universität wien

MASTERARBEIT | MASTER'S THESIS

Titel | Title

Exploring Nonlinearities in Stock Performance

verfasst von | submitted by

Kirill Tsatkhlanov BSc

angestrebter akademischer Grad | in partial fulfilment of the requirements for the degree of

Master of Science (MSc)

Wien | Vienna, 2025

Studienkennzahl lt. Studienblatt | Degree programme code as it appears on the student record sheet:	UA 066 974
Studienrichtung lt. Studienblatt | Degree programme as it appears on the student record sheet:	Masterstudium Banking and Finance
Betreut von | Supervisor:	Univ.-Prof. Mag. Günter Strobl PhD

ACKNOWLEDGEMENT

I hereby declare that this Applied Research Project, submitted to the University of Vienna in partial fulfillment of the requirements for the Master of Science degree, is my own original work, except where otherwise indicated and properly referenced. I acknowledge that AI-based tools were used to assist with text creation and refinement. All research design, empirical analysis, interpretation, and conclusions are entirely my own. This project has not been submitted, either in whole or in part, for any other academic degree or qualification.

Signed: TSATKHLANOV KIRILL

Student number: 01624965

Date: 18.09.2025

ABSTRACT

The goal of this thesis is to investigate if machine learning algorithms can provide better predictions of stock returns by capturing nonlinearities that might be overlooked by the traditional asset pricing models. The main emphasis of this study lies on the comparison of traditional linear regression implementations of the Capital Asset Pricing Model and the Fama-French Five-Factor model with nonlinear machine learning techniques, including Random Forests and extreme Gradient Boosting. Using the Kenneth R. French library, which offers monthly return data from twenty-five U.S. portfolios sorted by size and book-to-market characteristics, both types of models are evaluated using a five-year rolling window and Root Mean Squared Error (RMSE) as the primary metric. A Gibbons-Ross-Shanken test (GRS) and Diebold-Mariano test (DM) are also applied. The results show that in the chosen setting the machine learning models do not regularly outperform linear benchmarks. This finding suggests that while machine learning has the potential to model complex relationships, the specific algorithms applied here, together with the chosen parameters, may be insufficient. These results tell us the importance of model and parameter selection together with the correct choice of advanced or specialized machine learning methods to realize significant improvements in predictive accuracy within the asset pricing context.

ABSTRACT - DEUTSCH

Das Ziel dieser Masterarbeit ist, zu untersuchen, ob Algorithmen des maschinellen Lernens im Vergleich zu traditionellen finanzwirtschaftlichen Modellen genauere Schätzungen der Aktienrenditen liefern können, da lineare Modelle potenzielle Nichtlinearitäten möglicherweise übersehen. Der Schwerpunkt liegt auf dem Vergleich zwischen den klassischen Regressionsimplementierungen des Capital Asset Pricing Models (CAPM) und des Fama-French-Fünf-Faktoren-Modells einerseits sowie nichtlinearen Methoden des maschinellen Lernens wie „Random Forests“ und „extreme Gradient Boosting“ andererseits. Die Analyse basiert auf Daten aus der Kenneth-R.-French-Datenbank, die monatliche Renditen von fünfundzwanzig US-Portfolios enthält. Die Portfolien sind nach Unternehmensgröße und Buch-zu-Markt-Verhältnis sortiert. Beide Modelltypen werden anhand eines fünfjährigen Rollfensters evaluiert, wobei die Quadratwurzel des mittleren quadratischen Fehlers (Root Mean Squared Error) als primäre Bewertungskennzahl dient. Zusätzlich werden Gibbons-Ross-Shanken-Test (GRS) und Diebold-Mariano-Test (DM) angewendet. Die Ergebnisse zeigen, dass die eingesetzten Modelle des maschinellen Lernens in der gewählten Konfiguration die linearen Vergleichsmodelle nicht regelmäßig übertreffen. Das zeigt, dass ausgewählte maschinelle Methoden zwar grundsätzlich das Potenzial besitzen, komplexe Zusammenhänge abzubilden, möglicherweise aber nicht ausreichend sind. Die Ergebnisse erhöhen die Bedeutung einer sorgfältigen Modell- und Parameterauswahl. Sie deuten auch darauf hin, dass fortgeschrittenere oder spezialisierte Methoden des maschinellen Lernens erforderlich sein könnten, um signifikante Verbesserungen der Prognosegenauigkeit im Kontext der finanzwirtschaftlichen Modelle zu ermöglichen.

TABLE OF CONTENT

LIST OF TABLES

LIST OF FIGURES

1 INTRODUCTION

The existence of financial markets has been the fundamental pillar of prosperity in the modern global economy. Their evolution has played a crucial role in promoting economic development by efficiently allocating capital, hedging risk, and providing both short-term and long-term liquidity to those in need, as the ease of trading assets and securities has reached unbelievable heights. For instance, the total market capitalization of U.S. public companies has reached 62 trillion dollars, which is more than twice the size of the country's GDP (World Bank, 2024).

As the development of the financial markets continues, so does the understanding of how the assets and goods should be priced together with the assessment of their risk. The increasing complexity of financial markets has increased the demand for refined models that could explain the asset returns and help to predict future performance. To minimize this issue, the Capital Asset Pricing Model (CAPM) was introduced by Sharpe (1964), which firstly introduced the systematic approach by relating an asset's expected return to its exposure to market risk, as measured by the beta coefficient. With time ongoing, it had become evident that the CAPM had some limitations that don't fully allow uncovering the variation in stock return across different market segments. This has led to the expansion of the asset pricing theory, with the Fama-French models enhancing the CAPM by introducing additional factors, such as size, values, profitability and investment (Fama & French, 2015). These factors gave investors a more comprehensive understanding of how risk and return are related to each other in complex financial markets.

While these traditional models have significantly advanced the field, they are fundamentally linear in nature and may not fully capture the dynamic relationships in the financial data. Ongoing

developments in machine learning (ML) techniques have opened new opportunities for stock prediction in financial markets. Together with the advanced computational capabilities and the increasing availability of large datasets, the machine learning algorithms could overcome the limitation caused by the classical asset pricing theories. Unlike linear models, the machine learning algorithms can explain complex, nonlinear interactions between variables, potentially uncovering hidden patterns and providing more accurate predictions of stock returns. (Gu, Kelly, & Xiu, 2020; Krauss, Do, & Huck, 2017).

Although an increasing number of studies apply machine learning methods for return prediction, many focus on predictive accuracy using large sets of technical or unstructured features, often without incorporating economically grounded factor models. This leaves a gap in understanding whether machine learning algorithms can still uncover nonlinear relationships among theoretically sound risk factors and better explain stock performance compared to traditional methods.

The main goal of this research is to investigate if the machine learning algorithms can outperform the traditional asset pricing models – namely the Capital Asset Pricing Model (CAPM) and the Fama-French Five-Factor Model (FF5) - in predicting stock returns. To ensure a fair and consistent comparison, the study uses the same set of explanatory variables from the CAPM and FF5 models, such as market excess return, size, value, profitability, and investment factors, as inputs to the machine learning models. The central hypothesis is as follows:

"Machine learning algorithms will provide a lower root mean squared error (RMSE) in predicting stock returns compared to traditional linear regression models when applied to CAPM beta and Fama-French factor variables."

The analysis is conducted using CRSP portfolio data obtained from the widely accepted Kenneth R. French data library, which contains twenty-five portfolios formed on size and book-to-market ratios. The machine learning models applied include extreme Gradient Boosting (XGBoost), Random Forests, and k-Nearest Neighbors (kNN), which will be benchmarked against standard linear regression models fitted to the same data. If the machine learning models demonstrate superior predictive performance—measured primarily by a lower Root Mean Squared Error (RMSE)—the findings may contribute to the development of more flexible and robust risk modeling approaches and offer practical benefits for institutional investors, asset managers, and financial analysts.

In addition to its academic contribution, this thesis has practical relevance for the financial industry. The ability to accurately predict stock returns is critical for portfolio optimization, risk modeling, and investment strategy. By testing whether machine learning algorithms can outperform traditional models using the same factor inputs, this research offers potential insights for asset managers, institutional investors, and fintech developers seeking to enhance their predictive capabilities in increasingly complex and data-driven markets. However, the findings indicate that, within the examined setting, machine learning models were not able to consistently outperform traditional linear benchmarks. These results, together with potential explanations and implications, will be further investigated later on.

In the following sections, a literature review and a more detailed explanation of the used machine learning algorithms together with the chosen methodology will be provided. To give a better understanding, illustrative examples will be included to demonstrate how these algorithms work in practice. In the end the results of the analysis will be presented and discussed, alongside the limitations encountered during the writing of this master's thesis.

2 LITERATURE REVIEW

In this section, a more detailed look at the literature will be highlighted. The main strategy was to emphasize established and highly cited papers to ensure academic robustness and avoid relying on methods or findings that are not yet broadly validated. The chapter starts with an overview of classical asset pricing models, such as the Capital Asset Pricing Model (CAPM) and the Fama-French Five-Factor model (FF5). Further, it is followed by an analysis of the existence and implications of nonlinearities in the financial data. Finally, the section will examine the increasing significance of machine learning algorithms in modern financial analysis and asset pricing.

2.1 Classical Asset Pricing Models

The Capital Asset Pricing Model (CAPM) and the Fama-French Five Factor (FF5) model are the foundational asset pricing models that have played a crucial role in financial economics for many years. The CAPM itself, founded by Sharpe (1964), proposes a linear relationship between the expected return of an asset and its systematic risk measured by the beta. In simple words, the model indicates that the investors should be compensated for both the time value of money and the risk associated with a certain asset. (Fama & French, 2004). The Capital Asset Pricing Model has been widely used in both academic and practical applications, but it still has been criticized for the ideas the model is based on, as the assumptions of market efficiency and investors' rational behavior are often not fulfilled (Fama & French, 2004). In addition, several studies and empirical tests have shown that the CAPM may not fully explain all market phenomena, especially during periods of high market volatility or major shifts in the macroeconomic environment. (Jagannathan & Wang, 1996)

In contrast, the Fama-French Five-Factor (FF5) model, introduced by Fama and French (2015), expands the CAPM, which allows the model to better explain the cross-sectional difference in the stock returns due to its new multifactor structure. According to empirical research, the FF5 greatly reduces the pricing errors and does a better job in explaining the performance of portfolios due to the implementation of new additional factors such as size (SMB), value (HML), profitability (RMW) and investment (CMA) (Fama & French, 2016). Additionally, research like Barillas and Shanken (2018) shows that the CAPM statistically fully fits inside the FF5 model and fails to pass formal model comparison tests.

Despite these improvements, the Fama-French Five-Factor model still has some limitations. One of the major drawbacks, pointed out by the creators themselves, is that the value factor (HML) is mostly useless when investments and profits are taken into account (Fama & French, 2015). This issue raises questions about the multicollinearity and interpretability of the model. Furthermore, the FF5 also doesn't take into account well-known return patterns like momentum, an anomaly that has been detected in the markets around the world (Fama & French, 2016), which weakens the explanatory power of the model. It has also been argued that Fama-French Five-Factor model, while empirically effective, is driven more by data mining than economic theory (Hou, Xue & Zhang, 2015). Also, the performance of FF5 is inconsistent in different areas, especially when it comes to the developing countries (Cakici, Fabozzi & Tan, 2013). In addition to that, another working paper has revealed that both the CAPM and Fama-French model could fail to deliver solid results in the estimation of individual stock returns (Bartholdy, 2004).

