Mutual funds out-of-sample performance and machine learning

FINA0063-1 : Advanced Statistical Methods in Finance

Professor

J. Hambuckers

Authors

Lausanne Ericka Pesseuh Ngueffo - s218213

Wilfried Mvomo Eto - s226625

Academic year 2023 - 2024

Abstract

This study uses machine learning methodologies to discern pooled fund investment managers. Our results highlight the predictive power of machine learning models in predicting performance, revealing nuanced interactions between fund characteristics and future performance. By focusing on the application of machine learning to predict positive alpha, this research contributes to the growing importance of sophisticated predictive methods in improving portfolio management decisions.

Contents

1. Introduction

In this comprehensive exploration of machine learning methodologies in finance, specifically tailored for mutual fund selection and out-of-sample alpha prediction, we underscore the nonlinearity between covariates and its profound impact on forecasting alpha. Our approach integrates sophisticated machine learning techniques, emphasizing the versatile nature of these models and their ability to capture complex relationships within financial datasets.

The term "machine learning" within our context encompasses a wide spectrum, including (i) a diverse collection of high-dimensional models designed for statistical prediction, (ii) regularization methods for effective model selection and overfit mitigation, and (iii) efficient algorithms for exploring potential model specifications [2]. This broad definition allows us to harness the flexibility of high-dimensional machine learning models, surpassing traditional econometric techniques. It holds great promise in approximating the intricate data-generating process underlying equity risk premia. Notably, we address the challenge of overfitting through a commitment to stable out-of-sample performance, highlighting the significance of employing clever tools for optimal model specification.

In certain literature, identifying outperforming funds ex ante is notoriously difficult. Machine-learning methods, leveraging nonlinearities and interactions in the relation between fund characteristics and performance, can construct tradable long-only portfolios, earning significant out-of-sample alphas. Investors can achieve economically significant alpha by investing in active mutual funds, provided they have access to sophisticated prediction methods. The study explores the economic mechanism, revealing that machine learning helps select outperforming funds by identifying skilled managers and those not offset by diseconomies of scale [3].

Moreover, the research introduces a new set of benchmarks for measuring risk premia using machine learning. These methods demonstrate high out-of-sample predictive R^2 and significant economic gains for investors. Neural network forecasts outperform buy-and-hold strategies, showcasing the flexibility and effectiveness of machine learning in empirical asset pricing. The study emphasizes the importance of understanding the complexity of risk premium measurement and its application in predicting returns [1].

The asset management industry's enormous growth prompts a reassessment of actively-traded equity mutual funds. Utilizing an artificial neural network model, the study identifies fund flows and fund return momentum as key predictors of mutual fund outperformance. The model's predictions generate a substantial difference in out-of-sample performance, highlighting the economic significance of predicting fund performance. The results suggest that investors can detect skill and reallocate investments towards skilled managers, with flows gradually influencing fund return momentum [4].

Our multifaceted approach extends beyond traditional methodologies. First, we conduct an in-depth analysis of the features of mutual funds, navigating challenges within the Morningstar dataset, featuring 15 weakly correlated covariates. Additionally, we explore the augmentation of Sentiment factors and Momentum factor to enhance correlation power, creating additional predictors and deriving the realized alpha of interest.

Second, following variable selection through LASSO, our analysis extends to a comprehensive forecast using machine-learning methods, with a specific emphasis on the unique capabilities of

neural networks. These methods not only showcase their ability to capture intricate relationships but also demonstrate practical utility for portfolio managers. Detecting positive alpha emerges as a crucial goal for portfolio managers seeking to outperform the market.

Third, amidst our investigation into mutual fund research, a persistent observation emerges—negative risk-adjusted returns (alpha) after accounting for transaction costs. To tackle this challenge, we draw inspiration from the methodology outlined by Kaniel et al. [4]. We employ a repertoire of machine-learning methods, including the elastic net, gradient boosting, random forests, and neural networks, to forecast fund performance with a specific focus on alpha prediction. Elastic Net, in particular, stands out as a powerful machine learning model, combining the strengths of both LASSO and Ridge regression, providing a balance between variable selection and regularization.

