Department of Business and Management

Bachelor's Degree in

Management and Computer Science

Finance and Financial Technologies [SECS-P/09]

XAI Algorithms for Deep Learning Models in Asset Pricing

Candidate

Giulio Presaghi

287611

Supervisor

Prof. Michela Altieri

Academic Year

2024 / 2025

Abstract

In order to elucidate the rationale behind the stock predictions produced by four of the primary machine learning models utilized in the asset pricing domain, this thesis investigates explainable artificial intelligence applied to empirical asset pricing. The two primary cutting-edge algorithms used for this investigation are LIME and SHAP. The purpose of this work is to show how adding insights from explainable AI to such models enhances their interpretability and prediction ability. Furthermore, a range of explainable AI data visualization techniques are offered to present the results and the models' interpretations in a clear and effective way.

Aknowledgements

This work not only stands as a milestone in my career as a student, but also as a person living in this stimulating and evolving society. Indeed, after three wonderful years full of enjoyment, successes, failures and hard work, I finally summarized what I've learned from them in this thesis. There are so many people to thank. I'll do my best to list them all.

Thanks to my professor, Michela Altieri, who guided me through the writing of this thesis and has been a wonderful mentor and teacher all over this semester. Thanks to my Colleagues with who I shared lots of new experiences, excitement and fears all over this journey. Thanks to my family who made all of this possible. Thanks to my lifetime friends. Thanks to my grandfather, for having provided me with such a curiosity and perseverance. And lastly, thanks to Myself, with which I learned, lived and felt all of this.

Giulio Presaghi, Roma, 19/09/2025

Contents

Chapter 1

Introduction

With this thesis, I investigate asset pricing by integrating machine learning (ML) techniques with explainable artificial intelligence (XAI) algorithms. The primary objective is to demonstrate how XAI can effectively address the black-box problem of ML models, an issue of particular importance in financial applications.

While the predictive capabilities of ML models in forecasting asset returns are well recognized, owing to their ability to capture complex and non-linear relationships between the dependent variable (asset returns) and independent variables (factors), their lack of interpretability remains a significant concern. This is especially true for black-box models such as Neural Networks and K-Nearest Neighbors, where the internal decision-making processes are difficult to discern.

Traditionally, the asset pricing literature has employed techniques such as portfolio analysis and linear regression, as exemplified by the seminal work of Fama and MacBeth [1973]. These methods offer clear interpretability, allowing researchers to extract meaningful economic insights from estimated coefficients and results.

However, they exhibit notable limitations in terms of *accuracy* when it comes to predicting risk premia. Beyond linear regression, a common practice involves sorting assets into portfolios based on firm characteristics and analyzing the average

returns within each group, a sort of non-parametric regression. While this approach provides greater flexibility in modeling asset returns, it also introduces challenges related to precision and predictive reliability.

Establishing a link between risk premia and the factor structure requires an initial estimation of these quantities, a task fraught with complexity. Measuring expected returns is particularly difficult due to market efficiency, which causes most of the variation in returns to be driven by unforecastable events. Compounding this problem is the relatively small sample size of asset returns compared to the often large set of predictive variables. Structural breaks, regime shifts, and nonstationarity further erode the effective sample size. Additionally, the set of candidate conditioning variables is not only vast but also characterized by high multicollinearity. An added layer of complexity arises from the uncertainty surrounding the functional relationships that link these high-dimensional predictors to expected returns. Collectively, these issues create a low signal-to-noise environment, making the accurate estimation of risk premia significantly more challenging than standard prediction tasks encountered in fields like computer science.

To overcome the longstanding challenges in financial data analysis, the empirical finance literature has increasingly explored the application of machine learning (ML) techniques. Elements of the ML framework, such as variable selection and dimensionality reduction, have been fundamental to empirical asset pricing since the early development of the field. Initially, researchers relied on economic theory and parsimonious model specifications to impose structure on learning problems in financial markets. The standard practice of sorting stocks by firm characteristics, constructing equal- or value-weighted portfolios, and selecting a limited number of portfolios as factor proxies exemplifies this approach. These methodological choices were aimed at mitigating issues such as nonlinearity, low signal-to-noise ratios, and the curse of

dimensionality, common obstacles in modeling asset returns.

In recent decades, the statistics and machine learning communities have introduced a wide array of exploratory and predictive methods that complement traditional economic approaches. These data-driven tools address empirical complexities in asset pricing and offer economists enhanced capabilities to uncover robust and insightful patterns that theory alone might not predict. At the same time, empirical findings derived from these techniques can feed back into theory development, offering new perspectives that refine or extend existing economic models.

Despite their strengths, many ML methods, particularly black-box models, present interpretability challenges. These models often lack transparency in how predictions are generated, requiring users to rely on their computational efficiency and predictive performance without a clear understanding of their internal mechanisms. For individuals who place strong trust in advanced algorithms and large datasets, this may be sufficient for guiding investment decisions. However, in professional financial environments, where articulating the rationale behind investment strategies is essential, such opacity poses a serious limitation.

Therefore, using ML models without a clear grasp of their decision-making processes is problematic in finance. Investment strategies typically require not only empirical validity but also economic justification. A transparent understanding of how models arrive at predictions is critical for ensuring accountability, facilitating

communication with stakeholders, and supporting sound financial decision-making.

Chapter 2

Literature Review

This chapter presents the state-of-the-art explainable artificial intelligence (XAI) and machine learning (ML) techniques employed for financial analysis, alongside an overview of the current asset pricing landscape. The asset pricing literature has evolved significantly, transitioning from foundational models like the Capital Asset Pricing Model (CAPM) [Sharpe, 1964] to contemporary approaches leveraging machine learning [Gu et al., 2020]. To enhance predictive accuracy, identify key drivers and gain insights into the dynamics of asset prices, researchers have applied a broad range of empirical methodologies.

Historically, techniques such as linear regression, particularly the Fama-MacBeth framework, and portfolio analysis have been central to empirical finance. However, in the past decade, substantial efforts have been directed toward incorporating machine learning methods to improve predictive performance and overcome the limitations of traditional models.

2.1 Theoretical Foundations

The financial discipline of determining the most significant factors influencing the average movement of the values of the underlying assets and creating techniques for estimating and forecasting these prices is known as *Asset Pricing*. Because they offer an economic framework for characterizing the cross-sectional dependence structure of returns, the so-called *Factor Models* are used in this context to model asset returns. The Fama and French [Fama and French, 1993] and CAPM [Sharpe, 1964] were among the models that used observable macroeconomic and financial variables as risk factors.

The *Capital Asset Pricing Model* represents the first approach in this financial field [Sharpe, 1964]. Indeed, its simplicity, also represents its greatest drawback. Its unrealistic assumptions and its poor empirical performance made it perfect for being elected as a baseline for the future asset pricing models. This one-factor model, assumes a direct relation between return and risk, facilitating the forecast for the returns. Two of its main assumptions are: *"a riskless asset exists, and investors can borrow and lend unlimited amounts at the risk free rate"*; *"complete agreement of investors on asset return distribution"*[Sharpe, 1964] This model requires the parameter β to describe the association between systematic risk and stock rates of return. β measures the degree of co-movement between asset returns and market portfolio returns. In other words, it quantifies the systematic risk of an asset (the amount of risk that cannot be diversified away).

The Fama-French multi-factor models implemented significant improvements to the older ones. According to them, the assumption of the CAPM regarding the unrestricted borrowing and lending of any amount of a risk free asset is an unrealistic assumption [Fama and French, 2004]. Starting from the idea that a single factor, like

the market or aggregate consumption growth, is insufficient to explain differences in average stock returns, Fama and French developed a three factor model based on Market, SMB and HML factors. The first factor is the same as the one in the *CAPM* (the market factor); the other two are the size factor and the value factor. The first one, *SMB* (Small Minus Big), reflects the ability of small companies to outperform larger ones, while the second factor, *HML* (High Minus Low), expresses the tendency of value stocks to perform better than growth stocks.

However various criticisms arose from the conception of this model. One of them, stated by Chung and Schill [2006], is the lack of economic interpretation of the other two factors *SMB* and *HML*, while the Market factor is obviously a proxy for the risk. Fama and French claimed that those factors represent firm distress, while several other papers suggested that the predictive power of size and value factors is spurious [Berk, 1995]. An alternative explanation is given again by Chung and Schill [2006], claiming that *SMB* and *HML* are proxy for the part of risk not captured by the market factor.

Taking a step back, an important model in the panorama is surely the APT [Ross, 1976]. This model introduced innovation in the asset pricing panorama by understanding foundational economic concepts (like risk premia) without the need of an economic interpretation of the factors used. This model quickly became the most adopted from both academics and practitioners and it is recognized as a catalyst for the application of machine learning to empirical asset pricing, due to its ease of use of latent factor models for return prediction.

Over time, many factors that supposedly predict cross-sectional variation in expected returns have been documented by Harvey et al. [2016]. This vast number of factors, often referred to as the *Factor Zoo*, has faced criticism. Some doubt the

actual usefulness of the proposed factors [McLean and Pontiff, 2016], while others struggle with incorporating so many risk sources into models explaining these empirical findings. A very notable contribution to the factors' panorama surely comes from Jegadeesh and Titman [1993] who introduced the concept of *momentum* for US stock prices. A very important step, because of proved powerful predictive power for excess return.

2.2 Asset Pricing and Machine Learning

Machine Learning techniques are widely leveraged in the literature of Asset Pricing in various ways, improving the predictions made by traditional models. Firstly Machine Learning is used to select and eliminate abundant factors, identifying the most important ones. It is further employed to improve predictive and investment performance, often measured by out-of-sample R^2 and the Sharpe ratio. In addition to this, also Explainable AI techniques are applied in finance to address diverse challenges and enhance model's performance. Dealing with the challenge of selecting only impactful factors, several studies have proposed various ideas. Giglio and Xiu [2021] use dimensionality reduction methods to estimate and test factor pricing models.

One of these methods is *PCA (Principal Component Analysis)*, that basically consists in rotating the data space in order to reduce the dimensions. Their approach 'uses principal components of test asset returns to recover the factor space and additional regressions to obtain the risk premium of the observed factor'. Their model is considered a two-factors cross-sectional model, where the factors are the two Principal Components. In that way, all the selected predictors are 'weighted' and included in the prediction, overcoming the 'omitted predictors' problem. A variation of this approach is given by Kelly et al. [2019], who implemented *IPCA*, a more accurate dimensionality reduction method.

One of the first implementations of a tree-based model on this topic comes from Moritz and Zimmermann [2016], who applied it to the cross-section of stock returns. It is proved that these models achieve much more accurate results than those in the traditional literature.

Chen and Guestrin [2016] implemented, among the others, an ensemble model that proved to be the most used by academics and practitioners. XGBoost is an advanced algorithm that constructs models sequentially. Unlike Random Forest, where trees are created in parallel, XGBoost builds new trees to correct the errors made by previous trees. This gradual approach allows for continuous improvement of the predictions.

The deep learning applicability was discussed, among the others, by Heaton et al. [2016]. The paper explores the applicability of deep learning hierarchical models to financial problems, highlighting the point that deep learning can *'detect and exploit interactions in the data that are, at least currently, invisible to any existing financial economic theory.'*[Heaton et al., 2016]. A more recent analysis of Deep learning in asset pricing is given by Ye et al. [2024], who comprehensively reviews the application of machine learning (ML) and AI in finance, starting from summarizing the traditional asset pricing models and examining their limitations in capturing the complexities of financial markets to exploring how ML models, including supervised, unsupervised, semi-supervised, and reinforcement learning, provide versatile frameworks to address these complexities.

Another important report of the majority of ML methods in asset pricing is the one of Gu et al. [2020]. It offers a complete picture of the link between asset pricing and machine learning. A common takeaway from all these papers is that even though Deep learning methods perfectly fit some kind of financial application, they are not widely used in this field. This is because finance require a deep understanding of the

decision-making process, and neural networks lack of interpretability.

