Empirical Asset Pricing Using Explainable Artificial Intelligence

Umit Demirbaga[‡]

Yue Xu[§]

This version: April 23, 2024

[‡]Department of Medicine, University of Cambridge, United Kingdom; Department of Computer Engineering, Bartin University, Turkey. Email: udemirbaga@bartin.edu.tr.

[§]Department of Finance, Business School, Durham University. Email: xu.yue@durham.ac.uk.

Empirical Asset Pricing Using Explainable Artificial Intelligence

Abstract

This paper applies explainable artificial intelligence in empirical asset pricing to explain the reasoning behind stock return predictions made by various complex machine learning models. We use two state-of-the-art explainable AI methods, LIME and SHAP. Our findings indicate that the primary drivers in our model predictions are stock-level characteristics such as momentum, 52-week high, and volatility. We demonstrate large improvements in predictive power and investment performance when incorporating insights from explainable AI into model refinement, surpassing the performance of machine learning models without such explanations. In addition, we use a variety of data visualization methods within explainable AI to help institutional investors interactively communicate the inner workings of these models to stakeholders.

Keywords: Explainable Artificial Intelligence; Asset Pricing; Factor Zoo; Machine Learning; Investment Performance

JEL Classification: C52; C55; C58; G12; G17

This version: April 23, 2024.

I. Introduction

The 2023 Invesco Global Systematic Investing Study (Invesco (2023)) reveals that half of systematic investors have already incorporated artificial intelligence (AI) into their investment process, and many believe AI will reshape the investment landscape. However, this report highlights a major concern regarding AI adoption by institutional investors: the difficulty in explaining AI decision mechanisms to stakeholders. This is because many AI models are black-box models, and investors hardly trust an investment decision made by a model without understanding its mechanism. Therefore, there is an increasing need to interpret complicated AI models in the asset management industry.

Explainable AI (XAI), an emerging field in computer science, aims to bridge the gap between complex machine learning predictions and the need for economically meaningful interpretations. While machine learning models can capture complex non-linear relationships and vast predictor spaces, their practical use remains limited without understanding the economic rationale behind these patterns. Explainable AI offers deeper and more intuitive insights, aligning empirical findings with theory and propelling finance research into a new era of data-driven, yet economically coherent exploration. Explainable AI also enhances predictive accuracy after humans better understand the interpretations of black-box models.

We address a traditional problem in empirical asset pricing: determining which stock-level characteristics are most important for predicting stock returns. The finance literature identifies characteristics such as momentum, volatility, and profitability that can explain stock returns (Jegadeesh and Titman (1993); Ang et al. (2006); Fama and French (2006)). However, the question of which characteristics are most important in return prediction remains under debate (commonly referred to as the 'factor zoo'). The literature applies various methodologies to identify key drivers of return prediction but often faces challenges. Simple models often struggle with high-dimensional data, exhibit limited feature interactions, are prone to overfitting, and generally yield low predictive power. In contrast, complex

machine learning models, while powerful, tend to lack interpretability.

We use a machine learning approach combined with explainable AI to tackle both challenges. This allows us to 1) understand the most impactful characteristics from these complex models and 2) improve the machine learning performance of stock return prediction. Our process can also be directly applied by institutional investors to explain their investment decisions to stakeholders.

We use two state-of-the-art explainable AI techniques: local interpretable model-agnostic explanations (LIME) by Ribeiro et al. (2016) and SHapley Additive exPlanations (SHAP) by Lundberg and Lee (2017). They provide insights on the most significant charactersitics and their impacts on the prediction result. The main idea of LIME is to explain model predictions by approximating the complex model *locally* with a simpler, more interpretable model, typically achieved by minimizing a loss function. The idea of SHAP is to unify existing methods to have a unique solution through a game-theoretic approach. SHAP values provide *global* explanations consistent across all predictions. Both LIME and SHAP are model-agnostic that can be used by any model interpretations.

This paper uses a rich set of 209 firm-level predictive characteristics as features for cross-sectional stock return prediction, covering a span of 65 years from 1957 to 2022. We use these comprehensive characteristics to predict the next month's excess return of each stock. Four different types of machine learning models are selected, including XGBoost, decision tree, K-nearest neighbors, and neural networks. Our analysis shows that XGBoost performs the best for return prediction, achieving an out-of-sample R^2 of 0.761. This performance is followed by the decision tree, K-nearest neighbors, and finally neural networks.

To understand the inner working of these machine learning models, we first use LIME interpretations to understand the most influential characteristics in these models for a particular prediction. XGBoost and the decision tree highlight similar emphasis on features such as momentum, 52-week highs, and maximum returns to predict the stock's next month

return.[1] This underlines the importance of market trends and historical performance in shaping return predictions.

On the other hand, in LIME interpretations, K-nearest neighbors and neural network models shift their focus towards firm fundamentals as their main predictors. These models prioritize characteristics like earnings surprise and dividend initiations, as well as other signals like change in capital investment and intangible return.

We then analyze the insights gained from the SHAP analysis of the XGBoost model. According to SHAP results, the most significant factors in explaining stock returns include maximum returns, 52-week highs, 12-month and 6-month momentum, realized volatility, idiosyncratic risk, zero trades, and industry return of big firms.[2] Overall, the XGBoost model analyzed by SHAP primarily considers stock market information such as past price and volatility for its most impactful characteristics.

In addition, we uncover the complex and non-linear relationships in asset pricing through the SHAP summary plot, which offers unique insights through its 'swarm' of points, each representing the distribution of SHAP values for a specific feature across all data points, revealing patterns not evident in other methods. For example, we observe that the relationship between 52-week highs and predicted stock return is asymmetrical. While most positive 52-week highs values are indicative of a positive, albeit moderate, stock return, some predictions of high 52-week highs are associated with extremely high positive predicted stock returns.

We also quantify the relative importance of all characteristics in model predictions by using mean absolute SHAP values. In the XGBoost model, the top three features - maximum returns, 52-week highs, and 12-month momentum - demonstrate similarly significant impacts on stock returns, each with mean absolute SHAP values slightly above 0.03, cumulatively exceeding 0.09 for all three combined. In contrast, the aggregate mean absolute SHAP value of the other 200 signals is around 0.06, suggesting that the combined effect of these top three

[1] Momentum, 52-week highs, and maximum returns are proposed by Jegadeesh and Titman (1993); George and Hwang (2004); Bali et al. (2011).

[2] Idiosyncratic risk, zero trades, and industry return of big firms are proposed by Ali et al. (2003); Liu (2006); Hou (2007).

signals is about 1.5 times greater than that of the 200 other features.

We also analyze the SHAP force plot, which is used to understand the contribution of each feature to a specific prediction. This plot reveals that maximum return is the most influential characteristic. In addition, we can clearly observe that the 12-month momentum contributes to an increase in the prediction, while the 6-month momentum decreases the predicted return. SHAP force plots are interactive, visually showing the influence of signals on the output and enabling researchers to interactively examine how changing a signal's value affects the model's prediction.

Momentum is a key driver to predict cross-sectional returns, as identified by XGBoost in our explainable AI analysis. However, there is a contradiction in the 6-month momentum results by XGBoost compared to conventional momentum literature. Higher 6-month momentum, as modeled, are associated with lower predicted stock returns, contrasting with traditional findings by Jegadeesh and Titman (1993). On the other hand, 12-month momentum is consistent with the traditional findings. This implies a mid-term mean reversion or rebound effect in stock prices. It is important to realize that trained black-box models may not always align with established knowledge, but reveal more complicated non-linear relationships.

In the decision tree model, the top three characteristics - 52-week highs, 12-month momentum, and 6-month momentum - align with those identified in XGBoost. A slight difference from XGBoost is that the decision tree prioritizes 12-month momentum (with mean absolute SHAP values of 0.05) as the most impactful factor, as opposed to 52-week highs (0.03) and maximum returns (0.01) for predicting cross-sectional stock returns. Regarding the influence of the top characteristics, 12-month momentum and realized volatility positively predict stock returns. In summary, the decision tree model mainly selects features based on market information and past performance.

For K-nearest neighbors and neural networks, SHAP analysis reveals that they share similar most significant characteristics, which primarily related to to firm fundamentals.

These include short interest, revenue growth rank, firm age, and brand capital investment.[3]

Among the top characteristics identified by SHAP value in any model, there is large overlap with those found in the LIME analysis. Although there are minor differences, both SHAP and LIME results point to the same conclusion: the key drivers of cross-sectional returns are constructed by stock past performance in XGBoost and decision tree; while the key drivers are related to firm fundamentals identified in K-nearest neighbors and neural networks.

Our findings indicate that past performance and volatility related characteristics identified by these models are more compelling as primary indicators for predicting monthly stock returns for three reasons. Firstly, XGBoost and the decision tree have superior predictive power, as evidenced by their higher R^2 and MSE values, and also deliver higher Sharpe Ratios compared to the other two models. Secondly, the sparser distribution of dots in the SHAP summary plots suggests that the characteristics identified by K-nearest neighbors and neural network models do not predict stock returns as effectively as those identified in XGBoost and the decision tree. Lastly, the top characteristics recognized by K-nearest neighbors and neural networks have less impact on the models to predict stock returns compared to the sum of other features.

Different machine learning models select different important characteristics to predict stock returns. It is impossible for different machine learning black-box models to be consistent to find the most important characteristics. However, based on all evidence, we conclude that past performance are the key determinants of cross-sectional stock returns.

Our second goal of this paper is to improve predictive performance in asset pricing by integrating insights from explainable AI. We propose a novel methodology to improve our complex model performance. We first identify the most impactful characteristics in each model using explainable AI. Subsequently, we train each machine learning model and adaptively select tuning parameters based on the explainable AI results using the Sequential Least

[3]Short interest, revenue growth rank, firm age, and brand capital investment are proposed by Dechow (2001); Lakonishok et al. (1994); BARRY and STARKS (1984); Belo et al. (2014).

SQuares Programming (SLSQP) algorithm. The SLSQP algorithm iteratively adjusts the parameters to find the optimal solution within the defined constraints, and then iteratively refines the feature weights. Once the optimal set of parameters is found, we retrain each model with the entire training dataset using these parameters. Finally, we evaluate the final model's performance on a test dataset.

There is a clear and consistent pattern of performance improvement after applying explainable AI across four machine learning algorithms. In particular, the use of explainable AI insights leads to a significant increase in out-of-sample R^2 and a decrease in MSE. The XGBoost model exhibits the highest predictive power, achieving an R^2 of 0.815 in the explainable AI-tuned model. There is an approximately 2.3% increase in R^2 after using explainable AI in the XGBoost model. The models ranking second, third, and fourth in terms of predictive power are the decision tree, K-nearest neighbors, and neural networks, respectively. These models show R^2 increases of 7%, 35%, and 48% after the application of explainable AI.

We also assess the investment performance of these models using the Sharpe Ratio. Consistently, across all models, the Sharpe Ratio increases after integrating explainable AI into the machine learning model development. The XGBoost model, in particular, achieves the highest Sharpe Ratio, reaching 1.67, followed by the decision tree with a Sharpe Ratio of 1.635 when applying explainable AI. The increase in the Sharpe Ratio is approximately 3.2% for XGBoost and 2.32% for the decision tree. This finding reveals the benefits that explainable AI-integrated machine learning models can bring to asset management.

In addition, we address the time efficiency of these models, focusing on the machine learning testing times. Across all models, there is a remarkable reduction in testing times after applying explainable AI, with the decision tree showing more than a 50% decrease in testing time. This improvement in time efficiency suggests that faster machine learning algorithms, refined by explainable AI, can facilitate more timely and effective investment decisions.

Our paper makes a methodological contribution to identifying key characteristics in cross-sectional asset pricing. A significant strand of the literature, often referred to as the 'factor zoo,' is dedicated to developing frameworks to eliminate abundant and less impactful factors (Harvey et al. (2016); Kelly et al. (2019); Kozak et al. (2020)). For example, Harvey and Liu (2021) suggest a step-wise model selection approach for factor selection. Lettau and Pelger (2020) introduce a novel method that generalizes principal component analysis by incorporating a penalty for pricing errors in expected returns. Feng et al. (2020) propose a new methodology that considers model selection errors when assessing the contribution of new factors to asset pricing. Freyberger et al. (2020) use the adaptive group LASSO to select characteristics and estimate how chosen characteristics affect expected returns nonparametrically. In contrast to these approaches, our paper uses a different methodology, applying complex machine learning models and subsequently using state-of-the-art explainable AI methods to understand the most important stock-level characteristics. This approach leverages the black-box models to capture sophisticated relationships among factors, and also provides insights into the inner workings of these models so that we can use these insights to eliminate less impactful signals.

Our work also contributes to the performance improvement using machine learning models in asset pricing. There is growing research applying machine learning to asset pricing. For example, Gu et al. (2020) demonstrate that machine learning models significantly outperform simple linear models in terms of statistics and investment performance. Chen et al. (2023) use deep neural networks in asset pricing, showing that their method beats all benchmark approaches. In bond return predictability, Bianchi et al. (2021) find higher statistical and economic gains through machine learning methods. Similarly, Bali et al. (2021) apply machine learning to option return predictability and observe higher performance over simpler models. Our paper distinguishes itself by not only using machine learning but also incorporating insights from explainable AI to refine these models. This innovative approach enables us to achieve higher predictive accuracy and superior investment performance in stock return

prediction, demonstrating the added value of explainable AI in return predictability.

Our final contribution is to guide practitioners to interactively explain black-box interpretations to stakeholders. A common challenge in asset management is the difficulty of presenting complex model results in an easily understood and clear format to stakeholders. To address this challenge, our paper uses multiple data visualizations to showcase prediction results, while thoroughly explaining the details of each visualization. This approach not only simplifies the communication of complex data but also ensures that the results are accessible and interpretable to a broad audience, improving trust and confidence among stakeholders.

Different visualization serves different goals. In particular, LIME visualization highlights how each feature influences a specific outcome in the model; SHAP Summary Plot reveals the distribution and direction of each feature's impact on model predictions; SHAP Bar Plot of Mean Absolute Value highlights the relative importance of influential features in the model; SHAP Bar Plot offers insights into, in which direction, specific features affect the model's predictions; SHAP Force Plots illustrates how each feature shifts the prediction from the base value to the final outcome.

Finally, explainable AI has the potential to revolutionize finance by providing deeper and more transparent interpretations of complex machine learning models. One limitation of machine learning models that hinders widespread use in finance academia is the lack of interpretability. Explainable AI addresses this limitation, enabling future researchers to tackle unsolved research questions by uncovering intricate relationships in various finance domains. For example, there are some recent studies on applying explainable AI in fintech and risk management. Hadji Misheva et al. (2021) uses explainable AI to explain credit risk. Bussmann et al. (2021) use explainable AI in the fintech risk management to explain loan decisions. Ariza-Garzon et al. (2020) use SHAP values to assess several models for granting scoring in P2P lending.

The structure of this paper is as follows. Section II outlines the mechanisms of two state-of-the-art explainable AI methods, namely LIME and SHAP. Section III provides an

overview of our data, methodology, and selection of four machine learning models. Section IV analyzes the interpretations of LIME and SHAP techniques across these four models. Section V presents methodology and discusses results on the statistics and investment performance improvement of the four models after using XAI. Section VI concludes the paper. Additional details and supplementary information are in the Appendix.

II. Explainable Artificial Intelligence

In this section, we outline the mechanisms of two widely applied explainable AI methods - LIME and SHAP.

II.1. LIME

Local interpretable model-agnostic explanations (LIME) by Ribeiro et al. (2016) is a technique designed to explain the predictions of any classifier in a way that is both interpretable to humans and faithful to the model's decision-making process. The essence of LIME is to construct a transparent and interpretable simple model that captures the complex model's behavior around a specific prediction.

To achieve this, LIME first defines an interpretable representation for the data. For text, this could be a binary vector denoting the presence or absence of words, while for images, it might be a binary vector indicating the presence of super-pixels. Let $x \in \mathbb{R}^d$ represent the original data and $x' \in \{0,1\}^d$ be its interpretable representation. Let f denote an original complex model (e.g., neural networks), and let g denote the explanation model. Let $\Omega(g)$ denote the complexity (as opposed to interpretability) of the explanation model g.

The goal is to find an explanation model $g \in G$ from a family of interpretable models G (like linear models, decision trees, etc.), which acts over the interpretable representation. The model g should locally approximate the prediction of the original model $f : \mathbb{R}^d \to \mathbb{R}$, where $f(x)$ could represent the probability of x belonging to a certain class.

To ensure that the explanation model g is a good local approximation of the complex

original model f, LIME introduces a locality-aware loss function $L(f, g, \pi_x)$, where π_x is a proximity measure defining the locality around the instance x. The fidelity of g to f is maximized by minimizing this loss function, with the trade-off between fidelity and interpretability managed by a complexity measure $\Omega(g)$, which could be the depth of a decision tree or the number of non-zero weights in a linear model.

The optimization problem LIME solves is as follows:

$$\xi(x) = \arg\min_{g \in G} L(f, g, \pi_x) + \Omega(g), \tag{1}$$

where $\xi(x)$ is the explanation for instance x. To approximate the behavior of f, LIME generates new samples in the vicinity of x and evaluates the predictions of f on these samples. These samples are perturbed instances z, drawn from a distribution that prioritizes closeness to x, weighted by the proximity measure π_x.

For the explanation model g, LIME often employs sparse linear models, which are interpretable due to their simplicity. It uses locally weighted square loss as L, where $\pi_x(z) = \exp\left(-\frac{D(x,z)^2}{\sigma^2}\right)$ is an exponential kernel. D is a distance metric like cosine or Euclidean distance, and σ is a kernel width parameter.

To find such a model, LIME uses techniques like Lasso regression to select features and then fits a weighted linear model where the weights are determined by the proximity measure π_x. This methodology allows LIME to provide local explanations that are accurate around the prediction of interest.

Ribeiro et al. (2016) uses LIME in several applications including classification for product reviews. They show that LIME consistently provides over 90% recall for the models. Other methods such as parzen window method (Baehrens et al. (2010)) only achieves around 60% recalls. This demonstrates that LIME explanations are accurate representations of the models' decision-making processes.