Finally, we conduct an in-depth analysis of variable importance and marginal relations. The multifaceted framework empowers portfolio managers with insights into the intricate world of alpha prediction. By seamlessly integrating machine learning, variable selection methodologies, and tradable portfolio construction, our approach, inspired by Kaniel et al. [4], provides a comprehensive toolset for portfolio managers striving to detect positive alpha and make well-informed investment decisions.

2. Mutual-fund characteristics

i. Morningstar Dataset

In our exploratory analysis, we sourced the dataset from the Morningstar database. The dataset comprises 330,403 monthly observations related to mutual funds, spanning from March 2007 to September 2023. Funds are categorized into 8 groups based on similar categories and investing styles. The dataset includes 15 features, consisting of 6 quantitative and 6 qualitative features. To understand the relationships between these quantitative variables, we present the correlation matrix at this level in Figure 15.

All correlation values in this matrix are close to zero, except for 'Longest.Manager.Tenure' and 'Average.Manager.Tenure.' This is expected, as these metrics focus on assessing the sustainability of the manager over time. The predominance of weak linear relationships among the mutual fund characteristics hampers machine learning models in effectively fitting the data for better predictions on the target variable, as discussed in the subsequent paragraphs.

To overcome this limitation and align with the analysis by Kaniel et al.[4], we introduced additional macro-financial features, specifically incorporating the five Fama and French factors (2015) and momentum factors into the dataset. This augmentation aims to strengthen the linear relationships among certain quantitative features and also facilitates the generation of additional features, following a similar approach to Kaniel et al.'s article [4]. This will be further developed in the next paragraph.

ii. The Five Fama and French Factors (2015) and Momentum Factors

According to Wikipedia, the Fama-French three-factor model provides an empirical explanation for the expected return of a financial asset.

The Fama and French model aligns with the Arbitrage Pricing Theory (APT), asserting that the expected return of a financial asset is a linear function of its sensitivity to risk factors. It delineates the three key factors for consideration.

Estimates by Fama and French reveal that the model can account for a substantial portion (approximately 90%) of the monthly return variation in stocks on the NYSE, AMEX, and NASDAQ exchanges.

The Fama and French factors, in conjunction with momentum factors, play a pivotal role in financial modeling, capturing diverse aspects of stock and investment performance. The Fama-French five-factor model extends the original three factors (Market Risk (Mkt-RF), Size Factor (SMB), Value Factor (HML)) by incorporating two additional factors (Profitability Factor (RMW), Investment Factor (CMA)). This model offers a comprehensive framework for comprehending and elucidating stock returns.

In a factor model, the momentum factor is often represented as "MOM" and is computed as the return of past winners minus the return of past losers, providing insights into short-term trends in asset prices.

These additional variables are extracted monthly, allowing seamless integration with the Morningstar database. They can be accessed via the https://mba.tuck.dartmouth.edu/pages/faculty/

ken.french/data_library.html link. Subsequently, variables such as abnormal returns and betas associated with each macroeconomic factor obtained through linear regression, as well as the investment duration, can be created.

iii. Dealing with Outlying Observations

Before engineering new features, our dataset must undergo cleaning. Modeling data with outliers can lead to a corrupted covariance matrix, introducing bias into the analysis. In others words, outliers can cause significant variations in estimators during the modeling process. In this project, box plot is a tool that enables to detect outlying observations. One can observe figure 1, values beyond the upper extreme bar or below the lower extreme bar in the box plots highlights the presence of the outliers in the dataset. Any observation whose at least one variable value is greater than the upper extreme bar or smaller than the lower extreme bar of the corresponding variable box plot is considered an outlier.

$$\text{IQR} = \text{Q3} - \text{Q1}, \quad \text{Upper Bound} = \text{Q3} + (1.5 \times \text{IQR}), \quad \text{Lower Bound} = \text{Q1} - (1.5 \times \text{IQR}) \quad (1)$$

where Q3 and Q1 denote the third quartile and the first quartile, respectively.