To address this fact, recent studies have applied explainable AI techniques in finance. Demirbaga and Xu [2024] proposes one of the few implementations of explainable AI in asset pricing by adopting the most famous XAI algorithms (LIME and SHAP).

Finally, as reported in their survey, W. and C. [2023] show a *a systematic meta-survey of current challenges and future opportunities*, willing to address the future employability of explainable methods in various fields of interest.

The study of this thesis wants to contribute to the current research literature by applying XAI algorithms in asset pricing. thereby providing better explanations and improved performance. This works aims at enhancing the interpretability of Deep learning models offering a more comprehensive understanding of asset pricing dynamics.

Chapter 3

Explainable Artificial Intelligence

Understanding the rationale behind a model's prediction can be as important as the prediction's accuracy, particularly in financial applications. However, achieving high levels of accuracy with large-scale modern datasets, often requires the use of complex models, such as ensemble methods or deep learning architectures, that are difficult to interpret, even for domain experts. This results in a trade-off between

accuracy and *interpretability*. To address this issue, several methods have been introduced to aid users in understanding and interpreting the predictions generated by these sophisticated models.

In the field of finance and asset pricing, the ability to interpret a prediction model's output is of paramount importance. It fosters appropriate user trust, provides insights into how a model can be improved and supports a deeper understanding of the process being modeled. In traditional empirical financial research, simple models are often preferred for their ease of interpretation, even if they are less accurate than more complex models. However, the increasing availability of big data has amplified the advantages of using complex models, thus highlighting the trade-off between *accuracy* and *interpretability* of a model's output. Recently, a wide array of methods has been proposed to address this issue.

This chapter is dedicated to the description of the two most popular XAI algorithms, LIME and SHAP, which are the models employed in the research conducted with this thesis.

3.1 LIME

Local Interpretable Model-Agnostic Explanations (LIME), proposed by Ribeiro et al. [2016], is a technique developed to offer clear and faithful explanations of predictions made by any classification model. The key idea behind LIME is to construct a simple, interpretable model that approximates the behavior of a more complex black-box model in the vicinity of a specific instance.

LIME starts by defining an interpretable representation of the data. Let $x \in \mathbb{R}^d$ denote the original input, and let $x' \in \{0,1\}^d$ represent its interpretable version. Consider f as the original black-box model (e.g., neural networks), and g as the interpretable explanation model. The complexity of the explanation model is denoted by

$\Omega(g)$.

The goal is to find a model $g \in G$, where G is a class of interpretable models such as linear models or decision trees, which operates on x' and approximates the prediction of $f(x)$. To construct g, LIME selects important features using techniques such as Lasso regression, and fits a weighted linear model. The weights are derived from a locality-aware proximity measure π_x, which ensures that the explanation is centered around the point x.

To quantify how well g locally approximates f, LIME introduces a loss function $L(f, g, \pi_x)$, where π_x defines the locality around the instance x. The proximity function is defined as:

$$\pi_x(z) = \exp\left(-\frac{D(x,z)^2}{\sigma^2}\right) \tag{3.1}$$

where D is a distance metric such as Euclidean or cosine distance, and σ is a parameter that controls the width of the kernel.

The explanation model g is optimized by solving the following objective:

$$\mathscr{E}(x) = \arg\min_{g \in G} L(f, g, \pi_x) + \Omega(g) \tag{3.2}$$

Here, $\mathscr{E}(x)$ is the explanation corresponding to the instance x. In practice, LIME generates perturbed samples z in the neighborhood of x, evaluates $f(z)$, and assigns weights to these samples using $\pi_x(z)$, giving more importance to those closer to x.

In their experiments, Ribeiro et al. [2016] applied LIME across multiple domains, including product review classification. Their results demonstrated that LIME could achieve over 90% recall when identifying important features for model predictions, significantly outperforming alternatives such as the Parzen window method, which only achieved approximately 60% recall. This highlights LIME's capability to produce explanations that closely reflect the decision-making process of the

underlying model.

3.2 SHAP

SHAP (SHapley Additive exPlanations), introduced by Lundberg and Lee [2017], adopts a game-theoretic perspective to explain how machine learning models make predictions. The SHAP framework is model-agnostic, meaning it can be applied to any model, and it supports both global explanations (for the entire model) and local explanations (for individual predictions). Shapley values are defined as the average marginal contribution of a feature, evaluated across all possible orderings of input features [W. and C., 2023]. Game theory ensures that this formulation yields a unique solution when applied within the framework of additive feature attribution. SHAP provides a unified and principled measure of feature importance, which many other explainability techniques attempt to approximate. It quantifies how much each feature contributes to a prediction by comparing model outputs with and without that feature across all possible subsets.

In their foundational work, Lundberg and Lee [2017] introduce three key properties along with a theorem that together uniquely define the SHAP approach within the class of additive feature attribution methods. All of them are based on the fundamental Definition (**Definition 1**):

$$g(x) = \phi_0 + \sum_{i=1}^{M} \phi_i \cdot x_i' \tag{3.3}$$

where $z' \in [0,1]^M$, M is the number of simplified input features and ϕ_i is the feature attribution of feature i.

This Definition defines feature attribution methods as linear functions of binary variables and matching this definition means that the model attributes an effect ϕ_i to each feature, summing the effects of all feature attributions to approximate the

output $f(x)$ of the original model.

3.2.1 Properties of Additive Feature Attribution Methods

To ensure a unique solution within additive attribution methods, three desirable properties are proposed. Unlike SHAP, many existing explanation methods do not consistently satisfy these properties, which limits their reliability.

Property 1: Local Accuracy

$$f(x) = g(x) = \phi_0 + \sum_{i=1}^{M} \phi_i \cdot x'_i \tag{3.4}$$

The surrogate model $g(x')$ must exactly match the output of the original model $f(x)$ when $x = h_x(x')$, where $\phi_0 = f(h_x(0))$ represents the prediction with all interpretable inputs set to zero.

Property 2: Missingness

If a particular input feature is missing (i.e., $x'_i = 0$), it should receive no attribution:

$$x'_i = 0 \Rightarrow \phi_i = 0 \tag{3.5}$$

Property 3: Consistency

If a model changes in such a way that the marginal contribution of a feature does not decrease (relative to any input subset), the corresponding feature attribution should not decrease either.

Let $f_x(z') = f(h_x(z'))$, and let z'/i denote setting $z'_i = 0$. Then, for any two models f and f', if

$$f'_x(z') - f'_x(z'/\{i\}) \geq f_x(z') - f_x(z'/\{i\}) \tag{3.6}$$

for all $z' \in \{0,1\}^M$, then it must hold that

$$\phi_i(f', x) \geq \phi_i(f, x) \tag{3.7}$$

Satisfying all three of these properties leads directly to the following theorem.

Theorem 1

There exists a single explanation model g that adheres to **Definition 1** and satisfies Properties 1, 2, and 3. This model yields the Shapley values:

$$\phi_i(f, x) = \sum_{z' \subseteq x'} \frac{|z'|!\,(M - |z'| - 1)!}{M!} \left[f_x(z') - f_x(z' \setminus \{i\}) \right] \tag{3.8}$$

Here, $|z'|$ indicates the number of non-zero components in z', and $z' \subseteq x'$ denotes all vectors z' where the non-zero features are a subset of those in x'. Each ϕ_i represents the Shapley value of feature i.

As demonstrated by Young [1985], Shapley values are the only attribution method satisfying axioms that correspond to Local Accuracy and Consistency. This underlines the theoretical strength of SHAP over alternative approaches, many of which fail to meet these standards.

3.2.2 Kernel SHAP (Linear LIME + Shapley Values)

Lundberg and Lee [2017] further introduce Kernel SHAP as a model-agnostic approximation technique designed to estimate Shapley values efficiently. Unlike the full Shapley value computation, which is computationally expensive, Kernel SHAP provides a faster approximation that still respects the foundational properties of Local Accuracy, Missingness, and Consistency. It achieves this by combining the structure of LIME with the theoretical guarantees of Shapley values.

Rather than relying on heuristic parameter selection, the authors provide a principled method to derive these parameters based on the loss function minimized by

LIME. This leads to the formulation of the following theorem:

Theorem 2: Shapley Kernel

Under **Definition 1**, the parameters that ensure the solution to the LIME objective is consistent with Properties 1, 2, and 3 are:

$$\Omega(g) = 0 \tag{3.9}$$

The weighting kernel for the Shapley formulation is given by:

$$\pi_{x'}(z') = \frac{(M-1)}{\binom{M}{|z'|} \cdot |z'| \cdot (M - |z'|)} \tag{3.10}$$

The corresponding loss function is defined as:

$$\mathscr{L}(f, g, \pi_{x'}) = \sum_{z' \in Z} \pi_{x'}(z') \cdot \left(f(h_x(z')) - g(z')\right)^2 \tag{3.11}$$

This formulation allows solving for ϕ_i using the regularization term $\Omega(g)$, the Shapley-specific kernel $\pi_{x'}(z')$, and the squared loss $\mathscr{L}$.

Kernel SHAP's model-agnostic nature makes it widely applicable across various machine learning models, from basic linear regression to complex deep neural networks. This broad compatibility allows practitioners to gain meaningful insight into how individual features contribute to specific model predictions, regardless of the underlying architecture.

3.3 LIME vs SHAP

Both LIME and SHAP are model-agnostic techniques, meaning they can be applied to interpret a wide variety of machine learning models across different domains. Despite this shared flexibility, there are key differences in how they generate explanations.

LIME offers *local fidelity*, meaning it is designed to faithfully approximate the behavior of the original model in the vicinity of a specific prediction. Consequently, LIME produces instance-specific explanations, and the surrogate model it generates may vary significantly from one prediction to another. For every new input, LIME constructs a new interpretable model that locally mimics the behavior of the black-box model.

By contrast, SHAP values aim to provide *global consistency* in feature attributions. SHAP generates a single, unified set of feature importance scores that remain consistent across all predictions. Instead of building a separate explanation model for each instance, SHAP computes the contribution of each feature by averaging its marginal impact across all possible subsets of input features. This approach captures the overall influence of each feature, incorporating all possible interactions and combinations, thus offering a more comprehensive understanding of model behavior.

In summary, LIME is generally faster and easier to implement for generating localized explanations, making it useful for understanding individual predictions. On the other hand, SHAP, while computationally more demanding, delivers more robust and theoretically grounded insights into feature importance across the entire

dataset Dwivedi et al. [2023].

Chapter 4

Methodology, Data and Models

This chapter describes the methodology adopted and how the data used in this thesis is sourced, constructed, imputed and organized. It also reports the Models' Selection process with the four main models adopted for this study. Although the primary objective of the thesis is not to study the machine learning models themselves, it aims at providing a comprehensive explanation of the models used.

4.1 Methodology

I define the excess return of stock i at time $t+1$ as $R_{i,t+1}$, which can be expressed as:

$$R_{i,t+1} = E_t(R_{i,t+1}) + \varepsilon_{i,t+1} \tag{4.1}$$

where the conditional expected return is modeled as:

$$E_t(R_{i,t+1}) = g^*(z_{i,t}) \tag{4.2}$$

Here, $z_{i,t}$ denotes an N-dimensional vector of stock-specific characteristics at time t, which are used as predictor variables. Throughout this thesis, I refer to these characteristics interchangeably as *features* or *factors*. The function $g^*(.)$ represents

the true, but unknown, data-generating process, mapping these inputs to expected returns. Each machine learning model aims at approximating this function differently.