Besides all these model-specific critiques towards CAPM and FF5, a more structural limitation underlies beneath – their common reliance on the linear assumptions (Gu, Kelly, & Xiu, 2020). Nevertheless, the CAPM and FF5 models are still important tools in the field of asset

pricing theory, especially when they are combined or changed with more recent empirical findings and models. Their basic ideas are relevant today in both academic and real-world investment strategies.

2.2 Nonlinearities in Financial Data

In the past decades the existence of nonlinear patterns in financial markets has received a lot of attention in the scientific fields. As has been mentioned before, traditional asset pricing models like CAPM (Sharpe, 1964) and the Fama–French multifactor models (Fama & French, 1993; 2015) are based on the idea that risk exposures and expected returns are linearly related. However, more evidence shows that the relationship between risk factors and asset returns is dynamic and state-dependent, indicating nonlinear characteristics. In simple words, this could imply that the risk premium, which is linked to the market factors, may change in different market conditions or regimes what directly challenges the underlying linear assumptions of the CAPM and FF5. For instance, research by Adrian, Crump and Vogt (2019) demonstrates a nonlinear trade-off between risk and return due to investors' "flight-to-safety" behavior under stress, what leads to state-dependent pricing of risk that cannot be fully captured by linear factor models. The existence of such nonlinearities leads to a conclusion that the response of an asset's returns to changes in the market environment may be more complicated than it is stated by the traditional models (Adrian, Crump, & Vogt, 2019). In addition to that, Daniel and Titman (1997) also question the economic interpretability of factor-based models altogether, as from their point of view, these models may not capture the real sources of systematic risk, as they only correlate with certain observed market anomalies. Adrian, Etula, and Muir (2014) go even further with their critique by adding intermediary-based risk as a more logical explanation for cross-sectional returns, which is something that standard linear frameworks don't take into account.

Further support for the presence of nonlinearities in the financial data comes from Dittmar (2002), who created a nonlinear pricing kernel that provides a better fit for observed asset returns than the standard linear stochastic discount factors used in CAPM and arbitrage pricing theory (APT). In addition to that, Ang and Bekaert (2002) use regime-switching models to show that the mean and variance of asset values change between different economic regimes. This indicates that investor behavior is not linear and is state-dependent. More empirical studies back up these conclusions. For instance, Ghysels, Santa-Clara, and Valkanov (2005) have implemented a mixed data sampling (MIDAS) approach to analyze the trade-off between risk and return, which resulted in strong evidence of a nonlinear relationship that challenges the core linear assumption embedded in models like the CAPM.

In order to detect these nonlinear patterns, researchers have tried several techniques, such as chaos theory, threshold autoregressive models, and nonlinear cointegration frameworks. With these methods, researchers were able to find complex relationships and changing structures that linear models usually miss. For example, Hsieh (1991) was one of the first researchers that showed the chaotic and nonlinear patterns in U.S. stock returns. Further, Lim, Brooks and Hinich (2008) have demonstrated in their work the evidence of nonlinear dependence in Asian emerging stock markets, challenging the weak-form efficiency hypothesis. Also, methods such as Hamilton's (1989) Markov-switching models and nonlinear cointegration methods (Balke & Fomby, 1997) have been used to explore structural breaks and complicated long-term relationships in asset pricing data.

Taken together, these findings provide compelling evidence that the core linearity assumption in traditional asset pricing models is a significant limitation. The real-world behavior of asset returns often reflects complex, dynamic, and nonlinear relationships, which are better

captured through models that allow for interactions, thresholds, regime changes, and other nonlinear features. As such, there is a growing consensus that machine learning methods, regime-switching models, and nonlinear econometric techniques offer a promising direction for improving the understanding of financial markets.

2.3 Machine Learning

With each year the machine learning (ML) algorithms are becoming more important, as they are slowly becoming a vital part in the modern stock prediction due to their ability to overcome linear limitations and uncover complex relationships that the traditional asset pricing models often fail to do. Advanced machine learning techniques, such as deep neural networks (DNNs), long short-term memory networks (LSTMs), and recurrent neural networks (RNNs) can model time dependencies and high-dimensional interactions in financial data (Fischer & Krauss, 2018; Nelson, Pereira, & de Oliveira, 2017). These methods are especially good at recording complex market behaviors and making predictions more accurate where the economic environment is highly volatile.

However, while the advanced neural networks are very flexible, they are also not without flaws, as interpretability of results, overfitting risk and high computational demand are usual faced issues. On the other hand, ensemble learning methods like Gradient Boosting Machines (GBM) and Random Forests (RF) often deliver a more balanced result of performance and interpretability. This makes them very appealing to financial analysis (Chen & Guestrin, 2016; Gu, Kelly, & Xiu, 2020). These models are not as sophisticated as deep learning methods, but still, they can do better than linear models due to their ability to capture interactions among variables like momentum, volatility and fundamental indicators (Gu, Kelly, & Xiu, 2020).

Regarding the ensemble machine learning algorithms, research by Krauss, Do, and Huck (2017) reveals that Random Forests and Boosting methods consistently outperform common benchmarks such as logistic regression and even classic factor-based models when it comes to stock selection tasks. Furthermore, Bryzgalova, Pelger, and Zhu (2023) demonstrate that factor-model variables - such as size or book-to-market ratios - still remain valuable factors when incorporated into machine learning algorithms, highlighting that they continue to play an important role even in modern ML frameworks.

In general, most of the research that has been done on the implementation of machine learning algorithms for forecasting stock returns has been focused on very complicated and resource-intensive models, like deep neural networks. These studies often also include new or proprietary features that go much beyond the scope of traditional financial factors. However, there are not many studies that investigate whether less complex but still powerful models, such as Random Forests or Extreme Gradient Boosting (XGBoost), can deliver better predictions using the same standard financial inputs as the Capital Asset Pricing Model (CAPM) and the Fama-French Five-Factor (FF5) model. This opens a room for discussion and research opportunity – to find out if the better performance is due to the nonlinear modeling itself or the inclusion of new variables. Addressing this gap would give a better insight into how machine learning algorithms could improve the accuracy of predictions in a more interpretable and intuitive way.

3 METHODOLOGY

The main goal of this study is to get better stock prediction by implementing the Fama-French Five-Factor model and the Capital Asset Pricing Model parameters into machine learning algorithms. The process of gathering data and the adjustments used to get it ready for analysis are

covered in the first part. Subsequently, a concise explanation of the parameters used in the CAPM and Fama-French Five-Factor model is provided. Finally, the selected methodologies, with a focus on the machine learning algorithms, are explained in detail to determine their suitability for capturing nonlinearities in stock performance. In addition, validation methodologies are also added to this chapter.

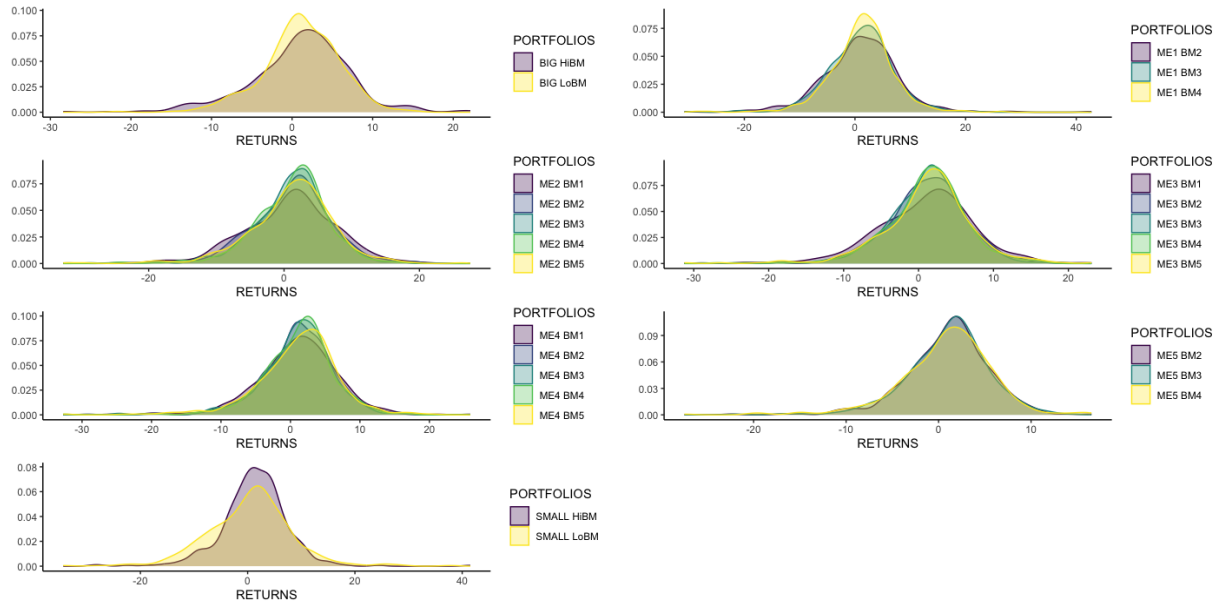
3.1 Data Preparation

To capture the nonlinear behavior between stock returns and desired parameters, the data was extracted from Kenneth R. French's website, which contains twenty-five portfolios formed on size and book-to-market ratios. The dataset itself is sourced from the Center for Research in Security Prices (CRSP) and offers high-quality historical financial data listed on major U.S. exchanges for scholarly research and real-world data analysis, making it an accurate and well-known source in the finance industry. For this study, the data provides 40 years of monthly observations beginning from 1984 to 2024.

The twenty-five portfolios are constructed based on two fundamental factors: market capitalization and book-to-market ratio. Stocks are categorized into five quintiles based on their market capitalization, and within each size quintile, stocks are further divided into five groups based on their book-to-market ratios. This two-dimensional classification results in a total of twenty-five portfolios, representing different combinations of size and value characteristics. All portfolio returns and factor values represent monthly relative percentage changes to the previous month. The distributions of portfolio returns are presented in Figure 1. The same foundation of the CRSP dataset was used to derive the parameters relevant for the Fama-French Five-Factor model and the CAPM. The CAPM uses the market return and the risk-free rate to calculate the expected

return based on the portfolio's beta, while the Fama-French model extends the CAPM by incorporating additional factors such as size (SMB), value (HML), profitability (RMW), and investment (CMA). A more detailed view of the classical asset pricing model will be outlined in the following chapter.

Figure 1: Distribution of Portfolios



3.2 Asset Pricing Models

The chapter delves into the fundamental asset pricing models that form the backbone of modern financial theory: the Capital Asset Pricing Model (CAPM) and the Fama-French Five-Factor model (FF5). These models are pivotal in understanding the relationship between risk and return, as they provide a foundational framework in finance for the estimation of an asset's expected returns. Despite the emergence of more complex and data-driven approaches in recent years, CAPM and FF5 continue to be broadly applied in both academic research and practical investment analysis.