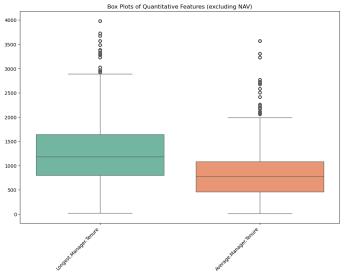


Figure 1: Box plots highlighting outliers in the 'Longest.Manager.Tenure' and 'Average.Manger.Tenure' variables.

After processing our dataset, we now have 177,033 observations and 20 features. In the following, we'll use the variable Return, representing Morningstar's calculation of total return. This is determined each month by taking the change in monthly net asset value, reinvesting all income and capital-gains distributions during that month, and dividing by the starting NAV. We'll also use the Five Factors Fama-French and Momentum to generate, using the linear regression model, the variables beta and monthly realized alpha.

iv. Realized Alpha

Firstly, the lagged values of these predictors are used to predict future excess returns (i.e., to predict the excess return $E(r_{it} - r_t^f)$, only x_{t-j} for $j > 0$ is used). This will be used as the dependent variable in the linear regression modeling process to estimate the realized alpha and betas, which will be monthly as they are extracted from monthly data. More precisely, we rely on the following equation for determining the monthly realized alpha for the i^{th} share class in the m^{th} month $\alpha_{i,m}$[3]:

$$\boldsymbol{\alpha_{i,m}} = \mathbf{r_{i,m}} - \hat{\boldsymbol{\beta}}_{\mathbf{MKT,i,m}}\mathbf{MKT_m} - \hat{\boldsymbol{\beta}}_{\mathbf{SMB,i,m}}\mathbf{SMB_m}$$

$$-\hat{\beta}_{\mathbf{HML,i,m}}\mathbf{HML_m} - \hat{\beta}_{\mathbf{RMW,i,m}}\mathbf{RMW_m} - \hat{\beta}_{\mathbf{CMA,i,m}}\mathbf{CMA_m} - \hat{\beta}_{\mathbf{MOM,i,m}}\mathbf{MOM_m} (2)$$

$\beta_{\mathbf{MKT_{i,m}}}$, $\beta_{\mathbf{SMB_{i,m}}}$, $\beta_{\mathbf{HML_{i,m}}}$, $\beta_{\mathbf{RMW_{i,m}}}$, $\beta_{\mathbf{CMA_{i,m}}}$, $\beta_{\mathbf{MOM_{i,m}}}$ are the factor loadings of the i-th share class excess return with respect to the FF5+MOM factors are estimated.These factor loadings represent the relationship between the excess return of the i-th share class and the FF5+MOM factors. We then consolidate the relevant summaries of all our quantitative covariates in Table 3, where we observe that the mean monthly realized alpha is equal to 0.2954%. Additional details on the features of our updated database are provided in Table 5, where we now have 30 characteristics.

Furthermore, we obtained a Pearson correlation matrix (Figure 16) where several covariates exhibit some correlation, although these values are not very strong. This enables machine learning algorithms to adapt to the data and achieve better model performance. Additionally, we include graphs illustrating the temporal evolution of our variables (Figure 17).

Although we can transform categorical data into numerical values, we will not use them in the modeling stage, as we did in the reference article. We will rely mainly on categorical variables as follows: the dependent variable is the Realized Alpha previously determined, and the other variables will be considered as predictors.

Given all the characteristics of our database, the question arises: which ones should be included in the modeling process to facilitate the adaptation of our models, capturing the nonlinearities among covariates, and obtaining effective results with greater learning efficiency? We will resort to LASSO which guides us in selecting the most impactful covariates for predicting Realized Alpha, considering the intricacies and nonlinearities inherent in our dataset.

3. Variable selection : LASSO

To address this, we will employ LASSO (Least Absolute Shrinkage and Selection Operator). LASSO is particularly well-suited for feature selection, helping us identify the most relevant covariates for predicting the Realized Alpha target variable. By applying LASSO regularization, we aim to introduce sparsity in the model, encouraging some coefficients to become exactly zero. This sparsity-inducing property of LASSO makes it a powerful tool for variable selection and handling high-dimensional datasets.