II.2. SHAP

SHAP (SHapley Additive exPlanations), proposed by Lundberg and Lee (2017), uses a game-theoretic approach to explain the inner workings of any machine learning models. SHAP values are model-agnostic that can be used by any machine learning model interpretations. SHAP defines the class of additive feature attribution methods, that unifies existing methods. The game theory results guarantee a unique solution apply to above methods. The SHAP value by Lundberg and Lee (2017) is a unified measure of feature importance that the existing methods approximate.

Let f denote the original prediction model and g the explanation model (the same notation in LIME). Local methods such as LIME are designed to explain the prediction $f(x)$ based on a single input x, employing simplified inputs x' that map to the original inputs through a mapping function $x = h_x(x')$. These methods aim to ensure $g(z') \approx f(h_x(z'))$ whenever $z' \approx x'$.

Definition 1 (Additive feature attribution methods): An explanation model is defined as a linear function of binary variables:

$$g(z') = \alpha_0 + \sum_{i=1}^{N} \alpha_i z'_i, \tag{2}$$

where $z' \in \{0,1\}^N$, N is the number of simplified input features, and α_i is the feature attribution of feature i.

Explanation models that match this definition attribute an effect ϕ_i to each feature, summing the effects of all feature attributions to approximate the output $f(x)$ of the original model. The six existing methods that match Definition 1 include LIME, DeepLIFT, Layer-wise relevance propagation, Shapley regression values, Shapley sampling values, and Quantitative input influence.

There are three desirable properties that can have the presence of a single unique solution in the class of additive feature attribution methods. However, the existing methods do not

all satisfy the following properties, making them non-optimal. The first property is local accuracy.

Property 1 (Local Accuracy):

$$f(x) = g(x') = \phi_0 + \sum_{i=1}^{M} \phi_i \cdot x'_i. \tag{3}$$

The explanation model $g(x')$ should match the original model $f(x)$ when $x = h_x(x')$, where $\phi_0 = f(h_x(0))$ is the model output with all simplified inputs toggled off.

The second property is missingness.

Property 2 (Missingness): If a feature is absent in the original input, it should have no attributed impact, formally:

$$x'_i = 0 \Rightarrow \phi_i = 0 \tag{4}$$

The third property is consistency. If a model changes resulting in an increase or maintenance of a simplified input's contribution, irrespective of the other inputs, the input's attribution should not decrease.

Property 3 (Consistency): Let $f_x(z') = f(h_x(z'))$ and $z' \setminus i$ denote setting $z'_i = 0$. This property states that for any two models f and f', if

$$f'_x(z') - f'_x(z' \setminus \{i\}) \geq f_x(z') - f_x(z' \setminus \{i\}). \tag{5}$$

for all inputs $z' \in \{0,1\}^M$, then $\phi_i(f', x) \geq \phi_i(f, x)$

Following all above three properties, it leads to Theorem 1.

Theorem 1: There exists only one possible explanation model g that follows Definition 1 and satisfies Properties 1, 2, and 3.

$$\phi_i(f, x) = \sum_{z' \subseteq x'} \frac{|z'|!(M - |z'| - 1)!}{M!} [f_x(z') - f_x(z' \setminus \{i\})] \tag{6}$$

where $|z'|$ is the number of non-zero entries in z', and $z' \subseteq x'$ represents all z' vectors

where the non-zero entries are a subset of the non-zero entries in x'. ϕ_i is known as Shapley values.

Young (1985) illustrate that Shapley values are the sole set of values satisfy three axioms similar to Property 1 and Property 3, so SHAP value can be a unified measure of feature importance. Other methods not based on Shapley values violate property 1 local accuracy and/or property 3 consistency.

SHAP values can be approximated directly using the Shapley sampling values approach, but Lundberg and Lee (2017) propose Kernel SHAP for approximating SHAP values with fewer evaluations of the original model to achieve similar approximation accuracy. It is a method that combines Linear LIME with Shapley values to approximate SHAP values without heuristic parameter selection. The method ensures the recovery of Shapley values, adhering to the properties of local accuracy, missingness, and consistency.

As discussed in the last section, LIME minimize the objective function:

$$\xi(x) = \arg\min_{g \in G} L(f, g, \pi_x) + \Omega(g),$$

The regularization term Ω, weighting kernel $\pi_{x'}$, and loss function L that solve the objective function and adhere to Properties 1-3 are derived by Lundberg and Lee (2017) as follows.

$$\Omega(g) = 0, \tag{7}$$

The weights for the Shapley kernel are:

$$\pi_{x'}(z') = \frac{(M-1)}{\binom{M}{|z'|}|z'|(|M-|z'|)}, \tag{8}$$

where $|z'|$ is the number of non-zero elements in z', and M is the total number of features.

The loss function L is a squared loss as:

$$L(f, g, \pi_{x'}) = \sum_{z' \in Z} [f(h_x(z')) - g(z')]^2 \pi_{x'}(z'), \tag{9}$$

We can then solve the Eq.(2) and find ϕ_i using above regularization term Ω, weighting kernel $\pi_{x'}$, and loss function L that are consistent with the Properties 1-3.

Kernel SHAP is model-agnostic, meaning that Kernel SHAP can be used to explain any machine learning model. It is a valuable feature because it allows practitioners to apply Kernel SHAP to a wide range of models, from simple linear regression to complex deep neural networks, to understand the contributions of each feature to the model's predictions.

II.3. LIME Versus SHAP

Both LIME and SHAP are model-agnostic that can be widely applied by different machine learning model interpretations in any field. There are differences between LIME and SHAP.

LIME provides local fidelity, which means it is accurate around the neighborhood of the prediction being explained. SHAP values offer global explanations that are consistent across all predictions. Therefore, LIME's explanations are specific to a single instance and may vary from one instance to another, while SHAP provides a single set of feature importances that are consistent across the entire dataset.

In particular, for each new prediction, LIME will potentially create a different simple model tailored to that particular instance, thus, the explanations provided by LIME can vary from one instance to another. In contrast, SHAP values are derived in a way that provide a consistent measure of feature importance that is based on the contribution of each feature across all possible combinations for the entire dataset. This approach does not create separate models for each instance; rather, it computes the average impact of each feature on the model's output, taking into account all possible feature interactions and combinations.

Finally, LIME is generally faster and more intuitive for local explanations, while SHAP is

computationally intensive but provides a more holistic understanding of feature importances (Dwivedi et al. (2023)).

III. Data, Methodology, and Machine Learning Models

III.1. Data

The monthly U.S. equity returns are sourced from the Center for Research in Security Prices (CRSP) database, including all U.S. firms listed on the NYSE, AMEX, and NASDAQ. The research period spans from March 1957 to December 2022, covering a comprehensive timeframe of 65 years. To calculate equity excess returns, we employ the 1-month Treasury-bill rate from CRSP as a proxy for the risk-free rate.

Our analysis employs a rich set of 209 firm-level predictive characteristics as features for stock return prediction. These characteristics include nearly all the signals documented in the extensive body of literature on cross-sectional stock returns.[4]

The dataset of firm-level characteristics exhibits a fair amount of missing observations. We systematically identify and replace missing data points with appropriate values to ensure the dataset's completeness and reliability, as discussed by Emmanuel et al. (2021). In particular, we uses the k-Nearest Neighbors (k-NN) imputation technique, which is the same as Demirbaga and Xu (2023). This approach relies on proximity-based imputation, where missing values in a multi-dimensional feature space are estimated by referencing data from their closest neighbors. We selected this technique due to its capacity to capture the inherent structure and relationships within the dataset, especially when the missingness mechanism is suspected to be non-random.

We allocate the dataset into three subsets: a training dataset consisting of 70% of the data, a validation dataset comprising 15% of the data, and a testing dataset accounting for 15% of the data.

[4]The construction of these 209 firm-level characteristics was undertaken by Chen and Zimmermann (2022) and can be accessed at https://www.openassetpricing.com/data.

III.2. Methodology

We define $R_{i,t+1}$ as the excess return of stock i at time $t+1$,

$$R_{i,t+1} = E_t(R_{i,t+1}) + \epsilon_{i,t+1}, \tag{10}$$

where

$$E_t(R_{i,t+1}) = g^*(z_{i,t}). \tag{11}$$

Our objective is to create a representation of $E_t(R_{i,t+1})$ using input variables that best predict the realized $R_{i,t+1}$ when applied out of sample. Let $z_{i,t}$ represent the N-dimensional vector of stock-level characteristics, which serve as input variables. we also name these stock-level characteristics as features in this paper. We assume that the conditional expected return, denoted as $g^*(.)$ is a function of these stock-level features. Each machine learning model has different functions.

III.3. Machine Learning Model Selections

In this paper, we select four different machine learning models. Among these four models, two of them are recognised as black-box model (XGBoost and neural networks), and two of them (decision tree and K-nearest neighbors) are more transparent but also complex models.

XGBoost is selected because it shows the highest accuracy and investment performance in asset pricing (Demirbaga and Xu (2023)). Neural network is recognized as the most commonly used black-box model that needs further understanding of inner workings. Demirbaga and Xu (2023) find that neural network has poor performance in predicting cross-sectional stock returns. It is important to discover the feature selections by neural networks to understand why neural network does not work well in asset pricing.

In addition, we select decision tree and K-nearest neighbors as comparison analysis. XAI techniques can also be beneficial in non black-box models in the following ways.

First, even in simpler models like decision tree, the relationships between features and

the target variable can be complex, especially with interactions between variables. XAI can help to uncover and clarify these relationships using simple visualizations.

Second, XAI can offer a more nuanced view of feature importance. While some models provide basic feature importance metrics, XAI techniques can give a more detailed picture, especially in terms of how different features interact with each other to influence the prediction.

Third, tools like LIME provide local interpretations of predictions, which means they explain individual predictions. This can be particularly useful in scenarios where you need to understand why the model made a specific decision for a particular instance, rather than just understanding the overall behavior of the model.

Fourth, stakeholders without a technical background might find the explanations provided by XAI tools more accessible and easier to understand than a machine learning model's interpretation.

To conclude, XAI can further enhance understanding, provide validation, and communicate insights in a more accessible way for any types of machine learning and deep learning models.

III.4. XGBoost

XGBoost (eXtreme Gradient Boosting) represents a scalable and distributed gradient-boosted decision tree (GBDT) algorithm. It is an ensemble learning method, which means it combines the predictions from multiple decision trees models for a final prediction. XGBoost uses a gradient boosting framework, which builds trees one at a time, where each new tree helps to correct errors made by the previous ones. It uses decision trees as base learners. Each tree is added to account for the shortcomings of the existing set of trees. The idea is to sequentially add trees, each one correcting the residual errors of the combined ensemble of all previous trees. The learning process involves minimizing a loss function through gradient descent. XGBoost also includes a regularization term in its loss function, which controls the complexity of the model and helps prevent overfitting. This makes XGBoost unique compared to traditional gradient boosting.

XGBoost offers several advantages, including high predictive accuracy and computational efficiency (Demirbaga and Xu (2023)). Its robustness against overfitting, achieved through regularization techniques, allows it to capture intricate data relationships effectively. However, it is worth noting that tuning XGBoost's hyperparameters can be computationally intensive, especially when working with extensive datasets.

XGBoost is often considered a 'black-box' model. First, since XGBoost is an ensemble of decision trees, the final prediction is the result of aggregating outputs from numerous trees. Tracing back the prediction to specific features becomes non-trivial when we have a large number of trees contributing to the output. Second, XGBoost can capture complex non-linear interactions between features. While this is beneficial for performance on complex datasets, it makes the decision-making process less transparent. To address these challenges, XAI methods such as LIME and SHAP can shed light on which features are most influential in the model's predictions.

III.5. Decision Tree

Decision tree is a supervised machine learning algorithm. Within a decision tree, features or attributes are represented by nodes, decision rules by branches, and outcomes or class labels by leaves. As the tree is constructed recursively, the algorithm selects the feature at each node that offers the optimal split based on a set of criteria, a process described in Kotsiantis (2013). The equation for a basic decision tree can be expressed as follows:

For classification:

$$\hat{y} = f(feature\_split); \tag{12}$$

And for regression:

$$\hat{y} = average(target\_values\_in\_leaf\_node), \tag{13}$$

where *feature_split* denotes how the data is split at each decision node based on a specific feature and its threshold, and *target_values_in_leaf_node* denotes the actual values or class labels of the target variable linked to the data points in a leaf node, and these are employed for making predictions.

III.6. K-Nearest Neighbors

The K-Nearest Neighbors (KNN) algorithm, also a supervised machine learning technique, is used for both classification and regression tasks. KNN makes predictions by either averaging the values or taking the majority class of the K-nearest data points in the feature space. This approach depends on the assumption that similar data points often share the same category or exhibit close numerical values. For classification, KNN identifies the K-nearest neighbors using a metric like Euclidean distance and assigns a class label based on the majority vote. For regression, it computes the mean of these neighbors' labels or values (Imandoust and Bolandraftar (2013)).

The KNN algorithm's formula as follows. For classification:

$$\hat{Y} = \arg\max \left(\sum_{i \in N} I(Y_i = y) \right); \tag{14}$$

and for regression:

$$\hat{Y} = \frac{1}{K} \sum_{i \in N} Y_i, \tag{15}$$

where N denotes the set of K-nearest neighbors, $\hat{Y}$ denotes the predicted class label, Y_i is indicative of the class label of the i-th nearest neighbor, and y is the class label projected for the specific data point under consideration. Moreover, the function I(âŃĚ) serves as an indicator function, assuming the value 1 when the enclosed condition holds true, and 0 otherwise. This formula is central to the functioning of the KNN classification method, determining the predicted class by aggregating the most frequent class label among the

K-nearest neighbors.

The main disadvantage of KNN is its sensitivity to the number of neighbours (K) and the distance metric. In addition, KNN demands significant computational resources, especially when dealing with large datasets. It is most effective when the data is evenly distributed across classes or when issues of class imbalance are properly addressed.

III.7. Neural Networks

Neural Networks (NNs) are a subset of machine learning models that draw inspiration from the human brain's structure and functions. NNs has interconnected nodes arranged in various layers: an input layer, one or more hidden layers, and an output layer by assigning weights to connections between nodes. Each node then uses an activation function to the sum of its weighted inputs (Guresen and Kayakutlu (2011)).

A feedforward neural network is as follows.

$$\hat{Y} = f(W^{(2)} \cdot f(W^{(1)} \cdot X + b^{(1)}) + b^{(2)}), \tag{16}$$

where f() representing the activation function, $\hat{Y}$ denotes the predicted output, X denotes 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 bias vectors for the hidden and output layers, respectively.

We implement neural networks with 5 layers with 32, 16, 8, 4, and 2 neurons. It has a relatively complicated structure than simpler neural networks with less layers so that it is a suitable black-box model to interpret using explainable AI.

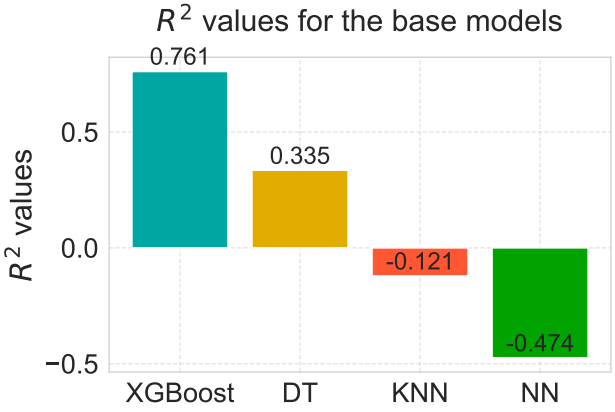
IV. The Interpretations of Explainable AI Techniques in Asset Pricing

This section analyzes the interpretations of explainable AI using LIME and SHAP. We first show the performance of base machine learning models. Based on the ranking of these model performance, we then use LIME to provide local interpretability for individual predictions of each model, offering insights into why certain stocks have specific predicted returns. We finally identify the highest impactful stock-level characteristics using SHAP values.

Figure I presents the out-of-sample R^2 values for each machine learning model prior to integrating explainable AI techniques. The results indicate that XGBoost is the top performer, achieving an out-of-sample R^2 of 0.761, with the decision tree, K-nearest neighbors, and neural networks following in performance.

Figure I: R^2 For Machine Learning Models

This figure presents the out-of-sample base R^2 values of each machine learning model. The sample period is from 1957 to 2022.



IV.1. Insights from LIME Analysis

We begin our analysis by using the LIME technique for explainable AI interpretations. Figure II, III, IV, and V show the LIME results. Each Figure presents an interpretation of characteristic influences on the predicted excess return. The horizontal bar in the left part

represents the range of possible predicted excess returns that the model could output, from the minimum to the maximum within the scope of the LIME. The color indicates whether the predicted value is closer to the minimum or maximum of the predicted range.

The feature contribution colors in the middle part are used to distinguish each impactful stock-level characteristics' influences on the model prediction. Features with orange bars indicate a positive impact on the predicted stock returns. This means that the presence or higher values of these features increase the predicted excess return for the next month. Features with blue bars have a negative impact on the predicted value. Their presence or higher values contribute to a decrease in the predicted excess return.

The sign next to a feature in the middle part indicates that the feature's sign contributes to an increase or decrease in the predicted return. For example, Figure II shows the mom6m's negative contribution value and its color, implying that when 6-month momentum is less than -0.12, it is associated with a positive predicted stock return. The length of the bar in the middle panel indicates the magnitude of the characteristic's impact. For example, in Figure II, the bar length of optionvolumne (0.03) is shorter than the mom6m bar (0.17), it suggests that optionvolumne is much less influential than mom6m.

The right part of the LIME figure shows the features and their corresponding values that contributed to the specific prediction. The Feature column lists the stock-level characteristics that the model used to make the prediction for excess returns at month $t+1$. The Value column shows the actual value each feature had for the specific stock at month t. The color is consistent with the color in the middle part, showing the positive or negative impact on the stock returns.

It is worth noting that these interpretations are specific to a particular model prediction and are based on the local approximation provided by LIME. In all our LIME figures by different ML models, we use the model prediction by the stock Colgate-Palmolive (CL) in December 2022 as an example.

We first analyze the LIME interpretation for XGBoost, depicted in Figure II, which is the

best-performing model among the four complex models. The left part of Figure II indicates that the XGBoost model predicts an average stock return of -0.07, which lies toward the lower end of the model's output range. The range of the predicted stock return varies between -0.65 and 2.80, suggesting that the features contribute to a slight decrease in the predicted returns.