3.2.1 Capital Asset Pricing Model

The Capital Asset Pricing Model (CAPM) is one of the foundational concepts in modern finance that describes the relationship between the asset's expected return and risk relative to the market. Developed independently by William F. Sharpe (1964), John Lintner (1965), and Jan Mossin (1966) as an extension of Harry Markowitz's Modern Portfolio Theory (1952), CAPM provides an equation to estimate the expected return of an asset based on its systematic risk, measured by beta β. The CAPM is based on several key assumptions. It assumes that investors are rational and risk-averse, seeking to maximize the expected utility of their wealth. They are assumed to have identical expectations about asset returns, risks, and correlations. The model also assumes frictionless markets without taxes, transaction costs, or restrictions on short selling. Additionally, all investors can borrow and lend freely at a common risk-free rate, and all assets are infinitely divisible and publicly traded. Of course, these assumptions are strong and expectedly often violated in real-world markets, but the CAPM still remains an important theoretical tool for analyzing equilibrium returns. The original formula is displayed below:

$$E(R_i) = R_f + \beta_i[E(R_m) - R_f]$$

where $E(R_i)$ is the expected return of the asset, R_f is the risk-free rate, β_i is the beta of the asset, representing its sensitivity to market movements, $E(R_m)$ is the expected return of the market and the term $E(R_m) - R_f$ represents the market risk premium.

One of CAPM's defining characteristics is its linearity, as it assumes a direct, proportional relationship between an asset's expected return and its systematic risk. The beta coefficient itself is a central component of the CAPM, as it quantifies the systematic risk of an asset — that is, the

portion of risk that cannot be diversified away. In practice, beta is calculated by regressing the asset's excess returns on the market's excess returns. Mathematically, beta corresponds to the covariance between the asset and market returns divided by the variance of the market returns.

$$\beta_i = \frac{Cov(R_i, R_m)}{Var(R_m)}$$

The CAPM regression is thus a powerful empirical tool because it allows the estimation of beta from historical data and the decomposition of an asset's return into systematic and idiosyncratic components. However, the model's empirical performance has been challenged, especially due to the presence of persistent anomalies and nonlinearities in returns, leading to the development of multifactor models such as the Fama-French models. Despite these advancements, CAPM continues to serve as a foundational model in finance, valued for its simplicity, theoretical elegance, and practical relevance in asset pricing, portfolio construction, and performance evaluation.

3.2.2 Fama-French Five Factor Model

The Fama-French Five-Factor Model (FF5), introduced by Fama and French (2015), is a significant extension of their earlier Three-Factor Model (1993) and represents one of the most influential empirical asset pricing models in modern finance. The model was developed to address persistent anomalies and pricing errors that the original Capital Asset Pricing Model and the Three-Factor model failed to fully explain. By incorporating additional factors related to profitability and investment, the FF5 model aims to provide a more accurate description of the cross-section of average stock returns.

The FF5 model assumes frictionless markets and rational investors, but it is more empirically driven than a strict theoretical assumption. Its development was motivated by empirical evidence that firm characteristics such as size, book-to-market equity, operating profitability, and investment patterns have systematic impacts on expected returns. Besides the market excess return, the FF5 model includes the size factor (SMB: returns of small minus large firms), the value factor (HML: returns of high minus low book-to-market firms), the profitability factor (RMW: returns of firms with robust minus weak profitability), and the investment factor (CMA: returns of firms that invest conservatively minus those that invest aggressively). The model can be expressed as:

$$E(R) = R_f + \beta_m\left[E(R_m) - R_f\right] + \beta_s SMB + \beta_h HML + \beta_r RMW + \beta_c CMA$$

where $E(R_i)$ is the expected return of the asset, R_f is the risk-free rate and β_i is the beta of the asset. Each factor in the FF5 model is calculated using portfolio sorts, typically conducted on stocks listed on the NYSE, AMEX, and NASDAQ exchanges.

To construct these factors empirically, Fama and French follow a certain sorting procedure. Firms are sorted into two groups based on market capitalization and then further sorted into three groups based on other criteria like book-to-market, profitability, or investment, forming portfolios for each dimension. Factor returns are then calculated using long-short portfolio strategies that average differences in returns across these categories. The data used to compute these factors is publicly available through Kenneth R. French's data library, which is a standard source in academic research (Fama & French, 2015).

Similar to the Capital Asset Pricing Model (CAPM), the Fama-French model assumes rational investors and efficient markets, but in addition to that, the model provides a more

comprehensive view of the factors that affect the asset return. While the Fama-French model provides a better explanation for the returns for a wider range of assets, the model has a higher degree of complexity that requires additional estimation of factors, bigger data and sophisticated analysis that is often challenging to implement. Despite its improved explanatory and predictive power, some studies argue that FF5 does not fully explain anomalies like momentum, leading to further developments in modern financial theory. Nevertheless, FF5 remains a widely used model in asset pricing, risk management, and portfolio construction, offering a more detailed framework for understanding stock returns.

3.3 Machine Learning Algorithms

In recent years the implementation of machine learning (ML) algorithms in financial modeling has gained significant attention due to their ability to capture complex and nonlinear relationships within the data. Traditional linear models that were described in the chapter before are the cornerstone of today's financial analysis. Nevertheless, these models, as already mentioned, assume only linear relationships between the parameters and the assets' returns that undervalue the potential hidden patterns given in the financial data.

In order to address this limitation, this study uses machine learning techniques to find out if nonlinearities could be captured and used for better stock prediction. To be more precise, algorithms such as extreme Gradient Boosting, Random Forests and k-Nearest Neighbors will be used, as they are known for their robustness in handling nonlinear interactions and also for good predictive power in dealing with various data, including financial data. By integrating the parameters from CAPM and Fama-French into the mentioned machine learning algorithms, the scope would be to determine the extent of nonlinearities in the data and if the chosen machine

learning algorithms would be able to deliver more accurate predictions of stock returns compared to standard linear regression. This approach not only enhances the understanding of stock performance but also offers potential improvements in investment strategies by leveraging the strengths of machine learning in financial analysis.

3.3.1 Gradient Boosting Machines

The Gradient Boosting Machines (GBMs) are part of the ensemble learning algorithms that have proven their efficiency in the last years and have become an important tool in machine learning for both regression and classification tasks. The boosting itself was founded by Yoav Freund and Robert Schapire in 1997 by the introduction of the Adaptive Boosting (AdaBoost). This method was designed to improve the performance of weak learners, typically decision stumps, by sequentially focusing on the hardest-to-classify instances, thereby creating a strong composite model (Schapire, 1999). The concept of AdaBoost was further developed by Jerome Friedman (2001), who introduced a gradient-descent-based formulation of boosting methods, where boosting was displayed as an optimization technique with a certain loss function that made it possible to train weak learners one after the other to result later in a powerful prediction model. This work by Friedman (2001) has become the foundational pillar for the further development of Gradient Boosting Machines (GBMs).

The core principle of the GBMs is not difficult to understand. It creates a group of weak learners one by one, usually shallow decision trees. Each new tree is trained to predict the residual error of the previously joined ensemble of decision trees, which allows it to improve predictions over time. The algorithm starts with an initial prediction, often the average of the target variable in the regression tasks. After that, the GBMs go iteratively through three main steps: Firstly, it

computes the residuals (errors) between the actual values and the current model's prediction. Further, in dependence of these errors, a new decision tree is fitted to the model that consecutively updates the predictions. The additive manner of the algorithm can be described in the following equation, where at each iteration t a new tree f_k is added to predict the output and minimize the objection function that will be discussed later in this section. Each unique tree assigns an optimal weight w_i to each leaf, which updates the prediction after a single iteration.

$$\hat{y}_i^{(t)} = \hat{y}_i^{(t-1)} + f_k(x_i)$$

Nevertheless, the study will focus on extreme Gradient Boosting (XGBoost), which is a more advanced boosting algorithm introduced by Tianqi Chen in 2016. The XGBoost adds new features and improvements to the previously mentioned gradient boosting algorithm that increase its performance and efficiency overall. The novelty of the method is the inclusion of regularization techniques that prevent overfitting and improve generalization. In addition to that, the approach also uses parallel processing to speed up the training process, which makes it highly efficient for large datasets. The extreme Gradient Boosting also has some built-in features that allow it to deal with missing values, what makes it more reliable in real-world situations, and to prune the decision trees based on a minimum loss reduction threshold. All this together makes XGBoost a powerful tool for predictive modeling and analysis (Chen, 2016).

The main regularized objective in XGBoost consists mainly of a loss function l that measures the difference between the predicted and actual values, and a regularization term Ω that penalizes the complexity of the model to prevent overfitting. The direct equation is following:

$$\mathcal{L}(\phi) = \sum_i l(y_i, \hat{y}_i) + \sum_k \Omega(f_k)$$

where y_i is the actual value of the i-th instance, $\hat{y}_i$ is the predicted value of the i-th instance, f_k represents the k-th tree in the ensemble, K is the total number of trees and Ω is a regularization term that penalized the complexity of trees. In this research XGBoost for regression is utilized. The loss function should be a differentiable convex loss function that measures the difference between the prediction and the target value. The most used common loss function is displayed below. Both the loss function and the regularization term could be represented as follows:

$$l(y_i, \hat{y}_i) = \frac{1}{2}(y_i - \hat{y}_i)^2 \text{ and } \Omega(f) = \gamma T + \frac{1}{2}\lambda||w||^2$$

where T is the number of leaves in the tree, while γ and λ are the regularization parameters. Unfortunately, the equation cannot be optimized with traditional methods, as it includes functions as parameters. To overcome this obstacle, the model will be trained in an additive manner. The formula will be rewritten in the following way:

$$\mathcal{L}^{(t)} = \sum_{i=1}^{n} l\left(y_i, \hat{y}_i^{(t-1)} + f_t(x_i)\right) + \Omega(f_t)$$

The exact optimization of the regularized objective in extreme Gradient Boosting is often time consuming and requires a lot of computational resources. In this case, XGBoost simplifies the derivation by approximating the change in the objective function $\mathcal{L}$, by using a second order Taylor expansion.

$$\mathcal{L}^{(t)} \simeq \sum_{i=1}^{n} [l\left(y_i, \hat{y}_i^{(t-1)}\right) + g_i f_t(x_i) + \frac{1}{2} h_i f_t^2(x_i)] + \Omega(f_t)$$

After solving the second-order Taylor approximation, an optimal weight for each leaf is derived to minimize the regularized objective. The g_i and h_i are the first and second derivatives

(gradients) of the loss function with respect to the previous prediction. This process is repeated iteratively for multiple boosting rounds, with each tree aiming to further reduce the residual error until the model achieves its full minimizing potential. The corresponding optimal weight could be written as follows:

$$w_j^* = -\frac{\sum_{i \in I_j} g_i}{\sum_{i \in I_j} h_i + \lambda} \text{ or simply } w_j^* = \frac{Sum\ of\ Residuals}{Number\ of\ Residuals + \lambda}$$

To get the optimal weight for each leaf, it is firstly necessary to determine the correct splitting of the decision tree. The splitting is decided on the gain or score a potential split can deliver. Higher gains indicate more informative splits and thus higher-quality tree structure. Once the splits are chosen, the optimal weight for each leaf is calculated.

$$Gain(q) = \frac{1}{2}\sum_{j=1}^{T} \frac{(\sum_{i \in I_j} g_i)^2}{\sum_{i \in I_j} h_i + \lambda} - \gamma T \text{ or simply } Gain(q) = \frac{1}{2}\frac{Sum\ of\ Residuals^2}{Number\ of\ Residuals + \lambda} - \gamma T$$

For better understanding, the example in Table 1 is demonstrated. The table represents a very simplistic example of a data set that is used from the Kenneth R. French website. To keep the demonstration straightforward, no regularization parameters and coefficients are applied. The dataset contains excess portfolio and market returns for four consecutive months, meaning that the risk-free rate has already been subtracted. The moving average is calculated as the mean of the current and previous month's excess market returns. All values are expressed in percentages.