The chosen features after LASSO regularization will not only simplify the model but also enhance its interpretability by focusing on the most influential variables. This approach aligns with the goal of capturing nonlinearities in the data while maintaining model efficiency.

LASSO is a linear regression technique that introduces a regularization term to the linear regression objective function. It uses the absolute values of the coefficients (L1 norm).Roughly speaking, LASSO looks for :

$$\hat{\beta}^{LASSO} = \underset{\beta_0,\beta}{arg\,min} \left[\sum_{i=1}^{n} \left(y_i - \beta_0 - x_i^T \beta \right)^2 + \lambda \sum_{j=1}^{p} w_j |\beta_j| \right]$$

where n and p denote the number of observations in the dataset and the number of covariates, respectively; λ represents the penalty parameter, and $(x_i, y_i)_{i=1,\dots,n}$ ($x_i \in \mathbb{R}^p$ and $y_i \in \mathbb{R}$) denotes the input-output pairs of the dataset.

i. Data Preparation

Previously, when building the linear regression model, we did not standardize our predictors in order to generate new variables and preserve the original information. Additionally, we did not emphasize the process of paratagging the dataset, which will be detailed in one of the next chapters.

When using LASSO, it is advisable to center and scale the variables in regularized regression so that the penalty coefficient λ acts uniformly across all regression coefficients. In other words, we standardize the predictors.

The transformation formula is given for each individual i and variable X_j:

$$z_{ij} = \frac{x_{ij} - \bar{x}_j}{\sigma_j}$$

Where $\bar{x}_j$ and σ_j are, respectively, the mean and the standard deviation (square root of variance) of the variable X_j, calculated on the formation sample. We do not focus on the process of splitting the dataset in this section, which will be detailed in one of the following sections.

ii. Lasso Path

When λ is too high, all the coefficients of the regression are zero, we have a illustration here ; when λ is too low, close to 0, we get the coefficients of the usual multiple linear regression. It is necessary to find the right balance and that is the whole difficulty of Lasso regression. The “Lasso path” tool can help us. It produces a graph that puts in relation to the different versions of α with the estimated coefficients. In this way, we see scenarios for solutions emerging (at meaning "different sets of selected variables") along the "Lasso path" (figure 2).

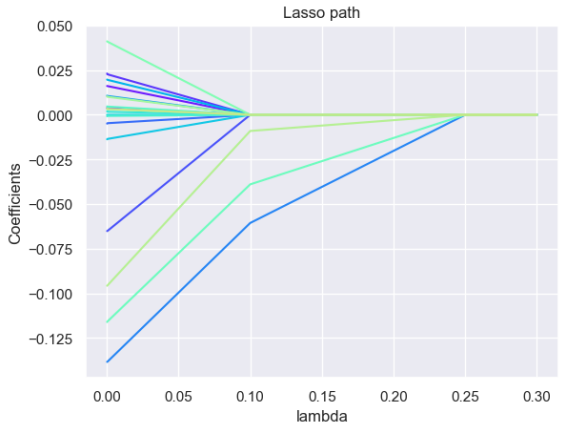


Figure 2: Path of the LASSO coefficients for a sequence of λ.

For $\lambda = 0.3$, all coefficients are effectively zero. For $\lambda = 0.15$, three of the coefficients are non-zero, corresponding to a scenario where only three explanatory variables are active in the regression. etc. In fact, for each value of λ, we can identify the number of selected variables (table 1).

λ	Number of non-zero coefficients
0.3000	0
0.2500	0
0.1000	3
0.0003	18
0.0002	18
0.0001	19

Table 1: λ vs the number of non-zero coefficients.

iii. Optimization via cross-validation

We have scenarios for solutions for different versions of λ. We don't know which one is the most effective at predicting. In this section, considering the temporal nature of our data as time series, we employ the LassoCV() cross-validation method. Additionally, we utilize another Python tool, TimeSeriesSplit (d), where d represents the number of parts serving to split the dataset, to ensure the preservation of the temporal order of the data during the identification process. The value of λ that minimizes the Mean Squared Error (MSE) will be considered the optimal solution for variable selection.