The middle part of the figure highlights the most impactful stock-level characteristics that significantly contribute to predicting stock returns. Among the positively influencing orange characteristics, mom6m (Jegadeesh and Titman (1993)), is the most important stock-level characteristic. "mom6m <= -0.12" implies that when the 6-month momentum is less than or equal to -0.12, it positively contributes to the stock return in the following month. This is somewhat counterintuitive, as negative momentum typically forecasts lower returns. However, given this specific condition (mom6m being less than -0.12), it may indicate a strong rebound effect or mean reversion in this prediction. Although existing literature has established a positive linear relationship between six-month momentum and future stock returns, our application of explainable AI reveals a negative impact when past 6-month returns are extremely low. This finding explains why momentum may not always be effective and suggests researchers to consider deeper and non-linear relationships in asset pricing.

Similiarly, "spinoff <= 0.00" indicates that the occurrence of a spinoff event is positively affecting the return, implying that stocks undergoing spinoffs are expected to perform better. "mom12moffseason <= -0.01" suggests that the slight negative value in the off-season 12-month momentum (Heston and Sadka (2008)) contributes positively, suggesting a similar rebound effect as observed with mom6m. "dcpvolspread <= -0.02" shows that a negative and lower dcpvolspread (debt-to-capital volatility spread) positively influences returns. The least impactful characteristic among the 5 characteristics that positively affect stock returns is optionvolume1. "optionvolume1 <= -323.38" suggests that a substantial negative value in option to stock volume has a large negative impact on the predicted return, potentially signifying uncertainty or negative market sentiment.

On the other hand, important characteristics that negatively predict the stock return (in

blue) include high52, divinit, mom12m,momseasonshort, and maxret. "0.65<high52<=0.82" means that positive and high values for the 52-week high (George and Hwang (2004)) contribute future returns negatively, possibly indicating that the stock that is at or near their 52-week high may be overvalued or due for a correction. "divinit<=0.00" implies that the stock that has initiated a dividend is expected to have slightly lower return. "-0.16<mom12m<=..." shows that, when past 12 month momentum (Jegadeesh and Titman (1993)) is negative, mom12m is associated to a lower future return. This could indicate that a longer-term downward trend in the stock return. "momseasonshort <=... " reflects that the negative average value for return seasonality last year has a negative effect on the return. Finally, "-0.05 < maxret <= -0.03" suggests that when maximum return (Bali et al. (2011)) is between -0.05 and -0.03, it is followed by negative next month returns. It indicates that positive maximum returns leads to positive future returns.

The right part displays the actual values of each feature that were input into the XGBoost model for prediction. The color scheme corresponds to the middle column, indicating the direction of a feature's influence on the prediction. For instance, when the value of "mom6m" is -0.19, which is less than -0.12, it pushes the prediction towards positive stock returns. Conversely, when the value of "high52" is 0.67, falling between 0.65 and 0.82 (the range of values used by the model to make decisions), the model adjusts the prediction towards negative stock returns. The other top 10 important characteristics exhibit similar behaviors.

To conclude, in this specific prediction, the features that positively influence future stock returns, are the occurrence of spinoff events, 12-month momentum, return seasonality from the previous year, and maximum return over a month. On the other hand, features that are associated with a negative prediction of returns include 6-month momentum, momentum excluding seasonal components, volatility spread, 52-week high, initiation of dividends, and the ratio of option to stock volume.

The LIME analysis primarily reveals a significant connection between future stock returns and market-level characteristics, such as momentum and volatility. This finding highlights

the complexity of relationships between various stock characteristics and their subsequent returns. The LIME results imply that both direct and inverse relationships are present, indicating the multifaceted nature of the stock market. Such results suggest that traditional linear models might not fully capture all the underlying dynamics and relationships inherent in stock market data.

Figure II: LIME Explanation for Model's Prediction for XGBoost

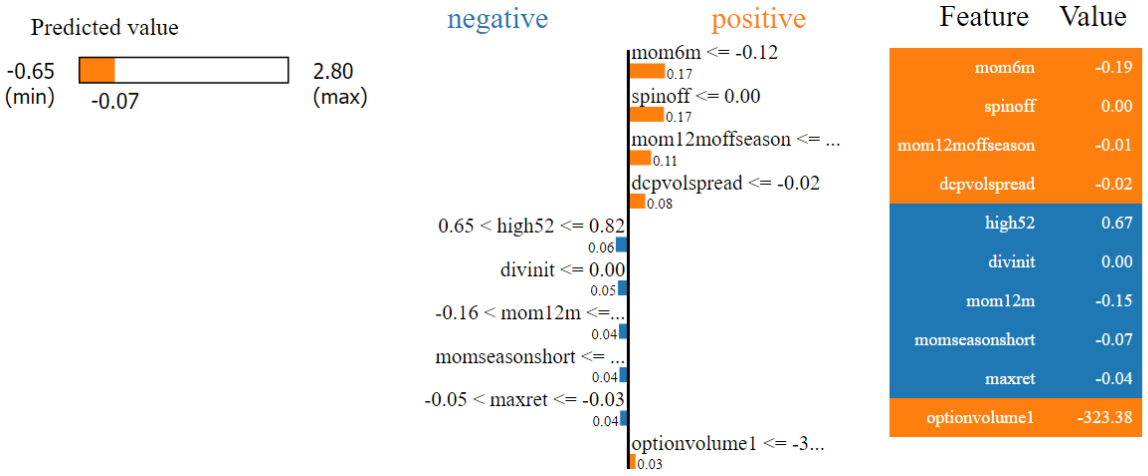


Figure III presents the LIME interpretation for decision tree, the second best performing model among the four machine learning models. The left part shows that the model predicts an average stock return of 0.01, which is near the lower end of the model's output range (-0.76 to 2.20). This average predicted stock return is slightly higher than -0.07 in the XGBoost prediction.

For the top 10 influential characteristics, we observe a similarity in feature predictions between the decision tree and XGBoost models. Both models share five of the top 10 impactful characteristics: maxret, cpvolspread, mom12m, mom6m, high52, and spinoff. Specifically, conditions such as maxret being higher than -0.03, cpvolspread being lower than 0.07, mom12m being within a certain range, mom6m ranging between 0.03 and 0.19, high52 falling between 0.65 and 0.82, and the occurrence of spinoff events are associated with negative stock returns. Both models show that 6-month momentum (mom6m) negatively affects future returns, while 12-month momentum (mom12m) has a positive impact. Moreover, they agree that high52

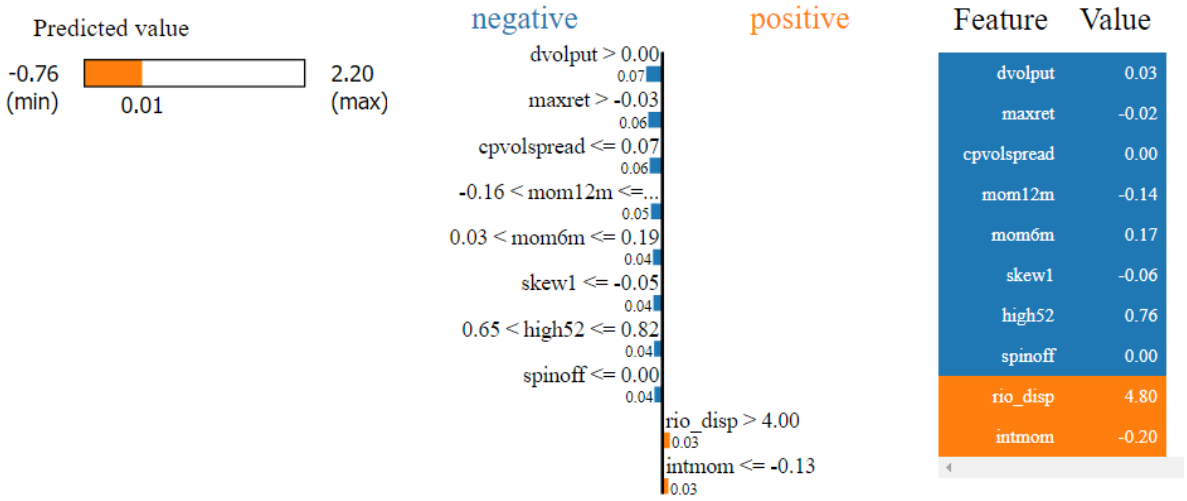
and maxret influence stock returns negatively within similar value ranges. However, the models differ in their predictions regarding spinoff events.

The other five key characteristics identified by the decision tree are dvolput, skew1, rio_disp, and intmom. It is clear that most characteristics predicted by the decision tree are market information variables such as momentum, skew1, and high52.

A main difference between the decision tree and XGBoost lies in the impact of their top 10 features. For instance, the most influential feature in XGBoost, mom6m, has a contribution of 0.17, whereas the top feature in the decision tree, dvolput, has a contribution of just 0.07. This indicates that XGBoost identifies features with more substantial impacts on predicted returns.

The difference between models could stem from the inherent modeling approaches: XGBoost captures complex non-linear relationships and interactions between features, whereas decision trees provide a more straightforward, hierarchical interpretation of feature influence. Each model's LIME result offers unique insights to have a more comprehensive understanding of the factors influencing stock returns.

Figure III: LIME Explanation for Model's Prediction for DT



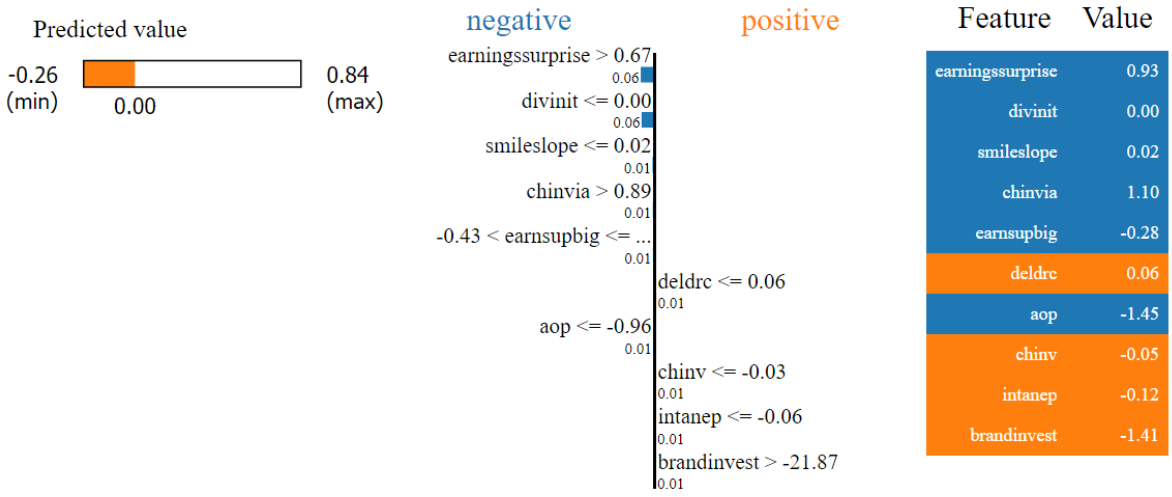
The third model is K-nearest neighbors, as shown in Figure III. The model predicts an average stock return of 0.00, with an output range from -0.26 to 0.84. This prediction suggests

that the combined effect of the features leads to a neutral expected return at $t+1$ across the sampled stocks.

The top 10 important features identified by K-nearest neighbors differ entirely from those selected by XGBoost and the decision tree. Specifically, an earnings surprise greater than 0.67 is associated with negative future returns. A firm initiating dividends is predicted to have negative future returns. Other characteristics influencing the K-nearest neighbors' model prediction include smileslope, chinvia, earnsupbig, deldrc, aop, chinv, intanep, and brandinvest.

In summary, the KNN model's LIME interpretation reflects the importance of financial indicators, ranging from earnings surprises to changes in profitability and investment. The model captures both traditional indicators such as earnings and dividends, as well as other factors like option pricing structures and intangible earnings. This interpretation of KNN mainly focuses on the firm accounting information on fundamentals to predict future returns. This is different from the LIME interpretations of XGBoost and decision tree, where their features are market information on past performance.

Figure IV: LIME Explanation for Model's Prediction for KNN

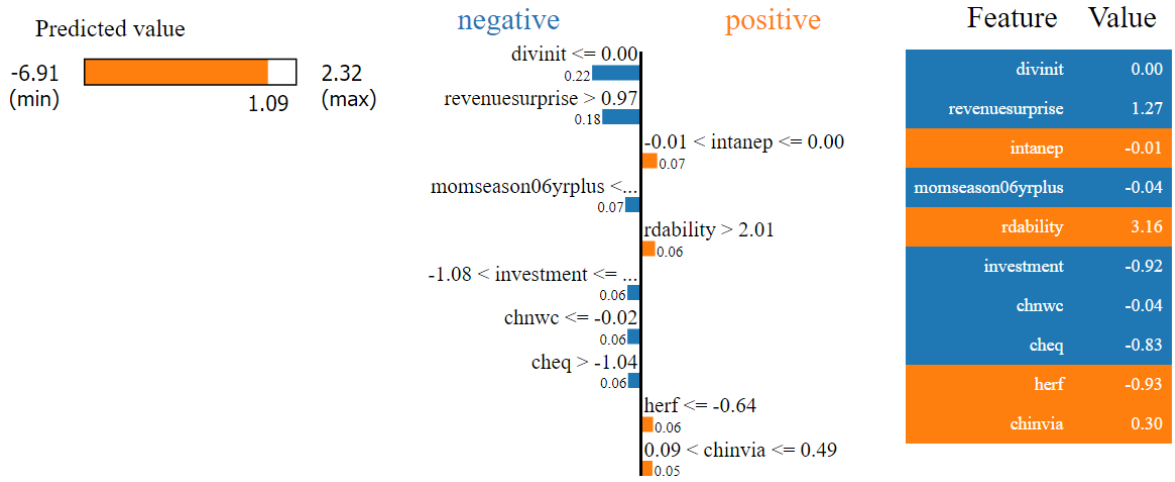


Finally, Figure V shows the LIME interpretation for the neural network model. The model predicts a stock return of 1.09, which is closer to the higher end of the model's output range that spans from -6.91 to 2.32. This suggests that, on average, the collective influence

of the features at time t is positive towards the predicted returns at $t+1$ across the sampled stocks.

For feature contributions, neural network emphasizes on the initiation of dividends, revenue surprise, intangible return using EP, return seasonality year 6 to 10, R&D ability, investment etc. The characteristics constructed by firm fundamental information are dominated in the neural network, which is similar to the K-nearest neighbors.

Figure V: LIME Explanation for Model's Prediction for NN



To conclude, the various machine learning models employed by LIME produce coherent visual representations for interpretation. Investors can understand which characteristics drive the predicted value of stock returns and to which extent these characteristics drive the prediction. If investors find some discrepancies in model predictions, they can use LIME analysis to understand the features to help them make decisions.

On the other hand, the models interpreted by LIME have focuses towards different types of characteristics. For example, XGBoost and decision tree emphasize more on past performance using market information, while K-nearest neighbors and neural network focus more on firm fundamentals using accounting information. However, given by the predictive power (R^2 and MSE) and Sharpe Ratio of XGBoost and decision tree are significantly higher than other two models, the features using market information identified by XGBoost and decision tree are more persuasive as leading characteristics to predict stock returns.

IV.2. Insights from SHAP Analysis

In this subsection, we discuss the insights from SHAP analysis, which provides global explanations. SHAP decomposes the prediction for each stock return into the sum of effects from each feature.

We first present SHAP summary plots for machine learning models in Figure VI, VII, VIII, and IX. SHAP summary plots interpret prediction results by visually showing each feature's impact on the model's predictions. Each horizontal bar represents a specific feature, and the corresponding color indicates the value of the feature. Blue indicates lower feature values, and red indicates higher feature values.

The length of each bar shows the magnitude of each feature's influence, which helps to emphasize the most important features. Each point on the plot represents a SHAP value for a feature for an individual prediction. The plot typically shows a 'swarm' of points, which represents the distribution of the SHAP values for each feature across all data points. This visualization identifies key drivers in the developed machine learning model to understand the model's inner workings.

X-axis shows the SHAP value, which represents the contribution of a feature to the difference between the actual prediction and the average prediction. For example, if a model predicts a stock's return and the average return is 2%, a SHAP value shows how much each feature contributes to moving the prediction away from this 2% average. Therefore, positive SHAP values indicates that the feature pushes the model's prediction higher, while negative SHAP values indicates that the feature pushes the model's prediction lower. In asset pricing, features represented by blue colors, are associated with lower predictions of stock returns, while features represented by red bars, contribute to higher predictions.

In the XGBoost model, as shown in Figure VI, we observe that the 52-week high (high52) is the most influential characteristic according to SHAP value interpretation. A low high52 value (indicated by blue dots) is associated with a lower predicted stock return, potentially even negative, whereas a high high52 value (represented by red dots) is associated with a

higher predicted stock return. Interestingly, the relationship between high52 and predicted stock return is asymmetrical. While most positive high52 values are indicative of a positive, albeit moderate, stock return, some predictions of high high52 values are associated with extremely high positive predicted stock returns.

The second and third important characteristics are 12-month momentum and 6-month momentum (mom12m and mom6m). High values of mom12m and mom6m (represented by red dots) are associated with slightly lower predicted stock returns, while low values of mom12m and mom6m (blue dots) correlate with relatively higher predicted stock returns. The impact of mom12m on the model output is greater than that of mom6m, especially on the positive side of return predictability. The result for 6-month momentum aligns with the findings in the LIME analysis, where mom6m is negatively related to predicted stock returns. The 6-month momentum indicated by XGBoost is counterintuitive compared to traditional findings (e.g., Jegadeesh and Titman (1993)), suggesting a mid-term mean reversion in stock prices.

In analyzing the SHAP summary plot, we focus on the distribution of SHAP values for each prediction, which enables us to spot points of extreme prediction. However, this doesn't directly indicate the sign of the prediction. For instance, while most predictions cluster around a SHAP value of zero, the frequency of these central predictions might greatly outnumber those with extreme SHAP values. To provide a clearer understanding, we will later present the mean of the SHAP values, which offers a more accurate interpretation of the average direction or sign of the predictions.

Maximum return over a month (maxret) and momentum without the seasonal part (mom12moffseason) are also top characteristics for return predictability and exhibit similar patterns to momentum. The subsequent important characteristics are realized volatility (realizedvol) and idiosyncratic risk (idiovolaht by Ali et al. (2003)). High realizedvol values have approximately zero impact on the model output, but low realizedvol values are linked with both negative and positive impacts.