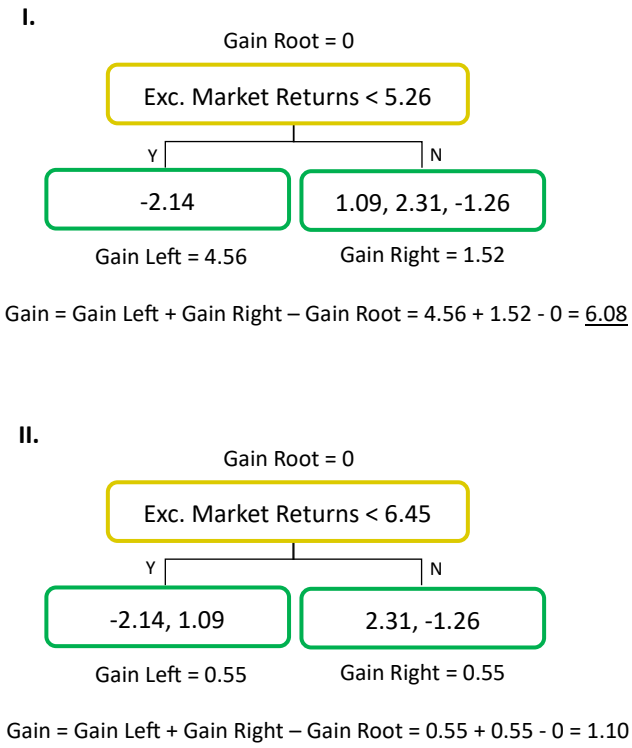
Table 1: Example XGBoost

NR.	DATE	EXCESS PORTFOLIO RETURNS	EXCESS MARKET RETURNS	RESIDUALS	MOVING AVERAGE
1	2024-05-01	8.23	4.48	-2.14	0
2	2024-06-01	11.45	6.03	1.09	5.26
3	2024-07-01	12.67	6.86	2.31	6.45
4	2024-08-01	9.11	7.99	-1.26	7.43

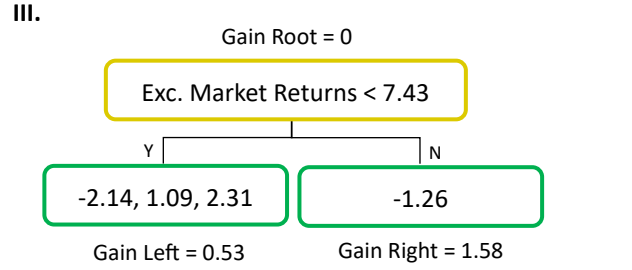
Initial Prediction	10.37

The extreme Gradient Boosting (XGBoost) algorithm begins by computing an initial prediction, which is the average of all excess portfolio returns in the table. This predicted value is then used to calculate the residuals for all observations. Residuals form the foundation for subsequent steps, as their correct separation or clustering directly impacts the final outcome. In the current approach, the mean (moving average) between pairs of residuals is calculated, which determines where the separation line should be constructed. The decision trees in Figure 2 illustrate the three possible scenarios. To determine which separation yields the best results, distinct XGBoost trees are generated and evaluated.

Figure 2: XGBoost Root Decision Trees



III.

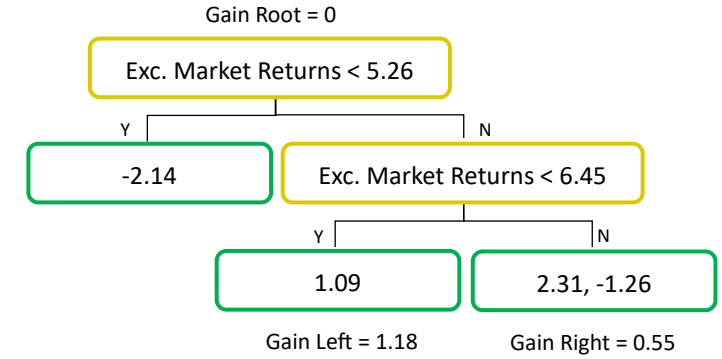


Gain = Gain Left + Gain Right – Gain Root = 0.53 + 1.58 - 0 = 2.10

For each leaf, a certain score or gain is calculated. The root leaf always has a score of zero if the initial prediction is set as the mean of all observations. Once this step is completed, the algorithm computes the gain for each unique tree. The tree with the highest gain is selected for further analysis. In this case, tree number one is chosen. It can be observed that the left leaf cannot be further divided, leaving the right leaf as the only option for further separation. The process continues by evaluating the remaining possible splits. Following the same procedure as before, the algorithm identifies the optimal tree with the highest gain. The root gain here equals the gain of the leaf that is being further split.

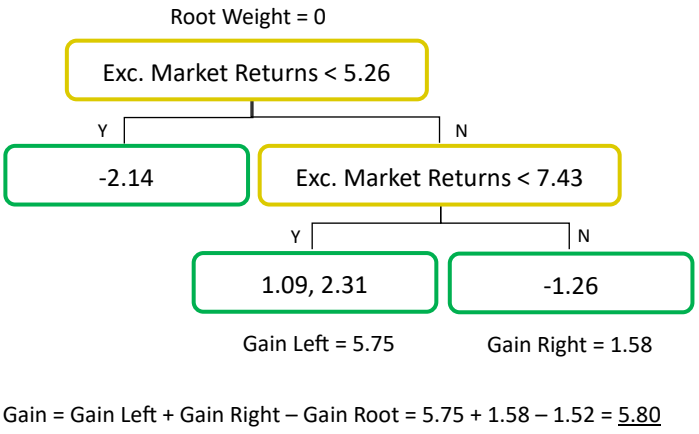
Figure 3: XGBoost Final Decision Trees

I.



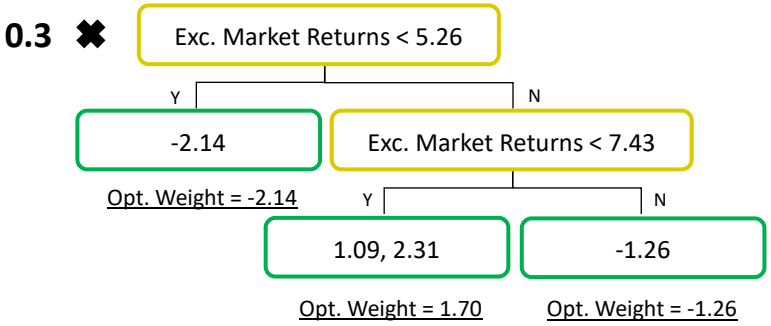
Gain = Gain Left + Gain Right – Gain Root = 1.18 + 0.55 – 1.52 = 0.21

II.



Gain = Gain Left + Gain Right – Gain Root = 5.75 + 1.58 – 1.52 = 5.80

Final



It is important to emphasize that in extreme Gradient Boosting, regularization parameters play a crucial role in controlling model complexity and preventing overfitting. In this example, for the sake of simplicity, these parameters were set to zero. However, in practice, tuning them can significantly enhance model performance. For instance, the gamma γ parameter acts as a pruning mechanism, ensuring that a split in the decision tree only occurs if it leads to a meaningful reduction in the loss function. Similarly, the lambda λ parameter introduces regularization on leaf gains, which helps to smooth the model and reduce variance. Moreover, the learning rate serves to shrink the contribution of each individual tree, further reducing the risk of overfitting by slowing

down the learning process. In this case, as can be seen in Figure 3, the learning rate is set to 0.3. To achieve optimal predictive performance, these hyperparameters must be carefully tuned.

Returning to the example, Table 2 below presents the updated predictions. To obtain these results, it is needed to take the initial prediction and add the product of the learning rate parameter and the optimal weight from the selected decision tree. This process reflects how boosting refines predictions by correcting the errors of the previous model. As shown in Table 2, the updated predictions are even closer to the actual values compared to the initial estimates. This confirms that each iteration is effectively reducing the prediction error, indicating that the model is learning in the right direction.

Table 2: Updated Predictions XGBoost

NR.	DATE	EXCESS PORTFOLIO RETURNS	EXCESS MARKET RETURNS	RESIDUALS	NEW PREDICTIONS
1	2024-05-01	8.23	4.48	-2.14	9.27
2	2024-06-01	11.45	6.03	1.09	10.87
3	2024-07-01	12.67	6.86	2.31	10.87
4	2024-08-01	9.11	7.99	-1.26	9.99

In summary, the XGBoost methodology provides a powerful and efficient framework for predictive modeling, particularly well-suited for capturing nonlinear patterns in financial data. Through iterative learning, regularization, and fine-tuned hyperparameters, the model effectively reduces prediction error while mitigating the risk of overfitting. Its flexibility and robustness make it a valuable tool for asset pricing applications and beyond.

3.3.2 Random Forests

Random Forests is another useful and popular ensemble algorithm that is also designed to complete classification and regression tasks. Similar to extreme Gradient Boosting, the algorithm utilizes combined decision trees in order to get predictions, but the way Random Forests get precise

results and overcome overfitting is totally different. The concept itself is based on the early work of Leo Breiman, who introduced the principles of Bagging (Bootstrap Aggregating) in 1996. The main idea of Bagging methodology is to create multiple versions of the predicted outcome that result from decision trees trained on bootstrapped datasets. The main emphasis in the Bagging algorithm lies on the variance minimization, where the final prediction would be the average of all prediction outputs (regression) or majority voting (classification) of these trees. If the prediction from the i-th tree is $T_i(x)$ and the total number of trees in the forest is N, then the overall prediction $\hat{y}$ for a new input x can be represented as:

$$\hat{y} = \frac{1}{N}\sum_{i=1}^{N} T_i\,(x)$$

Further, Breiman, together with Adele Cutler, extended the idea of Bagging by including one more step of randomness, which resulted into - Random Forests (Breiman, 2001). To be more detailed, the algorithm, similar to its predecessor - Bagging, consists of multiple decision trees that are trained on a certain amount of bootstrapped data, meaning that some data points are repeated while others are left out. In addition to that, to ensure even more diversity in decision trees and capture different aspects of the data, Random Forests chooses a random subset of parameters for its prediction. This method helps to make the decision trees less correlated with each other and reduces overfitting, making the model more robust.

To get a better understanding of how the Random Forests algorithm functions in practice, a simple illustrative example is provided below. Table 3 used in this example includes the first three Fama-French factors—excess market returns (Mkt-RF), SMB (Small Minus Big), and HML (High Minus Low)—along with the corresponding excess portfolio returns. All values are expressed in percentages.