We calculate the average to have a synthetic performance measure for every scenario. Then we display the table relating λ and the MSE in cross-validation.In the case at hand, the optimal λ value is 10^{-4} and its associated MSE value is 0.108 and we find out that none of the estimated coefficients for the variables is set to zero (table 4). We also note through this table that the Dividend feature.Yield has a positive relationship with the dependent variable Realized Alpha having an estimated coefficient of 4.973 i.e. an increase in yield is associated with an increase in dividend yield.. Therefore, all the quantitative variables will be used when modeling.

4. Modelling

i. Modelling and model performance strategies

We have a total of 20 quantitative variables, comprising 19 predictors and one dependent variable, Realized Alpha. It's important to note that these data are temporal, considering their temporal nature, and 177,033 observations. By adopting the approach proposed by [4], the dataset is partitioned into three distinct sets: 60% for training, 20% for validation, and another 20% for testing, all while maintaining the chronological order of observations. The process for training models and assessing their performances is as follows:

- Train the model over the training set.
- Evaluate it over the validation set (helpful for detecting overfitting or underfitting).
- Then, train the model again, but now over the training set + validation set.
- Finally, evaluate it over the test set.

The first, or "training," subsample is used to estimate the model subject to a specific set of tuning parameter values. The second, or "validation," sample is used for tuning the hyperparameters. Hyperparameter tuning amounts to searching for a degree of model complexity that tends to produce reliable out-of-sample performance. The validation sample fits are of course not truly out-of-sample because they are used for tuning, which is in turn an input to the estimation. Thus the third, or "testing," subsample, which is used for neither estimation nor tuning, is truly out-of-sample and thus is used to evaluate a method's predictive performance [1]. It should be noted that the selection process is reserved only for models with hyperparameters and its is crucial for optimizing various models derived from machine learning algorithms with multiple hyperparameters. These parameters enable the models to effectively capture nonlinearity among different variables, adapt well to observations, and enhance their predictive performance.

Besides, model performance is assessed by evaluating the **R-squared** of each machine learning model on the test set, as we are dealing with a regression problem [1][4][3]. Also known as the coefficient of determination (R^2), it represents the proportion of the variance in the dependent variable explained by the statistical or machine learning model. R^2 measures the goodness of fit of the model to the observed data, with a value of 1 indicating a perfect fit—meaning the model explains all the variability. A value of 0 indicates that the model does not explain any variability, while intermediate values represent the proportion of variability explained. By assuming that $(x_i, y_i)_{i=1,\dots,n}$ ($x_i \in \mathbb{R}^p$ and $y_i \in \mathbb{R}$) denotes the input-output pairs of the dataset, the R-squared is defined as follows:

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

where $\hat{y}_i$ is the predicted output and $\bar{y}$ is the mean of the actual values (y_i).

The loss function function is defined by :

$$Loss : \mathbb{R}^2 \to \mathbb{R}$$

$$(y_i, \hat{y}_i) \rightarrow \text{Loss}(y_i, \hat{y}_i) = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2$$

.

Typically by supposing the function f_θ:

$$f_\theta : \mathbb{R}^p \rightarrow \mathbb{R}$$

parameterized by θ is generated by a given machine learning algorithm, we look for the parameter such that:

$$\hat{\theta} = \underset{\theta}{\text{argmin}}\ \mathbb{E}\left(\text{Loss}(y_i, f_\theta(y_i))\right)$$

.

In the following, we focus on machine learning methods that generate the following models:

- Linear Regression and Elastic Net models;
- Random Forest and XGradient Boosting models;
- Neural Network model.

Linear Regression has no parameters and will serves as a reference. It is worth noting that there exists a relationship between R^2 and MSE for all the models mentioned above, except for the case of the neural network model. The neural network model is capable of capturing more complex relationships, and R^2 might not be the primary evaluation metric. Neural networks are typically assessed using the loss function, with R^2 being less interpretable.

ii. Machine learning methods

Linear regression

As previously observed, mutual funds generally exhibit non-linear relationships. Due to these weak correlations between predictors, a model such as the linear regression model is not effective in predicting future Realized Alpha. This is attributed to the fact that the linear regression model performs well in data with linear relationships between variables, which is not the case in mutual funds, as discussed in subsequent paragraphs. Furthermore, the linear regression model has no parameters to optimize, making it a reference model in this study.