The following significant characteristics are also market information-related, such as days with zero trades (zerotrade by Liu (2006)), industry return of big firms (indretbig by Hou (2007)), return skewness, and intermediate momentum (intmom by Novy-Marx (2012)), among others.

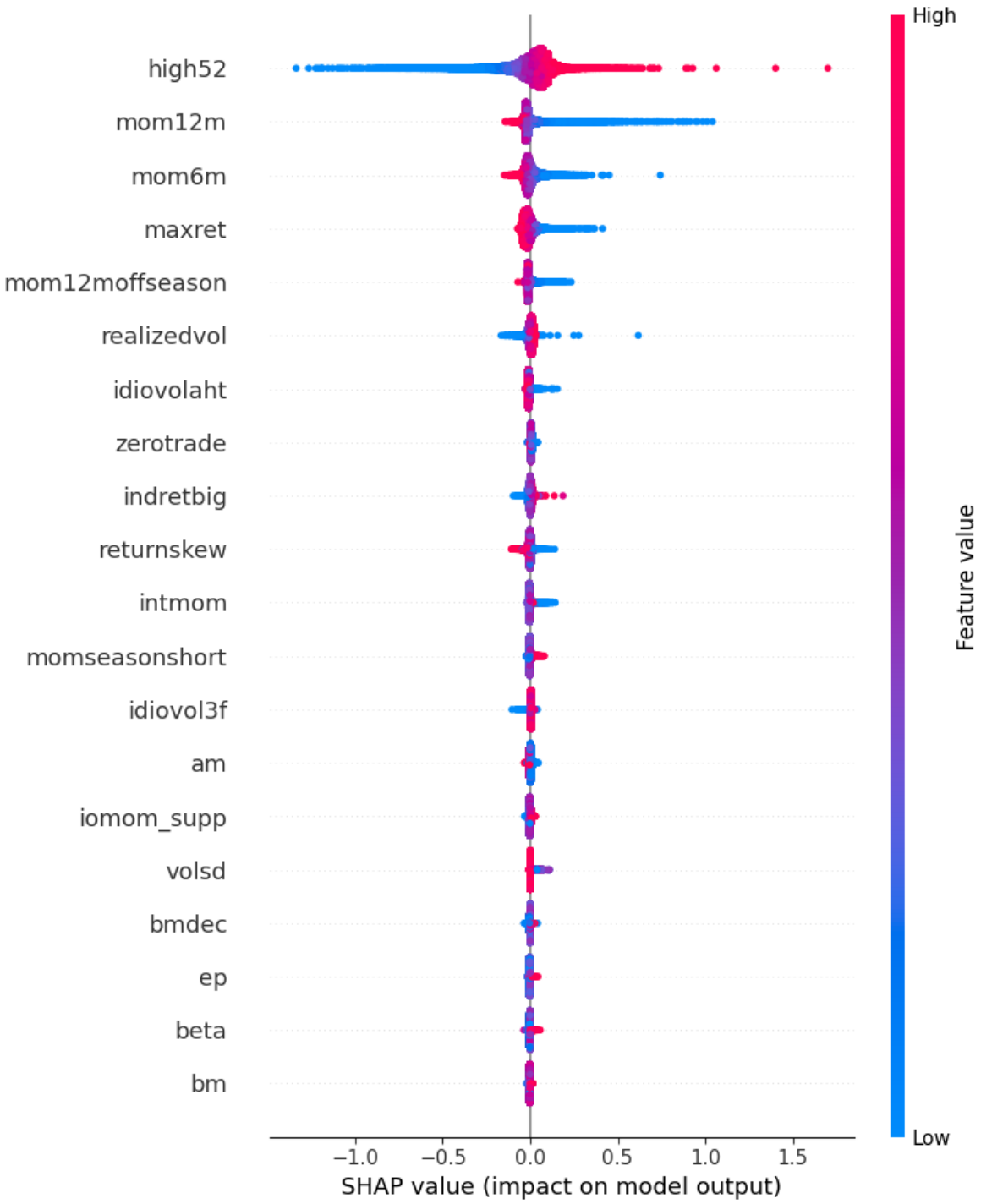
In the XGBoost model, similar to LIME, almost all top impactful characteristics use market-related information (such as price and volatility) rather than firm fundamentals to predict stock returns. Some exceptions are the total assets to market (am) and book to market (bmdec) ratios by Fama and French (1992), which, however, are also partially related to market information.

Among the top characteristics identified by SHAP value in the XGBoost model, there is considerable overlap with those found in the LIME analysis, including high52, mom12m, mom6m, maxret, etc. Although there are minor differences in interpretations, both SHAP and LIME results point to the same conclusion: the key drivers of cross-sectional returns are factors constructed from stock price information, such as momentum, 52-week high, and realized volatility.

The second machine learning model we examine is the decision tree, as shown in Figure VII. We find that the top three characteristics in the decision tree -high52, mom12m, and mom6m - are the same as those in XGBoost. Despite minor differences, a low high52 value (indicated by blue dots) is associated with a negative stock return, while a high high52 value predicts a positive stock return. In both the decision tree and XGBoost models, some instances of high high52 values can predict extremely high positive stock returns. In addition, in both models, extremely low values of mom12m and mom6m are associated with high stock returns, whereas high values of these characteristics tend to predict slightly lower stock returns. The primary difference between the two models is that the decision tree has more emphasis on low mom12m than mom6m in predicting high stock returns.

Return skewness (returnskew) is ranked fourth in the decision tree, in contrast to its tenth-place ranking in XGBoost. Our results suggest that a low returnskew value correlates

Figure VI: SHAP Summary Plot for XGBoost Model Interpretation



with a higher predicted stock return (indicative of a negative stock return), while a high high52 value (represented by red dots) is associated with a higher predicted stock return.

Figure VIII shows the SHAP summary for the K-nearest neighbors (KNN) model. The density of the dots along the zero line indicates the frequency of the features contributing to the model's output. Upon first observation, the distribution of dots along the x-axis appears sparse, indicating that the features identified by the KNN model may not predict stock returns as effectively as those in the XGBoost and decision tree models. This aligns with the comparatively lower predictive power of the KNN model.

The most significant characteristics indicated by SHAP in the KNN model are brand capital investment (brandinvest by Belo et al. (2014)), short interest (shortinterest by Dechow (2001)), and revenue growth rank (meanrankrevgrowth by Lakonishok et al. (1994)). A main pattern is that most characteristics are based on firm fundamentals and accounting information, such as investment, earnings, revenue, and firm age. This differs from the XGBoost and decision tree models, where most characteristics are related to stock performance and derived from market data.

Finally, Figure IX presents the SHAP summary for the neural network model's interpretation. The SHAP values of most characteristics are centered around zero, with the notable exceptions of earnings surprise of big firms (earnsupbig) and revenue surprise. Similar to the KNN model, most characteristics are based on firm fundamentals information. However, given the neural network's lower predictive power, these key factors are less reliable for asset pricing.

Figure VII: SHAP Summary Plot for DT Model Interpretation

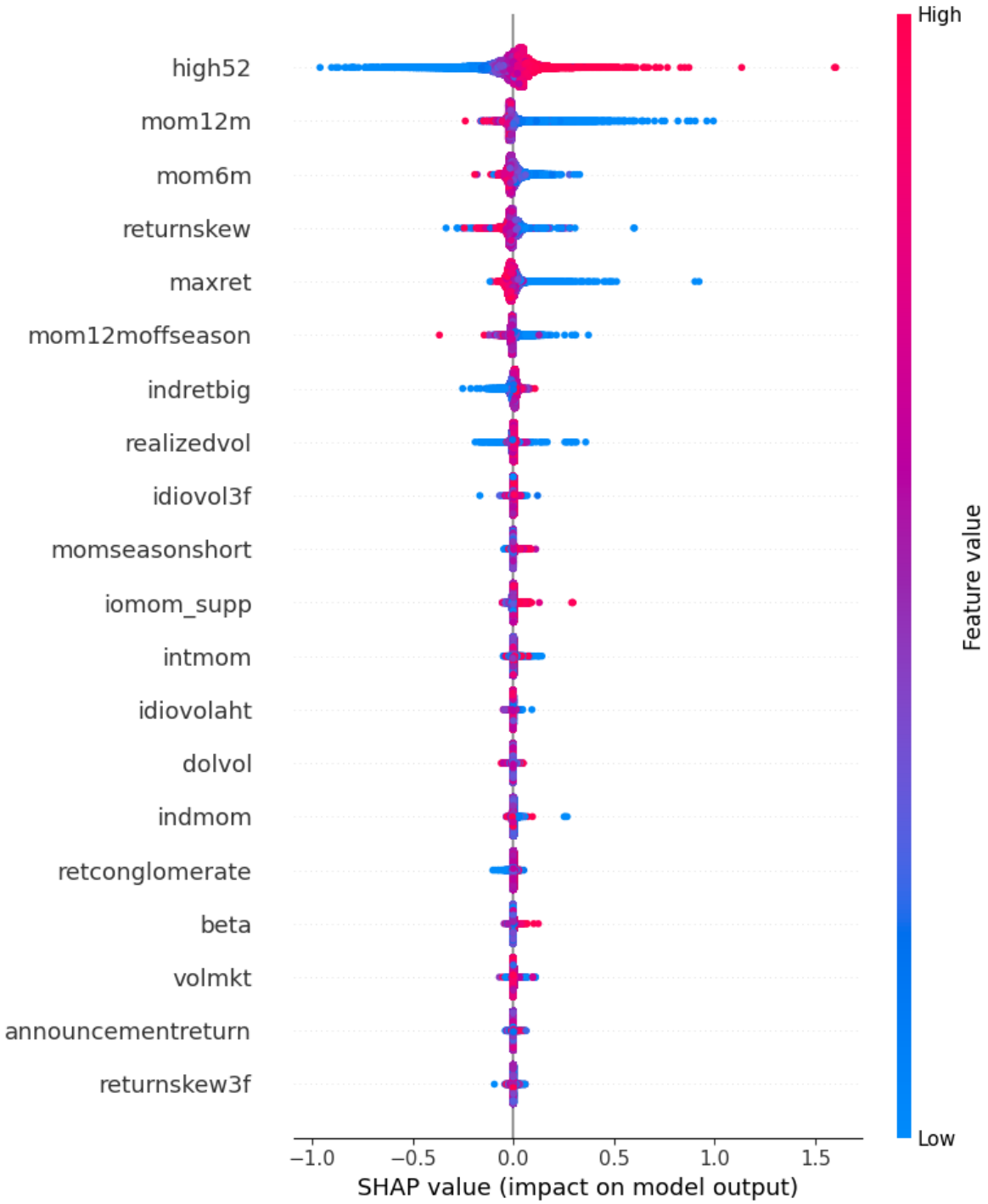


Figure VIII: SHAP Summary Plot for KNN Model Interpretation

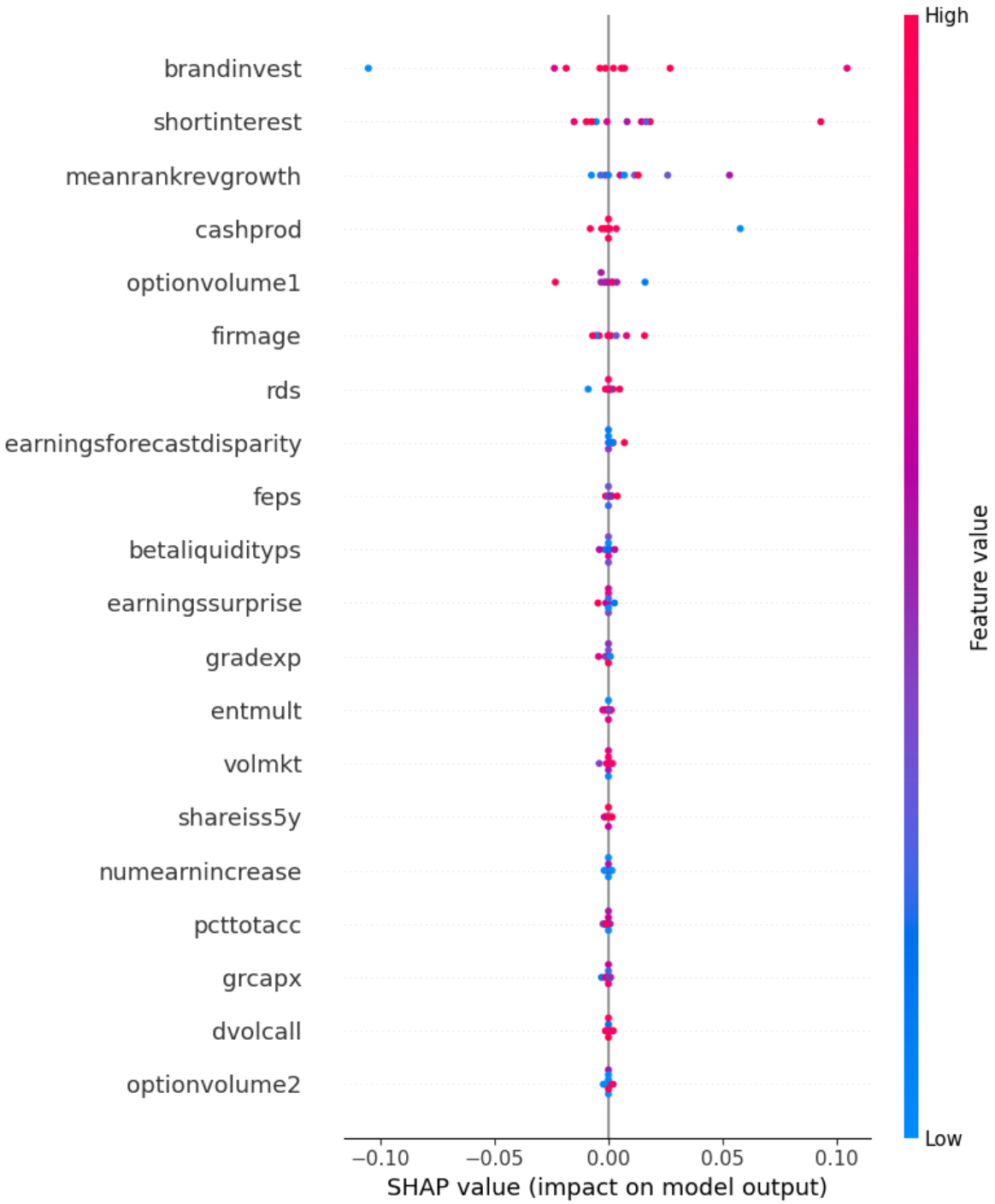
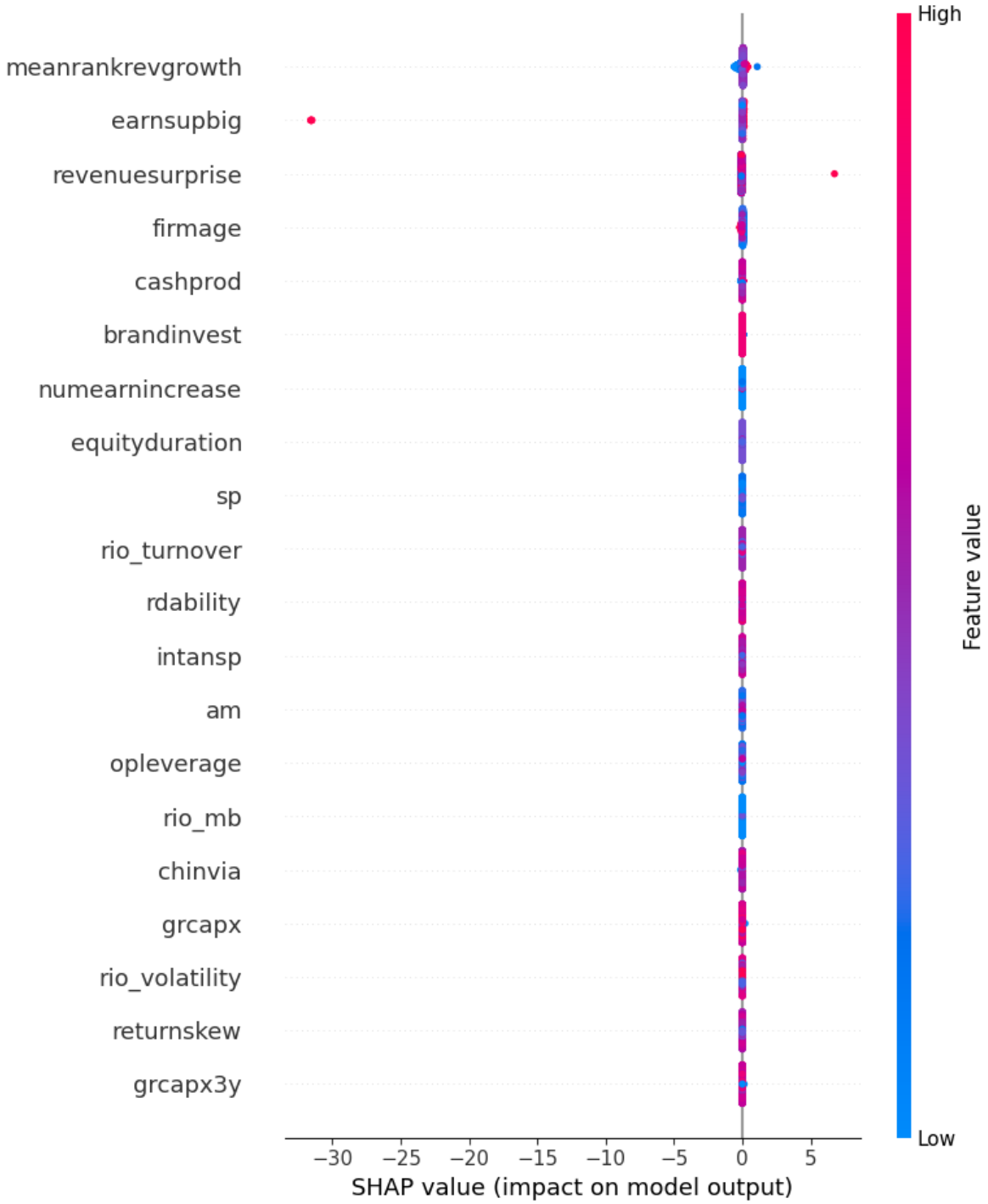


Figure IX: SHAP Summary Plot for NN Model Interpretation



We next study the relative importance of these stock-level characteristics to understand how their importance are compared with each other. We use the bar plots showing the mean absolute SHAP values in Figure X. The x-axis represents the mean absolute SHAP value for each feature. A higher absolute SHAP value indicates a larger average impact on the model output. The features are ordered by their mean SHAP value, with the feature having the largest mean SHAP value at the top and the smallest at the bottom. The bar at the bottom labeled "Sum of 200 other features" represents the cumulative impact of all other features not listed individually on the plot.