Table 3: Example Random Forests

NR.	DATE	MKT-RF	SMB	HML	EXCESS PORTFOLIO RETURNS
1	2024-05-01	7.99	3.51	-5.35	15.10
2	2024-06-01	1.22	1.04	-0.10	4.65
3	2024-07-01	-0.84	-1.39	4.07	-2.92
4	2024-08-01	-0.96	-0.10	3.72	-1.68
5	2024-09-01	5.09	-2.31	-0.96	4.10

As previously mentioned, the Random Forests algorithm begins by generating several bootstrapped samples from the original dataset. It is important to note that the number of bootstrap iterations is a user-defined parameter and can be adjusted depending on the desired level of accuracy and efficiency. To keep it simple, for this example, only three bootstrapped datasets are created. After generating these datasets, Random Forests introduces its distinctive mechanism of randomly selecting a subset of input variables at each node split. This technique is critical in reducing correlation between individual trees and improving the model's generalization capability. In the given example, only one column is excluded from each bootstrapped table. However, in practical applications, more features can be randomly omitted, depending on the settings defined in the hyperparameters. This level of control allows users to fine-tune the model's complexity and performance. Table 4 below illustrates how the algorithm operates. Each table represents a different randomly generated subset of rows, where duplicates may also occur. For each iteration, the algorithm selects only two out of the three available parameters, while the excluded parameter is highlighted in grey.

Table 4: Bootstrapped Samples

I.

NR.	DATE	MKT-RF	SMB	HML	EXCESS PORTFOLIO RETURNS
3	2024-07-01	-0.84	-1.39	4.07	-2.92
2	2024-06-01	1.22	1.04	-0.10	4.65
4	2024-08-01	-0.96	-0.10	3.72	-1.68
4	2024-08-01	-0.96	-0.10	3.72	-1.68
5	2024-09-01	5.09	-2.31	-0.96	4.10

II.

NR.	DATE	MKT-RF	SMB	HML	EXCESS PORTFOLIO RETURNS
1	2024-05-01	7.99	3.51	-5.35	15.10
2	2024-06-01	1.22	1.04	-0.10	4.65
2	2024-06-01	1.22	1.04	-0.10	4.65
5	2024-09-01	5.09	-2.31	-0.96	4.10
3	2024-07-01	-0.84	-1.39	4.07	-2.92

III.

NR.	DATE	MKT-RF	SMB	HML	EXCESS PORTFOLIO RETURNS
1	2024-05-01	7.99	3.51	-5.35	15.10
5	2024-09-01	5.09	-2.31	-0.96	4.10
5	2024-09-01	5.09	-2.31	-0.96	4.10
2	2024-06-01	1.22	1.04	-0.10	4.65
1	2024-05-01	7.99	3.51	-5.35	15.10

In the next step, the Random Forests algorithm constructs decision trees—somewhat similar in concept to the process used by algorithms like XGBoost. As a reminder, XGBoost builds its trees based on a specific objective function, typically involving a loss function and regularization terms. It computes optimal split points using gradient descent, aiming to minimize the error through carefully derived weights and values at each node. In contrast, Random Forests does not rely on a loss function to guide tree construction. Instead, it uses a simpler approach: it selects the best split at each node by evaluating the variance (in regression tasks) or the Gini impurity or entropy (in classification tasks). In the context of regression, which applies to this example, the algorithm assesses all possible splits across all available features and chooses the one that results in the lowest variance in the target variable. Lower variance within the resulting groups indicates a better, more informative split.

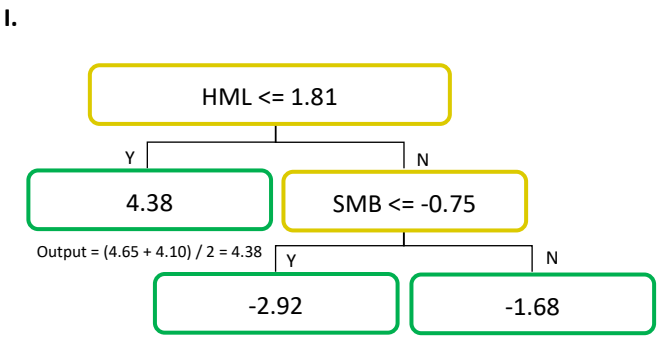
$$Var(t) = \frac{1}{N_t} \sum_{i \in t} (y_i - \bar{y}_t)^2$$

This splitting process continues recursively, constructing the tree from top to bottom. Similar to XGBoost, the moving average is calculated for each parameter to determine the optimal

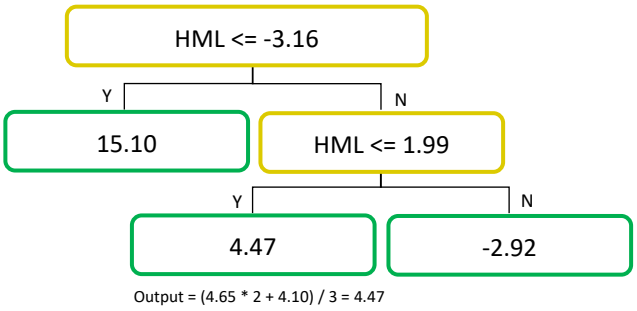
split. All possible combinations are tested, and the split that minimizes variance is chosen. In this case it is more complex, as multiple parameters are present, meaning that a greater number of candidate splits must be considered. Typically, the algorithm continues to split the data until one of two conditions is met: either no further meaningful splits can be made, or the tree reaches a predefined maximum depth. This maximum depth is another hyperparameter that can be manually specified to control the model's complexity and prevent overfitting.

The graph in Figure 4 illustrates a very simplified version of decision trees generated from bootstrapped samples. Once all the trees are constructed, the final prediction is made by aggregating the results from each individual tree. For regression problems, this typically means averaging the predictions from all trees. For instance, in Table 3, the prediction for the first observation would be 4.38 from the first tree and 15.10 from both the second and third trees, yielding an average of 11.52 as the final output. This ensemble approach is what gives Random Forests its strength—by combining the outputs of many individual trees (weak learners), it achieves high predictive accuracy and greater robustness.

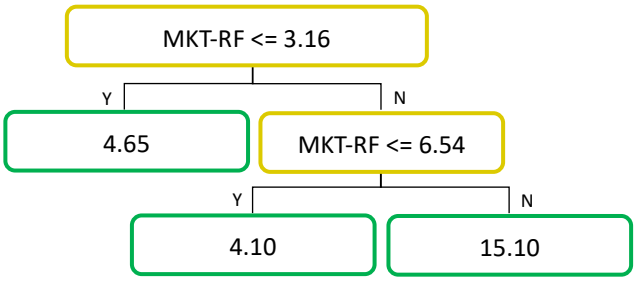
Figure 4: Random Forests Decision Trees



II.



III.



In general, Random Forests for regression are particularly effective because, through techniques such as bootstrap aggregation and random feature selection, they can capture complex, nonlinear relationships between the predictors and the target variable. Thus, they have become widely used in asset pricing and return prediction tasks, where relationships between financial factors and returns are often nonlinear and influenced by complex interactions.

3.3.3 k-Nearest Neighbors

The k-Nearest Neighbors (kNN) algorithm is a fundamental, non-parametric method used for both classification and regression tasks in machine learning. Originally introduced by Fix and Hodges (1951) in an unpublished U.S. Air Force technical report, it gained wider recognition following the influential work of Cover and Hart (1967), who analyzed its consistency and error

bounds in classification settings. In the context of regression, kNN operates by estimating the output value of a query point based on the average of its k nearest neighbors in the training dataset. These neighbors are identified according to a predefined distance metric, most commonly Euclidean. The selection of an appropriate distance metric also affects performance, with Manhattan, Minkowski, and Mahalanobis distances being alternatives to Euclidean distance. This property allows kNN to capture complex, nonlinear relationships in data without requiring a predefined functional form. To predict the value of a new data point x', the kNN regression algorithm follows these steps:

1. Compute the distance between x' and all training points x using a distance metric such as Euclidean distance:

$$d(x', x_i) = \sqrt{\sum_{j=1}^{m} (x'_j - x_{ij})^2}$$

   where x'_j and x_{ij} represent the j-th feature of the test and training point, respectively.
2. Identify the k closest training points based on the computed distances.
3. Compute the predicted value $\hat{y}$ as the average of the target values y_i of the k-nearest neighbors:

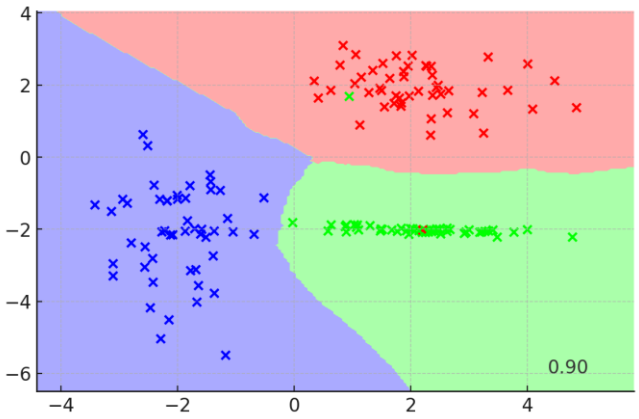
$$\hat{y}(x') = \frac{1}{k} \sum_{i \in \mathcal{N}_k} y_i$$

   where $\mathcal{N}_k$ is the set of indices of the k-nearest neighbors.

In order to demonstrate how the k-Nearest Neighbors algorithm works, Figure 5 is attached below. The colored regions represent the decision boundaries generated by the model based on the proximity of neighboring data points. For instance, if a new data point falls within the red region, its prediction would be based on the average or majority class of the red data points surrounding

it. As with other machine learning models such as Random Forests and XGBoost, kNN includes hyperparameters that must be selected carefully—most notably, the number of neighbors (k). The choice of (k) significantly influences model performance: too small a value may lead to overfitting, while too large a value can smooth out the decision boundaries excessively, reducing accuracy. In Figure 5, the model achieves a good accuracy, indicating reliable classification performance within the given dataset.

Figure 5: kNN Classification Example[1]



Overall, kNN is less sophisticated compared to other machine learning algorithms discussed in this study, such as Random Forests and XGBoost. While it is intuitive and easy to implement, kNN lacks the complexity and adaptability of more advanced models, especially when dealing with high-dimensional data or large datasets. Its performance is highly sensitive to the choice of hyperparameters and the presence of noise in the data. Nevertheless, it serves as a useful baseline model and provides valuable insights into the behavior of instance-based learning methods.

[1] The image shown in Figure 5 was generated using an AI image generation tool (OpenAI, DALL•E).