Following the training and validation of our linear model, the results obtained on the test set are grouped in the figure 14. As expected, the ratio of variance of Realized Alpha explained by this model is **0.51**, indicating moderate performance. It can also be seen from these results that some variables (HML, RMW, and betasSMB) have p-values greater than 5%, suggesting that they do not have a statistically significant predictive power on the dependent variable.

Random Forest and XGradient Boosting

These machine learning methods are comprehensive approaches. They involve combining the predictions of multiple models constructed using an ensemble algorithm, aiming to enhance predictive performance compared to the use of a single model [2].

One of the challenges in machine learning is effectively managing the trade-off between bias and variance. High variance coupled with low bias can result in overfitting, where the model fits the training set well but fails to generalize to unseen data. On the other hand, a large bias and low variance can lead to underfitting, where the model poorly adapts to the training set and consequently performs poorly on unobserved data. Machine learning methods to overcome these problems include ensemble methods such as Random Forest and XGBoost. While the former allows the independent growth of multiple models, with their predictions averaged to primarily decrease variance. Meanwhile, the latter grows several models sequentially, mainly decreasing bias [2].

Random Forest is a versatile learning algorithm with multiple hyperparameters for optimization. One of the most crucial parameters for controlling overfitting or underfitting is `max_depth` (maximum depth of the tree). It allows us to regulate the complexity of individual trees within the ensemble. Additionally, the algorithm involves selecting a random subset of attributes at each best division and utilizes bootstrap sampling to generate multiple learning samples[3]. These features contribute to the robustness and generalization capabilities of Random Forest. Similar to the previously mentioned learning algorithm, XGBoost (XGradient Boosting) also comes with hyperparameters. During the optimization phase, we will focus on the same `max_depth` hyperparameter as the previous algorithm and a fixed learning rate to 0.1. During this optimization phase, as explained in the modeling part, here are the results are depicted in the figure 12.

As soon as the `max_depth` hyperparameter is greater than 15, the models fit well on the training set and have better generalization on the unobserved data (validation set) with R^2 approaching 0.98 typically. Then after training on the combination of the training and validation set, we get the following result on the test set (figure 3):

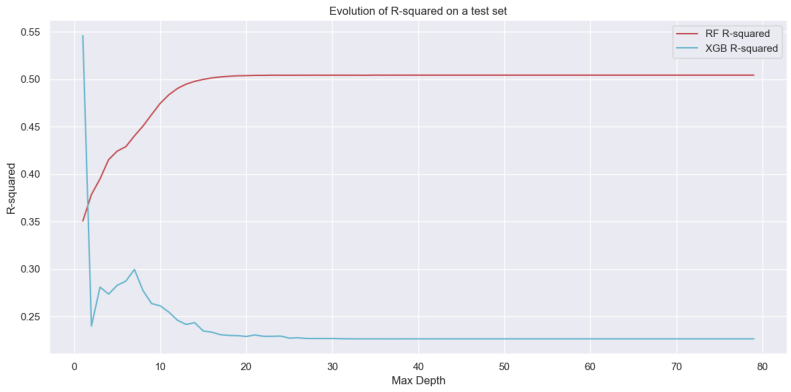


Figure 3: Evolution of R-squared on a test set (Random Forest and XGradient Boosting).

Figure 4: Comparison of true vs predicted outputs for Random Forest and XGBoost on the test set.

However, the results here are less satisfactory. The XGBoost model with **max_depth** = 1 reaches a maximum R^2 value of 0.55. In contrast, Random Forest models with **max_depth** greater than 20 reach a maximum R^2 value of approximately 0.51 (figure 3). It has been noticed that the `XGradient Boosting` model closely resembles that of the `Random Forest` in terms of predictions. However, a notable difference is observed in its ability to make better predictions of `Realized Alpha`, particularly for values below -10%. This distinction is evident when comparing the sky-blue dots in one plot to the red dots in the other plot, indicating that the `XGradient Boosting` model outperforms the `Random Forest` model in predicting `Realized Alpha` values below -10% (figure 4).

