrated that dividend growth rate predictability is feasible for the pre-war era, whereas the opposite holds for the post-war period.
- Zhu et al. (2018) demonstrated that the dividend growth time series exhibits predictive efficacy concerning forthcoming growth rates. They illustrated a favorable R-square value for the lag one dividend growth rate.

In light of this summary, I will conduct my study based on the pure dividends, by simply summing them over a year. I included the dividend yield despite the aforementioned issues since an LSTM model can

disentangle intricate relationships that are not captured by linear models. Finally, any lack of predictive power over the post-war period would not come as a surprise, as it will conform to previous research.

Theoretical framework

In this section, I'll go through all of the theoretical principles that I used in my research as well as the best practices. To begin, I will explain the variables used in my model (Dividend growth rate, stock returns, and dividend yield). Then, I will give an explanation of the model used to generate artificial data, presented by Ang and Liu (2007). Finally, I will go over a full explanation of a RNN and the mechanics behind an LSTM unit.

Variables

Three focal variables are at the heart of asset price variation: dividend growth rate, dividend yield, and stock returns. The log-returns of the aggregate stock market between t and t+1:

$$r_{t+1} = \log\left(\frac{P_{t+1} + D_{t+1}}{P_t}\right),$$

Where P is the price, and D is the dividend received by equity holders. The aggregate log dividend growth between t and t+1:

$$\Delta d_{t+1} = \log\left(\frac{D_{t+1}}{D_t}\right),$$

And the dividend price ratio at time t:

$$dp_t = \frac{D_t}{P_t}$$

The choice of these variables stems from the prior literature that showed a tight link between them. As indicated in Campbell and Shiller's (1988) log linearization of returns, price formation is defined by future dividends and the rate at which they will be discounted.

Cox-Ingersoll-ross model (CIR)

The CIR model is a stochastic differential equation (SDE), that aims to model the behavior of a mean reverting process. This model was first introduced by economists John Cox, Jonathan Ingersoll, and Stephen Ross in 1985. It is widely used in interest rate derivatives pricing.

Continuous form:

Let x(t) be the state variable that needs to be modeled. The SDE describes a mean reverting behavior:

$$dx_t = a\,(\,b - \,x_t)\,dt + \sqrt{x_t}\,\sigma\,dW_t$$

- x_t is the state variable.
- 'a' is the speed of reversion.
- 'b' is the long-term mean of x_t

- 'σ' represent the volatility of the random shocks of x_t
- W_t is a Wiener process

This model implies a reversion to the mean whenever x_t diverges from 'b'. And 'a' is the speed of reversion that multiplies the deterministic change in the quantity of x_t. The Wiener process W_t models a Gaussian random shock to x_t. The shock's volatility depends on the state variable, resulting in a non-constant diffusion term.

Discrete form:

The continuous form is tricky to calibrate. A discrete form would allow us to determine the statistics of the model.

$$\Delta x_t = a\,(\,b - x_{t-1}\,)\,\Delta t + \varepsilon_t\,(\,0\,, x_{t-1}\,\sigma^2)$$

As we can see, the change in x_t can't be regressed by a simple OLS since the error terms are heteroskedastic and are a function of the state variable. As the state variable increases, so does the volatility associated with it.

OLS form:

To regress this model using an OLS method, we have to have homoscedastic errors. By dividing by the $\sqrt{x_{t-1}}$ we obtain:

$$\frac{x_t - x_{t-1}}{\sqrt{x_{t-1}}} = \frac{a\,b}{\sqrt{x_{t-1}}} - a\sqrt{x_{t-1}} + \varepsilon_t\,(\,0\,, \sigma^2)$$

Which translates into homoscedastic error terms.

Generative model

An LSTM RNN suffers from high variance when trained with little data. In order to have a bigger data sample, I need to generate artificial data points that have the same properties of the historical dividend yield, dividend growth, and stock returns. Many models describe this relation in a linear fashion, which doesn't capture the real endogenous dynamics between my variables. Ang and Liu (2007) provided a model that describes the dynamics in an endogenous manner. The relation is described by a series of stochastic differential equations that are interrelated. Using the definition of return and a transversality assumption, they were able to derive the stochastic volatility, and the expected returns processes only by calibrating the dividend yield and the dividend growth into the historical data. Vice versa, by parametrizing 2 out of these 4 processes we can derive the remaining with strong restrictions on their dynamics. These derivations arise from a dynamical version of the Gordon model. This approach is similar to the present value model (Campbell and Shiller (1988)) but it's more tightly connected.

The continuous model:

Considering a variable x_t that describes the state of the economy, and follows a stochastic process:

$$\mathrm{d}x_t = \mu_x(x_t)\,\mathrm{d}t + \sigma_x(x_t)\,\mathrm{d}B_t^x,$$

With μ_{x_t} the drift and the σ_{x_t} the diffusion, which are functions of the state variable x_t . The dividend process is restricted by the presented process:

$$\frac{\mathrm{d}D_t}{D_t} = \left(\mu_d(x_t) + \frac{1}{2}(\sigma_{\mathrm{d}x}^2(x_t) + \sigma_d^2(x_t))\right)\mathrm{d}t + \sigma_{\mathrm{d}x}(x_t)\,\mathrm{d}B_t^x + \sigma_d(x_t)\,\mathrm{d}B_t^d,$$

For the sake of simplicity, they assumed that the Wiener processes, $\mathrm{d}B_t^x$ and $\mathrm{d}B_t^d$, are uncorrelated. The price of an asset is determined by the dividend and returns by the following formula:

$$\frac{\mathrm{E}_t[\mathrm{d}P_t] + D_t\,\mathrm{d}t}{P_t} = \mu_r\,\mathrm{d}t.$$

By iterating forward, we obtain the price of the asset:

$$P_t = \mathrm{E}_t\left[\int_t^T e^{-(\int_t^s \mu_r\,\mathrm{d}u)} D_s\,\mathrm{d}s + e^{-(\int_t^T \mu_r\,\mathrm{d}u)} P_T\right].$$

This formula describes the price formation at time t. In a more intuitive sense, it can be understood as the accumulation of anticipated dividends and price at time T, all discounted to time t. Iterating forward to infinity, the price P_t is only expressed as the expected infinite sum of discounted cash flows. That implies the transversality condition.

$$\lim_{T\to\infty} \mathrm{E}_t[e^{-(\int_t^T \mu_r\,\mathrm{d}u)} P_T] = 0$$

Under this condition, the existence of the dividend yield is preserved, and that holds almost surely.

Alternatively, by iterating the definition of returns to compute the stock price, we can determine the parameters of the return process. Suppose that D_t/P_t is a function of the state variable x_t, the cumulative return dR_t is implied by the application of Ito's lemma to the definition of return:

$$\begin{aligned}\mathrm{d}R_t = &\left(\frac{(\mu_x + \sigma_{\mathrm{d}x}\sigma_x)f' + \frac{1}{2}\sigma_x^2 f'' + 1}{f} + \mu_d + \frac{1}{2}(\sigma_{\mathrm{d}x}^2 + \sigma_d^2)\right)\mathrm{d}t \\ &+ \sigma_x(\ln f)'\,\mathrm{d}B_t^x + \sigma_{\mathrm{d}x}\,\mathrm{d}B_t^x + \sigma_d\,\mathrm{d}B_t^d.\end{aligned}$$

Empirical application

By employing this dynamic model, I can infer artificial data of dividend yield, dividend growth, return drift, and return volatility, calibrating just two of these processes.

Many studies suggest that dividend growth is not IID (see Lettau and Ludvigson, 2005; Bansal and Yaron, 2004). I will calibrate the dividend price ratio and the dividend growth rate with a heteroskedastic factor that is dependent on the dividend yield.

Let's $x_t = \frac{D_t}{P_t}$, with x_t following a mean reverting CIR model:

$$dx_t = \kappa(\theta - x_t)dt + \sigma\sqrt{x_t}\, dB_t^x$$

And the dividend growth follows:

$$d\, lnD_t = (\alpha + \beta\, x_t)dt + d\sqrt{x_t}\, dB_t^d$$

For the sake of simplicity, I considered that the wiener process of the dividend yield and the dividend growth are independent.

The return dynamics can be implied from the above calibration:

$$\mu_r(x) = \kappa + \alpha + \frac{(\sigma^2 - \kappa\theta)}{x} + \left(1 + \beta + \frac{1}{2}b^2\right)x,$$
$$\sigma_{rx}(x) = -\frac{\sigma}{\sqrt{x}}.$$

The dividend growth rate is predictable by dividend yield (see Ang and Bekaert (2007)). By running a simple OLS regression we can find the alpha and beta.

$$g_{t+1} = \alpha + \beta\, dy_t + \varepsilon_{t+1}.$$

Ang and Liu (2007) showed that the unconditional variance of the dividend growth rate can be matched by the formula below:

$$\mathrm{E}[(\ln D_s - \ln D_t - \mathrm{E}[\ln D_s - \ln D_t])^2] = \left(d^2 + \left(\frac{\beta\sigma}{\kappa}\right)^2\right)(s-t)\theta - \frac{\beta^2\sigma^2}{\kappa^3}(1 - e^{-\kappa(s-t)})\theta.$$

Calculating the variance from the historical data and having the regression parameters of alpha and beta and the dividend yield parameters, we can calculate d, the factor included in the conditional volatility of the dividend process.

Economical validity

Using a CIR model for the dividend price ratio models a mean reverting process that doesn't accept negative value, owing to the involvement of a square root component within the diffusion term. More intuitively, as prices go down the dividend yield goes up, increasing the conditional volatility of the process

which is very logical in financially distressed periods. As prices go down, volatility spikes due to uncertainty and fear.
Similarly, for the dividend process, uncertainty for future dividend payments is well modeled by including the dividend yield factor in the diffusion term.

Artificial neural network

Artificial neural network known also as deep learning is a sub-field of machine learning. It's a mathematical model that tries to predict future targets when it is well-trained and optimized. While the concept of using neural networks as a model is not new, the current surge in neural network applications can be attributed to the increased accessibility to substantial computational power, driven by advancements in semiconductor performance. It is inspired by the interconnectedness of the human neurons. A neural network is constituted of input, deep, and output layers and each layer contains many neurons. Within this section of the paper, I will delve into a comprehensive explanation of neural networks, then I will explain a more advanced setup called recurrent neural network (RNN) that can hold some memory. Lastly, I will explore the long short-term memory (LSTM RNN), which effectively addresses vanishing and exploding gradient problems.

Deep neural network

In Figure 1 - Neural Network (Zhou, V. (2020)), a graphical representation of a deep neural network (DNN) that have an input layer constituted of 4 input neurons (in blue), 2 output neurons (in green), and 2 deep layers containing 6 neurons each. In the following, I will explain each component of the DNN by defining essential jargon used by practitioners.