The primary objective is to construct an estimate of $E_t(R_{i,t+1})$ that, when applied to unseen data, closely predicts the realized return $R_{i,t+1}$. All models are trained with the shared goal of minimizing the prediction error, specifically using the *Mean Squared Error* (MSE) as the loss function. MSE is one of the most widely used performance metrics in regression tasks, as it quantifies the average squared difference between predicted and actual outcomes. It is computed as:

$$MSE = \frac{1}{N_{test}} \sum_{i=1}^{N_{test}} (r_{i,t+1} - \hat{r}_{i,t+1})^2 \tag{4.3}$$

where N_{test} denotes the number of observations in the testing sample, and the error is evaluated only on this out-of-sample data.

To further evaluate predictive performance, I compute the out-of-sample coefficient of determination, R^2_{oos}, which provides a panel-level summary of forecasting accuracy across both firms and time periods. It is defined as:

$$R^2_{oos} = 1 - \frac{\sum_{(i,t)\in T3} \left(r_{i,t+1} - \hat{r}_{i,t+1}\right)^2}{\sum_{(i,t)\in T3} r^2_{i,t+1}} \tag{4.4}$$

In this equation, $T3$ refers to the testing subsample, which is strictly reserved for evaluation purposes, meaning the data within $T3$ is never used for model training or hyperparameter tuning. R^2_{oos} summarizes the predictive performance by aggregating forecast errors across all stocks and time points in the out-of-sample period.

4.2 Data

I obtained the data for the daily European equity returns from the *Tidyfinance* library (*YahooFinance* dataset), including all european firms listed in the EuroStoxx50. The

research period spans from January 2010 to January 2025, covering a comprehensive timeframe of 15 years. This choice has been made in order to avoid having possible "noise" in the data panel due to too many recessions and market shocks.

The dataset's dimensions are of 134518 rows (after having handled missing values) and 17 columns, which comprises the target variable and a comprehensive set of stock-level predictive characteristics, selected from all the signals documented in the asset pricing literature, following the studies of Gu et al. [2020]. Apart from the five Fama and French [1993] factors, I constructed the stock-level predictors for the european stocks, including the short term reversal (mom1), stock momentum (mom12), momentum change (chmom), recent maximum return (maxret), long term reversal (mom36m), turnover (turn), dollar volume (dolvol), zero trading days (zerotrade), total return volatility (retvol) and the Risk Free (RF). All of them have been winsorized (except the Fama and French ones) in the data preprocessing phase.

For the purpose of training and testing the models, the dataset is partitioned into three distinct subsets: a training dataset comprising 70% of the total data, a testing sample accounting for the 15% and a validation sample of 15% too. This partitioning strategy ensures that the models can be trained, tested and validated on separate portions of the data, thereby providing a robust evaluation of their performance. Table 4.1 below reports the variables used for the analysis with their major descriptive statistics.

Starting with `ret` (Return, the target variable), we observe a small positive mean of 0.0006, suggesting a slight average gain, coupled with a relatively low standard deviation of 0.0177, implying limited volatility. Returns spanned a wide range, from a minimum of -0.4398 to a maximum of 0.3782, with half of the observations falling between -0.0079 and 0.0091, centered around a median of 0.0005. The `Mkt-RF` (Market Risk Premium) displays a mean of 0.0228, denoting an average excess re-

turn of the market over the risk-free rate, but with a substantially higher standard deviation of 1.033, reflecting greater market volatility compared to individual returns. Its values swung from -12.0 to 8.54, with its interquartile range from -0.49 to 0.56. The factor proxies SMB (Small Minus Big) and HML (High Minus Low) show small positive means of 0.0011 and 0.004 respectively, showing a slight average outperformance of small-cap stocks and value stocks within this dataset. Their standard deviations (0.4026 for SMB and 0.544 for HML) indicate their respective levels of variability. The RF (Risk-Free Rate) has a mean of 0.0057 and a low standard deviation of 0.0078, with a floor at 0.0, suggesting periods of very low or zero risk-free rates. CMA (Conservative Minus Aggressive) shows a slightly negative mean of -0.0009, potentially suggesting that aggressive investment strategies marginally outperformed conservative ones on average, with a standard deviation of 0.3174. Conversely, RMW (Robust Minus Weak) exhibits a positive mean of 0.0074, implying that stocks with robust profitability generally outperformed those with weak profitability, with a standard deviation of 0.3032. Momentum variables reveal interesting patterns: mom1m (Momentum 1-month) has a tiny positive mean (0.0006) and small standard deviation (0.0134). However, mom12m (Momentum 12-month) presents a substantial negative mean of -0.9696, suggesting a significant average reversal effect over a year, reinforced by a high standard deviation of 2.8295. Its values ranged from -7.6579 to 5.8106. In contrast, mom36m (Momentum 36-month) displays a positive mean of 0.2678, indicating a longer-term momentum effect, with a standard deviation of 0.3378. chmom (Characteristic Momentum) stands out with an extremely high standard deviation of 5.4481 despite a near-zero mean (0.0009), suggesting exceptionally large fluctuations. Finally, maxret (Maximum Return) has a mean of 0.032 and a standard deviation of 0.0149, detailing the average of the highest returns observed. ret_lagged (Lagged Return) mirrors ret with a similar mean of 0.0006 and standard deviation of 0.0138. dolvol (Dollar Volume) showcases incredibly

large figures. However, as it is also reported on the Table 4.1, the variable has been expressed in percentage (%) to ease the visualization. With a mean of 1.3261 and a standard deviation of 8.6178, it is highlighted how immense is the variability in trading activity. Its range extends from a minimum of 1.9417 to a maximum of 3.3629. The variable `zero_volume_day` is uniformly 0.0 across all its statistics, confirming no instances of `zero_volume_day` in the dataset. Lastly, `retvol` (Return Volatility), with a mean of 0.0153 and standard deviation of 0.0058, quantifies the average daily return volatility within a relatively tight range. The `turn` variable is identical to the `dolvol`. As reported in Appendix B, there is a very high correlation between those two factors. However, It has been decided to not eliminate one of them from the analysis for consistency purposes.

Table 4.1: Descriptive Statistics of 17 variables used for the thesis (comprehensive of the target variable). With a count of 134,518 observations, the dataset is robust. From `ret` to `ret_lagged` the values are reported in percentage to ease the visualization.

Variable	Count	Mean	Std Dev	Min	Median	Max
ret (%)	134518	0.06	0.0177	−0.3898	0.05	0.3782
Mkt-RF (%)	134518	2.28	1.033	−12.0	6.0	8.54
SMB (%)	134518	0.11	0.0426	−3.33	0.0	1.85
HML (%)	134518	0.4	0.544	−3.04	−2.0	4.38
RF (%)	134518	0.57	0.0078	0.0	0.0	0.02
RMW (%)	134518	0.74	0.3032	−1.73	2.0	2.67
CMA (%)	134518	−0.09	0.3174	−1.73	−1.0	1.31
mom1m (%)	134518	0.06	0.0134	−0.0258	0.06	0.0269
mom12m	134518	−0.9696	2.8295	−7.6579	−100.0	5.8106
chmom (%)	134518	0.09	5.4481	−12.7685	0.25	12.7324
maxret (%)	134518	3.2	0.0149	0.0135	2.85	0.0696
ret_lagged (%)	134518	0.06	0.0138	−0.0264	0.06	0.0277
mom36m	134518	0.2678	0.3378	−0.3022	0.2421	0.9836
turn	134518	1.3261	8.6178	1.9417	1.1281	3.3629
dolvol	134518	1.3261	8.6178	1.9417	1.1281	3.3629
zero_volume_day	134518	0.0	0.0	0.0	0.0	0.0
retvol	134518	0.0153	0.0058	0.0075	0.0141	0.029

In the dataset, there are numerous missing observations, which pose a significant challenge for conducting machine learning. Handling missing values is an essential

data preprocessing procedure to address the absence of information within a dataset. This involves systematically identifying and replacing missing data points with appropriate values to maintain the dataset's completeness and integrity or to just drop the entries with missing values in order to avoid bias.

There are several options to address this problem. However, I decided to drop the rows with missing values to avoid any sort of bias that could have arisen with an imputation technique like the K-Nearest Neighbors (KNN) imputation technique or the cross-sectional median value. This choice has been driven by the common practice in financial fields to avoid imputation of values in order to maintain the robustness of the dataset and to avoid bias.

4.3 Model Selection

In this thesis, I employed four distinct machine learning models. Among them, two are considered black-box models, namely XGBoost and Neural Networks, while the remaining two models, Decision Tree and K-Nearest Neighbors (KNN), are more interpretable, yet still involve a certain level of complexity.

XGBoost was chosen due to its superior predictive accuracy and strong empirical performance in asset pricing contexts. Neural Networks, being widely adopted and recognized as a prototypical black-box model, are included for further investigation into their internal decision-making processes. The Decision Tree and K-Nearest Neighbors models were selected to facilitate comparative analysis and to offer varying degrees of interpretability alongside predictive performance.

Explainable Artificial Intelligence (XAI) techniques are valuable not only for black-box models but also for more *transparent* ones. First, even relatively simple

models can exhibit complex relationships between features and the target variable, particularly when interactions or non-linear effects are present. XAI methods help in disentangling and visualizing these complexities. Second, while some models offer inherent measures of feature importance, XAI approaches can yield a more refined and granular understanding, especially in revealing how combinations of features contribute to specific outcomes. Third, tools such as LIME provide local interpretability, offering explanations at the individual prediction level. This is crucial in situations where understanding the rationale behind a specific prediction is more important than grasping the model's overall behavior. Fourth, for stakeholders lacking a technical background, the insights generated through XAI tools tend to be more intuitive and accessible than traditional model summaries or raw statistical outputs.

In summary, XAI not only enhances interpretability for complex models but also serves as a valuable tool for validating and communicating insights across a range of machine learning algorithms, including even simple linear models, by making their inner workings more transparent and understandable.

4.3.1 XGBoost

XGBoost (eXtreme Gradient Boosting) is a scalable and distributed implementation of the Gradient Boosted Decision Trees (GBDT) algorithm. As an ensemble learning method, it combines the outputs of multiple decision trees to generate a final prediction. Unlike traditional boosting techniques, XGBoost builds trees sequentially, where each successive tree aims at correcting the residual errors of the previous ones. This is achieved by minimizing a specified loss function using gradient descent. Additionally, XGBoost incorporates a regularization component in its objective function, which penalizes model complexity and helps mitigate overfitting. This regularization feature distinguishes XGBoost from conventional gradient

boosting implementations.

XGBoost presents several advantages, including exceptional predictive performance and computational efficiency. Its regularization mechanisms enhance generalization capabilities, enabling the model to effectively capture complex, non-linear relationships within the data. Nevertheless, it should be noted that hyperparameter tuning in XGBoost can be computationally demanding, particularly with large-scale datasets.

XGBoost is often classified as a "black-box" model due to its complexity. First, being an ensemble of potentially hundreds of decision trees, the final prediction results from an aggregated output, making it challenging to trace the influence of individual features. Second, its ability to model sophisticated non-linear feature interactions, while advantageous for accuracy, adds to the model's opacity. To overcome these interpretability challenges, Explainable AI (XAI) tools such as LIME and SHAP are employed. These methods provide insights into feature importance and help elucidate the underlying mechanisms driving the model's predictions.

4.3.2 Decision Tree

A Decision Tree is a supervised machine learning algorithm that can be used for both classification and regression tasks. It models decisions and their possible consequences by partitioning the input space into distinct regions based on feature values. In a decision tree, each internal node represents a test on a feature, each branch corresponds to the outcome of that test, and each leaf node holds a prediction, either a class label (in classification) or a numerical value (in regression).

The construction of the tree is performed recursively by selecting, at each node, the feature and corresponding threshold that result in the best possible data split

according to a given impurity criterion (such as Gini impurity, entropy, or variance reduction). The general mathematical representation of predictions in a decision tree can be formalized as:

$$\hat{y} = f(\text{feature\_split}) \tag{4.5}$$

for classification tasks, and

$$\hat{y} = \text{avg}(\text{target\_values\_in\_leaf\_node}) \tag{4.6}$$

for regression tasks, where *feature_split* represents the rule used to divide the data at each decision node based on a particular feature and threshold. The term *target_values_in_leaf_node* refers to the actual target values associated with the data points that fall into a given leaf node, and their average is used to produce the predicted output in regression settings.