3.4 Validation

To compare the performance of the linear models with that of the machine learning algorithm, it is essential to select an appropriate evaluation metric. Unfortunately, the standard R-squared is primarily designed for assessing the goodness-of-fit in linear models and may not provide meaningful insights when applied to nonlinear machine learning models. Therefore, as previously mentioned, this study will use the Root Mean Squared Error (RMSE) to compare the final results across models. The formula is displayed below:

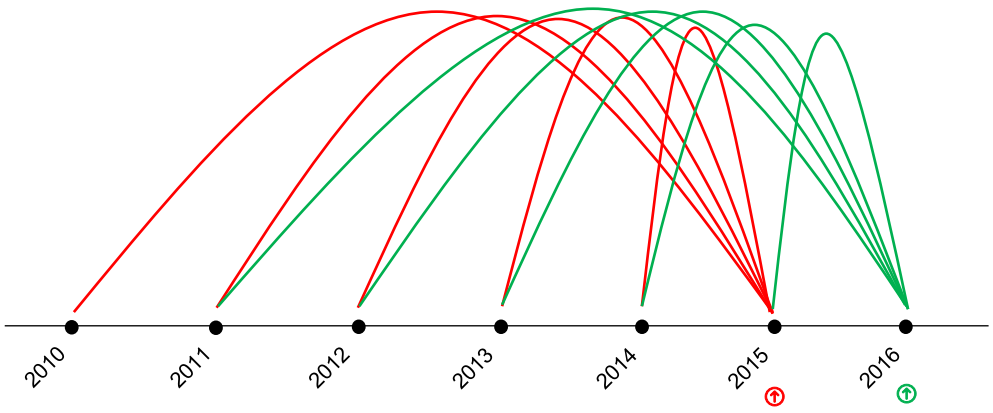
$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}$$

To make sure the results are consistent and reliable across both machine learning models and traditional asset pricing models, the same out-of-sample validation method will be used. In this case, a rolling window is applied, where five years of past data are used to predict returns of the upcoming year. This method is better suited for financial data than the well-known k-fold cross-validation methodology that is usually used for the training of machine learning algorithms. Since stock returns follow a time sequence, it's important to respect that order. A rolling window keeps the timeline intact, making the prediction setup more realistic—just like how forecasts work in real life. On the other hand, k-fold cross-validation randomly splits the data, which can mix past and future information in a way that wouldn't happen in real markets (Bergmeir, C., Benítez, J. M, 2012).

The validation steps include the following steps: (1) For each OOS sample predicted year a RMSE is calculated, resulting in thirty-five RMSE values for a single portfolio. (2) This procedure is done across all portfolios, generating twenty-five RMSE results for each year. (3) To summarize

the results, the median RMSE is derived for each year, providing a single robust error measure. A graphical illustration of a five-year rolling window is presented in Figure 6.

Figure 6: Rolling Window Validation Illustration



To evaluate the joint performance of asset pricing models, this study implements the Gibbons, Ross, and Shanken (1989) test. This statistical procedure assesses whether a set of intercepts from time-series regressions of portfolio excess returns are jointly equal to zero. Under the null hypothesis, if an asset pricing model fully explains expected returns, then all pricing errors (alphas) should be zero. The GRS test statistic is formally defined as:

$$GRS = \frac{T}{N} * \frac{(T - N - L)}{(T - L - 1)} * \frac{\hat{\alpha}' \hat{\Sigma}^{-1} \hat{\alpha}}{1 + \bar{f}' \sum_f^{-1} \bar{f}}$$

where $\hat{\alpha}$ is a vector of estimated intercepts, $\hat{\Sigma}$ is the residual covariance matrix, $\bar{f}$ is the mean of the factor returns, Σ_f is the factor return covariance matrix, N is the number of portfolios, T is the number of time periods and L is the number of factors used in the model. This formulation applies directly to traditional linear factor models such as the Capital Asset Pricing Model (CAPM) and the Fama-French Five-Factor (FF5) model, where the parameters are estimated via regressions on portfolio excess returns (Gibbons, Ross, & Shanken, 1989). Under the null hypothesis, the GRS

statistic follows and F- distribution with N and $T - N - L$ degrees of freedom. The corresponding p-value is obtained from the distribution. If the value is below the 5 % threshold, this will indicate that the null hypothesis is rejected, implying that the model fails to fully explain expected returns.

Unfortunately, the GRS test is unsuitable for assessing the performance of machine learning algorithms. This is primarily due to the fact that machine learning models are typically non-parametric and do not generate explicit pricing errors (alphas) in the same way as linear regressions do. The GRS test relies on the presence of a clearly specified linear model with interpretable intercepts, residual variance-covariance structures, and assumptions regarding normality and homoskedasticity. Since machine learning techniques such as Random Forests, XGBoost, and k-Nearest Neighbors do not estimate alphas or follow a strict functional form, applying the GRS test to these models would violate their assumptions and lead to misleading inferences.

Given the absence of a direct analogue to the GRS test in nonlinear settings, an alternative evaluation tool must be employed. To address this gap, the Diebold-Mariano (DM) test is implemented as a suitable method to statistically compare the forecast accuracy of competing models. Although the DM test does not test pricing errors or factor misspecification, it provides a formal statistical test for the null hypothesis of equal predictive accuracy between two forecasting models. In this context, the DM test helps determine whether the predictive performance of a machine learning model is significantly different from that of a linear benchmark, based on out-of-sample forecast errors such as the Root Mean Squared Error (Diebold & Mariano, 1995).

It is an asymptotic test that operates by computing the loss differential between the forecast errors of two models over a given forecast horizon. Let $e_{1,t}$ and $e_{2,t}$ denote the forecast errors from

the models at time t. A typical loss function is the squared error loss, $L(e_t) = e_t^2$. The loss differential at each time point is defined as:

$$d_t = L(e_{1,t}) - L(e_{2,t})$$

Under the null hypothesis of equal predictive accuracy, the expected value of d_t is zero. The test statistic is given by:

$$DM = \frac{\bar{d}}{\sqrt{\frac{2\pi \hat{f}_d(0)}{T}}}$$

where $\bar{d}$ is the sample mean of d_t, T is the number of observations, and $\hat{f}_d(0)$ is an estimate of the spectral density of d_t at frequency zero, which accounts for possible autocorrelation in the loss differential series. To be more precise, the spectral density describes how the variance of a stationary time series is spread across different frequencies. At frequency zero, it captures the long-run variability, which is equivalent to the variance of the loss differential plus all of its autocovariances. In practice, this quantity is estimated by heteroskedasticity- and autocorrelation-consistent (HAC) estimators (Newey & West, 1987).

While the DM test does not replace the role of the GRS test in validating linear asset pricing models, it provides a viable and statistically sound approach to evaluate whether the machine learning models offer meaningful improvements in predictive accuracy. This distinction is critical when comparing fundamentally different modeling approaches, and it enables a more balanced and rigorous performance evaluation in this study. Similar comparative methodologies using the DM test to evaluate machine learning algorithms in financial forecasting have been employed by authors such as Gu, Kelly, and Xiu (2020).

4 RESULTS DISCUSSION

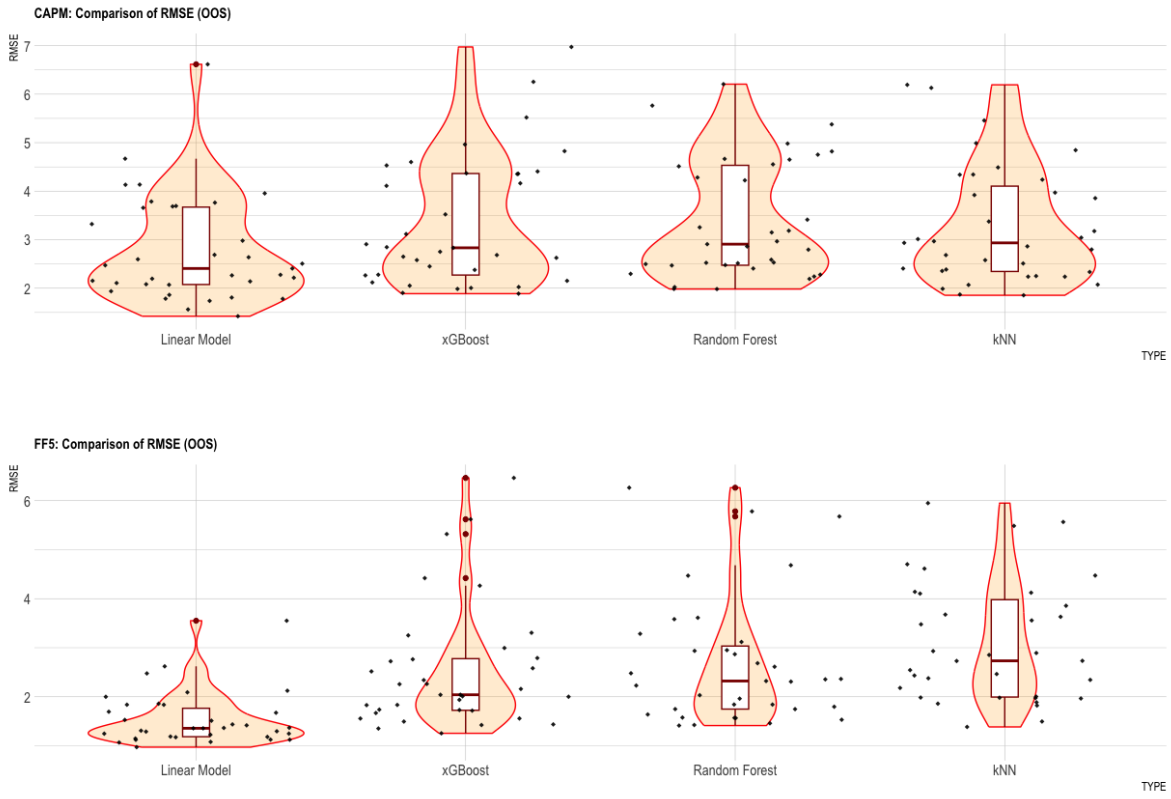
This chapter analyzes the results obtained from the methodologies discussed in the previous chapter. It first examines the outcomes of CAPM and Fama-French linear regressions before moving on to the performance of machine learning algorithms. All computations were conducted using the R programming language.

The graph in Figure 7 presents a violin plot and a boxplot that show the distribution of Root Mean Squared Error (RMSE) values for both – the linear approach and the machine learning algorithms using the parameters only from the Capital Asset Pricing Model (CAPM) or the Fama-French Five-Factor (FF5) model. The violin plot is used to display the density of the RMSE distribution, where the boxplot's main role is to highlight the interquartile range and median. In addition to that, all observations in the form of black data points are also included in the figure. All these plots are helpful to determine the spread and the skewness of the RMSE values across all models. As can be seen from the first row, where only the CAPM factor was used, the distribution of RMSE values is somehow similar across all models. The linear regression displays a bit better result, with the lower median of observations compared to those of the machine learning algorithms. The results given by extreme Gradient Boosting, Random Forests and kNN are nearly indistinguishable from each other. Nevertheless, even with the linear model showing better results, the difference is not that big, indicating that for the CAPM, the predictive power of machine learning algorithms, to some extent, is comparable with the classical linear regression. However, for the FF5, the linear regression clearly outperforms all machine learning models. It can be explicitly seen that the RMSE values are tightly clustered around the median, with few extreme values observed. All three machine learning algorithms have a much broader RMSE distribution

with a higher median and less accurate predictions, with kNN having by far the worst outcome from all models.

Overall, it can be concluded from the figure that the chosen machine learning techniques do not offer a clear advantage under the CAPM framework and clearly underperform when compared to the FF5 linear model. This suggests that the linear relationships embedded in the FF5 model are effectively captured by traditional regression, whereas machine learning models may introduce unnecessary complexity that fails to improve, and may even degrade, predictive performance.