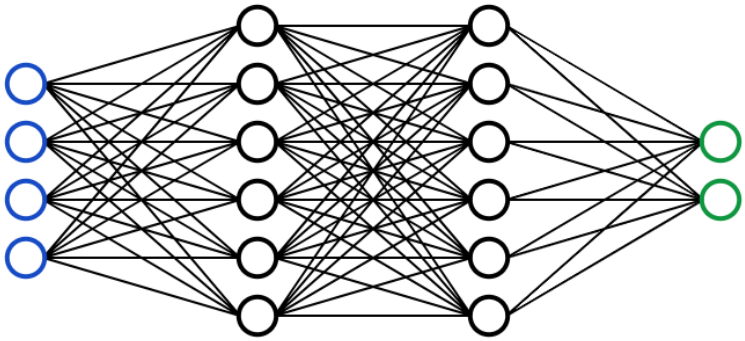


Figure 1 - Neural Network (Zhou, V. (2020))

-Feature: A feature is a variable that is believed to have predictive power over the intended target. It's determined by the architect of the neural network.

-Data matrix: It’s an (n x m) matrix used as a container of the input and output data. There exist 'm' features and 'n' examples. Taking Figure 1 - Neural Network (Zhou, V. (2020)) as an example, we have 6 features in our matrix data, 4 out of 6 are used to predict the remaining 2.

-Example: An example is a full information set describing a single instance of the data matrix (one line of the matrix). The number and the quality of examples play a substantial role in the model's efficiency.

-Layer: A layer constitutes a collection of neurons. The representation above illustrates four layers, with two of these being considered deep layers.

- Neuron: A neuron is a circle represented in the Figure 1 - Neural Network (Zhou, V. (2020)) above. The input neurons contain the example data, the output neurons contain the predictions, and the inside neurons contain calculated numbers. These calculated numbers are obtained by applying an activation function to the linear combination of numbers inside the neurons of the prior layer, plus a bias term.

- Batch size: A batch refers to a portion of the training data, upon processing of which the model updates its parameters using the gradient descent algorithm. For instance, a dataset of 1000 with a batch size of 10 will need 100 updates to pass through all the data and complete an epoch.

- Epoch: An epoch is a concept used to signify that the model has completed one full pass through the entire training data.

A neural network is a black box that takes some features as input and gives a prediction as an output. Every blue neuron represents a feature that is assumed to have a predictive capability over the output feature in green. For instance, let's assume that the first output neuron should predict whether it's a dog or a cat, and the second output neuron should predict if it's a male or a female. The 4 input neurons would take characteristics such as weight, size, face, and eye shape. The output layers should generate a number between 0 and 1 determining the probability of each predicted feature.

Data split

It's a common practice to divide the available data into three sets. The training, validation, and testing sets. As its name suggested, the training set constitutes the portion of data used to train the model. The validation serves as a way to assess the generalization capabilities of the model during its training process. Finally, the test dataset serves to derive the out-of-sample predictability power value.

We avoid relying solely on validation dataset metrics for model evaluation, as these metrics might be selectively chosen by the architect to showcase the lowest values among all the alternatives, introducing a potential downward bias in the R-square.

Forward propagation

This process goes from left to right. Each neuron contains a calculated number obtained by applying an activation function to the linear combination of numbers inside the neurons of the prior layer, plus a bias term. The weights that participate in the linear combination can be randomly chosen at initialization. Activation functions squash the linear combination input, producing an output between [0,1] for the sigmoid function (like logistic regression), and between [-1,1] for the tanh function. The none linear activation function is a crucial component to capture none linear relationships. The model predictions are

given in the last layer. These predictions are compared with the real target value by using a cost function. In my case, I've used the mean squared error (MSE) to penalize large deviations from the real values.

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2$$

Backpropagation

Our ultimate goal is to minimize the cost function which is a function depending on the initialized weights. This is achieved by adjusting these weights by the appropriate magnitude to ensure a well-trained model. By calculating the gradients of the cost function, we get the direction of the steepest ascent. To progress along the path of the steepest descent, the weights should be subtracted by the gradient vector scaled by α. The learning rate coefficient serves to mitigate the problems of overshooting gradient and slow learning process. Once the weights have been updated, this procedure should be iteratively performed over multiple epochs to approach the global minimum of the cost function. To robustly fit the data, the number of epochs shouldn't be less or more than a certain number (problem-specific). Over-training the model causes an overfitting (high variance) problem, where the model predicts the training set with high precision by capturing the noise process, but performs poorly on unseen data. On the other hand, an under-fitting (high bias) problem occurs when the model is trained for a few epochs. In this case, the model will be naive, unable to correctly predict either the training set or the test set.

Recurrent Neural Network (RNN)

The traditional neural network explained in the prior section is used for classification, regressions, and fraud detection problems. However, it can't be used to predict a time sequence, due to the absence of a memory mechanism that enables the model to use prior information of the series. RNN comes as a solution for such problems, with an architecture similar to the DNN, but with a small adjustment on the level of the neuron input. In a RNN the output of the neuron is taken again to be included in the linear combination of the next example. In this manner, historical information influences both the training and prediction stages for forthcoming examples.

Mechanism

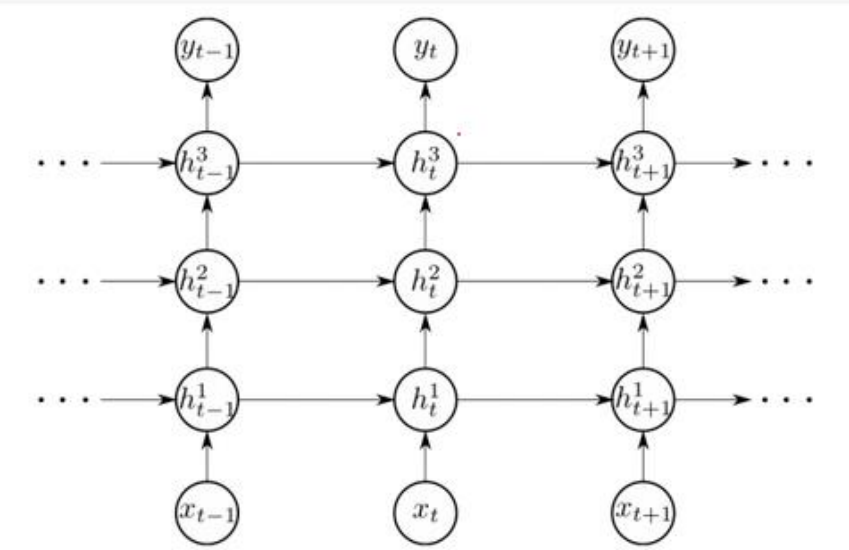


Figure 2 - Stacked RNN (Lambert, J. (2014))

$$h_t^{(l)} = f_h^{(l)}\left(h_t^{(l-1)}, h_{t-1}^{(l)}\right) = \phi_h\left(W_l^T h_{t-1}^{(l)} + U_l^T h_t^{(l-1)}\right)$$

The Figure 2 - Stacked RNN (Lambert, J. (2014)) above represents a stacked RNN. In a RNN we can include multiple lags to predict multiple steps in the future. Each row corresponds to a hidden layer, and each column represents the next time step, showing the evolution of the neurons along the lagged steps.

Let's take h_t^2 as an explicative example. h_t^2 is determined by the prior layer at time t denoted as h_t^1, and by the output of the same neuron at time step t-1, denoted as h_{t-1}^2. Considering one neuron per layer, we multiply each component with its corresponding weight W_l, U_l, and apply an activation function to it. As a consequence, lagged information is integrated into the prediction of the next step t+1.

Vanishing exploding gradient

Recurrent neural networks are useful for sequence analysis, but they come with the problem of vanishing or exploding gradient, which makes the optimization phase very difficult. I'll explain this problem by giving an example of a simple RNN made of 1 input neuron, 1 deep neuron, and 1 output neuron, illustrated in Figure 3 - Single RNN. In this example, we will train the model to predict the $n + 1^{th}$ data point based on all the past sequence of data $x_1, x_2, x_3, \ldots, x_n$).

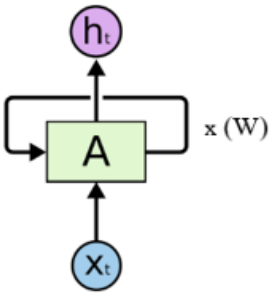


Figure 3 - Single RNN

When the model processes the second data point it will incorporates the first data point into the linear combination through the depicted feedback loop, which contributes with a factor of W* x_1. When the model take the 3rd data point, the 1st data point will also enter the linear combination but with a factor of $W^2 * x_1$. When we reach the n^{th} data point, the participation of x_1 in the prediction of the $n + 1^{th}$ point is $W^n * x_1$. This factor will also enter the loss function and the gradients. The problem comes when n becomes very large and W is smaller or larger than 1 causing vanishing/exploding gradients respectively.

In the case of a vanishing gradient, the $W^n * x_1$ factor will be very small, which in turn will make the gradient very small, causing very slow training of the model. This issue prevents us from attaining the global minimum, or even worse, could lead to converging to an unknown local minimum.

In the case of an exploding gradient, $the\, W^n * x_1$ factor will be very large, which in turn makes the gradient very large. In this scenario, the update will go beyond the necessary weight adjustment. This problem is milder compared to the vanishing problem because the training loss will increase when plotted against the number of epochs, making it easily recognizable. This problem is resolved by applying a long short-term memory RNN, which will be explained in the next section.

Long Short-Term Memory (LSTM)

An LSTM is a much more complex variant of RNN. Keeping everything similar to a RNN, an LSTM contains a structure within each neuron. That allows to have long-term memory called the cell state, a short-term memory called the hidden state, and a series of decision gates inside the structure.