4.3.3 K-Nearest Neighbor

The K-Nearest Neighbors (KNN) algorithm is a supervised machine learning method applicable to both classification and regression problems. KNN operates under the assumption that observations that are close in feature space tend to have similar target values or belong to the same class. It makes predictions by identifying the K nearest data points to a given input and then aggregating their outcomes: through majority voting for classification or averaging for regression.

For classification tasks, KNN assigns a class based on the most frequent class label among the K nearest neighbors, typically using distance metrics such as Euclidean distance. For regression tasks, it predicts a value by computing the average of the numerical target values of these neighbors.

The mathematical formulation of the KNN algorithm is as follows. For classification:

$$\hat{Y} = \arg\max_{y} \left(\sum_{i \in N} I(Y_i = y) \right) \tag{4.7}$$

and for regression:

$$\hat{Y} = \frac{1}{K} \sum_{i \in N} Y_i \tag{4.8}$$

where N denotes the set of the K-nearest neighbors, $\hat{Y}$ is the predicted output, and Y_i represents the target value of the i-th neighbor. In the classification formula, $I(Y_i = y)$ is an indicator function that equals 1 if the class label of the i-th neighbor is equal to y, and 0 otherwise. This process ensures that the predicted class corresponds to the most common label among the neighbors.

While KNN is simple and intuitive, it has certain limitations. Its performance is highly sensitive to the choice of K and the distance metric used. Furthermore, the algorithm can be computationally expensive, especially on large datasets, since it requires storing the entire training set and calculating distances to all training points at prediction time. KNN performs best when the data is well-distributed and class imbalance is adequately handled.

4.3.4 Neural Networks

Neural Networks (NNs) are a subset of machine learning models inspired by the structure and function of the human brain. NNs consist of interconnected nodes (or neurons) organized in layers: an input layer, one or more hidden layers, and an output layer. Each connection between nodes is associated with a weight, and each node applies an activation function to the weighted sum of its inputs to produce an

output.

A basic feedforward neural network can be mathematically represented as:

$$\hat{Y} = f\left(W^{(2)} \cdot f\left(W^{(1)} \cdot X + b^{(1)}\right) + b^{(2)}\right) \tag{4.9}$$

where $f(\cdot)$ denotes the activation function, $\hat{Y}$ is the predicted output, X represents the input data, $W^{(1)}$ and $W^{(2)}$ are the weight matrices for the hidden and output layers respectively, and $b^{(1)}$ and $b^{(2)}$ are the corresponding bias vectors.

In this thesis, following the study of Demirbaga and Xu [2024], it has been implement a neural network with five layers comprising 32, 16, 8, 4, and 2 neurons, respectively. This architecture is relatively more complex than shallower neural networks, making it a suitable black-box model for interpretation using Explainable Artificial Intelligence (XAI) techniques. The layered structure and non-linear transformations increase its representational power while simultaneously making its internal decision process less transparent, highlighting the necessity for XAI tools like SHAP and LIME.

Chapter 5

Results

This chapter reports the models' interpretations using the XAI algorithms: LIME and SHAP. Firstly, the performance of base machine learning models across the

dataset is presented and accordingly ranked. LIME algorithm is then employed to provide local interpretability for one single prediction per model, showing which factors influence the most, and with what magnitude, the prediction of a certain stock's return. Additionally, the most impactful stock-level characteristics on a global level are identified using SHAP values. Finally, the performance results after utilizing insights from XAI, in particular the SHAP values, are compared to the performance prior to using XAI insights.

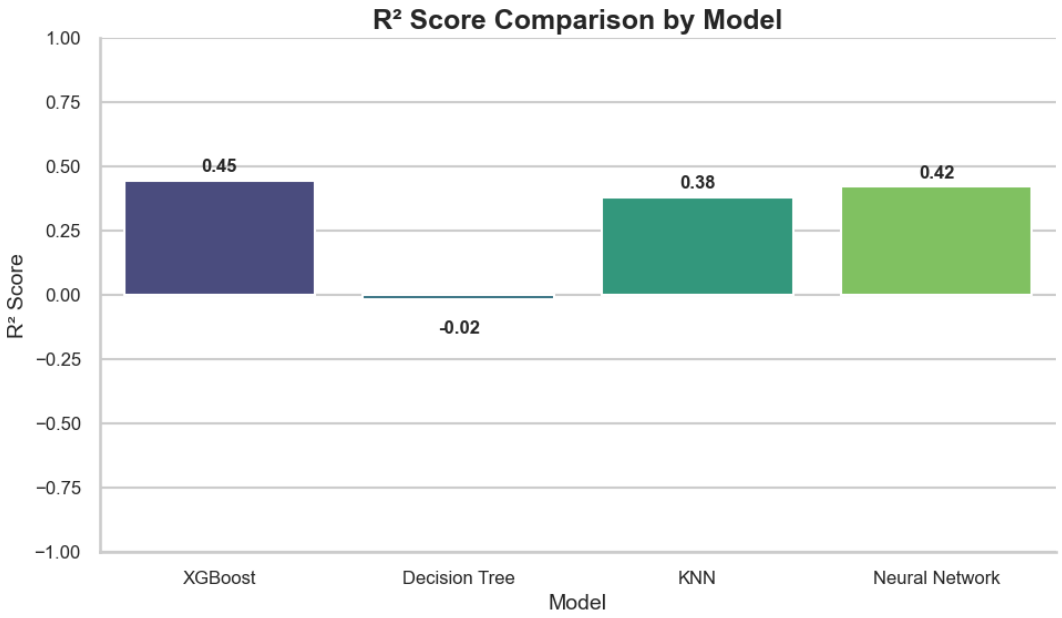
5.1 R^2 Comparison between Machine Learning Models

Figure 5.1, presents the out-of-sample R^2 values for each machine learning model across all the dataset before integrating XAI techniques. The graph shows that XGBoost is the top-performing model, achieving an out-of-sample R^2 of 0.45. Following, there are Neural Networks, K-Nearest Neighbor and Decision Tree. However, the largest out-of-sample R^2 value for Decision tree is -0.017, which is far below the optimal R^2 threshold. This could be due to the relative simplicity of the model, making it ineffective for such complex non-linear predictions.

5.2 Insights from LIME Analysis

One key advantage of employing Local Interpretable Model-agnostic Explanations (LIME) lies in its ability to focus on specific instances of the model's predictions. Each explanation produced by LIME pertains to a single observation within the dataset, in this context each instance corresponds to a particular date.

Figure 5.1: R^2 for Machine Learning Models. The best score is gained by XGBoost with an R^2 of 0.45. NN and KNN obtained respectively 0.42 and 0.38. Decision Tree is the worst performer with a $R^2 < 0$



To facilitate the interpretation of the visualizations presented in this section, LIME explanation figures utilize color-coded bars to represent the contribution of individual features to the model's prediction. Features shown with orange bars indicate a positive contribution to the predicted excess stock return. This implies that the presence or higher magnitude of these features increases the forecasted return for the subsequent day. In contrast, features represented by blue bars denote a negative contribution, suggesting that their presence or larger values reduce the predicted excess return.

The symbol adjacent to each feature in the central section of the figure indicates the direction in which that feature influences the prediction, whether it contributes to an increase or decrease in the predicted return. For instance, consider a case where the feature `Mkt-RF` displays a positive contribution value and is colored orange. This implies that when `Mkt-RF` exceeds 0.52, it is associated with a higher predicted stock return. The length of each bar reflects the magnitude of the feature's impact. For

example, the bars corresponding to `Mkt-RF` and `maxret` reveal that `Mkt-RF`, with a contribution of 0.01, has a more substantial influence than `maxret`, which has a contribution of 0.00.

The horizontal bar on the left side of the figure represents the full range of possible predicted excess returns the model could output, as determined by the scope of LIME. The color gradient across this bar indicates whether the predicted value is closer to the lower or upper end of this range.

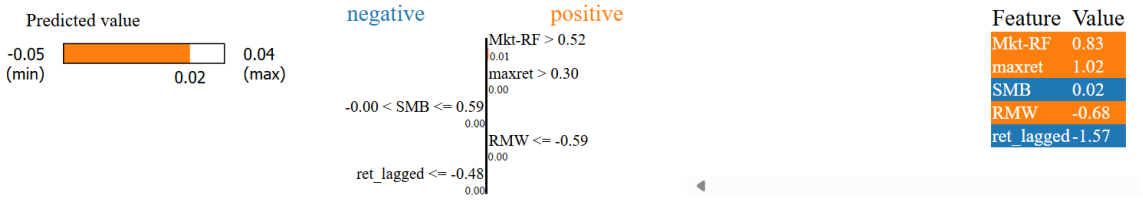
The right side of the LIME figure displays the individual features and their corresponding values that influenced the specific prediction. The *Feature* column lists the relevant stock-level factors used by the model to generate the prediction for excess returns at time t_1. The *Value* column shows the actual value of each feature at time t_0 for the specific stock under consideration. The color coding remains consistent with the central section, representing whether the feature had a positive or negative effect on the prediction.

The LIME explanation for the XGBoost model is considered the most significant due to XGBoost's superior predictive performance relative to the other models. This visualization highlights the most impactful firm-level characteristics that contributed substantially to the prediction of stock returns.

Firstly, the LIME explanation for the model's prediction for XGBoost can be found in Figure 5.2.

As a result from Figure 5.2, `Mkt-RF` emerges as the most important characteristic among the positively influencing factors. `Mkt-RF` is a feature that comes from Fama and French [1993] and it is related to the overall market risk premium. It measures the additional return investors expect for taking on the risk of investing in the stock

Figure 5.2: LIME explanation for Model's prediction for XGBoost. The most influential factor is `Mkt-RF` with values > 0.52, while the least important is `ret_lagged`<=-0.48. The only two negatively influencing factors are `SMB` with values -0.00< `SMB` <=0.59 and `ret_lagged`



market compared to a risk-free asset. Specifically, the condition `Mkt-RF` > 0.52 implies that when the market risk premium is greater than 0.52, it positively contributes to the stock return in the following day. This can be interpreted as investors considering high values of risk premium as higher chances for companies to yield returns.

Additionally, the maximum return `maxret` feature ranks as the second most impactful factor. Under the condition `maxret` > 0.30, this factor positively contributes to the following day's return, with an impact level of approximately 0.01. This measure quantifies the maximum single day return a stock had during a recent period, usually the past month.

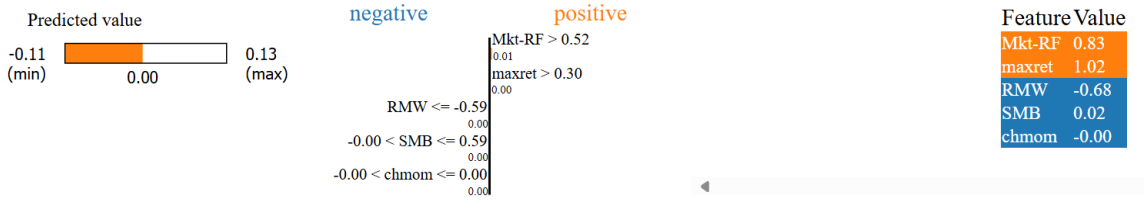
Furthermore, the `SMB` factor influences the return's prediction in a positive way with the condition that -0.00< `SMB` <=0.59.This measure quantifies the maximum single day return a stock had during a recent period, usually the past month. This could be interpreted with the idea that companies with relatively low size premium have a higher chance of yielding a high return.