It's important to note that this plot uses the mean absolute value, which focuses on the magnitude of impact while ignoring the direction (positive or negative). It provides a clear assessment of the *relative importance* of individual features. This is different from the SHAP summary plot, where identifies return predictability patterns by observing the distribution of SHAP values for each feature.

The first plot from XGBoost shows that maxret is the most influential characteristic in significantly predicting stock returns, with an impact on the model's output of around 0.03. High52 ranks as the second most crucial feature with a slightly lower impact of approximately 0.03. Other characteristics with high impacts include mom12m, mom6m, realizedvol, ep, and others.

The mean absolute SHAP value indicates the relative importance of features. For example, maxret has roughly three times (0.03/0.01) the impact on predicting stock returns compared to zerotrade.

Moreover, the cumulative impact of all other features (0.06) is higher than that of any individual feature. This suggests that, while the top features have a significant impact, the model's predictions are also considerably influenced by the collective effect of smaller features.

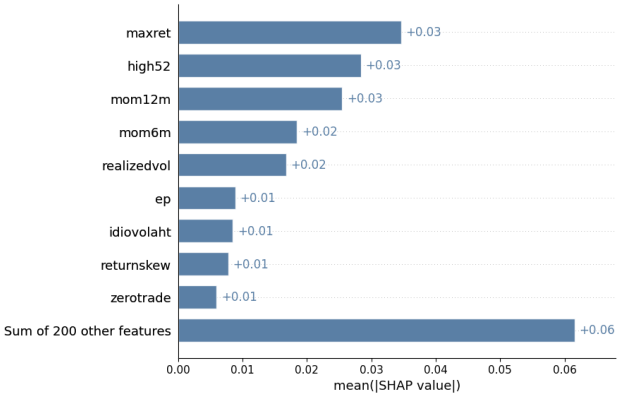
The second plot from the decision tree model shows that 12-month momentum is the most impactful characteristic with a mean absolute SHAP value of 0.05, followed by high52 (0.03) and maxret (0.01). The sum of all other features is 0.04, which is less than the top

feature, mom12m. Compared to XGBoost, the decision tree model's most important features contribute more to its predictions.

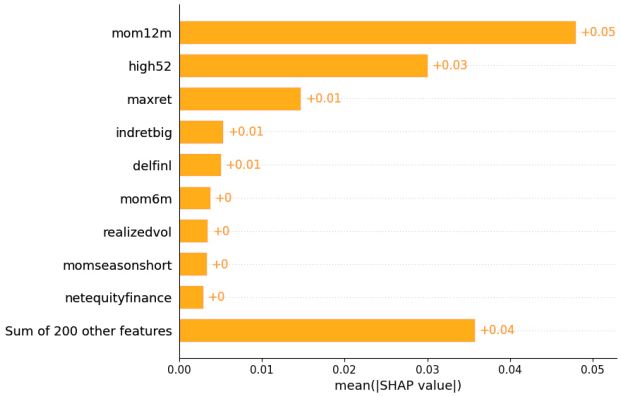
The third and fourth plots in Figure X present the relative importance of features in the K-nearest neighbors and neural network models. Their top features do not contribute as significantly to stock return prediction as those in XGBoost and the decision tree. Specifically, both shortinterest and meanrankrevgrowth have mean absolute SHAP values of only 0.01 in KNN. In the neural network model, firmage and equityduration exhibit mean absolute SHAP values of 0.02 and 0.01, respectively. However, the sum of other features in both KNN and neural network is 0.04 and 0.05, indicating that these models do not identify particularly impactful characteristics.

In conclusion, maxret, high52, and mom12m are the top three characteristics in both XGBoost and the decision tree for predicting stock returns, and their importance is relatively similar. In KNN and neural networks, meanrankrevgrowth and firmage emerge as particularly important characteristics. The sum of the 200 other features in all models also demonstrates strong importance.

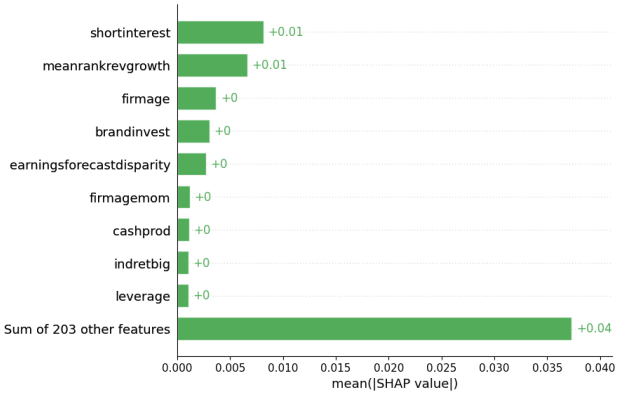
Figure X: SHAP Bar Plot



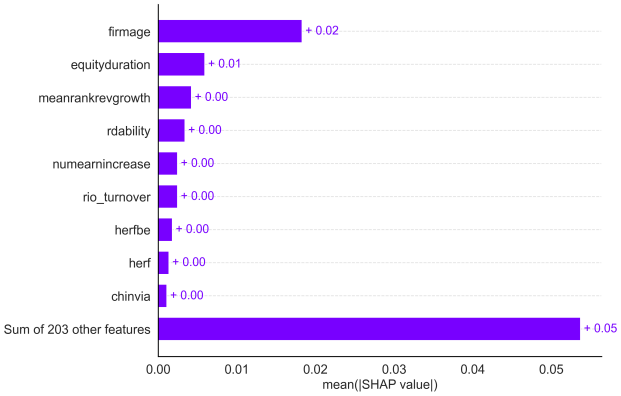
(a) XGBoost



(b) Decision Tree



(c) K-Nearest Neighbors



(d) Neural Networks

After assessing the relative importance of features, we compare the top 20 features of each model, including their real values and signs. We focus on the real SHAP value, which shows not only the magnitude but also the direction of each feature's impact on the prediction. This analysis helps us understand which how significant features increase or decrease the predicted stock returns.

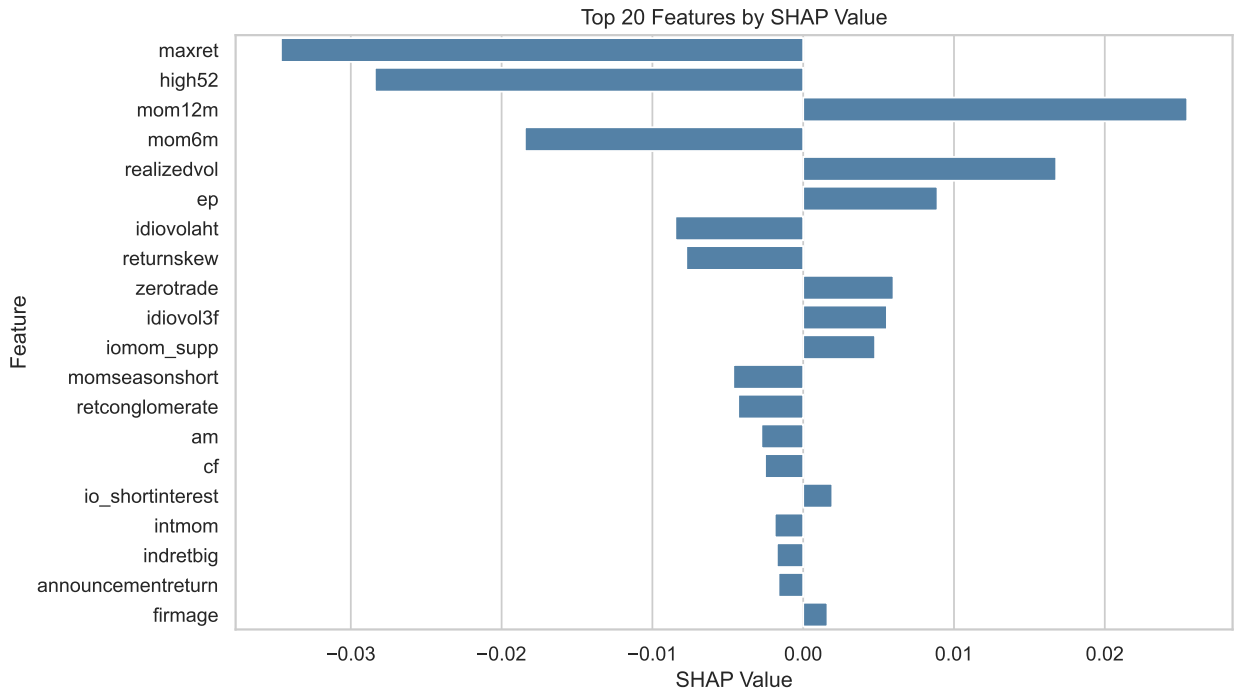
Figure XI presents the top 20 features, ranked by their SHAP values, from four machine learning models used for predicting stock returns. The first figure reveals the top 20 features in the XGBoost model. Here, maxret exhibits the most substantial positive SHAP value, associated with a lower predicted stock return. High52 has a slightly lower impact on the predicted returns and negatively affects stock returns. Mom12m, ranking third, generally increases the prediction, whereas the fourth feature, mom6m, reduces the prediction of stock returns.

The second figure of decision tree suggests that mom12m is the most important characteristic, positively predicting stock returns, followed by high52 and maxret, which negatively predict returns. The signs of the top three characteristics align with those in XGBoost. Mom6m ranks sixth in the decision tree, with a positive sign, increasing the predicted stock returns. This interpretation is opposite from the result in XGBoost, but is consistent with the literature on momentum (Jegadeesh and Titman (1993)).

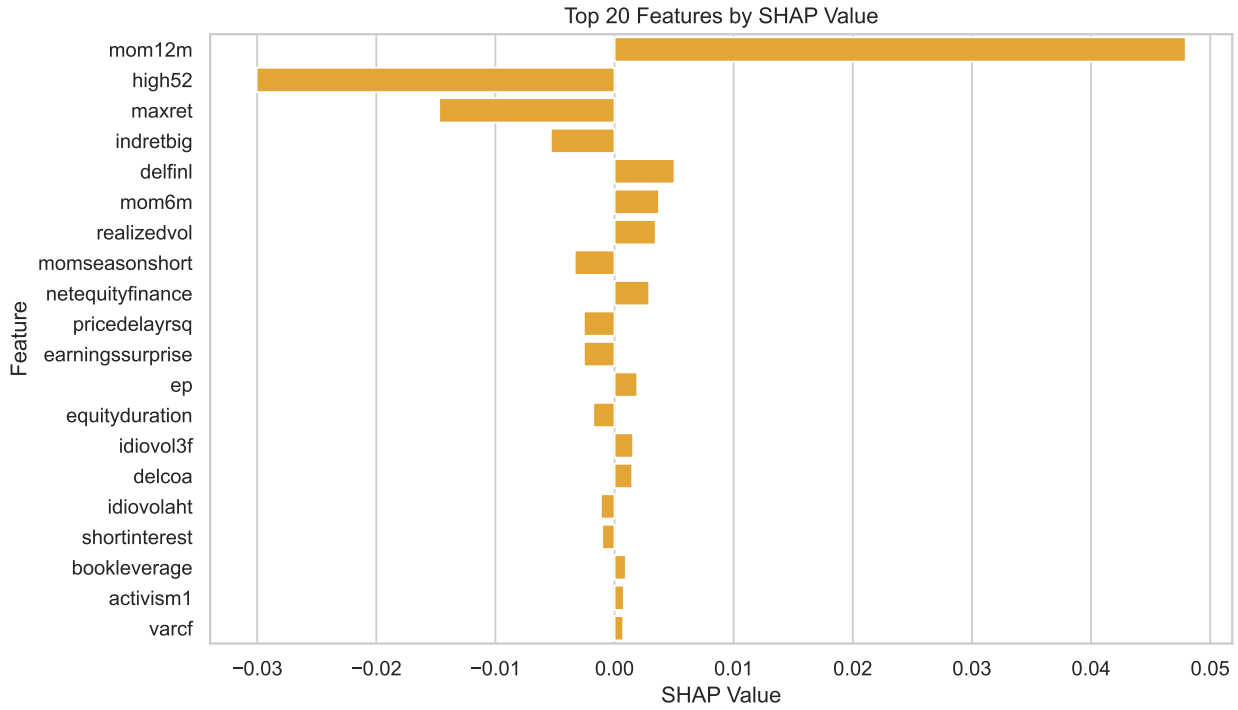
We also present the relative importance and signs of features in K-nearest neighbors and neural networks. In K-nearest neighbors, shortinterest positively predicts stock returns, while firmage, meanrankrevgrowth, earningsforecastedisparity, and brandinvest negatively predict returns. In neural networks, revenuesurprise and cashprod lower the prediction, whereas firmage, equityduration, and meanrankrevgrowth increases it.

In summary, the top features in XGBoost and the decision tree that positively predict stock returns are 12-month momentum and realized volatility, while maxret and high52 negatively predict stock returns.

Figure XI: Top 20 Features by SHAP Value

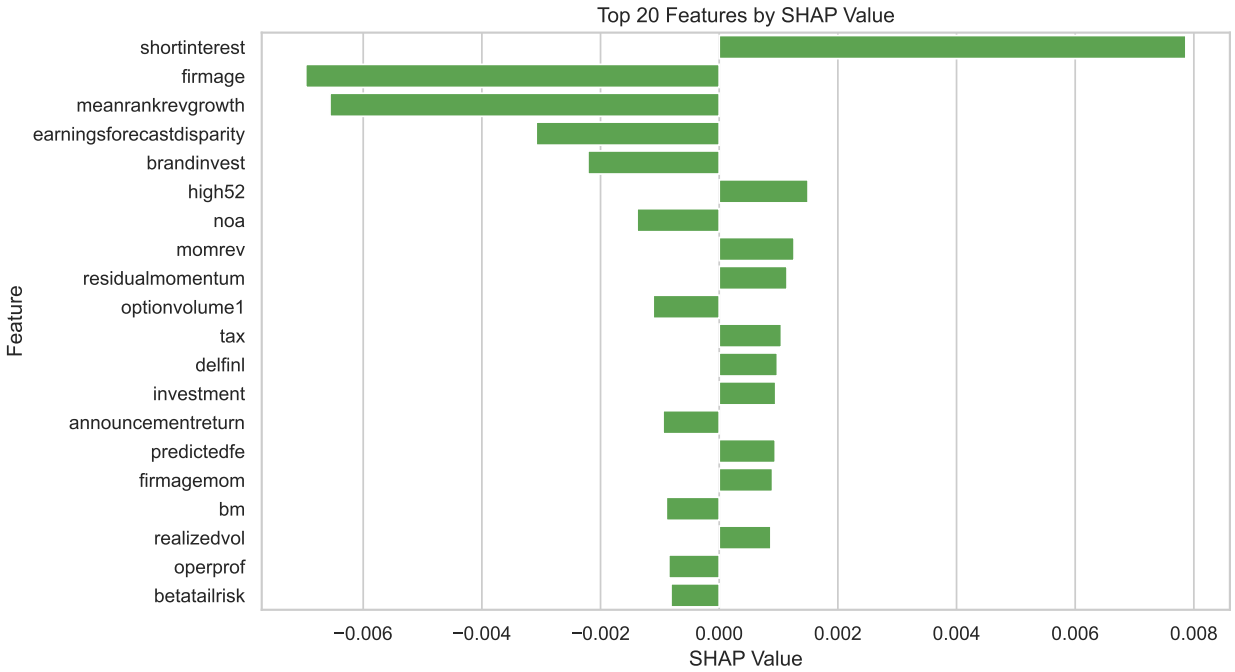


(a) XGBoost

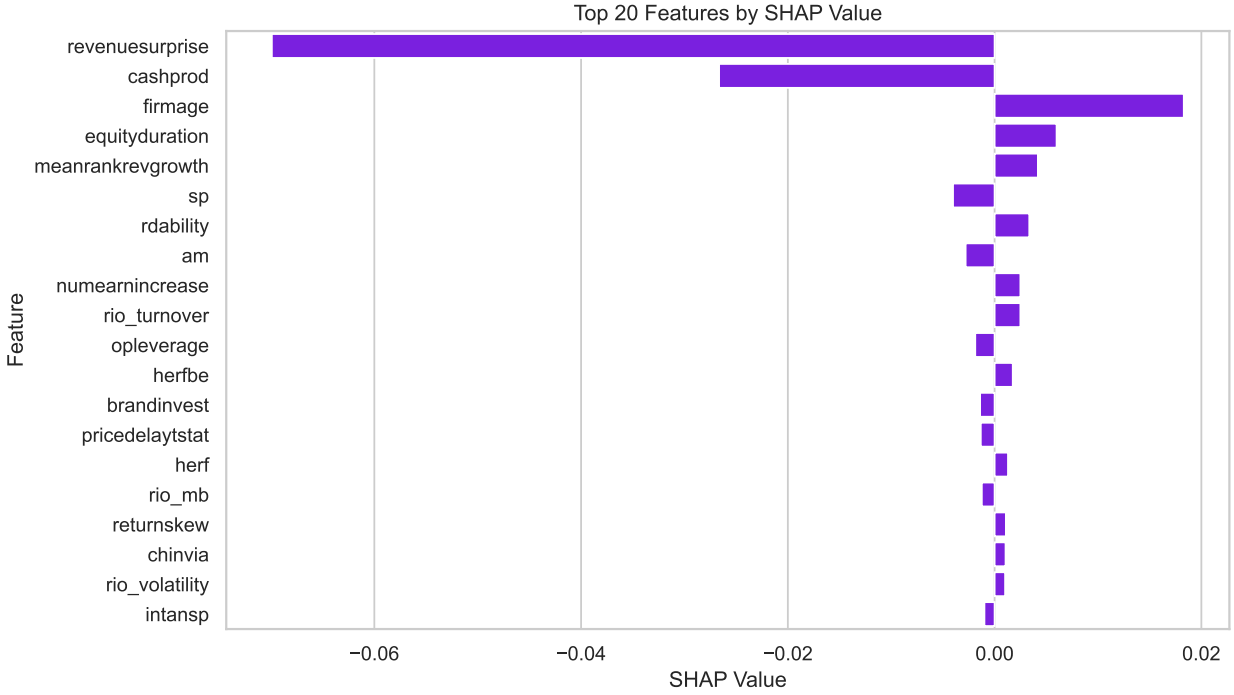


(b) Decision Tree

Figure XI: Top 20 Features by SHAP Value (Continued)



(c) K-Nearest Neighbors



(d) Neural Networks

Finally, we analyze the SHAP force plot in Figure XII. A SHAP force plot is used to understand each feature's contribution to a specific prediction, similar to LIME analysis.