Notation:

- C_t, h_t the cell state and the hidden state respectively
- x_t the input vector
- b_f , b_i , b_c , b_o the bias vector
- W_f , W_i , W_c , W_o the parameters matrices
- σ , tanh (the sigmoid and hyperbolic tangent activation functions respectively)

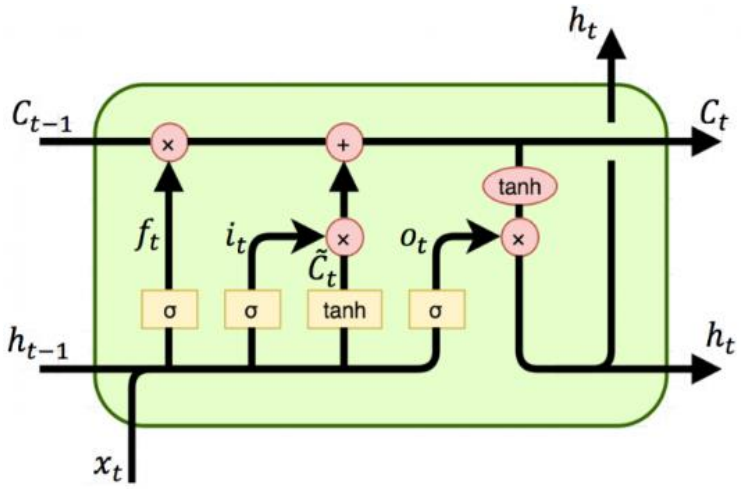


Figure 4 - LSTM unit architecture (dProgrammer lopez. (2019, April 6))

- $f_t = \sigma\,(W_f\,.[\,h_{t-1}\,, x_t\,] + b_f)$
- $i_t = \sigma\,(W_i\,.[\,h_{t-1}\,, x_t\,] + b_i)$
- $o_t = \sigma\,(W_o\,.[\,h_{t-1}\,, x_t\,] + b_o)$
- $\widetilde{C_t} = tanh\,(W_c\,.[\,h_{t-1}\,, x_t\,] + b_c$
- $C_t = f_t\ .\ C_{t-1} + i_t\,.\widetilde{C_t}$
- $h_t = o_t\,.\tanh(C_t)$

As presented in Figure 4 - LSTM unit architecture (dProgrammer lopez. (2019, April 6)), the architecture of a neuron is composed of a series of gates. A neuron receives 3 inputs: the data x_t, the long memory from C_{t-1} , and the short memory h_{t-1} .

The first gate that calculates f_t is called the forget gate. The activation function of this gate is a sigmoid that gives a number between 0 and 1, which gives the percentage of the long memory to be remembered.

Second, the combination of $\widetilde{C_t}$ and i_t is called the input gate which determines the amount added/subtracted from C_{t-1} * f_t . Based on the short memory and incoming input, it determines what should be retained from them for incorporation into the long-term memory. $\widetilde{C_t}$ is the long-term memory to be remembered from h_{t-1} and x_t. The percentage of the potential memory to be remembered from $\widetilde{C_t}$ is i_t.

Finally, o_t gives the percentage that should be remembered from the updated version of the long memory, which will be used to obtain the new short-term memory, h_t.

Although the cell state is modified many times, we notice that there are no weights or biases that can modify it directly, thereby avoiding the issues of vanishing or exploding gradients.

Methodology

This section will comprehensively outline the model and detail all the assumptions that have been employed in this research. I'll start by explaining the inputs of the LSTM model, the structure of the neural network, and all the fine-tuned hyperparameters. Subsequently, I'll proceed to the data section, discussing the chosen variables and the artificial data, as well as a justification for the generated data's compatibility.

In this work, I've used an LSTM model that takes two sequences of data, the dividend yield and the dividend growth rate, to predict the dividend growth rate at time t+1. Despite the possibility of taking the derived artificial drift and diffusion terms of returns, I've chosen to omit these variables since they are derived from the characterization of the dividend yield and dividend growth process. This implies that the information included within the latter is the same as that stored by artificial returns data. This step will make our model easier to train without restraining any useful information.

Dividend smoothing is a practice done by business managers that deteriorates the predictability of the dividend growth rate, but the literature argues that the dividend smoothing ability of managers is imperfect over the business cycle (see, for example, Gertler & Hubbard, 1993; Bernanke, Gertler, and Gilchrist, 1996). Thus I've used a 10 lag sequence as input for the DP and DG ratios, which covers a whole business cycle. This indicates that I require 10 lags to forecast the dividend growth rate for the subsequent step t+1.

Cost function

For model evaluation and training, I employed the mean squared error of predictions as the chosen cost function. The MSE is very sensible to high deviation from the real values, severely punishing the model by increasing the MSE considerably.

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

The use of the MSE will result in more smoothed and less volatile predictions, which will be discussed in the results section of this article.

Optimizer

After the forward propagation, comes the backward propagation which is a recursive process that calculates the gradient, and updates the model weights using an optimizer. An optimizer is the algorithm responsible for updating the parameters. Many algorithms are available at my disposal, including the most known like the gradient descent, gradient descent with momentum, and the RMSprop. However, the best optimization algorithm that studs up against criticism is the "Adam" optimizer, standing for adaptive moment estimation, proposed by Kingma and Ba (2014). It represents a fusion of the gradient descent with momentum and the RMSprop algorithms. The "Adam" optimizer is used to converge more quickly into the global minimum without being trapped in a local one. In the subsequent section, I will clarify the algorithm of the "Adam" optimizer.

At time t, the algorithm will calculate the gradient of the MSE:

$$\nabla_W MSE(W) = \left(\frac{\partial MSE(W)}{\partial W_1} \quad \frac{\partial MSE(W)}{\partial W_2} \quad \dots \quad \dots \quad \frac{\partial MSE(W)}{\partial W_n}\right)$$

After calculating the gradient, it will update the exponential weighted moving average of the gradient in the style of the gradient descent with momentum algorithm:

$$V_{\nabla_W}{}^{new} = \beta_1 . V_{\nabla_W}{}^{old} + (1 - \beta_1) . \nabla_W$$

$V_{\nabla_W}{}^{new}$ is the updated vector of smoothed gradient, that serves to dampen big gradients that go in the wrong direction, while keeping unchanged the gradient that goes in the right direction. The exponentially weighted moving average of the gradient takes the $\frac{1}{1-\beta_1}$ prior steps of the gradient when calculating the $V_{\nabla_W}{}^{new}$.

The second block of the "Adam" optimizer comes from the RMSprop algorithm, which calculates the S_{∇_W} illustrated below. The explanation of the role of S_{∇_W} will be explained later.

$$S_{\nabla_W}{}^{new} = \beta_2 . S_{\nabla_W}{}^{old} + (1 - \beta_2) . (\nabla_W)^2$$

A correction of V_{∇_W} and S_{∇_W} is needed, since these are exponentially weighted moving averages that are initialized at zero at time t=0. This implies a downward bias due to the initial condition. To correct this bias, the 'Adam' optimizer uses a corrected version of V_{∇_W} and S_{∇_W} , illustrated below:

$$V_{\nabla_W}^{corrected,new} = \frac{V_{\nabla_W}{}^{new}}{(1-\beta_1{}^t)} \quad ; \quad S_{\nabla_W}^{corrected,new} = \frac{S_{\nabla_W}{}^{new}}{(1-\beta_2{}^t)}$$

As β_1 and β_2 are less than 1, $\beta_1{}^t$ and $\beta_2{}^t$ converge to zero. It implies that the first iterations are corrected, and the rest will not be affected because they will be divided by 1.

As I've already clarified, the "Adam" optimizer is a blend of the momentum gradient descent and the RMSprop algorithms. The LSTM weight vector, denoted as W, will be updated according to the below equation:

$$W_{new} = W_{old} + \alpha . \frac{V_{\nabla_W}^{corrected,new}}{\sqrt{S_{\nabla_W}^{corrected,new}} + \varepsilon}$$

The hyper parameters α, β_1, β_2, and ε are initialized as follows:

- α = *0.001*
- β_1 = *0.9*
- β_2 = *0.999*
- ε = 10^{-8}

These are the standard hyperparameters, that have proven to be the most beneficial to my model after many trial and error. The ε factor is included to conserve the stability of the update when $S_{\nabla_W}^{corrected,new}$ is near zero, which guaranty a none explosive update of the weights vector W.

This optimizer ensures a smooth update process, by incorporating an exponentially weighted moving average, denoted as $V_{\nabla_W}^{corrected,new}$. That helps smoothen the optimization when $V_{\nabla_W}^{corrected,new}$ becomes significant, since it will be tempered by $\sqrt{S_{\nabla_W}^{corrected,new}}$ which also becomes substantial simultaneously.

The “Adam” optimizer shows its resilience in different neural network types and architecture, thanks to its dynamic updating system that adjusts the gradient updating steps accordingly, without the need to fine-tune the learning rate, α.

Batch size

The batch size corresponds to the number of training examples that are intended to be incorporated in each iteration of the weight update. Three ways of learning emerge from the batch size choice: Online learning, mini-batch Adam, Full batch Adam.

When the batch size is equal to 1, the weights are updated for each training example. In this case, the model is faced with high variance problems, which means that the model overfits the training data and memorizes the noise process. In addition, the scheme taken by the “Adam” optimizer becomes very noisy and takes a lot of time on its way down in the loss landscape. It might never converge to the global minimum due to its stochasticity. Finally, it doesn’t get the benefits of parallelism, leading to inefficient GPU utilization. This corresponds to online learning

The full batch approach involves utilizing all the available training examples to perform one weight update for the model. As reported by Keskar et al. (2017), this approach also suffers from drawbacks, like sensitivity to outliers, difficulty in escaping large minima, high computation cost, and inability to generalize to unseen data known under the term "Generalization gap".

Ultimately, an optimal batch size, neither too large nor too small, proves to be the most effective during the training phase. Besides the computational efficiency achieved through mini batch Adam, it enables the neural network to navigate the multi-dimensional landscape of the loss function without getting trapped to sharpe minima. Instead, the network can converge towards flatter minimizers.

Despite the tempting loss values offered by sharp minimizers when training with full batch size, it might be prone to the "Generalization gap" described in Figure 5 - A sharp minimum compared to a flat minimum (Keskar, N. S. et al. (2017)).

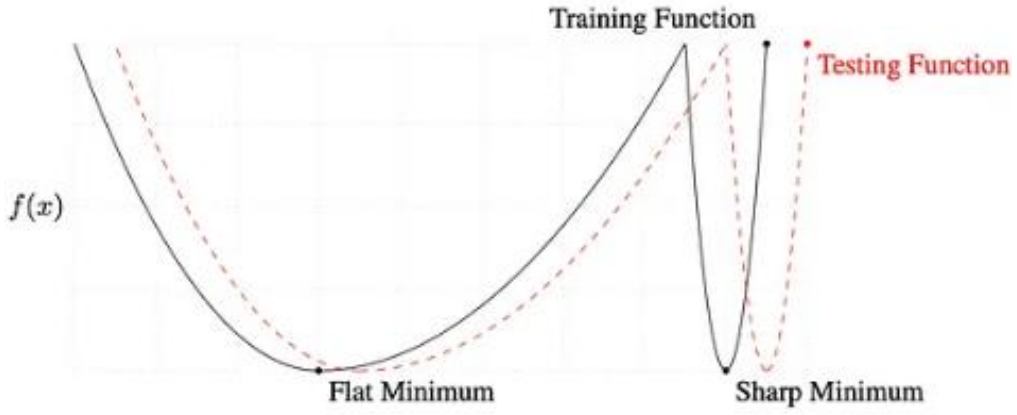


Figure 5 - A sharp minimum compared to a flat minimum (Keskar, N. S. et al. (2017))

By assuming that the link between features and labels in out-of-sample data is akin but not identical to that observed in the training, the sharpe minimizers will produce peaks in the cost function (red dotted line) for the unseen data. Contrarily, a flat minimum will yield a low loss value for unseen data.