Then, the Robust Minus Weak `RMW` factor influences positively the prediction with the condition `RMW` <=-0.59. While the lagged return `ret_lagged` influences negatively the prediction of returns having the relationship of `ret_lagged` <=-0.48, but with a lower magnitude than the others.

Secondly, I turn to the LIME explanation for the Decision Tree model in Fig-

ure 5.3

Figure 5.3: LIME explanation for Model's prediction for Decision Tree. The only difference here in respect to the XGBoost explanation are the chmom and RMW factors. The first with relation -0.00< chmom<=0.00 and negatively influencing the prediction and RMW, this time, influencing negatively.

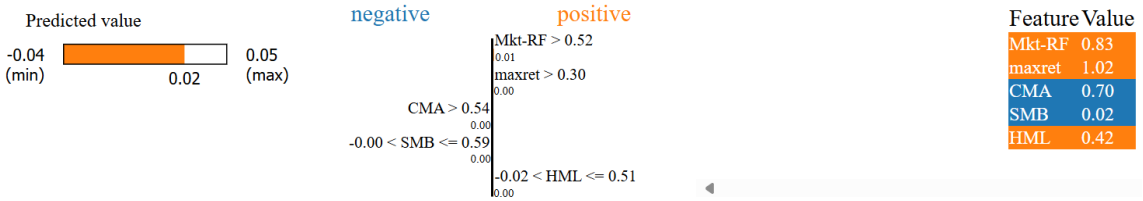


As mentioned earlier, Mkt-RF remains an important factor here, exhibiting the same directional impact as observed with XGBoost. The same is to say for maxret and SMB, that influence the prediction of the Decision Tree model with the same magnitude than XGBoost. Two differences are identifiable on the last two factors, that for the Decision Tree explanation are: Robust Minus Weak RMW, still coming from Fama and French [1993]; and Momentum Change chmom. RMW compares the returns of firms with high (robust) operating profitability to those with low (weak) operating profitability. It influences negatively on the prediction with the condition: *RMW* <=-0.59. A negative value associated with this measure suggest that Robust firms are outperforming Weak firms. Regarding the chmom factor, only a positive value (chmom > 0.00) will positively affect the prediction. Specifically *Momentum* itself refers to the tendency of securities that have performed well in the past to continue performing well in the near future, and vice versa for those that have performed poorly. Indeed the Momentum Change looks at the change in this Momentum.

Thirdly, I move forward to the LIME explanation for the KNN model.

Figure 5.4 highlights the similarity of factors in comparison with the other models and that they all are influencing positively the prediction of the return except for the CMA and SMB. Indeed the difference in the prediction for the KNN model is the presence of the variables CMA and HML, with relations CMA > 0.54 and -0.02<HML<=0.51.

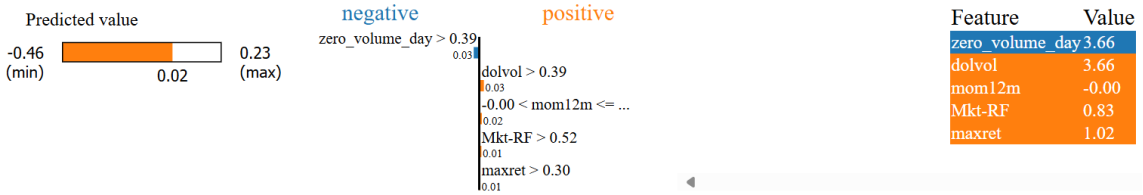
Figure 5.4: LIME explanation for Model's prediction for KNN. Practically identical to the XGB's one, except from the the two variables *HML* and *CMA*



Another difference is the predicted value's range that here is between -0.04 and 0.05 instead of -0.05 and 0.04 of the XGB's, while for the Decision Tree we have a completely different range values: -0.11 and 0.13.

Lastly, in the LIME explanation for the Neural Network model shown in Figure 5.5, it is registered a change in variables' importance.

Figure 5.5: LIME explanation for Model's prediction for Neural Networks. It shows a very different situation from the others. `zero_volume_day` is the most influential variable with a magnitude of 0.03 and negative influence on the prediction. Then `dolvol` and `mom12m` are influencing positively followed by the two factors that in the previous models were in poleposition.



Indeed, the most impactful factors for the prediction of the Neural Network are surprisingly `zero_volume_day`, influencing negatively the prediction, that it assumes values > 0.39. It is interesting to note that its contribution to the model is 0.03, a relatively high value in respect to the other models' values. Then the `dolvol` factor, along with `mom12m`, `Mkt-FR` and `maxrret` present the same values. Also here, there is a change in magnitudes, indeed for the positively influencing factor, the contribution magnitude reaches a maximum value of 0.03 and a minimum of 0.01 (relatively higher that the ones in the other models' explanations).

With LIME explanations for various machine learning models, I can identify the complex relationships between factors and the target variable, which is the one-

day-ahead return in this thesis. These explanations not only reveal the positive or negative relationships between dependent and independent variables, but also the specific conditions for the factor values. This comprehensive and detailed figure allows you to identify the most significant factors and understand the reasons behind the movements in returns at specific times of interest. This capability is one of the most valuable advantages of the LIME technique.

This chapter highlighted some of the most important factors from the LIME results. I selected a few and provided an economic rationale for their impacts, but there are many other factors that remain to be explained. These relationships can be easily explored in the LIME figures. I have demonstrated how to interpret and explain the LIME result figures.

Overall, the LIME results showed relatively many matches across different models leading to a general convergence between variable's importance, Although maintaining some differences.

5.3 Insights from SHAP Analysis

In this section, SHAP (SHapley Additive exPlanations) approach is used to analyze the important predictive features and their relationships with future stock returns. The primary distinction between SHAP and LIME lies in the scope of their interpretability: SHAP provides a *global* explanation of the model, whereas LIME offers *local*, instance-specific interpretations. SHAP is applied only to the XGBoost and Decision Tree models in this analysis. Due to the high computational demands, SHAP was not implemented for the KNN and Neural Network models.

Two types of SHAP visualizations are presented: Summary Plots and Dependence Plots. The first aims at showing the ranking of the variables based on their importance for the prediction. The second plot shows the effect of a single feature in the

model (the most impactful feature).

Figure 5.6 and Figure 5.7 display the SHAP summary plots for the XGBoost and Decision Tree models, respectively. SHAP summary plots facilitate the interpretation of predictions by visualizing the average impact of each feature on the model's outputs. Each horizontal bar represents a feature, and the color gradient reflects the actual value of that feature across instances (blue indicates low values, while red represents high values).

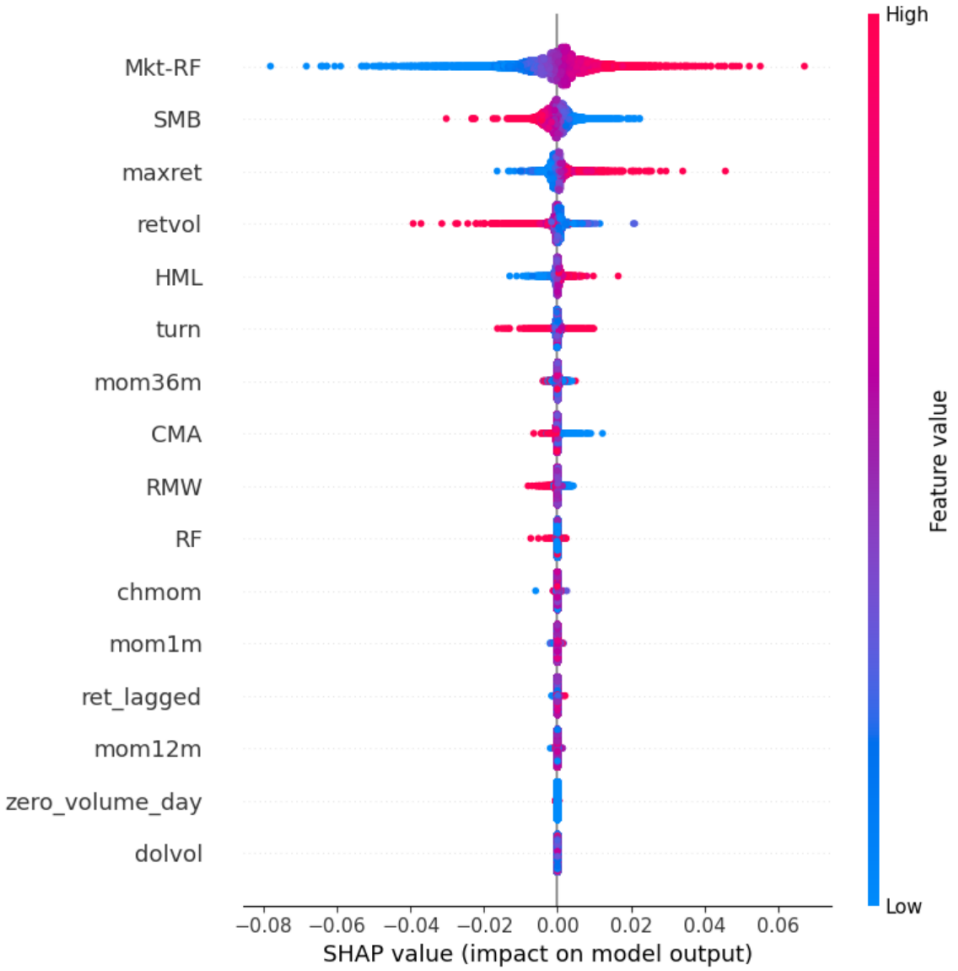
The length of each bar indicates the magnitude of the feature's contribution to the model's predictions. Each point represents a SHAP value for a particular feature in a specific observation, producing a 'swarm' effect across the plot. This distributional visualization reveals both the importance and the directionality of each feature's influence across the entire dataset.

The X-axis represents SHAP values, which quantify the contribution of each feature to the deviation of a model's prediction from the baseline (i.e., the mean prediction). For example, if the model's average predicted return is 2%, then a positive SHAP value indicates that the feature contributes to increasing this prediction, while a negative value decreases it. In the context of asset pricing, features with blue coloration are associated with lower predicted returns, whereas features with red coloration are linked to higher predicted returns.

Figure 5.6 presents the summary plot for the XGBoost model. Many of the impactful variables in this SHAP analysis correspond to those identified in the earlier LIME analysis.