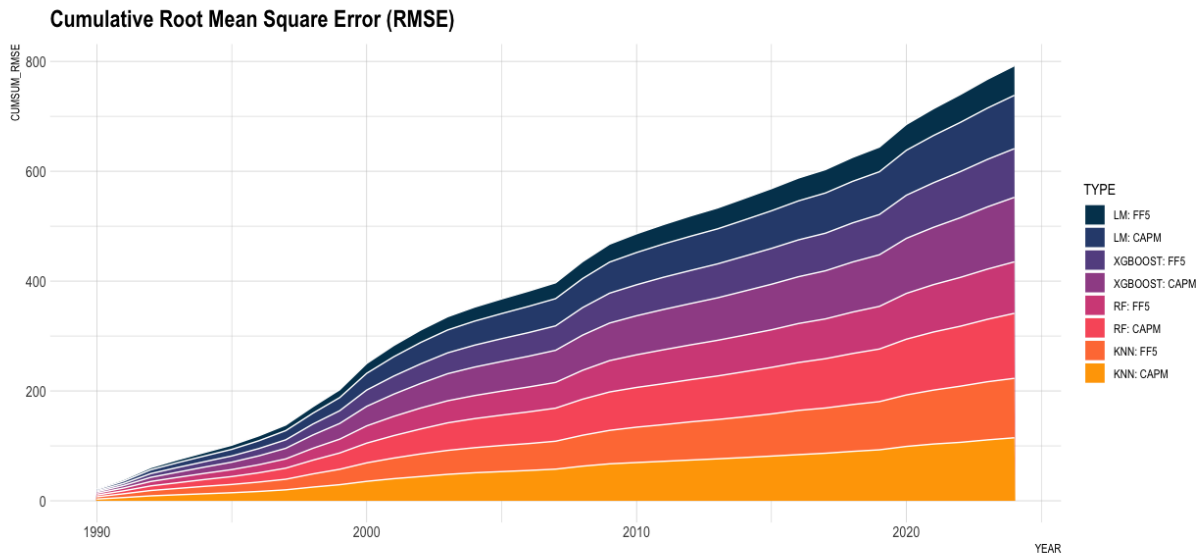
Figure 7: Boxplots of RMSE per Model



To give the final statement about the performance of machine learning algorithms and the linear regression, Figure 8 illustrates the evolution of the cumulative Root Mean Squared Error

(RMSE) over time, beginning from 1990 to 2024. On the graph, each colored line in the stacked area represents the aggregated RMSE year by year, where the vertical axis line indicates the cumulative values and the horizontal axis the timeline itself. From the plot it becomes explicit that the RMSE throughout the whole period has nearly a linear development, what means that the error accumulates at a constant rate. The steady growth shows that each model performs consistently over the years. The final cumulative results on the right side of the plot indicate that linear models (LM) – especially the Fama-French Five-Factor model - have the lowest RMSE, meaning that they have produced a more accurate prediction compared to other models. Even the machine learning algorithms have delivered higher cumulative errors, regardless of whether they used CAPM or FF5 factors, they are still not much worse than their linear counterparts.

Figure 8: Cumulative RMSE over time



However, as in Figure 7 the plot further supports the conclusion that chosen machine learning algorithms like extreme Gradient Boosting, Random Forests and k-Nearest Neighbors (kNN) have failed to outperform the linear models in stock prediction. It has been learned that adding

complexity through nonlinear algorithms does not improve performance and may even lead to worse results.

Nevertheless, even if the performance of the chosen machine learning algorithms was not better than linear models, it is still interesting to investigate if they were still able to detect any nonlinear patterns in the data. The Figure 9 and Figure 10 below display the plot of predicted and actual values for the extreme Gradient Boosting and the linear regression in the CAPM and FF5 settings. Similar plots of other machine learning methods could be found in the Appendix. In both cases a straight line is perfectly fitted to the results, whereas only in XGBoost a minimal nonlinear trend observed that is mainly caused by outliers in the data. Even here the machine learning models were not able to uncover meaningful nonlinear relationships in the data, reinforcing that in this case, with this chosen data, the return-generating process behaves in an essentially linear manner.

Figure 9: Diagnostic Plots (Linear Model)

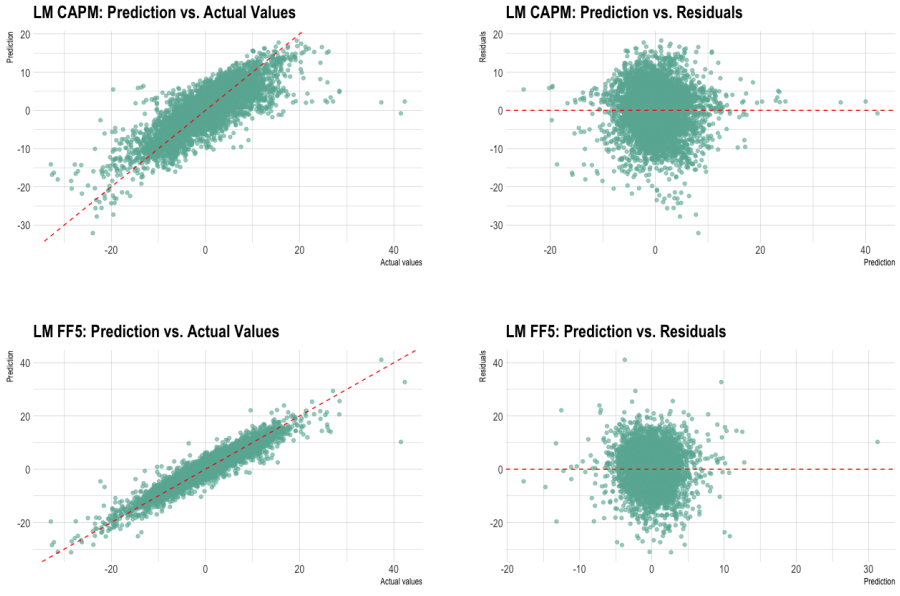


Figure 10: Diagnostic Plots (XGBoost)

In addition, the GRS test results are presented in Table 5. The main goal was to examine whether the intercepts (alphas) of time-series regressions across all portfolios are jointly equal to zero, which would indicate that the model fully explains expected returns (Gibbons, Ross, & Shanken, 1989). The test statistics obtained were 0.58 for the CAPM and 0.18 for the FF5 model, with corresponding p-values of 0.95 and 1. Since both p-values are above the significance threshold of 5 %, the null hypothesis of zero pricing errors cannot be rejected in both cases, suggesting that models adequately price all twenty-five portfolios. The lower GRS statistic for the FF5 model further supports its relatively stronger performance, though the difference compared to the CAPM is not statistically significant given the very high p-values.

Table 5: GRS-Statistics Results

Model	GRS-Statistics	p-value
CAPM	0.58	0.95
FF5	0.18	1

Since the GRS test is not applicable to machine learning algorithms due to their non-parametric and often nonlinear structure, the Diebold-Mariano (DM) test was employed to compare the predictive accuracy of the machine learning models with those of the linear benchmarks. The DM test evaluates whether there is a statistically significant difference in forecast errors between two models. A significant threshold of 5 % is also applied. Here, the test compares the RMSE-based loss functions over time, where a negative test statistic indicates that the linear models outperform the machine learning models. The results in Table 6 show consistently negative and statistically significant DM test statistics for all comparisons. For instance, when comparing machine learning models to the FF5 benchmark, values of –23.30 (XGBoost), –25.03 (Random Forests), and –32.65 (kNN) were obtained. Similar outcomes are observed against the CAPM benchmark. All ML models had a p-values very close to zero what indicates the strong rejection of the null hypothesis.

Table 6: Diebold-Mariano-Test Results

Model	XGBoost	Random Forests	kNN
CAPM	-20.31	-20.97	-18.09
FF5	-23.30	-25.03	-32.65

This finding contrasts with the common expectation that machine learning can outperform traditional methods. It indicates that either the nonlinearities in the data are not strong enough to justify the use of complex models, or that the machine learning models used—despite their flexibility—were not adequately tuned or structured to capture the true return-generating process. Additionally, it may reflect that linear asset pricing models, while simple, remain highly effective under the specific empirical conditions of this study.

While machine learning algorithms offer the flexibility to model complex relationships, their performance in this study did not surpass that of traditional linear models, such as the CAPM and the Fama-French Five-Factor Model. Although the nonlinear models provided reasonable predictive power, they failed to deliver consistently superior results. This outcome can be attributed to several key factors. One possible explanation lies in the quality of the dataset used for the analysis. The stock return data sourced from Kenneth R. French's database is already well-structured and exhibits relatively strong linear relationships with the established risk factors. As a result, identifying meaningful nonlinear patterns becomes inherently challenging. In financial markets, nonlinearity often emerges under conditions of market stress, anomalies, or inefficiencies, none of which were significantly present in this dataset. Consequently, while nonlinear machine learning models have the potential to detect complex dependencies, they may struggle to provide additional value when applied to a dataset where linear assumptions hold relatively well.

Another important consideration is the choice of machine learning algorithms. While the models used in this study, such as kNN, offer nonlinearity, they may not be the most advanced or specialized variants available. More sophisticated techniques, such as deep learning architectures or neural networks, could potentially improve performance. However, these approaches often require significantly more computational power and introduce a higher degree of model complexity, which may not be necessary in this case—especially when traditional linear models are already performing well and providing interpretable results.

Additionally, a different data source could have been explored to test whether alternative datasets would reveal stronger nonlinearities. However, this approach presents its own challenges. Lower-quality data could introduce noise, making it harder to extract meaningful relationships.

Moreover, financial markets are influenced by numerous external factors, such as macroeconomic conditions, policy changes, or investor sentiment, which may not be fully captured in any given dataset. While machine learning models might be able to capture such externalities if they appear in the training data, there is no assurance that these patterns would persist in the future.

5 CONCLUSION

The main goal of this thesis was to explore the potential of machine learning algorithms to capture nonlinear relationships in stock returns by applying variables from the Capital Asset Pricing Model and the Fama-French Five-Factor Model as inputs. Algorithms such as extreme Gradient Boosting, Random Forests and k-Nearest Neighbors had the task of outperforming traditional linear regression methods in delivering a more accurate prediction of asset returns.

Despite the theoretical appeal of machine learning for modeling complex, nonlinear patterns in financial data, the empirical results presented in this study indicate that the stated hypothesis is rejected, as the selected ML algorithms failed to outperform the linear benchmarks. In fact, the models not only underperformed in predictive accuracy but also failed to detect any meaningful nonlinearities in the dataset. This suggests that the chosen machine learning methods—while widely used in many domains—may not be well-suited for this type of financial modeling, at least not in their current configuration or when using traditional factor-based inputs.

In a broader context, these findings imply that either the underlying financial data does not exhibit the kind of nonlinear structure that machine learning models are best at exploiting, or that the specific algorithms used in this study lack the complexity or flexibility required to capture such patterns. More sophisticated models, such as deep learning or neural networks, may be better

equipped to uncover subtle interactions and structures within the data. These models have shown promise in other areas of financial prediction and could be a valuable avenue for future research.