No rule implies a specific size for the mini-batch, instead, it is a trial-and-error procedure. In my case, I chose a mini-batch size of 12.

Network structure

A neural network structure will be determined by the number of input, output, and deep neurons. The input neurons will comprise two sequences, each consisting of 10 lagged values of the dividend yield and dividend growth rate. The output layer is the future step t+1 of the dividend growth rate.

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                 Output Shape              Param #
=================================================================
 lstm (LSTM)                  (None, 10, 100)           41200

 lstm_1 (LSTM)                (None, 10, 50)            30200

 lstm_2 (LSTM)                (None, 10, 30)            9720

 lstm_3 (LSTM)                (None, 10)                1640

 dense (Dense)                (None, 1)                 11

=================================================================
Total params: 82,771
Trainable params: 82,771
Non-trainable params: 0
_________________________________________________________________
```

Table 1 - Structure of the LSTM RNN

Table 1 - Structure of the LSTM RNN resumes my model structure. Since there are no established guidelines for the number of layers and neurons, and no existing neural network models related to the subject, it will be a trial-and-error problem. I've tried many different structures, in which I based my choice on the lowest validation loss. I've settled on the structure above, with 4 deep layers. The initial layer contains 100 neurons, followed by the second layer with 50 neurons, the third with 30 neurons, and the fourth with 10 neurons. A cumulative count of 82,771 parameters for training. By having a decreasing number of neurons shaped like a funnel, the model learns complex dynamics between inputs, and compresses them to the most relevant in the final layers. It allows the earlier layers to learn simple dynamics, while deeper ones can extract more abstract relations. Furthermore, it is less computationally intensive compared to allocating a larger number of neurons, which would lead to an increased count of trainable parameters.

Regularization

Regularization techniques are used to prevent overfitting and enhance generalization capabilities. Many techniques at our disposal can be used including L1 lasso regression, L2 ridge regression, and the dropout technique. I have opted for early stopping regularization, where the model halts its learning process after the validation loss exceeds its prior values for a predefined number of epochs consecutively. I've chosen 80 epochs to visualize the valid/train loss as a function of epochs and manually chose the optimal number of epochs for training (details in the results section).

Data

My data, whether real or artificial, will be given a lot of importance in this study because it was a barrier for a lot of potential neural network studies on this subject.

I used the Stock market data for the S&P 500 extracted from Robert Shiller's book, *Irrational Exuberance* [Princeton University Press 2000, Broadway Books 2001, 2nd ed., 2005]. The data can be downloaded by clicking on the following link **U.S. Stock Markets 1871-Present and CAPE Ratio**.

Among a plethora of financial data, we can extract the monthly stock price alongside the corresponding dividend information. From this data, I constructed the yearly dividend yield and dividend growth rate without reinvestment from 12-1871 till 12-2022. The choice of reinvestment strategy is based on Chen, L. (2009) in which he showed that reinvested dividends are contaminated by returns. The reinvested dividend behaves like returns with a correlation factor of 0.65 compared to 0.18 for the non-reinvested dividends, thus destroying information held by the pure dividends.

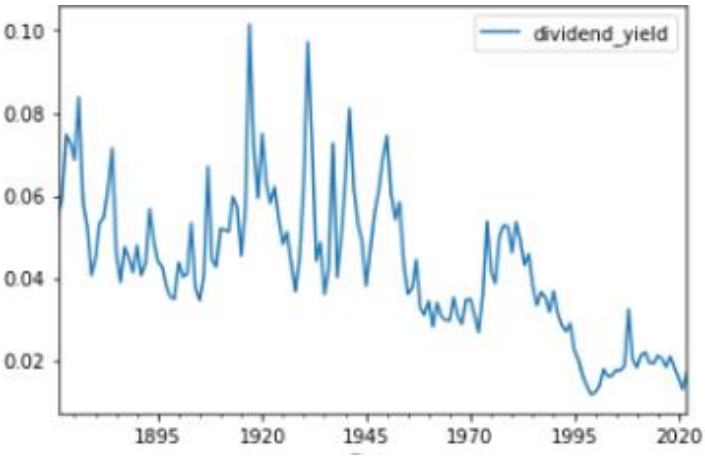


Chart 1 - Historical dividend yield

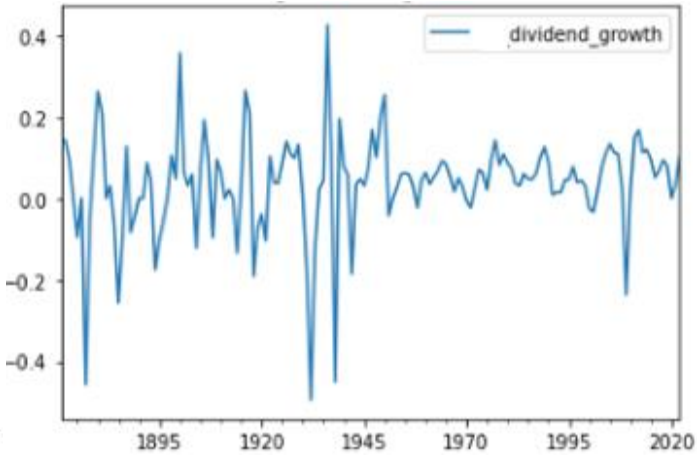


Chart 2 - Historical dividend growth rate

	dividend_yield	ln_dividend_growth
count	152.000000	151.000000
mean	0.042906	0.036759
std	0.017701	0.120632
min	0.011682	-0.494696
max	0.101471	0.426519
kurtosis	0.279611	5.745710
skewness	0.455448	-1.302313

Table 2 - Descriptive statistics of the historical data

In the Charts 1 and 2 above, the historical dividend yield and dividend growth rate can be observed spanning the years from 1871 to 2022, with a descriptive statistics table that shows the mean, standard deviation, kurtosis, and the skewness. These statistics will become a benchmark to assess the validity of the artificial data. The dividend yield and dividend growth have high standard deviations, especially during the distressed times of major financial crisis, and the dividend growth is very leptokurtic and negatively skewed. As shown in the table, as well as the complimentary density distributions (Appendix) of these two datasets, we can conclude that the presented distributions are not normal.

The most attractive observation in chart 1 and 2, is the structural shift of the datasets after the second world war. The dividend growth rate witnessed much lower volatility than the prewar period, and a big downshift in the dividend yield mean.

	ln_dividend_growth	dividend_yield
ln_dividend_growth	1.000000	-0.130223
dividend_yield	-0.130223	1.000000

Table 3 - Pre-war historical correlation

	ln_dividend_growth	dividend_yield
ln_dividend_growth	1.000000	0.247217
dividend_yield	0.247217	1.000000

Table 4 - Post-war historical correlation

Table 3 - Pre-war historical correlationand Table 4 - Post-war historical correlation30, also show a shift in the correlation between dividend growth rate and the dividend yield. During the prewar period, the correlation of -0.13 aligns well with findings in financial literature. It is consistent with the concept that higher dividend growth is accompanied by price increases, leading to lower dividend yields. After the war, the whole logic is reversed for which there is no theoretical explanation.

The reported shifts can be caused by smoothing and manipulation of dividends as Chen, Da, and Priestley (2012) have reported. The consequences of these shifts will have a big impact on my model's ability to predict post-war dividend growth rates (developed in the results section).

Artificial data

Training an LSTM neural network demands a big dataset to get reliable results. However, the collected data from 1871 to 2022 gives 151 data points, which is not enough. As a consequence, I used the model proposed by Ang and Liu (2007) to generate data. I calibrated the dividend yield using a CIR model, but dealing with the continuous form of the CIR model posed challenges. Thereby, I made the calibration in its discrete form presented below:

$$\frac{x_t - x_{t-1}}{\sqrt{x_{t-1}}} = \frac{a\,b}{\sqrt{x_{t-1}}} - a\sqrt{x_{t-1}} + \varepsilon_t\,(0, \sigma^2)$$

	(-a)	ab
Coeff	-0.09061	0.003653
std error	0.041611	0.001615
t-stat	-2.17753	2.261969
p-value	0.031015	0.025146
Sigma	0.046037	

Table 5 - Coefficients of the OLS regression

Table 5 summarizes the results of the OLS regression. The 'a' coefficient was found to be 0.09, while the 'a.b' coefficient stood at 0.00365. Notably, their p-values are significant at 0.0031 and 0.0251 respectively. The simulated data will be generated from the SDE below, using Python.

$$dx_t = 0.09061\,.\,(\,0.040318 - \,x_t)\,dt + \,0.046037.\sqrt{x_t}\;\mathrm{d}B_t^x$$

Next, the dividend growth rate will be modeled using a stochastic differential equation dependent on the dividend yield. The drift term will be derived through a regression of the dividend growth rate on the lagged dividend yield. The outcome of this analysis is presented in Table 6.

$$d\;lnD_t = (\alpha + \,\beta\;x_t)dt + d\sqrt{x_t}\;\mathrm{d}B_t^d$$

	coef	std err	z	P>|z|	[0.025	0.975]
Intercept	0.1729	0.028	6.268	0.000	0.119	0.227
np.exp(shifted_ln_dividend_yield)	-3.1611	0.705	-4.484	0.000	-4.543	-1.780

Table 6 - Coefficients of the OLS regression of the dividend growth on lagged dividend yield

Alpha (α) is equal to 0.1729 and beta (β) equal to -3.1611 with respective z-scores of 6.26 and -4.484, which shows high significance. The coefficient 'd' is a function of b, β, σ, κ, and θ (formula in the theoretical framework section). Having established these coefficients, the calculated value of 'd' is 0.4974, and the simulated dividend process is outlined below.

$$d\;lnD_t = (0.1729 - \,3.1611x_t)dt + 0.4974\sqrt{x_t}\;\mathrm{d}B_t^d$$

After the calibration of my generative model, I did a simulation on Python (Code snippet 2) of 2000 data points, which is considered a small dataset size for training an LSTM. The computational power at my disposal is limited, making it difficult to train the model on a larger dataset. Table 7 represents some statistics of the artificial data, with the accompanying graphs and plots in the appendix 2.

	Divdiend_yield	ln_dividend_growth
count	2000.000000	2000.000000
mean	0.042780	0.038492
std	0.023834	0.125761
min	0.000474	-0.599663
max	0.142910	0.414968
kurtosis	1.539585	1.299729
skewness	1.111878	-0.825574

Table 7 - Descriptive statistics of the artificial data

The generated data catches well the historical mean of the dividend yield and dividend growth rate, only with a 0.29% and 4.7% difference respectively. Similarly, the standard deviation of the generated data differs from the historical standard deviation by 12.46% for the dividend yield, and 4.25% for the dividend growth rate. Although there are significant differences in the magnitudes of kurtosis and skewness, the model is still able to achieve the correct signs. The significant difference between the maximum and minimum values didn't come as a surprise, given the greater likelihood of obtaining extreme values during the simulation of 2000 data points in comparison to the 152 historical points.