The most influential factor is the market risk premium, `Mkt-RF`. Higher values of `Mkt-RF` tend to increase future returns, and lower values similarly influence predic-

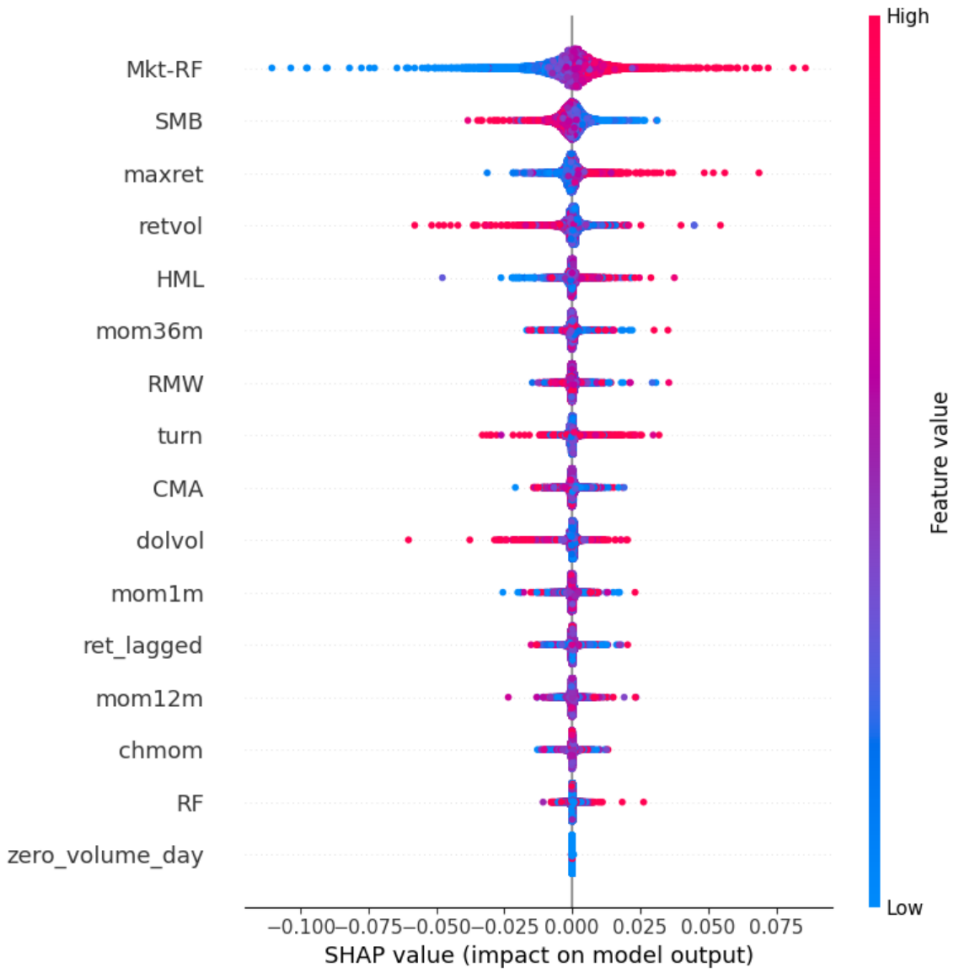
Figure 5.6: SHAP Summary Plot for XGBoost Model Interpretation. The majority of variables shown are consistent with those highlighted by the LIME explanation, indicating not only local but also global importance. The relative contribution magnitudes are also comparable to those found in the LIME interpretation.



tions in the opposite direction, consistent with the findings from LIME. Other factors such as `SMB` and `maxret`, also show the same importance and the same relation as in LIME analysis. This suggests that these variables are important not only at the local level but also in a global context. Nevertheless, some variables, like `retvol`, are not present in the LIME analysis, showing a slight divergence from the two explanations.

Figure 5.7 shows the summary plot for the Decision Tree model. Again, SHAP values largely reflect the importance patterns observed in the LIME interpretations.

Figure 5.7: SHAP Summary Plot for Decision Tree Model Interpretation. The global feature importance slightly aligns with the LIME explanation, although with some divergence in the contributing factors.

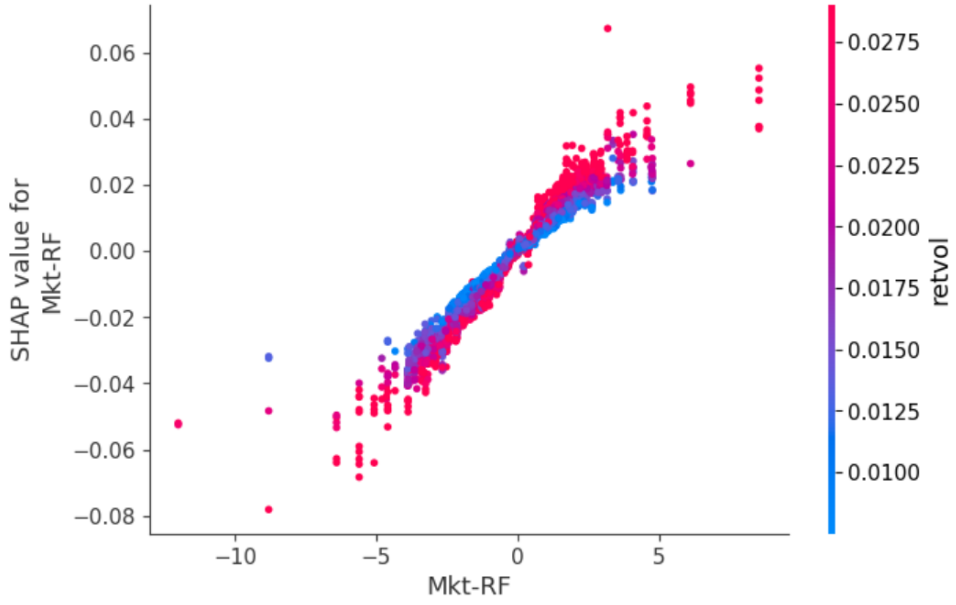


As in the XGBoost model, the most impactful feature is `Mkt-RF`, though its influence is slightly more sparsed. Interestingly, factors such `RMW` and `chmom` are not ranked in a high position here, contrary to the LIME explanation that finds them very impactful. This indicates that while these variables may be significant for some individual predictions (local importance), they don't hold considerable weight in the model's overall behavior (global importance).

Figure 5.8 presents the Dependence Plot, a particular visualization that shows how the value of a given factor affects the model's output highlighting also a potential interaction with another factor. On the *y* axis, the SHAP value for the factor is plotted

while on the x axis the real value of the factor is reported. On the right side there is a scale of values that shows the potential interaction with another factor. Its value is represented by the colour. In this case the most impactful feature (`Mkt-RF`) is taken as the variable to inspect. The potential interactor turns out to be `retvol`.

Figure 5.8: SHAP Dependence Plot for XGBoost on the most relevant factor `Mkt-RF` with insights on a potential interaction with `retvol`. `Mkt-RF` presents a positive impact on the prediction due to the slope of the graph. However, this is also due to the interaction of the `retvol` that shows an inversely proportional relation with the `Mkt-RF`.

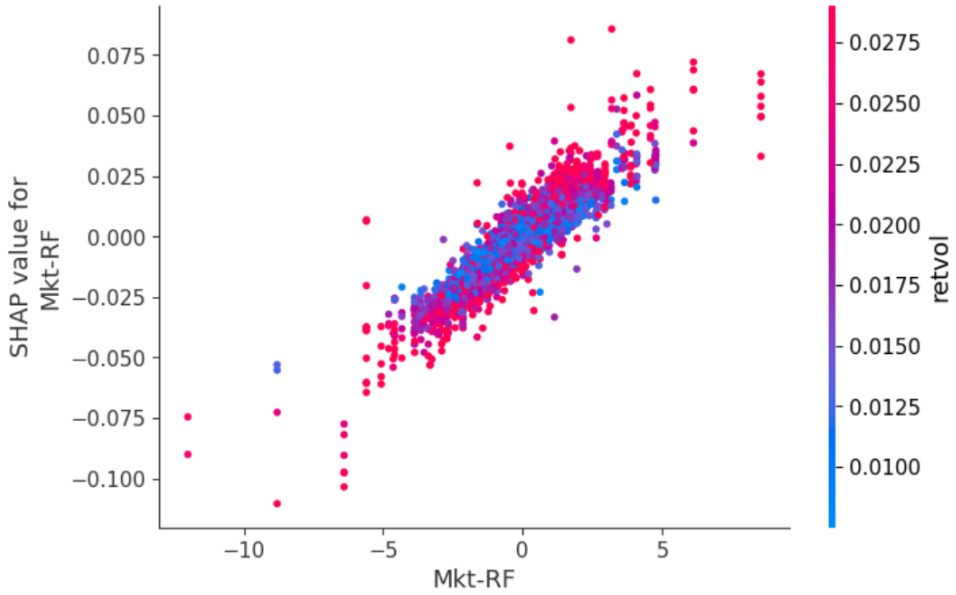


It is clearly shown that the bigger the value for the `Mkt-RF`, the more impacts the model's output. This insight is gathered through the fact that a bigger real value yields a bigger SHAP value in the analysis. However, by analyzing also the color in the graph, it is shown that the interaction with `retvol` is very powerful. Indeed, for higher values of `Mkt-RF`, with a high value of `retvol`, the first factor has a better impact on the output. The opposite case is true: for a lower value of `Mkt-RF`, with a low value of `retvol`, the first factor has better impact on the output.

Figure 5.9 shows the Dependence Plot for the Decision Tree analysis.

The same analysis we did for the XGB case, could be proposed here. Indeed the

Figure 5.9: SHAP Dependence Plot for Decision Tree on the most relevant factor `Mkt-RF` with insights on a potential interaction with `retvol`. As in the XGB case `Mkt-RF`'s real value is positively correlated with the respective SHAP values. The difference here is the slightly more sparseness of the points.



positive gradient of the curve shows the positive correlation between the real values and the SHAP values of the `Mkt-RF`. Furthermore, the interaction with `retvol` is the same as in the previous analysis. The only difference is the sparseness of the points in the graph, potentially signifying that in the Decision Tree analysis there is not the same strength in the interaction.

5.4 Performance Improvement After Using XAI

Explainable AI isn't just about understanding the inner-workings of machine learning models, it also helps enhance their performance. When models are more interpretable, they often perform better. In the asset management industry, this translates into higher investment yields.

After having identified the most impactful stock-level features using XAI techniques, the insights gained are used to fine-tune the two machine learning models analyzed with SHAP (XGBoost and Decision Tree). This section first outlines the

approach taken to train and evaluate the models using those XAI findings. It then shows how prediction accuracy improved as a result.

5.4.1 Model Performance Comparison

In order to establish the importance of Explainable AI (XAI) in improving model performance, we evaluate three different versions of each machine learning model using standard performance metrics.

First, we obtain the base performance for each model version, referred to as *Base*. This version relies solely on the initial training dataset, without any hyperparameter tuning or XAI guidance. The *Base* metric serves as a foundational benchmark, reflecting how well the model explains the variability in stock returns using its default settings. This allows for subsequent refinements to be evaluated relative to this initial performance.

Second, we assess model performance after applying standard hyperparameter tuning techniques, denoted as *Tuned*. Hyperparameter tuning is a conventional method used to improve predictive performance by optimizing parameters that are not directly learned during training. By comparing the *Base* and *Tuned* versions, we quantify the benefits of optimization strategies that do not incorporate any XAI insights.

Finally, we introduce a third version, referred to as *Tuned (XAI)*, which incorporates adaptive tuning guided by explainable AI. In this approach, the SHAP (SHapley Additive exPlanations) method is employed to determine global feature importance. SHAP values are selected as the benchmark for XAI results because they provide consistent, model-wide explanations that reflect the influence of each feature across all predictions. Furthermore, the results obtained using SHAP were largely consistent with those derived from LIME.

The top 7 features with the highest SHAP values, those with the most substantial and consistent contributions to model predictions, are selected. The model is then retrained on the training dataset taking in consideration these features and an "adaptive tuning" procedure is applied. This procedure adjusts tuning parameters specifically for the top 7 SHAP-identified features while leaving less important features relatively unchanged.

To perform this "adaptive tuning", we employ the GridSearch method along with the *Mean Absolute SHAP Importance*, differently as proposed in recent literature Demirbaga and Xu [2024] and Leander Weber [2023]. GridSearch is a powerful fine-tuning technique very effective for a vast plethora of models. It is used here to tune the hyperparameters basing itself on a predefined grid with all the possible values for hyperparameters.

The final *Tuned (XAI)* model thus represents a version of the machine learning model that incorporates insights from XAI directly into its training pipeline. If the performance of the *Tuned (XAI)* model surpasses both the *Base* and *Tuned* versions, this provides empirical evidence that XAI not only enhances model transparency but also contributes to improved predictive accuracy.

Model performance is evaluated using two standard metrics: the coefficient of determination (R^2) and the Mean Squared Error (MSE). These metrics quantify the model's ability to explain variance and its average predictive error, respectively.

This methodology offers a comprehensive and systematic framework for model development. By integrating explainability (via SHAP) with adaptive tuning (via GridSearch), it ensures that the final models are both accurate and interpretable. This aligns with the growing demand for transparent and accountable AI applications,

particularly in the asset management industry.