SHAP force plots serve different purposes from other SHAP plots. They visually represent the push and pull of features on the model's output, making it easy to see how each feature's value shifts the base prediction to the final output. Often interactive, these plots allow researchers to explore the impact of changing a feature's value on the model's prediction. Thus, they are well-suited for presenting the model's findings to stakeholders and auditing individual predictions.

The central horizontal line represents the expected model output, while the colored bars on either side show the direction and magnitude of each feature's impact on the prediction. Features that push the prediction higher (increase the stock returns) are shown in red, with the length of the red bar indicating the magnitude of the feature's positive impact. Conversely, features that push the prediction lower (decrease the stock returns) are shown in blue. In addition, the feature value, such as maxret = -0.01752, refers to the actual value of the feature, not its impact.

Each feature has a value representing its actual contribution in this specific instance. For example, a blue mom6m indicates that the 6-month momentum for this particular prediction has a value that increased the predicted outcome by that SHAP value amount. The base value is the average output of the model (stock returns) over the training dataset, which is the starting point for adding each feature's impact. The output value (f(x)) is the actual prediction for the specific instance after all feature impacts have been applied, calculated as the sum of the base value and all individual SHAP values.

Figure XII presents the SHAP force plot for the XGBoost model. It shows that the average stock returns are 0.008954 in this trained XGBoost model, with a prediction of -0.04 after all feature impacts. Maxret is the most impactful characteristic, followed by high52 and mom6m. Mom12m is also significant, increasing the prediction. This indicates that maximum return over a month, 52-week high, and 6-month momentum are associated with lower predicted

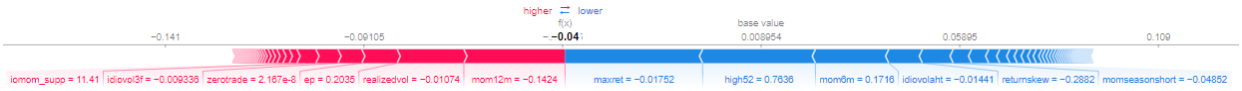
returns, while 12-month momentum correlates with higher returns. Features like zerotrade and ep, with shorter bars, have smaller impacts, suggesting they don't significantly change the prediction compared to a feature like maxret. This finding aligns with the SHAP summary plot and the LIME analysis.

The second plot, Figure XII, shows the decision tree's result. The base and predicted values are very close, at 0.008 and 0.01, respectively. Mom12m has the highest impact on predicting stock returns and contributes positively. This is followed by high52 and maxret, which are also the most impactful in XGBoost.

The third plot presents the SHAP force plot for K-nearest neighbors, which differs significantly from the previous models in that the most impactful characteristics contribute less. For example, the blue bar length of meanrankrevgrowth in KNN is much shorter than the blue bar length of maxret in XGBoost. The key characteristics driving the prediction are also completely different, with the KNN model selecting firm fundamentals as main features, compared to XGBoost's focus on momentum using past return information.

Finally, the SHAP force plot for the neural network shows a pattern distinct from the other three models. In this trained neural network model, the average stock returns are 1.159, with a prediction of 1.09 after all feature impacts. Among the top characteristics, revenuesurprise has the most significant impact in lowering the prediction, followed by cashprod, while firmage notably increases the prediction.

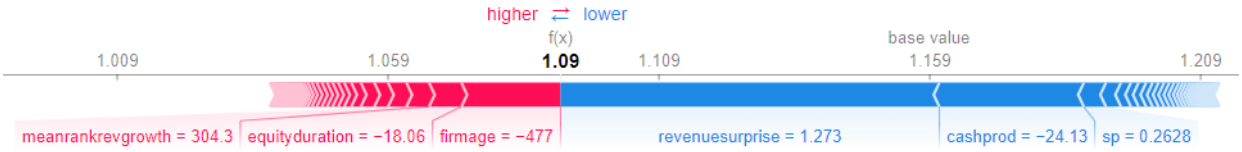
Figure XII: SHAP Force Plots



(a) XGBoost

(b) Decision Tree

(c) K-Nearest Neighbors



(d) Neural Networks

V. Performance Improvement After Using Explainable AI

Explainable AI is used not only to understand inner workings, but also to leverage such interpretations to improve the machine learning model performance. This is because more understandable models can achieve better performance. In the asset management industry, the improved model performance leads to higher investment outcomes.

After establishing the most influential stock-level characteristics through XAI techniques, we use these XAI-derived insights to refine our machine learning models. This section first presents the methodology used to train and test the machine learning models based on the XAI results. Subsequently, we demonstrate the improvement in both predictive accuracy and investment performance attributable to the application of XAI.

V.1. Training, Validation, And Testing Based on Explainable AI

We first identify the most important characteristics using SHAP. This analysis employs the SHAP method as the benchmark of explainable AI results to evaluate the importance of each feature within our model. We favor SHAP values as they provide a global explanation consistent across all single predictions. Moreover, the results derived from LIME largely align with those from SHAP. We use the result of top 20 features that exhibit the highest SHAP values due to their clear and substantial influence on the model's predictive performance.

After identifying top 20 features using SHAP values, we train the machine learning model using the training dataset. We then adaptively select tuning parameters based on the top 20 features identified by explainable AI, while do not tune for the less important features. We use the Sequential Least SQuares Programming (SLSQP) algorithm, a powerful optimization technique to determine the optimal feature weights for enhancing the predictive performance of our model based on XAI interpretations. The SLSQP algorithm refines feature weights in an iterative process to identify the most effective solution within predefined constraints, in this case, the Mean Squared Error (MSE), a key performance metric for our predictive model.

The SLSQP algorithm's ability to handle both constrained and unconstrained optimization problems, made it an important component in the fine-tuning of feature weights to enhance the overall predictive accuracy of our model.

Algorithm 1 in Appendix A.3 shows the SLSQP algorithm. In particular, given a list of feature names, training data, target data, initial feature weights, and constraints, the algorithm initializes optimization variables with the provided initial weights (Line 3). Subsequently, a custom objective function is defined (Line 5), and constraint definitions are established (Line 7). The SLSQP algorithm iteratively adjusts the parameters to find the optimal solution within the defined constraints, and then iteratively refines the feature weights. Finally, the optimal feature weights are extracted from the optimization results (Line 11). The final step in our process is to retrain the model on the full training dataset using the optimal parameters found by SLSQP algorithm, and assess the performance of the final model on the test dataset.

Our methodology presents a comprehensive approach to model development and refinement, leveraging XAI for feature importance, followed by rigorous training, tuning, and feature weight adjustment. This process ensures that our model is not only accurate but also interpretable, aligning with the growing need for transparency in AI applications in the asset management industry.

V.2. Model Performance Comparison

In order to establish the importance of XAI in improving model performance, we use performance metrics of three models trained in slightly different ways.

We first obtain the base performance measure for each machine learning model, which we name it *Base*. *Base* only relies on the training dataset and do not do hyperparameter tuning and do not consider XAI results. This metric serves as the baseline measure of performance, indicating how well the initial model explains the variability of the stock returns. By establishing this foundational benchmark, we can assess the incremental value provided

by subsequent refinements.

Second, we measure the hyperparameter tuning performance, which we name it *Tuned*. Hyperparameter tuning is a standard method to optimize machine learning models. By comparing the performance of the base model with the performance post-hyperparameter tuning, we can quantify the benefit derived from optimization techniques that do not directly incorporate XAI insights.

Finally, we measure the model performance after adaptive tuning based on XAI, which we name it *Tuned (XAI)*. The procedure of *Tuned (XAI)* is described in the previous subsection. It represents the model's performance after adjusting parameters based on XAI findings. If the performance for the *Tuned (XAI)* model outperforms the previous two ones, it provides empirical evidence that the interpretability provided by XAI contributes to more than just understanding the model - it directly improves predictive performance.

The model's predictive performance is evaluated using the coefficient of determination (R^2) and mean squared error (MSE). We also evaluate investment performance using Sharpe Ratio (SHARPE (1964)). Given that we use the test dataset to evaluate performance, they are naturally out-of-sample performance.

V.3. Performance Results

Figure XIII displays the out-of-sample R^2 in ascending order. The bottom of the figure indicates that XGBoost achieves the highest R^2 compared to the other three machine learning models. The base performance of XGBoost exhibits an out-of-sample R^2 of 0.761. The tuned XGBoost model attains an R^2 value of 0.796. The tuned (XAI) model result achieves an even higher R^2 of 0.815, which is an increase of approximately 2.3% from the tuned model to the tuned model using XAI to determine optimal feature weights.

The ML model with the second-highest out-of-sample R^2 is the decision tree, which has a base R^2 of 0.335, a tuned R^2 of 0.426, and a tuned (XAI) R^2 of 0.457. The decision tree also exhibits a significant increase in predictive power after applying explainable AI techniques.

This is an increase of R^2 for approximately 7% from the tuned model to the tuned (XAI) model. An increase in accuracy of 7% in return predictability in the financial market can have a substantial positive impact on investment performance.

For our third machine learning model, K-nearest neighbors, the base model's R^2 value is negative. After tuning, the KNN model achieves an R^2 of 0.253. The R^2 value further increases to 0.341 after we establish the important characteristics using explainable AI and subsequently tune the model based on these insights. The performance improvement is around 35% from the tuned model to the tuned (XAI) model.

Finally, the base neural network's R^2 value is -0.474, which improves to -0.125 after tuning, and further improves to -0.065 upon tuning using XAI. The increase is around 48% from the tuned model to the tuned (XAI) model. However, the neural network still exhibits the poorest performance with a negative R^2 value after XAI tuning, suggesting that neural networks may not be suitable for return predictability. Despite this, the substantial increase in R^2 values indicates that XAI can significantly enhance model accuracy.

To conclude, the consistent improvements observed in various models, from their base versions to tuned and then to tuned (XAI) iterations, support the hypothesis that a deeper understanding of a model's inner workings enhances its predictive power.

Another metric assessing the performance of machine learning models is the mean squared error (MSE), which calculates the average of the squared differences between observed and predicted values. Figure XIV presents the MSE results. All models' base MSEs are higher than their tuned MSEs, and the tuned model MSEs are higher than those tuned based on XAI, implying that understanding the inner workings using XAI techniques can improve model accuracy by decreasing the mean squared errors. This finding is also consistent with the results using out-of-sample R^2 presented in Figure XIII. Among the four machine learning models, XGBoost has the lowest MSE, reaching as low as 0.003 when tuned using XAI interpretations, suggesting that XGBoost possesses the highest predictive power.

After analyzing the preditive power, we investigate the investment performance using

Figure XIII: R^2 Comparison

This figure presents a comparison of the out-of-sample R^2 values among base, tuned, and tuned (XAI) models for four machine learning algorithms, arranged in ascending order by the R^2 value. The sample period is from 1957 to 2022.

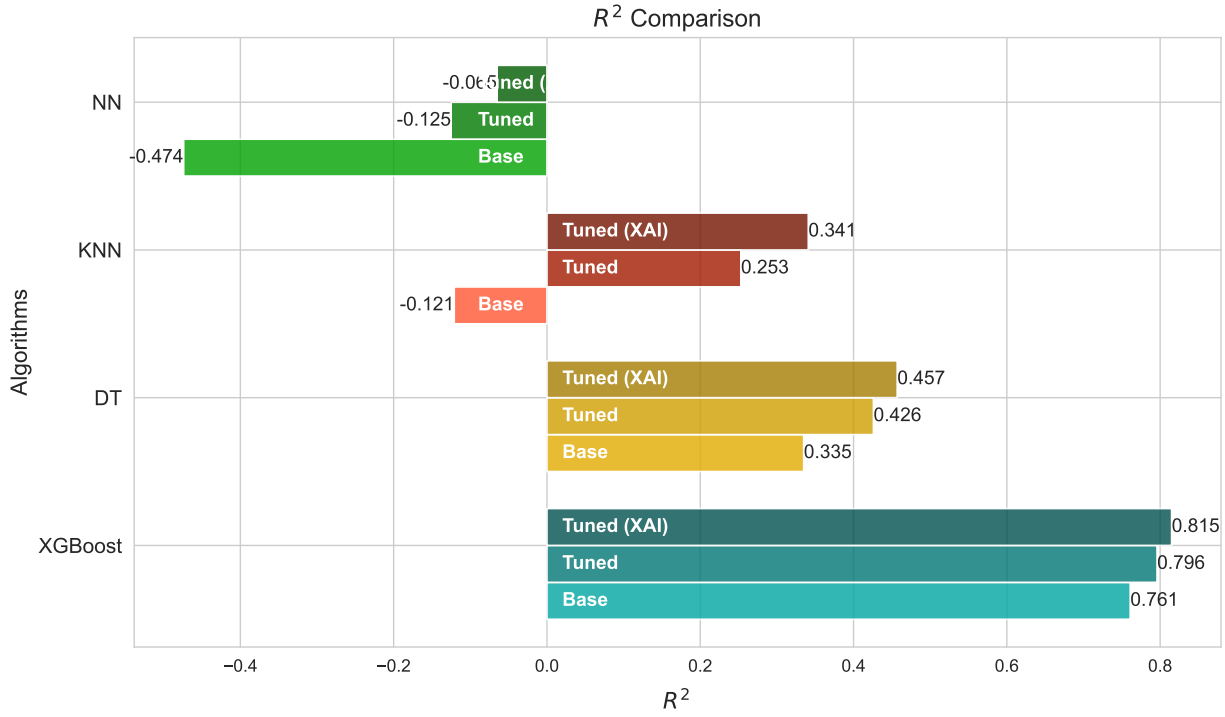
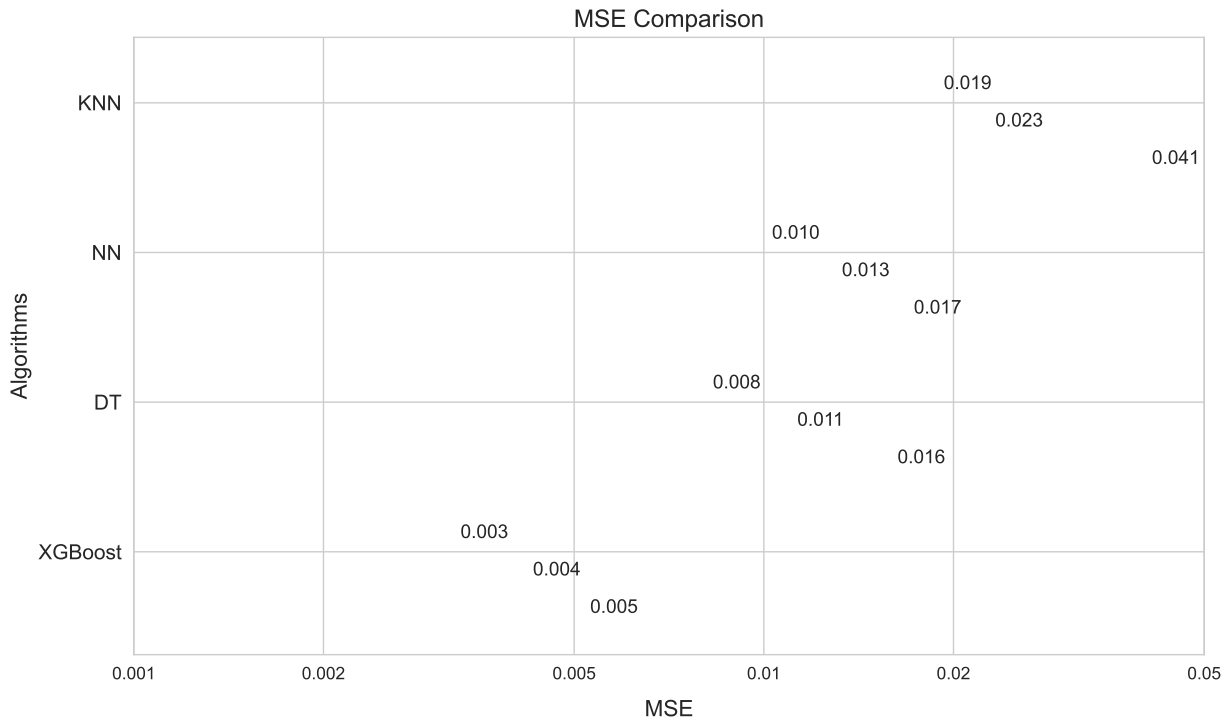


Figure XIV: MSE Comparison

This figure presents a comparison of the out-of-sample MSE values among base, tuned, and tuned (XAI) models for four machine learning algorithms, arranged in ascending order by the MSE value. The sample period is from 1957 to 2022.



different tuned machine learning models to obtain a more thorough understanding of the effectiveness of these machine learning algorithms in predicting stock returns.

We construct portfolios that use the machine learning predictions to make stock selections. On a monthly basis, each machine learning model is used to forecast stock returns for the upcoming month. We then use these predictions to categorize stocks into deciles, constructing ten portfolios based on the predicted excess returns from each specific machine learning model. There is a monthly rebalance for the different ten portfolios based on our machine learning prediction. We finally equal-weight each portfolio and calculate each portfolio's excess return and standard deviation.

We implement a long-short portfolio strategy by buying the stocks with the highest predicted returns (decile 10) and by shorting the stocks with the lowest predicted returns (decile 1). This strategy enables us to assess the practical investment performance of these machine learning models. We then use Sharpe Ratio for the long-short portfolios to measure the investment performance of our models. Sharpe Ratio, proposed by SHARPE (1964), is a commonly used metric to measure risk-adjusted returns. The higher Sharpe Ratio, the better investment performance. The tuned Sharpe Ratio is calculated based on the tune (XAI) model, in which we select tuning parameters adaptively using the results on XAI.