	ln_dividend_growth	Divdiend_yield
ln_dividend_growth	1.000000	-0.530809
Divdiend_yield	-0.530809	1.000000

Table 8 - Correlation coefficient of the artificial data

The correlation between artificial data sequences is more significant than their historical peers as shown in Table 8. This big difference was anticipated, as a theoretical framework suggests a significant negative correlation, which would serve as a hypothetical reference. It's the result of the simulated data that respects the boundaries given by Ang and Liu (2007).

Train/val/test

We avoid relying solely on validation dataset metrics for model evaluation, as these metrics might be selectively chosen by the architect to showcase the lowest values among all the alternatives, introducing a potential downward bias in the R-square. My split is 62.5% for the training data, 18.75% for the validation, and the similarly for the test data.

Normalization

Before the training phase, we have to scale down our data (features and labels) to squash them between 0 and 1. This transformation will help us to get a more stable training phase, that mitigates the risk of encountering large gradients. This transformation is named normalization:

$$X_{new} = \frac{X - X_{min}}{X_{max} - X_{min}}$$

I utilized the "MinMaxScaler()" object from "sklearn.preprocessing," employing its ‘fit’ and ‘transform’ methods. Now, my data is ready to be fed to the model.

In a nutshell, I will train the model using two artificial datasets, each containing 10 lagged values: the first for dividend growth rate and the second for dividend yield. The objective is to predict the dividend growth for the subsequent time step (t+1). For optimization, I used the "Adam" optimizer in conjunction with the mean squared error (MSE) cost function.

Results

During the training phase, I collected the training and validation loss in a Python object called 'list', and plotted them against the epochs. Normally, both training and validation losses should exhibit a decline following each epoch. But at a certain point, the validation loss begins to ascend, signaling the overfitting of the model. At that moment, I stopped the training and I saved my model corresponding to the lowest validation loss.

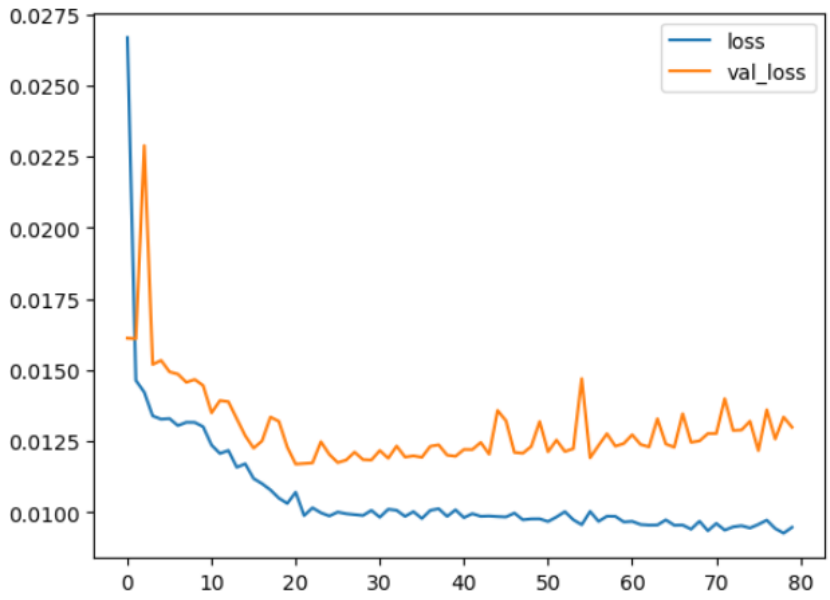


Chart 3. 1 - Training and validation MSE of the model through epochs

Chart 3. 1 represents the plots of training and validation losses for 80 epochs. The losses are very random and contain a lot of oscillations, which is not very surprising due to the nature of the data, which is generated from a model that contains a random component. Additionally, it's not usual to have a validation loss lower than the training loss, which is the case for the first epoch. This phenomenon can arise due to the initialization of weights and biases, which might yield unpredictable outcomes during the initial epoch. Additionally, simpler examples within the validation set could contribute to this occurrence.

I chose to save my model when it reaches the 21st epoch, as it yielded the lowest validation loss of 0.0117 and a training loss of 0.0107 corresponding to a marginal difference of 9.3%. However, this doesn't necessarily guarantee strong generalization capability to unseen data, even though the validation loss is computed on unseen data. The unbiased loss is computed on the test dataset, yielding a promising result of 0.0119 compared to 0.0117 for the biased estimator. This translates into good generalization capabilities.

All the calculated metrics are computed on artificial data, with none being based on historical data from 1882 till 1945.

The MSE for the historical data is 0.128, indicating a marginal deviation of 7.56% from the loss observed on the testing set. Achieving these outcomes showcases the promising potential of this model, even

though it has not been exposed to historical data. However, this result doesn't give any indication of whether it outperforms a benchmark or any results of previous studies.

Re-scaled data

Computing the R-square metric proposed by Goyal and Welch (2008) gives us the ability to compare our model to prior methodologies. To do so, we should re-scale back the predicted data, by reverse engineering the normalization formula:

$$X = X_{new}\,(X_{max} - X_{min}) + X_{min}$$

With $X_{max,historical}$ = 0.4265 and $X_{min,historical}$ = -0.4946

Table 9 shows the statistic of the re-scaled predictions and historical values of the dividend growth rate from 1882 till 2022 including 1882.

	Prediction rescaled	Real values rescaled
count	141.000000	141.000000
mean	0.065511	0.037894
std	0.074686	0.113740
min	-0.197495	-0.494696
max	0.181431	0.426519
kurtosis	1.031781	6.135800
skewness	-0.877572	-1.178816

Table 9 - Descriptive statistics of the predictions and the historical dividend growth rate

The predicted data's mean and standard deviation deviate significantly from the actual values. The big difference in the mean is tied to the divergence in the post-war predictions, which is evident in Chart 3. 2. The big discrepancy in the standard deviation didn't come as a surprise, considering the model's training with the MSE loss function, which penalizes substantial fluctuations. This characteristic compels the model's predictions to exhibit lower MSE and subsequently reduced volatility. This is also shown in the minimum and maximum, which are not as extreme as their historical peers. Kurtosis and skewness are far from the historical ones but the signs are correctly predicted.

Chart 3. 2 represents the historical values plotted in orange, the predicted values are in blue and the green line is the mean of the historical dividend growth rate.

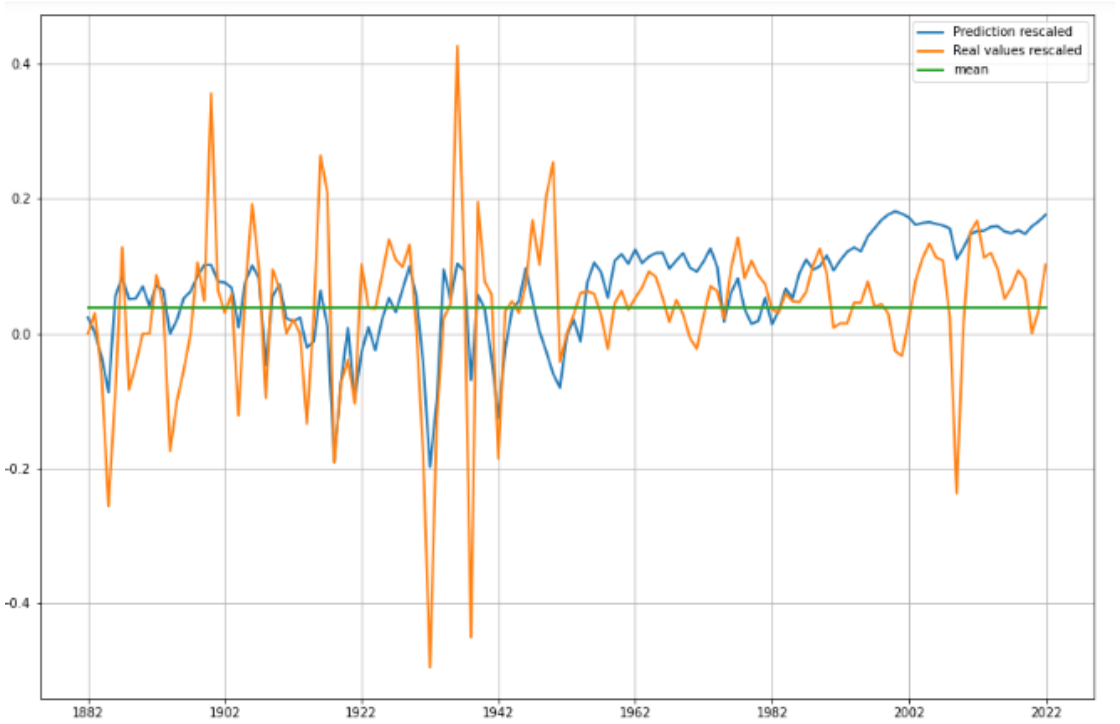


Chart 3. 2 - Dividend growth rate predictions plotted against the historical dividend growth rate

The predictability during the pre-war period is noticeably superior compared to the post-war era. The predictions are most often higher than historical values from the beginning of 1945, which explains why the mean of the predictions came much higher than the historical one. In order to assess whether a predictive model is stable and well-defined, Goyal and Welch (2008) proposed the out-of-sample R-square (R^2_{OOS}):

$$\Delta\overline{d_t} = \frac{1}{q}\sum_{i=1}^{q}\Delta d_t$$

$\Delta\overline{d_t}$ is the mean of the dividend growth rate, with q denoting the start of the forecasting period. In my case, $\Delta\overline{d_t}$ should be calculated as the mean of the first 10 observations. However, a more realistic choice would include the whole sample, which adversely affects the R^2_{OOS} of my model.

$$R^2_{OOS} = 1 - \frac{\sum_{t=q+1}^{T}(\Delta d_t - \Delta\hat{d}_t)^2}{\sum_{t=q+1}^{T}(\Delta d_t - \Delta\bar{d}_t)^2}$$

This statistic compares the ability of the historical mean over my model's ability to predict the target values. When R^2_{OOS} takes a negative value, the mean outperforms my model. Vice versa, when the R^2_{OOS} is positive, my model outperforms the historical average in predicting dividend growth rate.

Then, I applied the Clark and West (2007) adjusted mean square prediction error (MSFE), to assess the statistical significance of the R^2_{OOS} . Under the null hypotheses, the predictive power of my model is similar to the historical mean's ability.

$$f_t = (\Delta d_t - \Delta\bar{d}_t)^2 - [(\Delta d_t - \Delta\hat{d}_t)^2 - (\Delta\bar{d}_t - \Delta\hat{d}_t)^2]$$

Then I regressed f_t on a constant and calculated the corresponding Newey-West standard error. Subsequently, I computed the p-value of the upper tail using a normal distribution. The Newey-West standard error known also as Heteroscedasticity and Autocorrelation Consistent standard error (HAC), estimate the standard error of the regression coefficient in the potential presence of heteroscedasticity and autocorrelation within residuals.

The results of the R^2_{OOS} are presented in Chart 3. 3 with an expanding window from 1882 till 2022 at a yearly frequency:

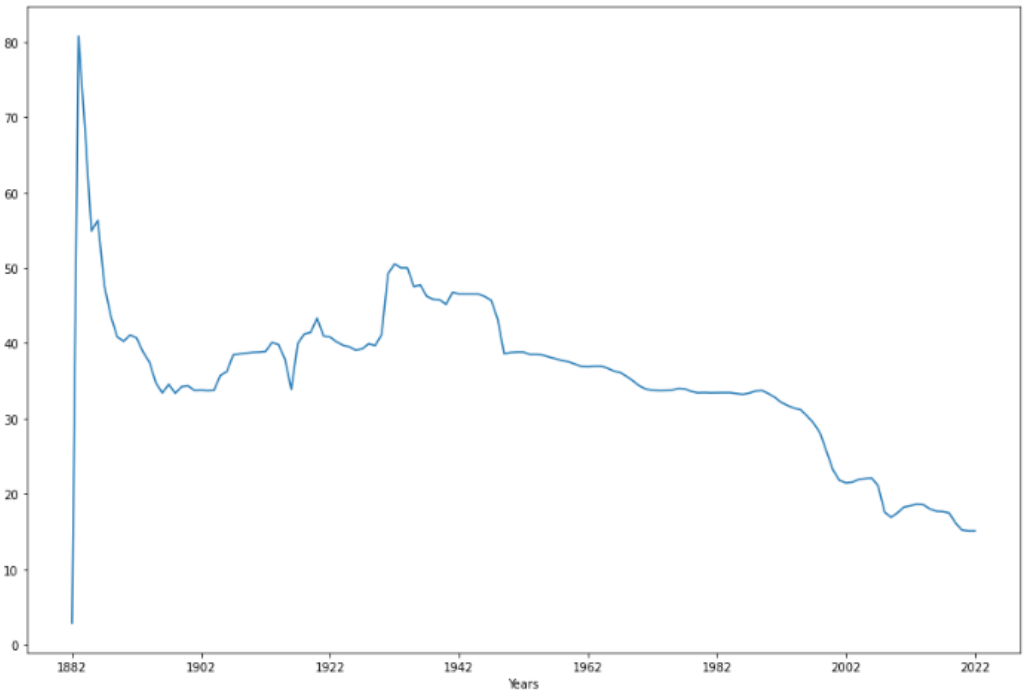


Chart 3. 3 - Cumulative R_OOS^2 of the predictions

The first spike in R^2_{OOS} won't be taken into consideration, because it's only calculated over the first two predictions. For the prewar period, the R^2_{OOS} is bounded between 35 and 50 and degraded after the second world war. The R^2_{OOS} spanning the whole prediction period from 1882 till 2022 stands at 15.09, with a p-value of 0.7%. This level of significance is notably high, surpassing the 1% threshold. But this significance might be driven by the values of the pre-war period. To have more comprehensive results, I divided the set into two distinct periods, the pre-war and post-war periods.

For the prewar period, the R^2_{OOS} stands at 46.52%, demonstrating significance at the 1% level, as indicated by a p-value of 0.18%. For the post-war period, beginning from 1946 till 2022, the R^2_{OOS} stands at -121.57. Notably, my model performed worse than the historical mean within this period. This result supports the findings of Chen (2009), Chen, Da, and Priestley (2012), and Zhu et al. (2018), claiming that manipulation and smoothing combined with shifts in corporate financial policies blur the predictability of the dividend growth rate.

My model showed a big predictability power over the benchmark for the pre-war period. This result was unexpected, given that the model was fine-tuned using synthetic data and did not have exposure to historical data.

Discussion