5.4.2 Performance Results

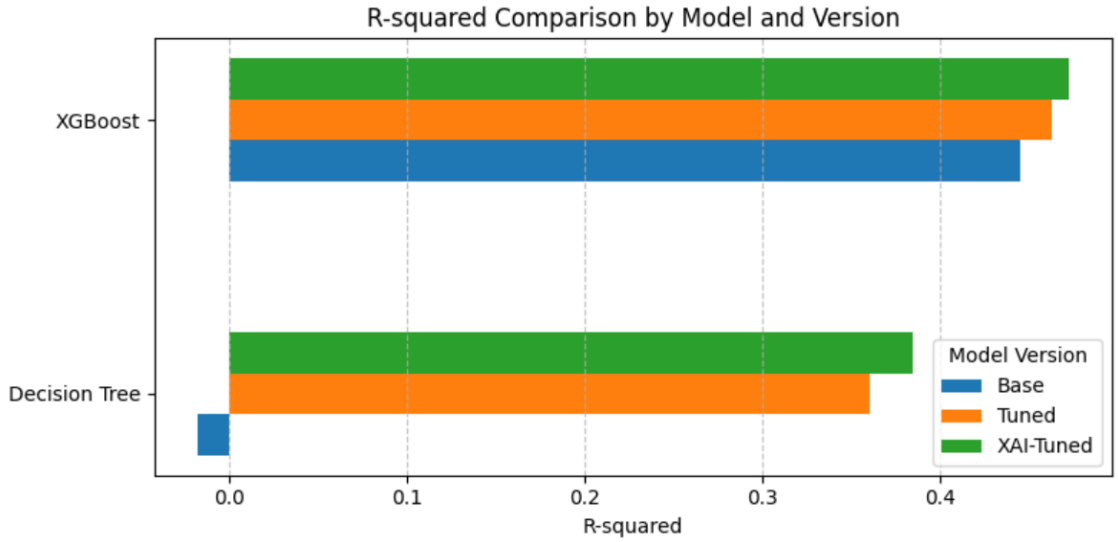
Figure 5.10 reports the out-of-sample coefficient of determination (R^2) for the three versions of each model. For the XGBoost model, the *Base* version achieves an R^2 of 0.445. Following hyperparameter tuning, the *Tuned* version improves slightly to 0.463. However, the *Tuned (XAI)* model results in an even higher R^2 of 0.472, overperforming both the baseline and the *Tuned* model.

A similar trend is observed for the Decision Tree model. The *Base* model exhibits a negative out-of-sample R^2 of -0.017, indicating that the model performs worse than a simple mean predictor. After standard tuning, the *Tuned* version achieves a substantial improvement, with an R^2 of 0.360. Finally, the *Tuned (XAI)* version slightly improves, with an R^2 of 0.384.

These results underscore not only the effectiveness of standard hyperparameter tuning in improving model performance, as evidenced by the consistent R^2 gains from the *Base* to the *Tuned* models, but also the effectiveness of the Fine-tuning with SHAP insights. Indeed, the *Tuned (XAI)* versions yield further improvements in R^2. In fact, they exhibit enhanced predictive performance, suggesting that while XAI enhances interpretability, it also improves generalization or predictive accuracy when used for adaptive tuning.

Another performance metric used to assess the models is the Mean Squared Error (MSE), which calculates the average squared difference between predicted and actual values. Figure 5.11 presents the MSE values for all model versions. As expected, both models' *Base* versions exhibit higher MSE values compared to their

Figure 5.10: Out-of-sample R^2 comparison across model versions. If standard tuning improves performance, XAI-guided tuning enhances further the performance for both models R^2, particularly for the Decision Tree model.

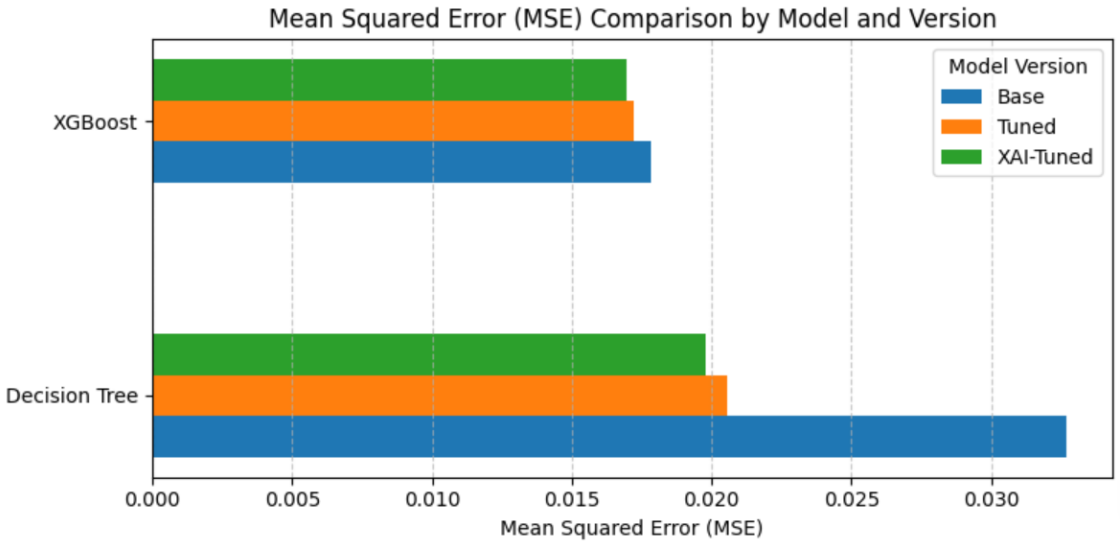


Tuned counterparts.

The *Tuned (XAI)* versions once again enhance the performance, showing the lowest MSE among all model versions. This confirms the previous finding from the R^2 analysis, using XAI for adaptive feature tuning reduces predictive error. These results highlight that model interpretability and accuracy likely align.

Among the two machine learning models, XGBoost achieves the lowest MSE, reaching a minimum of 0.000172 in the *Tuned* version. In contrast, the *Tuned (XAI)* version of XGBoost records a slightly lower MSE of 0.000169. These results reaffirm that being the explainability critical for understanding model behavior, it also guarantees enhanced predictive performance, particularly when directly integrated into the tuning process.

Figure 5.11: Mean Squared Error (MSE) comparison across model versions. The *Tuned* versions exhibit a low error, however the *Tuned (XAI)* versions overperform the standard tuning further reducing the MSE.



Chapter 6

Conclusions

This thesis aimed at demonstrating the potential of the implementation of XAI in machine learning models for asset pricing. Of the two main problems I wanted to address with this study, both have been successfully proved by empirical work: the enhancement in interpretability and in predictive performance. Indeed, through the use of LIME and SHAP, Explainable AI identifies the key factors influencing

the most the prediction of the stock return, enhancing the interpretability of such complex models. On the other hand, the approach for the performance improvement of the models in the hyperparameter tuning phase adopting SHAP explanations, has been proved to be effective. Using GridSearch, the models are fine tuned on the validation set basing on the top 7 factors. Then, the models are retrained and tested on the full dataset maintaining the hyperparameters selected before.

XAI confirms itself as an important tool for a future where machine learning models have to be transparent and interpretable by human. This opens a new research on various fields, especially in finance. As an example, machine learning can be used to incorporate lots of signals of different data type and from a variety of different data sources. XAI can be used to clarify and explain the importance of each signal in order for the financial operation to be transparent in front of stakeholders and clients. Furthermore XAI could also be implement in the Mergers and Acquisitions field to understand the dynamics behind the process. The broad application of XAI stands to revolutionize understanding and decision-making processes across different areas of finance.

Bibliography

Jonathan Berk. A critique of size-related anomalies. *Review of Financial Studies*, 8 (2):275–286, 1995. doi: 10.1093/rfs/8.2.275.

Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. *arXiv preprint arXiv:1603.02754*, 2016. URL https://arxiv.org/abs/1603.02754.

Y. Peter Chung and Michael J. Schill. Asset pricing when returns are nonnormal: Fama-french factors versus higher-order systematic comoments. *Journal of Business*, 79(2):923–940, 2006. doi: 10.1086/499143.

Umit Demirbaga and Yue Xu. Empirical asset pricing using explainable artificial intelligence. *SSRN Electronic Journal*, 2024. doi: 10.2139/ssrn.4680571. URL https://ssrn.com/abstract=4680571.

Rudresh Dwivedi, Devam Dave, Het Naik, Smiti Singhal, Omer Rana, Pankesh Patel, Bin Qian, Zhenyu Wen, Tejal Shah, Graham Morgan, and Rajiv Ranjan. Explainable ai (xai): Core ideas, techniques, and solutions. *ACM Computing Surveys*, 55(9):835, 2023. doi: 10.1145/3561048. URL https://doi.org/10.1145/3561048.

Eugene F. Fama and Kenneth R. French. Common risk factors in the returns on stocks and bonds. *Journal of Financial Economics*, 33(1):3–56, 1993.

Eugene F. Fama and Kenneth R. French. The capital asset pricing model: Theory and evidence. *Journal of Economic Perspectives*, 18(3):25–46, 2004.

Eugene F. Fama and James D. MacBeth. Risk, return, and equilibrium: Empirical tests. *Journal of Political Economy*, 81(3):607–636, 1973.

Stefano Giglio and Dacheng Xiu. Asset pricing with omitted factors. *Journal of Political Economy*, 129(7):1947–1990, 2021. doi: 10.1086/714090. URL https://doi.org/10.1086/714090.

Shihao Gu, Bryan Kelly, and Dacheng Xiu. Empirical asset pricing via machine learning. *The Review of Financial Studies*, 33(5):2223–2273, 2020. doi: 10.1093/rfs/hhaa009. URL https://doi.org/10.1093/rfs/hhaa009.

Campbell R. Harvey, Yan Liu, and Heqing Zhu. ... and the cross-section of expected returns. *Review of Financial Studies*, 29(1):5–68, 2016. doi: 10.1093/rfs/hhv059.

J. B. Heaton, N. G. Polson, and J. H. Witte. Deep learning in finance. *arXiv preprint arXiv:1602.06561*, 2016. URL https://arxiv.org/abs/1602.06561.

Narasimhan Jegadeesh and Sheridan Titman. Returns to buying winners and selling losers: Implications for stock market efficiency. *Journal of Finance*, 48(1):65–91, 1993. doi: 10.1111/j.1540-6261.1993.tb04702.x.

Bryan T. Kelly, Seth Pruitt, and Yinan Su. Machine learning in asset pricing. *Journal of Financial Economics*, 134(3):501–524, 2019. doi: 10.1016/j.jfineco.2019.03.015. URL https://doi.org/10.1016/j.jfineco.2019.03.015.

Alexander Binder Wojciech Samek Leander Weber, Sebastian Lapuschkin. Opportunities and challenges of xai-based model improvement. *Information Fusion*, X:Y–Z, 2023. doi: 10.1016/j.inffus.2022.02.023. URL https://doi.org/10.1016/j.inffus.2022.02.023.

Scott M. Lundberg and Su-In Lee. A unified approach to interpreting model predictions. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper/2017/file/8a20a8621978632d76c43dfd28b67767-Paper.pdf.

R. David McLean and Jeffrey Pontiff. Does academic research destroy stock return predictability? *Journal of Finance*, 71(1):5–32, 2016. doi: 10.1111/jofi.12365.

Benjamin Moritz and Tom Zimmermann. Tree-based conditional portfolio sorts: The relation between past and future stock returns. *SSRN Electronic Journal*, 2016. doi: 10.2139/ssrn.2740751. URL https://ssrn.com/abstract=2740751.

Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. Why should i trust you? explaining the predictions of any classifier. *arXiv preprint arXiv:1602.04938*, 2016.

Stephen A. Ross. The arbitrage theory of capital asset pricing. *Journal of Economic Theory*, 13(3):341–360, 1976.