Fig. XV shows the Sharpe ratios for base models, tuned models, and XAI-tuned models based on XAI interpretations. All models have the lowest Sharpe ratio for the base models and the highest Sharpe ratio for the XAI-tuned models. This suggests that XAI is highly effective in improving the models to achieve higher investment performance.

Among the four machine learning models, XGBoost has the highest Sharpe Ratio, followed by the decision tree with a slightly lower ratio. Specifically, the Sharpe Ratio of the XGBoost base model is 1.568, which increases to 1.618 after tuning, and reaches 1.670 after XAI-based tuning. Second, the decision tree achieves a Sharpe ratio of 1.635 after XAI-based tuning. The difference in investment performance between XGBoost and the decision tree is marginal. This aligns with the interpretations from SHAP and LIME analyses, which indicate that

XGBoost and the decision tree identify similar characteristics as key features for prediction.

K-nearest neighbors have a much lower Sharpe Ratio, at 0.762 for the XAI-tuned model, while the neural network has a negative Sharpe Ratio. The investment performance of the different tuned models is consistent with the R^2 and MSE results. It suggests that the key characteristics selected by K-nearest neighbors and neural networks are not particularly effective for return predictions.

The Sharpe Ratios achieved using XAI techniques to understand the importance of feature selection are significant. Frazzini et al. (2018) found that the Sharpe Ratio of Warren Buffett's Berkshire Hathaway is 0.79, while the U.S. market's Sharpe Ratio is 0.49. The Sharpe Ratio of 1.67 achieved by XGBoost is much higher than the top performer Berkshire Hathaway, indicating that the application of machine learning models in conjunction with XAI offers substantial benefits in asset management.

In summary, the observed enhancements across various models - from their base forms through to their tuned and XAI-based iterations - supports the hypothesis that a better understanding of a model's internal mechanisms significantly increases its predictive power and investment performance. This performance improvement underscores the important role of XAI in refining machine learning models, in the context of asset pricing, where model interpretations can lead to more precise predictions and higher investment performance.

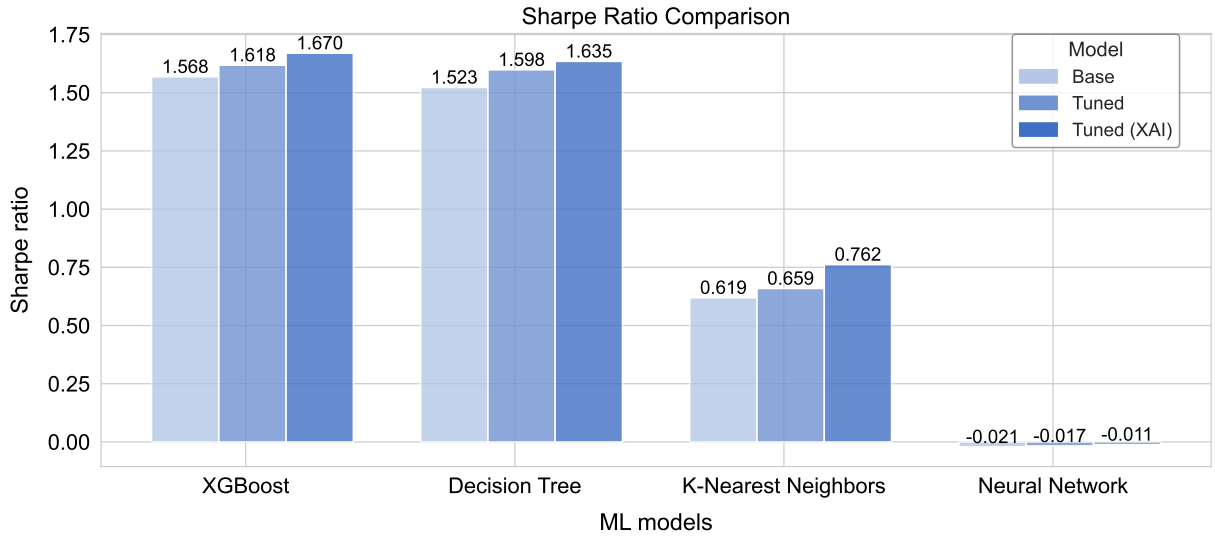
Finally, we compare the testing times among the base, tuned, and XAI-tuned versions of four machine learning models. The fast-paced changes in financial markets requires real-time responses. A quicker machine learning algorithm can enable valuable insights more actionable. For robustness and statistical reliability, we run each algorithm five times and analyze the resulting testing times using standard deviations, the same as Demirbaga and Xu (2023).

Figure XVI presents the testing times for the four machine learning models in ascending order. We find that all models exhibit substantial time savings after tuning and demonstrate further increased time efficiency after tuning based on XAI interpretations.

In particular, the decision tree has the lowest testing time among all four models (0.051s).

Figure XV: Sharpe Ratio Comparison

This figure presents a comparison of the out-of-sample Sharpe Ratios among base, tuned, and tuned (XAI) models for four machine learning algorithms, arranged in ascending order by the Sharpe Ratios. The sample period is from 1957 to 2022.



The testing time decreases to 0.039s after tuning and further diminishes to 0.024 seconds after XAI-based tuning. This reduction is more than half when using XAI interpretations. Halving the testing time can significantly impact investment performance by enabling more prompt investment decisions.

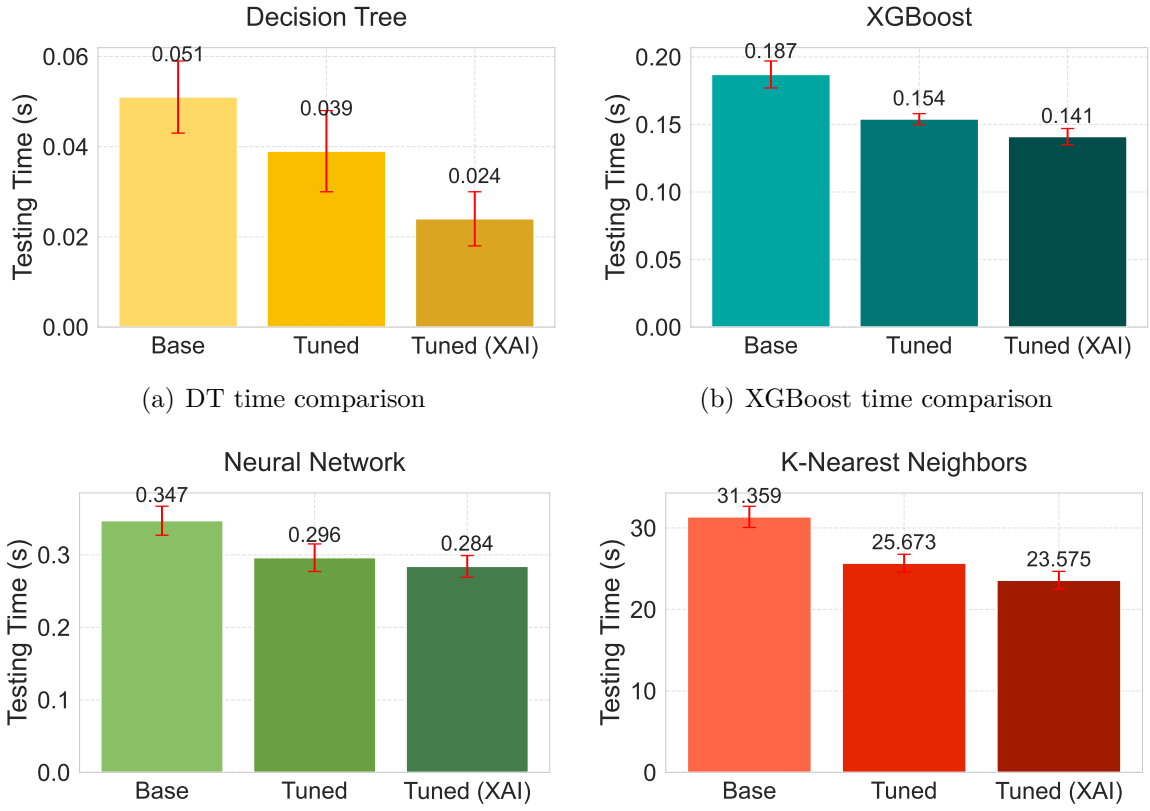
The second most time-efficient model is XGBoost. Its testing time decreases from 0.187 seconds for the base model to 0.154 seconds for the tuned model, and to 0.141 seconds for the XAI-tuned model, which represents an approximate 25% decrease in testing time.

The neural network is slightly more time-intensive than XGBoost but also reduces its testing time after XAI-based tuning. The K-nearest neighbors model is among the most time-consuming models (31.359s for the base) but also decreases its time to 23.575s for the XAI-tuned model.

To conclude, the consistent results of improved investment performance and reduced testing times suggest that applying XAI in investments can significantly enhance investment returns for investors and improve time efficiency, enabling more timely investment decisions.

Figure XVI: Testing Time Comparison

This figure presents a comparison of the testing time among base, tuned, and tuned (XAI) models for four machine learning algorithms, arranged in ascending order by the testing time. The sample period is from 1957 to 2022.



(a) DT time comparison

(b) XGBoost time comparison

(c) NN time comparison

(d) KNN time comparison

VI. Conclusion

This paper shows the substantial potential of integrating machine learning models with explainable AI in asset pricing. Explainable AI, through LIME and SHAP analyses, enhances the interpretability of complex models and helps us identify key determinants of stock returns, such as momentum and volatility. Our approach of using explainable AI to refine and enhance machine learning models has improved their predictive accuracy and overall investment performance.

Explainable AI opens doors for future research in various finance areas. For example, investors can use machine learning models to incorporate thousands of signals derived from different data sources, including textual and satellite data. Explainable AI can clarify which signals are important and explain these decisions to stakeholders. In addition, with the growing importance of environmental, social, and governance (ESG), explainable AI can interpret how different ESG factors influence investment decisions. It's also valuable for understanding the dynamics in mergers and acquisitions in corporate finance. The broad application of XAI stands to revolutionize understanding and decision-making processes across different areas of finance.

References

Ali, A., L. S. Hwang, and M. A. Trombley (2003). Arbitrage risk and the book-to-market anomaly. *Journal of Financial Economics 69*(2), 355–373.

Ang, A., R. J. Hodrick, Y. Xing, and X. Zhang (2006). The Cross-Section of Volatility. *The Journal of Finance LXI*(1), 259–299.

Ariza-Garzon, M. J., J. Arroyo, A. Caparrini, and M. J. Segovia-Vargas (2020). Explainability of a Machine Learning Granting Scoring Model in Peer-to-Peer Lending. *IEEE Access 8*, 64873–64890.

Baehrens, D., T. Schroeter, S. Harmeling, M. Kawanabe, K. Hansen, and K. R. Müller (2010). How to explain individual classification decisions. *Journal of Machine Learning Research 11*, 1803–1831.

Bali, T. G., H. Beckmeyer, M. Moerke, and F. Weigert (2021). Option Return Predictability with Machine Learning and Big Data. *SSRN Electronic Journal 36*(202), 3548–3602.

Bali, T. G., N. Cakici, and R. F. Whitelaw (2011). Maxing out: Stocks as lotteries and the cross-section of expected returns. *Journal of Financial Economics 99*(2), 427–446.

BARRY, C. B. and L. T. STARKS (1984). Investment Management and Risk Sharing with Multiple Managers. *The Journal of Finance 39*(2), 477–491.

Belo, F., X. Lin, and M. A. Vitorino (2014). Brand capital and firm value. *Review of Economic Dynamics 17*(1), 150–169.

Bianchi, D., M. Büchner, and A. Tamoni (2021). Bond Risk Premiums with Machine Learning. *Review of Financial Studies 34*(2), 1046–1089.

Bussmann, N., P. Giudici, D. Marinelli, and J. Papenbrock (2021). Explainable Machine Learning in Credit Risk Management. *Computational Economics 57*(1), 203–216.

Chen, A. Y. and T. Zimmermann (2022). Open Source Cross-Sectional Asset Pricing. *Critical Finance Review*.

Chen, L., M. Pelger, and J. Zhu (2023). Deep Learning in Asset Pricing. *Management Science*.

Dechow, P. M. (2001). Short-sellers, fundamental analysis, and stock returns. *Journal of Financial Economics 61*(1), 77–106.

Demirbaga, U. and Y. Xu (2023). Machine Learning Execution Time in Asset Pricing. *SSRN Electronic Journal*.

Dwivedi, R., D. Dave, H. Naik, S. Singhal, R. Omer, P. Patel, B. Qian, Z. Wen, T. Shah, G. Morgan, and R. Ranjan (2023). Explainable AI (XAI): Core Ideas, Techniques, and Solutions. *ACM Computing Surveys 55*(9).

Emmanuel, T., T. Maupong, D. Mpoeleng, T. Semong, B. Mphago, and O. Tabona (2021). *A survey on missing data in machine learning*, Volume 8. Springer International Publishing.

Fama, E. F. and K. R. French (1992). The Cross-Section of Expected Stock Returns. *The Journal of Finance 47*(2), 427–465.

Fama, E. F. and K. R. French (2006). Profitability, investment and average returns. *Journal of Financial Economics 82*(3), 491–518.

Feng, G., S. Giglio, and D. Xiu (2020). Taming the Factor Zoo: A Test of New Factors. *Journal of Finance 75*(3), 1327–1370.

Frazzini, A., D. Kabiller, and L. H. Pedersen (2018). Buffett's Alpha. *Financial Analysts Journal 74*(4), 35–55.

Freyberger, J., A. Neuhierl, and M. Weber (2020). Dissecting Characteristics Nonparametrically. *The Review of Financial Studies 33*(5), 2326–2377.

George, T. J. and C. Y. Hwang (2004). The 52-week high and momentum investing. *Journal of Finance 59*(5), 2145–2176.

Gu, S., B. Kelly, and D. Xiu (2020). Empirical Asset Pricing via Machine Learning. *Review of Financial Studies 33*(5), 2223–2273.

Guresen, E. and G. Kayakutlu (2011). Definition of Artificial Neural Networks with comparison to other networks. *Procedia Computer Science 3*, 426–433.

Hadji Misheva, B., A. Hirsa, J. Osterrieder, O. Kulkarni, and S. Fung Lin (2021). Explainable AI in Credit Risk Management. *SSRN Electronic Journal* (September), 1–16.

Harvey, C. R. and Y. Liu (2021). Lucky factors. *Journal of Financial Economics 141*(2), 413–435.

Harvey, C. R., Y. Liu, and H. Zhu (2016). âŃŕ and the Cross-Section of Expected Returns. *Review of Financial Studies 29*(1), 5–68.

Heston, S. L. and R. Sadka (2008). Seasonality in the cross-section of stock returns. *Journal of Financial Economics 87*(2), 418–445.

Hou, K. (2007). Industry information diffusion and the lead-lag effect in stock returns. *Review of Financial Studies 20*(4), 1113–1138.

Imandoust, S. B. and M. Bolandraftar (2013). Application of K-Nearest Neighbor (KNN) Approach for Predicting Economic Events : Theoretical Background. *Int. Journal of Engineering Research and Applications 3*(5), 605–610.

Invesco (2023). 2023 Invesco Global Systematic Investing Study. Technical report.

Jegadeesh, N. and S. Titman (1993). Returns to Buying Winners and Selling Losers: Implications for Stock Market Efficiency. *The Journal of Finance 48*(1), 65–91.

Kelly, B. T., S. Pruitt, and Y. Su (2019). Characteristics are covariances: A unified model of risk and return. *Journal of Financial Economics 134*(3), 501–524.

Kotsiantis, S. B. (2013). Decision trees: A recent overview. *Artificial Intelligence Review 39*(4), 261–283.

Kozak, S., S. Nagel, and S. Santosh (2020). Shrinking the cross-section. *Journal of Financial Economics 135*(2), 271–292.

Lakonishok, J., A. Shleifer, and R. W. Vishny (1994). Contrarian investment, extrapolation, and risk. In *Journal of Finance*, Volume 2, pp. 273–316.

Lettau, M. and M. Pelger (2020). Factors That Fit the Time Series and Cross-Section of Stock Returns. *The Review of Financial Studies 33*(5), 2274–2325.

Liu, W. (2006). A liquidity-augmented capital asset pricing model. *Journal of Financial Economics 82*(3), 631–671.

Lundberg, S. M. and S. I. Lee (2017). A unified approach to interpreting model predictions. In *Advances in Neural Information Processing Systems*, Volume 2017-Decem, pp. 4766–4775.

Novy-Marx, R. (2012). Is momentum really momentum? *Journal of Financial Economics 103*(3), 429–453.

Ribeiro, M. T., S. Singh, and C. Guestrin (2016). "Why should i trust you?" Explaining the predictions of any classifier. *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining 13-17-Augu*, 1135–1144.

SHARPE, W. F. (1964). CAPITAL ASSET PRICES: A THEORY OF MARKET EQUILIBRIUM UNDER CONDITIONS OF RISK. *The Journal of Finance XIX*, 425–442.

Young, H. P. (1985). Monotonic solutions of cooperative games. *International Journal of Game Theory 14*(2), 65–72.

Appendix

A.1. Correlations Among Top Features

We show the correlations among top features by different machine learning models using heatmaps. The heatmap is a valuable tool to understand the relationships between features. Figure A.1 visualizes the relationship between the top 20 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 a 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 A.2, A.3, and A.4 show the same heatmaps but with different color visulisations.

We find that XGBoost and decision tree have the similar top 20 features. This is consistent with the XAI results in Section IV. Momentum with different horizons have high correlations. Volatility and return-related characteristics (i.e., high52 and maxret) are also highly correlated. For K-nearest neighbors and neural networks, characteristics are mostly accounting-related and are not highly correlated.

One would argue that some top features are highly correlated in XGBoost, such as mom12 and mom6m, exhibit high correlation. In our research, we have chosen not to exclude correlated features prior to applying machine learning models. Removing these features beforehand could lead to inconsistencies between models, as it might result in the exclusion of a feature that is more influential in one model over a correlated counterpart in another. Our black-box models are not affected by correlation among features in the following ways.

First, due to their non-linearity, black-box models often operate in a non-linear space. They can capture complex relationships and interactions among variables without relying on linear assumptions, making them more robust to multicollinearity.