In my attempt to predict future dividend growth, I introduced a whole new methodology that was never used, namely LSTM RNN. The lack of data was the main obstacle faced by other researchers. This article demonstrates that employing a model to create artificial data and feeding it to a neural network can outperform the benchmark with a notable significance level.

Despite adopting an entirely new approach, the distilled findings within the theoretical framework draw to me a guiding path through this research. It also illuminated distinctions between reinvested dividends and the non-reinvestment strategy, along with shedding light on the conceivable vulnerabilities the model could encounter concerning post-war dividend growth. The similarities and opposition in the literature played an important qualitative judgment to obtain such results.

In the data generation section, I only reported one of two generative models to obtain the data. The first-order VAR model used in (Engsted & Pedersen, 2010) seemed very naïve and didn't model any endogenous relation between dividend yield and dividend growth series. It structured the dividend yield and dividend growth as basic AR(1) models, introducing a correlation applied to the shocks of these two time sequences. Conversely, the model presented by Ang and Liu (2007) demonstrated an ability not only in capturing the initial and secondary central moments of the DG, but also in catching the right orientations of skewness and kurtosis.

The big challenge came with the training of the neural network. After trial and error, the reported structure of the model was chosen from hundreds of combinations. The number of layers and neurons was a time-exhaustive process to find a good structure for the model. The choice of the activation functions, a pivotal decision in conventional neural networks, did not pose an issue in my scenario. This is because the LSTM RNN has an unchangeable pre-specified activation functions. For the optimizer, I chose the "Adam" optimizer, a choice rooted in industry best practices. This choice implies a dynamic gradient descent approach, eliminating the need for a separate fine-tuning step for the learning rate.

Answer to the research question

In defiance of the established notion that predicting the dividend growth rate is largely implausible, I was able to predict the dividend growth rate, achieving superior predictions for the pre-war era. As a result, I was able to verify the validity of the model proposed by Ang and Liu (2007). This is due to the model's capability to make reliable predictions during the pre-war era, despite its training only relied on synthetic data generated from their proposed model.

My approach presents many advantages and disadvantages. These advantages are:

- Ability to outperform the approaches presented in prior research for the pre-war period, with a s ignificant R^2_{OOS} at the 1% significance level.
- The model was able to catch extreme drawdowns in the dividend growth rate before the onset o f a financial crisis.
- An LSTM RNN is a non-parametric model for predictions. It can catch complex behaviors between the input series, as opposed to conventional time series models.
- The ability for an LSTM to model non-stationary time series.

Conversely, some disadvantages should be presented:

- The learning phase can be very long, especially for an LSTM RNN.
- Unlike a conventional mathematical model, a neural network operates as an enigmatic entity, processing inputs and producing outputs devoid of insight into the underlying dynamics.
- The hyperparameter tuning phase can be very long. It is also prone to small mistakes that can create big unexplainable errors in results.
- This approach involves a dual-layer error potential. When the model has prediction ability, both t he generative component and the LSTM are considered accurate and valuable. Conversely, if the outcomes lack alignment with the intended objectives, one of two issues could be at play: either the generative model fails to grasp real-world dynamics, or the LSTM RNN requires further calibration. This significant quandary might compel researchers to prioritize improving the LSTM RNN, even if the generative model isn't accurate.

Obstacles and difficulties

The main difficulties during the training of a neural network are the hyperparameter tuning, batch size, and number of epochs, … Any small change can distort the model. Also, finding quality data for training and testing is a crucial process in modeling. In my case, the caliber of the generative model and the outcomes of hyperparameter tuning contribute to the aforementioned dilemma.

The second obstacle is specific. The limited computing power that I have on my hardware was a big drawback. I encountered RAM overload, leading to crashes when using large batch sizes. That’s why I only generated 2000 data points, which simplifies model training without encountering resource issues.

Conclusion

Executive summary

Predicting the dividend growth rate presents a challenging task for finance practitioners. An important literature that presents trends, oppositions, and similarities has tackled this area without a clear conclusion. The absence of a substantial historical dataset size stood as a big difficulty against neural network modeling.

It is now possible to create synthetic data that is a reliable alternative to historical data using stochastic differential equations, as proved by Ang and Liu (2007). By effectively addressing the complexities of data challenges, an LSTM neural network can undergo iterative training, refining its performance over multiple attempts.

The result for the pre-war period is promising, with an R^2_{OOS} of 46.52% at a significance level of 1%, beating the benchmark set by Goyal and Welch (2008). For the post-war period, the model wasn't able to perform well, which raises questions on the validity of the generative model. A possible anomaly might be present in the historical dividend yield, potentially altering its behavior in the postwar period. I delve further into potential alternatives to surpass this problem in the “limitation and suggestion” section.

Managerial implications

Many managers disregard the importance of the dividend growth rate and capitulate on the equity premium prediction. But, if the prediction of the expected dividend growth is constant, then a simple regression of the returns on the dividend yield should deliver a good result, potentially necessitating a substantial revision of prevailing financial theories.

I introduced an innovative framework that possesses the capability to spark inspiration for a wide range of professionals, including asset managers, risk analysts, asset allocators, equity researchers, and numerous others. Once this neural network is successfully trained, the process of generating predictions becomes straightforward. These predictions can then be employed to develop a corrected dividend yield, effectively filtering out fluctuations tied to expected dividend growth. This paves the way for constructing a predictive model aimed to estimate stock returns.

Hence, I urge managers to objectively weigh the advantages and disadvantages of this approach, enabling them to make an informed choice about delving into this realm. Based solely on the insights gleaned from my research, I would recommend considering an exploration of this avenue.

Theoretical implication

Regarding theoretical consequences, I was able to set a predictive model for the dividend growth that challenges many researchers' claims that the dividend growth rate is iid.

In addition, I obtained confirming results with Chen, L. (2009), that there is a reversal in the dividend growth predictability between pre and post-war periods. Dividend smoothing, different sets of firms in the market index, and reduced dividend growth volatility might be the cause behind these findings.

12 442 words

Finally, I was able to introduce a new approach to tackle dividend predictability which plays a big role in financial valuation which future research can opt it as their milestone beginning for any new attempt on this matter. I will give a series of recommendations for future research that can enhance the predictability of the post-war period.

What I've learned

Through this research thesis, I have acquired proficiency in employing LSTM neural networks, a more intricate architecture compared to DNN. This implies that my learning encompasses the utilization of DNNs, RNNs, and a distinct variant of RNNs equipped with a specialized unit designed to retain both long and short-term memory. At the beginning of my research, it seemed very complicated, but after trying and learning from my mistakes I acquired valuable knowledge about neural networks. Also, I have gained extensive expertise in the quantitative aspect of financial markets. The underlying factors that drive price formation, namely anticipated dividend growth, and returns, are integral to the dynamics of market fluctuations. Through a comprehensive review of the literature, I have been provided with a roadmap for my research journey, encompassing established predictive models for returns and dividend growth. This knowledge equips me with a comprehensive toolkit to excel in roles related to asset management.

Limitations and suggestions for future research

My approach is confronted by many limitations:

- The suitable structure (number of layers, number of neurons, batch size …) is heavily dependent on the training dataset, which can be different from mine since it's generated from a stochastic process. That's why the training phase is also a stochastic process.

- The restrain posed by the limited computational power has led me to use a sample of 2000 data points, which is deemed relatively modest when training a neural network.