William F. Sharpe. Capital asset prices: A theory of market equilibrium under conditions of risk. *Journal of Finance*, 19(3):425–442, 1964.

Saeed W. and Omlin C. Explainable ai (xai): A systematic meta-survey of current challenges and future research directions. *Knowledge-Based Systems*, 2023. URL https://www.sciencedirect.com/science/article/pii/S0950705123000230.

Junyi Ye, Bhaskar Goswami, Jingyi Gu, Ajim Uddin, and Guiling Wang. From factor models to deep learning: Machine learning in reshaping empirical asset pricing. *arXiv preprint arXiv:2403.06779*, 2024. URL https://arxiv.org/abs/2403.06779.

H. P. Young. Monotonic solutions of cooperative games. *International Journal of Game Theory*, 14:65–72, 1985.

Appendix

A. Variables' Construction

In this thesis, a relatively small set of variables is employed to capture both firm-specific characteristics and systematic risk factors essential for modeling and predicting stock returns. These variables are constructed based on widely accepted methodologies in empirical asset pricing. In particular, by following the work of Gu et al. [2020] and Demirbaga and Xu [2024]. The dependent variable, ret, is defined as the daily return of a stock, calculated as the percentage change in closing price from one day to the next. This serves as the primary measure of asset performance. The formula for daily return is:

$$ret_t = \frac{P_t - P_{t-1}}{P_{t-1}} \tag{6.1}$$

Where P_t is the closing price on day t and $P_(t-1)$ is the closing price on day t-1. The Mkt-RF variable represents the excess return of the market over the risk-free rate and is a foundational element of models such as CAPM and the Fama-French multifactor frameworks.

$$Mkt - RF_t = R_{m,t} - R_{f,t} \tag{6.2}$$

Where $R_(m,t)$ is the return on the market portfolio at time t and $R_(f,t)$ is the risk-free rate at time t. SMB (Small Minus Big) and HML (High Minus Low) are size and value factors respectively, capturing cross-sectional variation in returns related to firm size and book-to-market ratios. SMB is typically constructed by taking the average return of small-cap portfolios minus the average return of big-cap portfolios. HML is typically constructed by taking the average return of high book-to-market portfolios (value stocks) minus the average return of low book-to-market portfolios (growth stocks). Profitability (RMW) and investment (CMA) factors extend the Fama-French model into its five-factor form. RMW (Robust Minus Weak) is constructed by taking the average return of robust profitability portfolios minus the average return of weak profitability portfolios. CMA (Conservative Minus Aggressive) is constructed by taking the average return of conservative investment portfolios minus the average return of aggressive investment portfolios. Momentum-related measures include mom1m, mom12m, and mom36m, computed as cumulative returns over 1, 12, and 36 months respectively, based on rolling return windows. For a cumulative return over N months:

$$momNm = \prod_{i=1}^{N}(1 + ret_i) - 1 \tag{6.3}$$

Where ret_i is the monthly return for month i.

chmom, or change in momentum, is derived by differencing momentum values between periods. For example, the change in 1-month momentum:

$$chmom_t = mom1m_t - mom1m_{t-1} \tag{6.4}$$

maxret is defined as the maximum single-day return in a prior window (typically a month) and it is often associated with "lottery-like" stock behavior. For a 20-day (approx. 1 month) window ending on day t:

$$maxret_t = max(ret_{t-19}, ret_{t-18}, ...ret_t) \tag{6.5}$$

ret_lagged refers to the one-day lag of the return variable, included to control for potential autocorrelation.

$$ret\_lagged_t = ret_{t-1} \tag{6.6}$$

dolvol is the product of stock price and trading volume, measuring liquidity in dollar terms.

$$dolvol_t = P_t * V_t \tag{6.7}$$

Where P_t is the stock price on day t and V_t is the trading volume on day t. zero_volume_day is a binary indicator equal to one when a stock has no trading volume on a given day, highlighting illiquidity.

$$zero\_volume\_day_t = \begin{cases} 1 \\ 0 \end{cases} \tag{6.8}$$

Where the factor has the value of 1 when V_t = 0 and 0 when $V_t > 0$, where V_t is the trading volume on day t.

Finally, retvol is the standard deviation of past returns over a short rolling window, serving as a proxy for realized volatility. For a rolling window of N days:

$$retvol_t = \sqrt{\frac{1}{N-1}} \sum_{i=0}^{N-1} 1(ret_{t-i} - ret_a)^2 \tag{6.9}$$

Where ret_a is the average return over the N-day window.

B. Correlation Among Factors

In this section the correlations among the factors and the top-7-factors used by different machine learning models are shown using heatmaps. The heatmap is a valuable tool to understand the relationships between features.

Figure 6.1 visualizes the relationship between the top 7 features used in the XGBoost model. Strong red squares along the diagonal show that each feature is perfectly correlated with itself. Off-diagonal squares that are red or blue indicate significant positive or negative correlation between different features. Darker shades of red suggest strong positive correlations, meaning that as one feature increases, the other tends to increase as well. Conversely, darker shades of blue indicate strong negative correlations, meaning that as one feature increases, the other tends to decrease. Figure 6.2, Figure 6.3, Figure 6.4 report the same heatmaps but for the other models. Figure 6.5 shows the same graph for all the 17 factors used in the analysis.

Figure 6.1: Correlation Heatmap of the Top 7 Factors: XGBoost

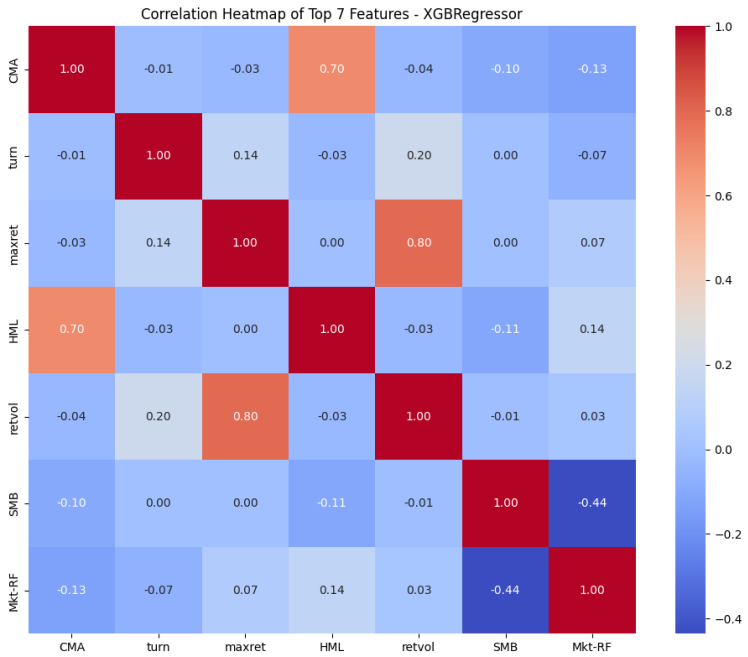


Figure 6.2: Correlation Heatmap of the Top 7 Factors: Decision Tree

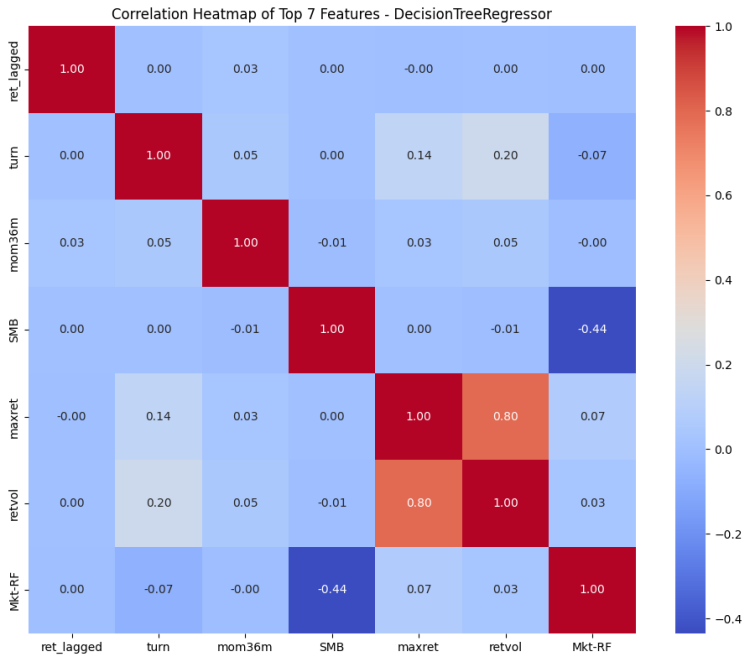


Figure 6.3: Correlation Heatmap of the Top 7 Factors: K-Nearest Neighbor

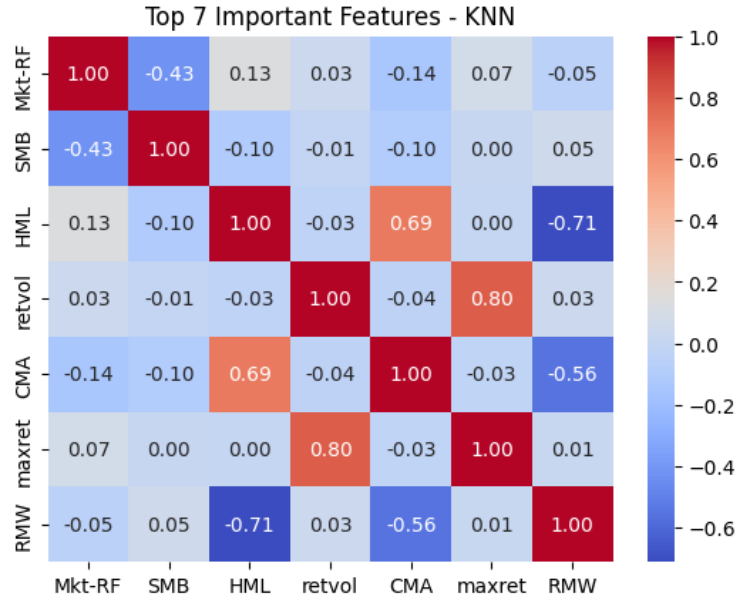


Figure 6.4: Correlation Heatmap of the Top 7 Factors: Neural Network

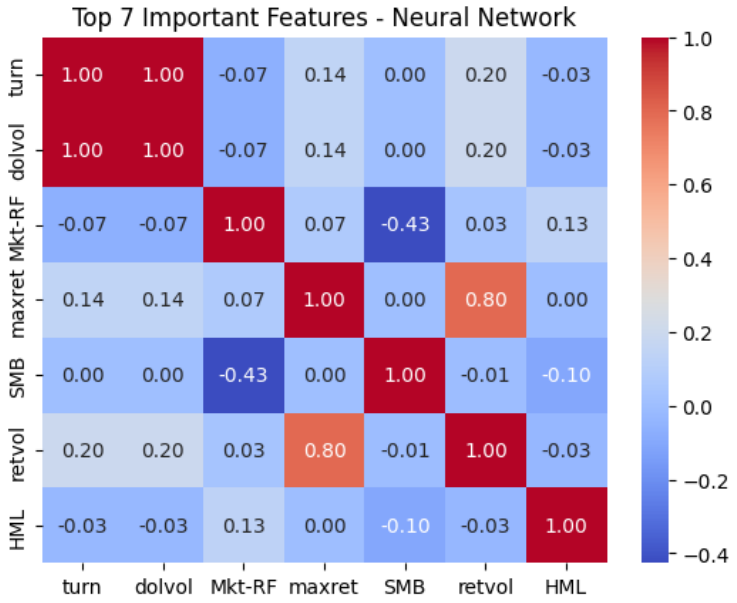


Figure 6.5: Correlation Heatmap of the 17 Factors (including the target variable)