Second, many black-box models have built-in mechanisms for feature selection. For example, decision trees can naturally select relevant features based on their information gain. This inherent feature selection helps the model focus on the most informative variables and can reduce the impact of multicollinearity.

Third, some black-box models incorporate regularization techniques to prevent overfitting. Regularization penalizes overly complex models, and this can help control the influence of correlated variables.

Finally, black-box models often have the ability to implicitly perform feature engineering through the learning process. For example, neural networks with hidden layers can automatically learn representations that capture the underlying patterns in the data, including handling correlated features.

Figure A.1: Heatmaps-XGBoost

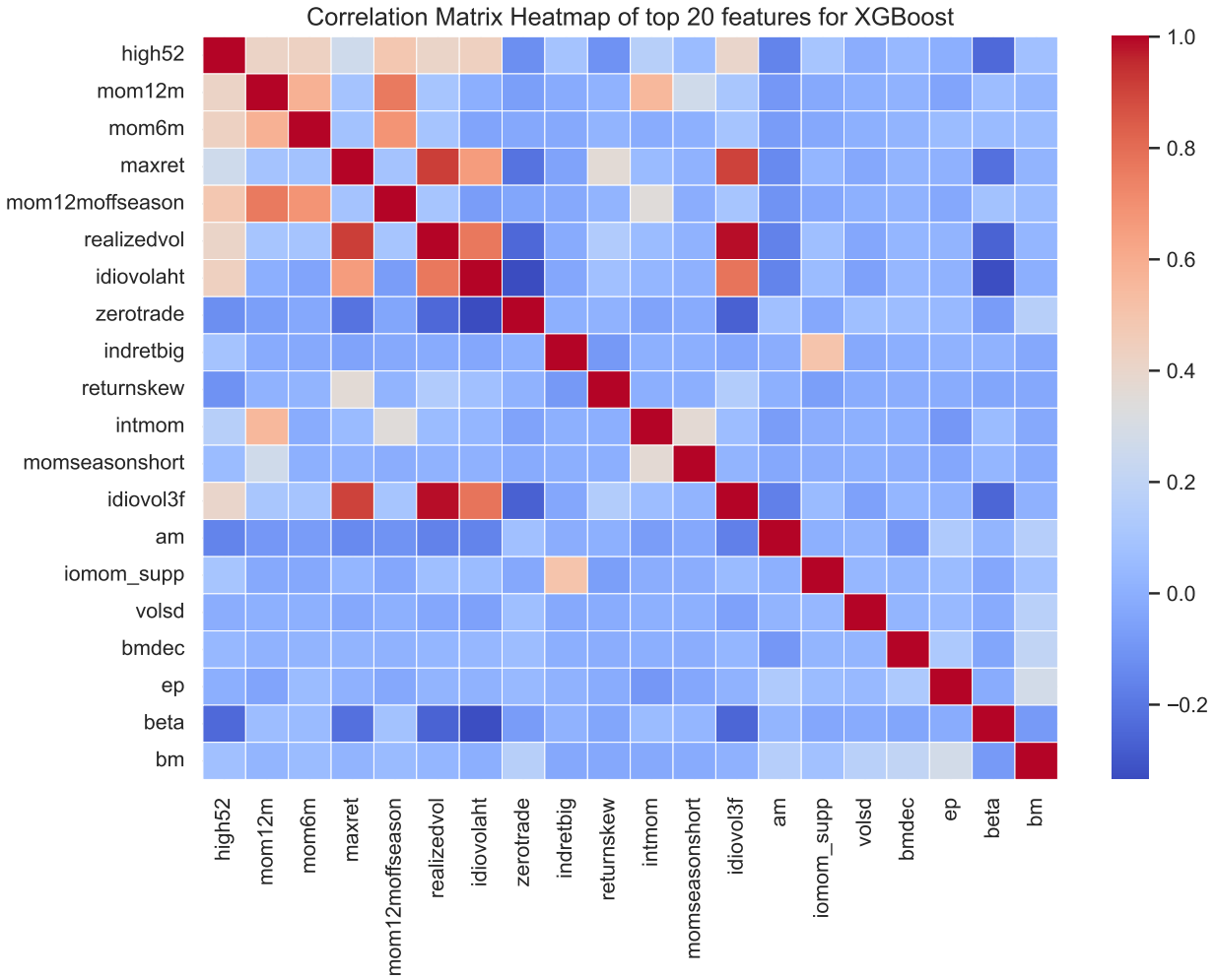


Figure A.2: Heatmaps-Decision Tree

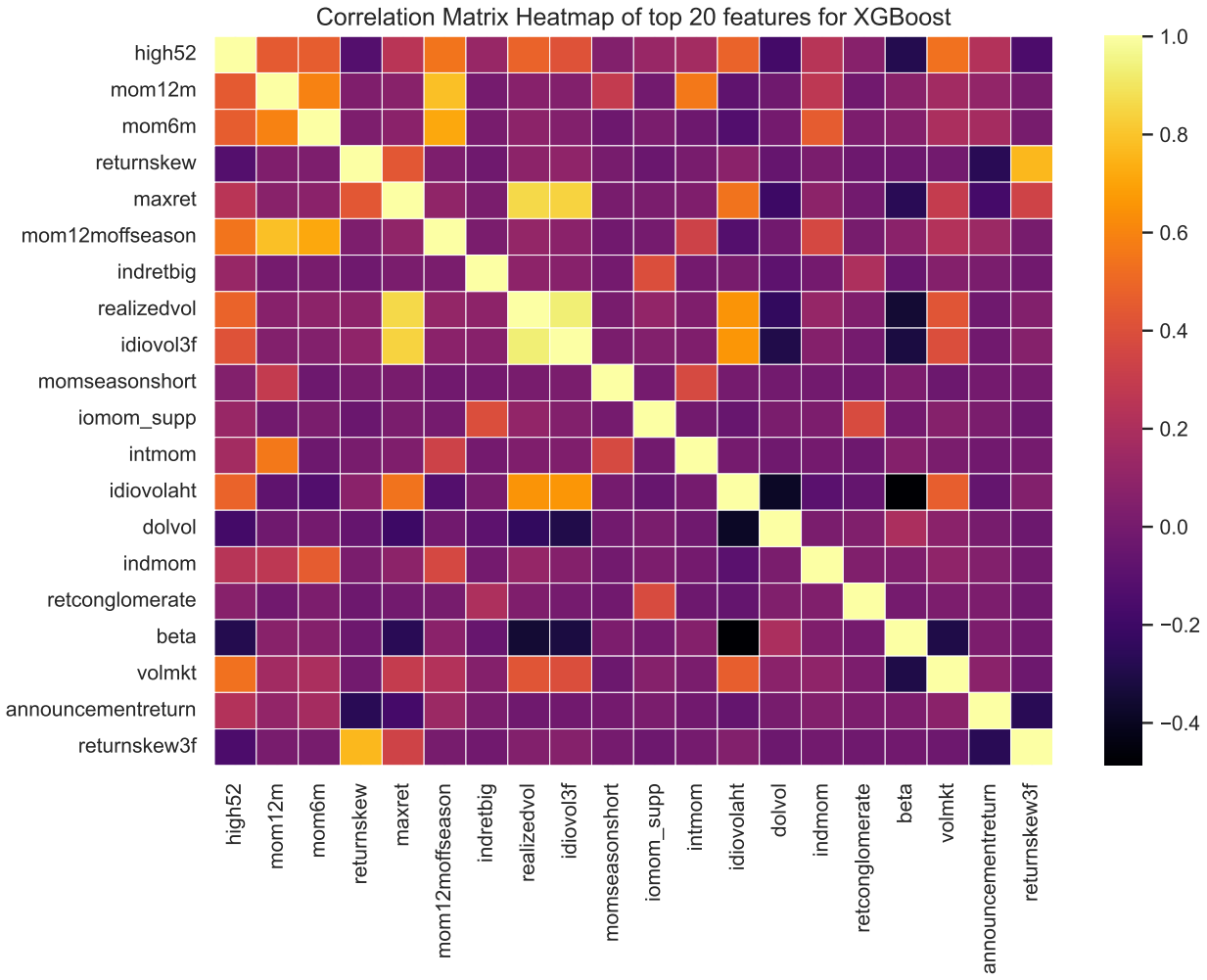


Figure A.3: Heatmaps-KNN

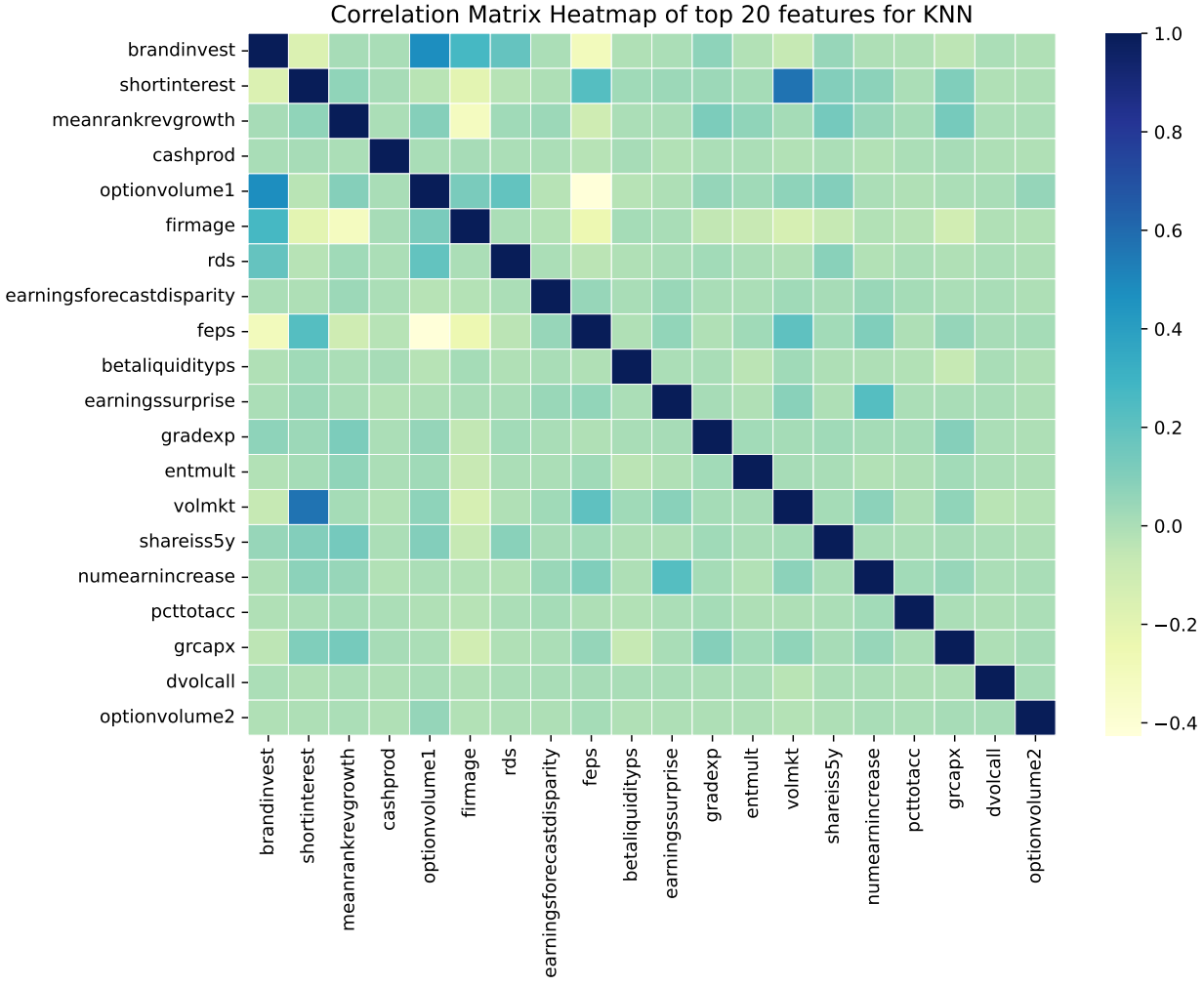
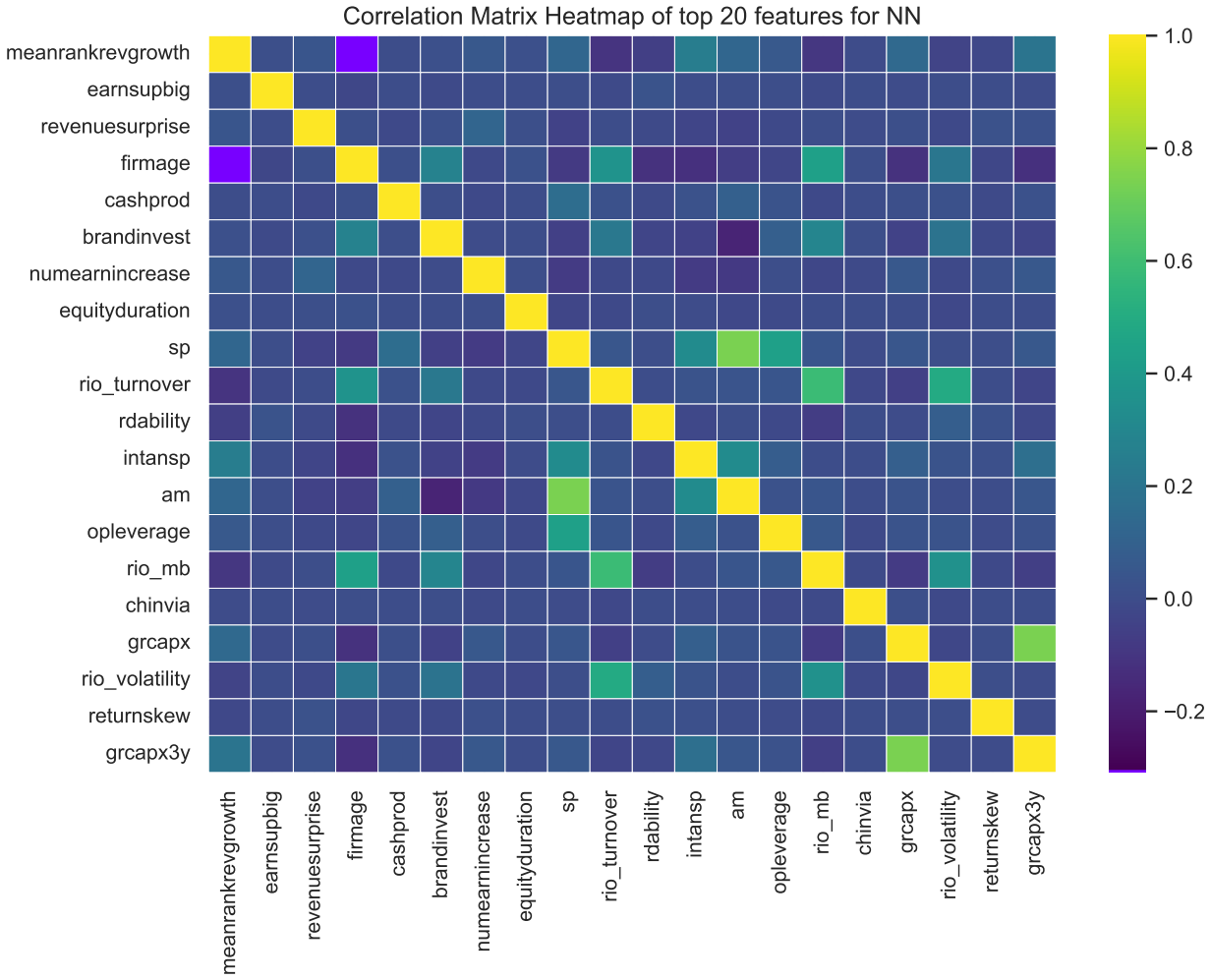


Figure A.4: Heatmaps-NN



A.2. Feature Weight Optimization - SLSQP Algorithm

Algorithm 1: Feature Weight Optimization using SLSQP

Data: feature_names[]: List of feature names
$\boldsymbol{X}_{\text{train}} \in \mathbb{R}^{n_{\text{train}} \times m}$: Training data matrix
$\boldsymbol{y}_{\text{train}} \in \mathbb{R}^{n_{\text{train}}}$: Training target data
$\boldsymbol{X}_{\text{test}} \in \mathbb{R}^{n_{\text{test}} \times m}$: Testing data matrix
$\boldsymbol{y}_{\text{test}} \in \mathbb{R}^{n_{\text{test}}}$: Testing target data
$\boldsymbol{w}_{\text{init}}[]$: Initial feature weights
constraints[]: Constraint definitions

Result: $\boldsymbol{w}_{\text{opt}}[]$: Optimal feature weights

Function `optimizeFeatureWeights(`*feature_names*$[], \boldsymbol{X}_{train}, \boldsymbol{y}_{train}, \boldsymbol{X}_{test}, \boldsymbol{y}_{test}, \boldsymbol{w}_{init}[]$, *constraints*$[]$`):`
  //Initialize optimization variables
  $\boldsymbol{w}_{\text{opt}}[] \leftarrow \boldsymbol{w}_{\text{init}}[]$
  //Define custom objective function to minimize
  DefineObjectiveFunction(feature_names$[], \boldsymbol{X}_{\text{train}}, \boldsymbol{y}_{\text{train}}, \boldsymbol{X}_{\text{test}}, \boldsymbol{y}_{\text{test}}$)
  //Define constraints for optimization
  DefineConstraints($\boldsymbol{w}_{\text{opt}}[]$)
  //Perform the optimization using SLSQP
  result $\leftarrow$ SLSQP(objective_function, $\boldsymbol{w}_{\text{opt}}[]$, constraints$[]$, method $=' SLSQP'$)
  //Extract the optimal feature weights
  $\boldsymbol{w}_{\text{opt}}[] \leftarrow$ result.x[]
  return $\boldsymbol{w}_{\text{opt}}[]$ Optimal feature weights
//Define the objective function for feature weight optimization
DefineObjectiveFunction(feature_names$[], \boldsymbol{X}_{\text{train}}, \boldsymbol{y}_{\text{train}}, \boldsymbol{X}_{\text{test}}, \boldsymbol{y}_{\text{test}}$)
//Define constraints for optimization
DefineConstraints($\boldsymbol{w}_{\text{opt}}[]$)