- The average and standard deviation of the synthetic data align quite closely with their historical peers. The model successfully produced data that follows the intended skewness and kurtosis patterns, although the scale of these latter two aspects doesn't accurately reflect reality. Consequently, the credibility of the generative model could be brought into question.

- The dividend yield has undergone a shift in behavior since the end of World War II, accompanied by a decrease in the volatility of the dividend growth rate. These two shifts are the potential cause of poor post-war predictability.

To surpass these limitations, I have a series of recommendations:

- To begin, use a powerful machine in order to explore different neural network structures.

- Try to use the data that is adjusted to inflation. It is worth to integrate inflation into the equation which might have a predictive power.

- As I have demonstrated, the data dynamics had shifted by the end of WW2. Thereby, it is worth calibrating 2 generative models, each based on the corresponding statistics of its timeframe.

- In the paper presented by Ang and Liu (2007), different generative models can be used. In my case, I've characterized the dividend yield to obtain the dividend growth series. In the context of the same paper, it is worthwhile to explore the effectiveness of other models that have been presented, given their potential relevance to our subject.

- Generative Adversarial Network (GAN) is an AI model that can simulate data based on the historical statistics of the original dataset. Its ability to generate multiple time series for different variables (dividend yield, stock returns, dividend growth rate) in an endogenous way, paves the way to include a bigger set of time series variables. Variables like stock variance, default yield spread, and the payout ratio can be added to the existing features. These recommendations are based on the results obtained by Zhu et al. (2018), showing a predictive power of these features over the dividend growth rate.

- Zhang et al., (2019) presents an Attention-based LSTM model for financial time series prediction. It was first used in image recognition and graphic transformation. I suggest utilizing this model, which is capable of assigning weights to the most crucial data points that exert the greatest influence on the future development of the dividend growth rate. Subsequently, this weighted input can be fed into the neural network.

Bibliography

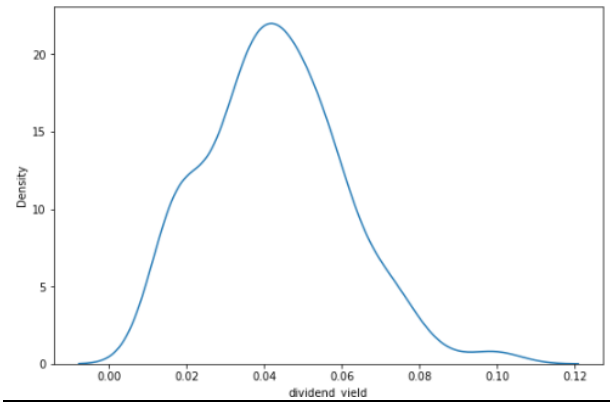
- Ang, A., & Bekaert, G. (2007). Stock Return Predictability: Is it There? The Review of *Financial Studies,* 20(3), 651-707. https://doi.org/10.1093/rfs/hhl021
- Gertler, M., & Hubbard, R. G. (1993). Corporate Financial Policy, Taxation, and Macroeconomic Risk. *The RAND Journal of Economics*, *24*(2), 286–303. https://doi.org/10.2307/2555763
- Bernanke, B., Gertler, M., & Gilchrist, S. (1996). The Financial Accelerator and the Flight to Quality. *The Review of Economics and Statistics*, *78*(1), 1–15. https://doi.org/10.2307/2109844
- Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.
- Keskar, N. S., Mudigere, D., Nocedal, J., Smelyanskiy, M., & Tang, P. T. P. (2017). On Large-Batch Training for Deep Learning: Generalization Gap and Sharp Minima. *arXiv preprint arXiv:1609.04836*. https://doi.org/10.48550/arXiv.1609.04836
- Shiller, R. J. (2000). Irrational exuberance. [Princeton University Press 2000, Broadway Books 2001, 2nd ed., 2005].
- Clark, T., & West, K. (2007). Approximately normal tests for equal predictive accuracy in nested models. *Journal of Econometrics, 138*(1), 291–311.
- Chen, L., Da, Z., & Priestley, R. (2012). Dividend Smoothing and Predictability. *Management Science*, *58*(10), 1834–1853. http://www.jstor.org/stable/41686885
- Zhang, X., Liang, X., Zhiyuli, A., Zhang, S., Xu, R., & Wu, B. (2019). AT-LSTM: An Attention-based LSTM Model for Financial Time Series Prediction. *IOP Conference Series: Materials Science and Engineering, 569*(5), 052037. doi:10.1088/1757-899X/569/5/052037
- Lettau, M., & Van Nieuwerburgh, S. (2008). Reconciling the Return Predictability Evidence. *The Review of Financial Studies*, *21*(4), 1607–1652. http://www.jstor.org/stable/40056863
- Lettau, M., & Ludvigson, S. C. (2005). Expected returns and expected dividend growth. *Journal of Financial Economics, 76*(3), 583-626. https://doi.org/10.1016/j.jfineco.2004.05.008
- van Binsbergen, J. H., & Koijen, R. S. J. (2010). Predictive regressions: A present-value approach. The Journal of Finance, 65(4), 1439–1471. http://www.jstor.org/stable/40864916
- Campbell, J. Y., & Shiller, R. J. (1988). The Dividend-Price Ratio and Expectations of Future Dividends and Discount Factors. *The Review of Financial Studies*, *1*(3), 195–228. http://www.jstor.org/stable/2961997
- Cochrane, J. H. (2008). The Dog That Did Not Bark: A Defense of Return Predictability. *Review of Financial Studies, 21*(4), 1533-1575.
- Fama, E. F., & French, K. R. (1988). Dividend yields and expected stock returns. *Journal of Financial Economics, 22*, 3-27.
- Lewellen, J. W. (2004). Predicting returns with financial ratios. *Journal of Financial Economics, 74*, 209-235.
- Goyal, A., & Welch, I. (2008). A comprehensive look at the empirical performance of equity premium prediction. *Review of Financial Studies, 21*, 1455-1508.

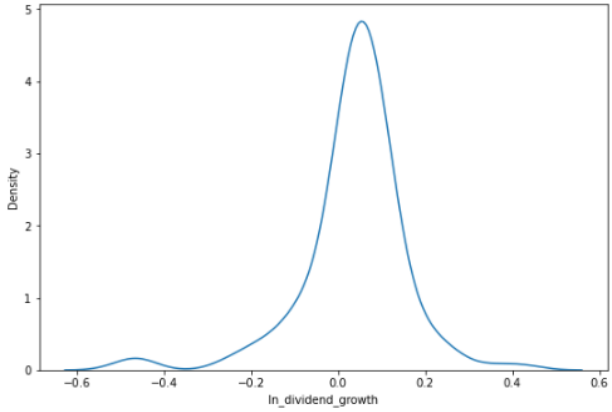
- Stambaugh, R. F. (1999). Predictive regressions. *Journal of Financial Economics, 54*(3), 375-421.
- Golez, B. (2014). Expected Returns and Dividend Growth Rates Implied by Derivative Markets. *The Review of Financial Studies, 27*(3), 790-822. doi:10.1093/rfs/hht131
- Lacerda, F., & Santa-Clara, P. (2010). Forecasting Dividend Growth to Better Predict Returns.
- Zhu, M., Chen, R., Du, K., & Wang, Y.-G. (2018). Dividend growth and equity premium predictability. *International Review of Economics & Finance, 56*(C), 125-137.
- Rangvid, J., Schmeling, M., & Schrimpf, A. (2014). Dividend Predictability Around the World. *The Journal of Financial and Quantitative Analysis*, *49*(5/6), 1255–1277. http://www.jstor.org/stable/43862856
- Chen, L. (2009). On the reversal of return and dividend growth predictability: A tale of two periods. *Journal of Financial Economics, 92*(1), 128-151. doi:10.1016/j.jfineco.2008.04.004
- McCulloch, W. S., & Pitts, W. (1943). A logical calculus of the ideas immanent in nervous activity. *Bulletin of Mathematical Biophysics, 5*, 115-133. https://doi.org/10.1007/BF02478259
- Rosenblatt, F. (1958). The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review, 65*(6), 386–408. https://doi.org/10.1037/h0042519
- Rumelhart, D., Hinton, G., & Williams, R. (1986). Learning representations by back-propagating errors. *Nature, 323*, 533-536. https://doi.org/10.1038/323533a0
- Hinton, G., Osindero, S., & Teh, Y.-W. (2006). A Fast Learning Algorithm for Deep Belief Nets. *Neural Computation, 18*(7), 1527-1554. doi:10.1162/neco.2006.18.7.1527
- Krizhevsky, A., Sutskever, I., & Hinton, G. (2012). ImageNet Classification with Deep Convolutional Neural Networks. *Neural Information Processing Systems, 25*. doi:10.1145/3065386
- Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... & Polosukhin, I. (2017). Attention is all you need. In *Advances in Neural Information Processing Systems* (Vol. 30, pp. 5998-6008).
- Menzly, L., Santos, T., & Veronesi, P. (2004). Understanding Predictability. *Journal of Political Economy*, *112*(1), 1–47. https://doi.org/10.1086/379934
- Bansal, R., & Yaron, A. (2004). Risks for the Long Run: A Potential Resolution of Asset Pricing Puzzles. *The Journal of Finance*, *59*(4), 1481–1509. http://www.jstor.org/stable/3694869
- Gabaix, X., Krishnamurthy, A., & Vigneron, O. (2007). Limits of Arbitrage: Theory and Evidence from the Mortgage-Backed Securities Market. *Journal of Finance, 62*(2), 557-595.
- Cochrane, J. H. (1992). Explaining the variance of price-dividend ratios. *Review of Financial Studies, 5*(2), 243–280.
- Cochrane, J. H. (2001). *Asset Pricing*. Princeton University Press, Princeton.
- Ang, A., & Liu, J. (2007, January). Risk, Return and Dividends. *NBER Working Paper No. w12843*. Retrieved from SSRN: https://ssrn.com/abstract=958495

- Campbell, J., & Shiller, R. (1998). Valuation ratios and the long-run stock market outlook: An update. Working Paper, Harvard University.
- Cox, J. C., Ingersoll, J. E., & Ross, S. A. (1985). A Theory of the Term Structure of Interest Rates. Econometrica, 53(2), 385–407. https://doi.org/10.2307/1911242
- Victor Zhou. (2020, February 9). Neural Networks from Scratch. Victor Zhou. https://victorzhou.com/series/neural-networks-from-scratch/
- Lambert, J. (2014). Stacked RNNs for Encoder-Decoder Networks: Accurate Machine Understanding of Images.
- dProgrammer lopez. (2019, April 6). RNN, LSTM & GRU: Recurrent Neural Network (RNN), Long-Short Term Memory (LSTM) & Gated Recurrent Unit (GRU).
- Nelson, C. R., & Kim, M. J. (1993). Predictable Stock Returns: The Role of Small Sample Bias. The Journal of Finance, 48(2), 641–661. https://doi.org/10.2307/2328916
- Goyal, A., & Welch, I. (2003). Predicting the Equity Premium with Dividend Ratios. Management Science, 49(5), 639–654. http://www.jstor.org/stable/4133989
- Engsted, T., & Pedersen, T. Q. (2010). The dividend–price ratio does predict dividend growth: International evidence. Journal of Empirical Finance, 17(4), 585-605. https://doi.org/10.1016/j.jempfin.2010.01.003

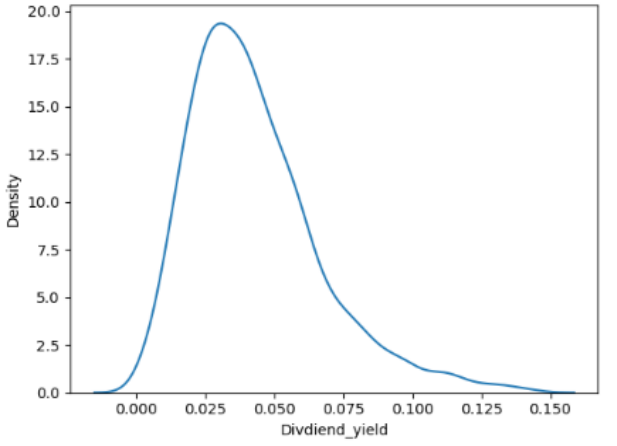
Appendix

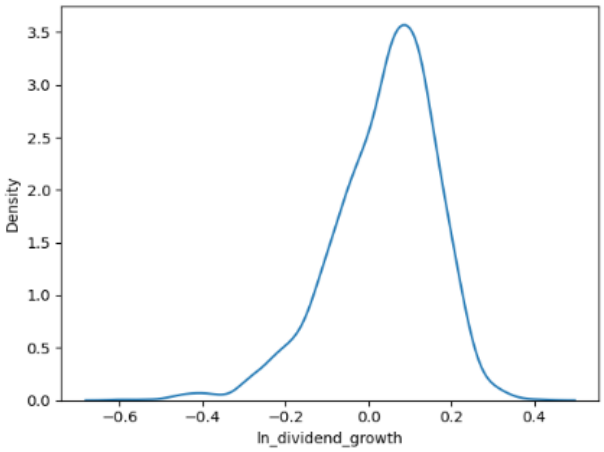
Appendix 1 – Kernel Density Plots of the historical data



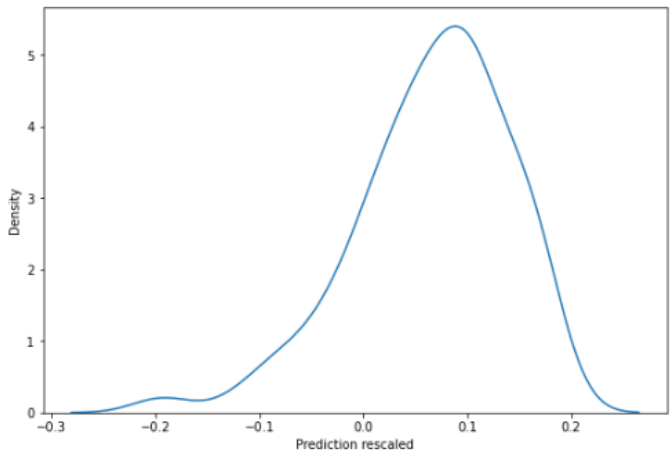


Appendix 2 - Kernel Density Plots of the artificial data





Appendix 3 – Kernel Density Plot of the predicted dividend growth rate



Appendix 4 – Code snippet