Furthermore, as discussed in the limitations section, the use of the Kenneth R. French twenty-five portfolios dataset may itself impose structural constraints on the analysis. These portfolios are constructed based on linear factor frameworks and may inherently favor linear explanatory models. Consequently, applying nonlinear models to such pre-aggregated data could obscure potential patterns that might be visible at the individual stock level or when using more flexible, tailor-made factor structures. Future research could focus on developing new sets of factors explicitly designed for use with nonlinear models, rather than relying on traditional ones originally built for linear approaches such as CAPM or FF5.

In summary, while this study did not find evidence supporting the superiority of machine learning models over linear methods in the context of predicting asset returns using established factor models, it provides valuable insights into the challenges and limitations of applying such techniques in financial modeling. It highlights the importance of both algorithm selection and data structure when attempting to detect nonlinearities in financial markets. Future studies that incorporate raw stock-level data, alternative factor constructions, or more advanced algorithms may yield more promising results.

6 BIBLIOGRAPHY

Adrian, T., Crump, R. K., & Vogt, E. (2019). Nonlinearity and flight-to-safety in the risk-return trade-off for stocks and bonds. *The Journal of Finance*, *74(4)*, 1931–1973. https://doi.org/10.1111/jofi.12776

Adrian, T., Etula, E., & Muir, T. (2014). Financial intermediaries and the cross-section of asset returns. *Journal of Finance*, *69(6)*, 2557–2596. https://doi.org/10.1111/jofi.12189

Ang, A., & Bekaert, G. (2002). International Asset Allocation with Regime Shifts. *Review of Financial Studies*, *15(4)*, 1137–1187. https://doi.org/10.1093/rfs/15.4.1137

Balke, N. S., & Fomby, T. B. (1997). Threshold Cointegration. *International Economic Review*, *38(3)*, 627–645. https://doi.org/10.2307/2527284

Barillas, F., & Shanken, J. (2018). Comparing asset pricing models. *Journal of Finance*, *73(2)*, 715–754. https://doi.org/10.1111/jofi.12607

Bartholdy, J., & Peare, P. (2004). Estimation of expected return: CAPM vs. Fama and French. *International Review of Financial Analysis*, *14(4)*, 407–427. https://doi.org/10.1016/j.irfa.2004.10.009

Bergmeir, C., & Benítez, J. M. (2012). On the use of cross-validation for time series predictor evaluation. *Information Sciences*, *191*, 192–213. https://doi.org/10.1016/j.ins.2011.12.028

Breiman, L. (1996). Bagging predictors. *Machine Learning*, *24(2)*, 123–140. https://doi.org/10.1007/BF00058655

Breiman, L. (2001). Random forests. *Machine Learning*, *45(1)*, 5–32. https://doi.org/10.1023/A:1010933404324

Bryzgalova, N., Pelger, M., & Zhu, J. (2023). Forest through the trees: Building cross-sections of stock returns. *The Review of Financial Studies, 36(5)*, 2119–2175. https://doi.org/10.1111/jofi.13477

Cakici, N., Fabozzi, F. J., & Tan, S. (2013). Size, value, and momentum in emerging markets. *Emerging Markets Review*, *16*, 46–65. https://doi.org/10.1016/j.ememar.2013.03.001

Chen, T., & Guestrin, C. (2016). XGBoost: A scalable tree boosting system. *Proceedings of the 22nd ACM SIGKDD*, 785–794. https://doi.org/10.1145/2939672.2939785

Cover, T., & Hart, P. (1967). Nearest neighbor pattern classification. *IEEE Transactions on Information Theory, 13(1)*, 21–27. https://doi.org/10.1109/TIT.1967.1053964

Daniel, K., & Titman, S. (1997). Evidence on the characteristics of cross-sectional variation in stock returns. *Critical Finance R*eview, *1(1)*, 103–127. https://doi.org/10.2307/2329554

Diebold, F. X., & Mariano, R. S. (1995). Comparing predictive accuracy. *Journal of Business & Economic Statistics*, *13(3)*, 253–263. https://doi.org/10.2307/1392185

Dittmar, R. F. (2002). Nonlinear Pricing Kernels, Return Predictability, and the Cross-Section of Equity Returns. *Journal of Finance*, *57(1)*, 369–402. https://doi.org/10.1111/1540-6261.00425

Fama, E. F., & French, K. R. (1993). Common risk factors in the returns on stocks and bonds. *Journal of Financial Economics*, *33(1)*, 3–56. https://doi.org/10.1016/0304-405X(93)90023-5

Fama, E. F., & French, K. R. (2004). The Capital Asset Pricing Model: Theory and Evidence. *Journal of Economic Perspectives*, *18(3)*, 25–46. https://doi.org/10.1257/0895330042162430

Fama, E. and French, K. (2015). A five-factor asset pricing model. *Journal of Financial Economics*, *116(1)*, 1-22. https://doi.org/10.1016/j.jfineco.2014.10.010

Fama, E. F., & French, K. R. (2016). Dissecting anomalies with a five-factor model. *Review of Financial Studies*, *29(1)*, 69–103. https://doi.org/10.1093/rfs/hhv043

Fischer, T., & Krauss, C. (2018). Deep learning with long short-term memory networks for financial market predictions. *European Journal of Operational Research*, *270(2)*, 654–669. https://doi.org/10.1016/j.ejor.2017.11.054

Fix, E., & Hodges, J. L. (1951). Discriminatory analysis, nonparametric discrimination: Consistency properties. *Technical Report No. 4, USAF School of Aviation Medicine*, Randolph Field, Texas. https://doi.org/10.2307/1403797

Friedman, J. H. (2001). Greedy function approximation: A gradient boosting machine. *Annals of Statistics*, *29(5)*, 1189–1232. https://doi.org/10.1214/aos/1013203451

Freund, Y., & Schapire, R. E. (1997). A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, *55(1)*, 119–139. https://doi.org/10.1006/jcss.1997.1504

Ghysels, E., Santa-Clara, P., & Valkanov, R. (2005). There is a risk-return trade-off after all. *Journal of Financial Economics, 76(3)*, 509–548 https://doi.org/10.1016/j.jfineco.2004.03.008

Gibbons, M. R., Ross, S. A., & Shanken, J. (1989). A Test of the Efficiency of a Given Portfolio. *Econometrica*, *57(5)*, 1121–1152 https://doi.org/10.2307/1913625

Gu, S., Kelly, B., & Xiu, D. (2020). Empirical asset pricing via machine learning. *The Review of Financial Studies*, *33(5)*, 2223–2273. https://doi.org/10.1093/rfs/hhaa009

Hamilton, J. D. (1989). A new approach to the economic analysis of nonstationary time series and the business cycle. *Econometrica*, *57(2)*, 357–384. https://doi.org/10.2307/1912559

Hou, K., Xue, C., & Zhang, L. (2015). Digesting anomalies: An investment approach. *Review of Financial Studies*, *28(3)*, 650–705. https://doi.org/10.1093/rfs/hhu068

Hsieh, D. A. (1991). Chaos and nonlinear dynamics: Application to financial markets. *Journal of Finance*, *46(5)*, 1839–1877. https://doi.org/10.1111/j.1540-6261.1991.tb04646.x

Jagannathan, R., & Wang, Z. (1996). The Conditional CAPM and the Cross-Section of Expected Returns. *The Journal of Finance*, *51(1)*, 3–53. https://doi.org/10.1111/j.1540-6261.1996.tb05201.x

Krauss, C., Do, X. A., & Huck, N. (2017). Deep neural networks, gradient-boosted trees, random forests: Statistical arbitrage on the S&P 500. *European Journal of Operational Research*, *259(2)*, 689–702. https://doi.org/10.1016/j.ejor.2016.10.031

Lim, K.-P., Brooks, R. D., & Hinich, M. J. (2008). Nonlinear serial dependence and the weak-form efficiency of Asian emerging stock markets. *Journal of International Financial Markets, Institutions and Money*, *18(5)*, 527–544. https://doi.org/10.1016/j.intfin.2007.08.001

Lintner, J. (1965). The valuation of risk assets and the selection of risky investments in stock portfolios and capital budgets. *The Review of Economics and Statistics*, *47(1)*, 13–37. https://doi.org/10.2307/1924119

Markowitz, H. (1952). Portfolio selection. *The Journal of Finance*, *7(1)*, 77–91. https://doi.org/10.2307/2975974

Mossin, J. (1966). Equilibrium in a capital asset market. *Econometrica*, *34(4)*, 768–783. https://doi.org/10.2307/1910098

Nelson, D. M., Pereira, A. C. M., & de Oliveira, R. A. (2017). Stock market's price movement prediction with LSTM neural networks. *Proceedings of the International Joint Conference on Neural Networks (IJCNN)*, 1419–1426. https://doi.org/10.1109/IJCNN.2017.7966019

Newey, W. K., & West, K. D. (1987). A Simple, Positive Semi-Definite, Heteroskedasticity and Autocorrelation Consistent Covariance Matrix. Econometrica, 55(3), 703–708. https://doi.org/10.2307/1913610

Schapire, R. E. (1999). Theoretical views of boosting and applications. *In Proceedings of the 10th International Conference on Algorithmic Learning Theory (ALT '99)*, 13–25. https://doi.org/10.1007/3-540-46769-6_2

Sharpe, W. F. (1964). Capital asset prices: A theory of market equilibrium under conditions of risk. *The Journal of Finance*, *19(3)*, 425–442. https://doi.org/10.2307/2977928

World Bank. (2024). GDP (current US$) [Data set]. *World Bank Open Data*. https://data.worldbank.org/indicator/CM.MKT.LCAP.CD

7 APPENDIX

Figure 11: Diagnostic Plots (RF)

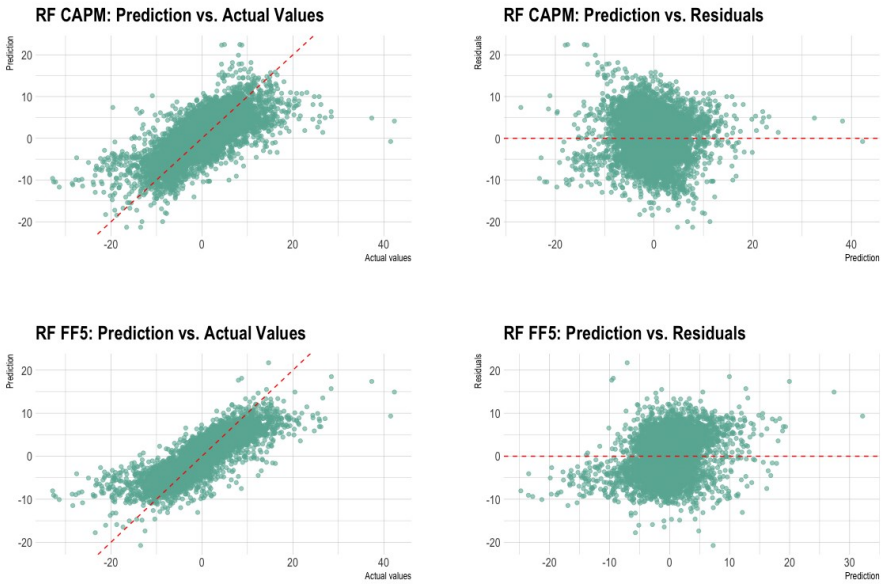


Figure 12: Diagnostic Plots (kNN)