```
# Regression code with the necessary libraries

import pandas as pd
import statsmodels.formula.api as smf


# Perform a linear regression of ln dividend growth on ln dividend yield with White standard errors
model = smf.ols('ln_dividend_growth ~ np.exp(shifted_ln_dividend_yield)', data=df["12-1872":"12-2022"]).fit(cov_type='HC0')

# Print the summary of the regression results
print(model.summary())

# get the variance of the residuals
residual_var = results.scale

# print the summary statistics of the regression
print(results.summary())
print('Residual Variance:', residual_var)
```

Code snippet 1:Regressing the dividend growth rate on the lagged dividend yield

```
import numpy as np
import matplotlib.pyplot as plt

# Define CIR parameters
mu = 0.0429   # Long-term mean
sigma = 0.046037   # Volatility
kappa = 0.09061   # Mean-reversion speed
theta = 0.040318   # Mean-reversion level

#Define log dividend growth parameters
alpha=0.1729
beta=-3.1611
b=0.4974

# Define time parameters
T = 2000  # Time horizon
N = 2000  # Number of time steps
dt = T/N   # Time step
```

```
# Initialize CIR process
x0 = 0.042    # Initial value
g0 = -0.2307   # Initial value

x = np.zeros(N+1)
G = np.zeros(N+1)

x[0] = x0
G[0]= g0


# Simulate CIR process
for i in range(1, N+1):
    dBx = np.sqrt(dt)*np.random.normal()
    dBd = np.sqrt(dt)*np.random.normal()
    x[i] = x[i-1] + kappa*(theta - x[i-1])*dt + sigma*np.sqrt(x[i-1]) * dBx
    G[i] = (alpha + beta * x[i-1])*dt + b * np.sqrt(x[i-1]) * dBd
```

Code snippet 2: Generating artificial data for the dividend growth and dividend yield

```
# Normalize the data

# Import the MinMaxScaler object
from sklearn.preprocessing import MinMaxScaler

# Fit the training data
scaler = MinMaxScaler()
scaler.fit(train)

# Apply the transform method
scaled_train = scaler.transform(train)
scaled_validation = scaler.transform(validation)
scaled_test = scaler.transform(test)
```

Code snippet 3: Normalizing the data

```
# Defining my neural network structure

from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense,LSTM, Dropout
from tensorflow.keras.optimizers import Adam

# define model
model = Sequential()

# LSTM RNN structure
model.add(LSTM(100,return_sequences=True,recurrent_regularizer=None,input_shape=(length,scaled_train.shape[1])))

model.add(LSTM(50,return_sequences=True,input_shape=(None,100) ,recurrent_regularizer=None))

model.add(LSTM(30,return_sequences=True,input_shape=(None,50) ,recurrent_regularizer=None))

model.add(LSTM(10,input_shape=(None,30), recurrent_regularizer=None))


# Final Prediction (one neuron per feature)
model.add(Dense(1))
```

Code snippet 4: Defining the LSTM RNN structure

```
# Compile the model
model.compile(optimizer='Adam', loss='mse')

# Callbacks for early stopping
from tensorflow.keras.callbacks import EarlyStopping
early_stop = EarlyStopping(monitor='val_loss',patience=150)
validation_generator = TimeseriesGenerator(scaled_validation,scaled_validation[:,1],
                                           length=length, batch_size=batch_size)
```

Code snippet 5: Model compilation and definition of the 'early_stop' function

```
# Learning phase

model.fit(generator,epochs=150,
                  validation_data=validation_generator,
                 callbacks=[early_stop])
```

Code snippet 6: Initialization of the training phase

```
# Predicting the future dividend growth rate

n_features_ = scaled_ds.shape[1]
test_predictions_ = []

first_eval_batch = scaled_ds[:length]
current_batch = first_eval_batch.reshape((1, length, n_features))

for i in range(len(scaled_ds)):

    if i+10 < 151:
        # get prediction 1 time stamp ahead ([0] is for grabbing just the number instead of [array])
        current_pred = model_final.predict(current_batch)[0]

        # store prediction
        test_predictions_.append(current_pred)

        # update batch to now include prediction and drop first value
        current_batch = np.append(current_batch[:,1:,:],scaled_ds[i+10].reshape((1,1,n_features)),axis=1)
```

Code snippet 7: Predicting the dividend growth rate

```
# HAC standard errors

import numpy as np
import pandas as pd
import statsmodels.api as sm


# Define your dependent and independent variables
y = df['ft']["1945":"2022"]
X = sm.add_constant(df['cste']["1945":"2022"])  # Adding a constant term

# Fit the regression model
model = sm.OLS(y, X)
results = model.fit()

# Calculate Newey-West standard errors with one-sided (upper tail) test
cov_type = 'HAC'
cov_kwds = {'maxlags': 35}  # You can adjust the maxlags as needed
cov = results.get_robustcov_results(cov_type=cov_type, **cov_kwds).cov_params()

# coefficient estimates and standard errors
coef = results.params
std_err = np.sqrt(np.diag(cov))

# Calculate t-statistics
t_stat = coef / std_err

# Calculate p-values for the upper tail test
from scipy.stats import t
df_resid = results.df_resid
p_values = 1 - t.cdf(np.abs(t_stat), df=df_resid)

# Print the results
result_summary = pd.DataFrame({'Coefficient': coef, 'Standard Error': std_err, 'T-statistic': t_stat,
                               'One-sided p-value': p_values})
print(result_summary)
```

Code snippet 8: Newey-West standard error calculation