Elastic Net

Regularization is often employed to alleviate overfitting in datasets with a large number of predicting variables. The elastic net of Zou and Hastie (2005) uses both 1-norm and 2-norm regularization terms to shrink the size of the estimated parameters [3]. The objective function for the elastic net, with two regularization terms, is:

$$\hat{\beta}^{EN} = \underset{\beta_0,\boldsymbol{\beta}}{arg\,min} \left\{ \frac{1}{2} \sum_{i=1}^{n} \left(y_i - \beta_0 - \mathrm{x}_i^T \boldsymbol{\beta}\right)^2 + \lambda \left[\sum_{j=1}^{p} \left(\frac{1}{2}(1-\alpha)\beta_j^2 + \alpha|\beta_j| \right) \right] \right\}$$

with α in the range [0;1]. The 1-norm term can be used to control the sparsity of the estimated parameter vector and the 2-norm term to increase its stability. For the case with $\alpha = 0$, the objective function in includes only the 2-norm term, and thus, elastic net is equivalent to ridge regression, which provides a dense estimator of the parameter vector. If, on the other hand, $\alpha = 1$, the objective function includes only the 1-norm term, and a Least Absolute Sum of Squares Operator (LASSO) regression is performed, which provides a sparse estimator.

When training the models generated by the Elastic Net algorithm on the training set and evaluating them on the validation set, we obtain an R^2 strictly less than 0.3 on the validation set, as shown in Figure 5.

Subsequently, we evaluate these models on the test set following the evaluation strategy to the previously defined . Notably, we observe the best R^2 approaching 0.53 when the α value is set to 0, indicating the use of a Ridge regression model. Hence we get the predicted Realized Alpha on the test set. We can also see that the predicted Realized Alpha values do not correspond to

the true values by the presence of much more observed red dots representing these true Realized Alpha values. (figure 6)

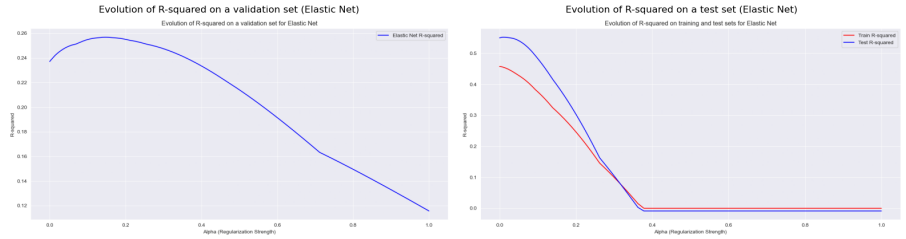


Figure 5: Evolution of R-squared on a validation set and test set (Elastic Net).

Figure 6: Comparison of true vs predicted outputs for Elastic Net on the test set.

Results are moderately satisfactory as suggested by linear regression and so all these learning algorithms capture the nonlinearity of the predictors on average in order to generate better predictions on the Realized Alpha.Interestingly, Random Forest demonstrates comparable performance to Linear Regression in this study, suggesting the potential for further optimization through adjustments to its hyperparameters. As we progress, our attention will turn to an even more potent yet challenging-to-train learning algorithm, specifically the neural network. .

Neural Network

Neural Network architecture We consider a neural model for regression tasks with a fully connected architecture. The input layer has M neurons, corresponding to the number of input features. Each hidden layer is designed with a consistent number of neurons (P), and the output layer consists of a single neuron for predicting continuous values since we handle a regression problem.

The model comprises L hidden layers, each with P neurons, where L is the number of hidden layers. Hyperparameters include the learning rate (α), the number of hidden layers (L), the number of neurons per hidden layer (P), and the training iterations (I). Robustness against outliers is addressed through loss functions like Huber or weighted least squares. Data normalization, utilizing methods like Z-score or range normalization, aids convergence. The Xavier initialization method is applied for weight initialization.

The core objective during training is to minimize the loss function, which measures the disparity between the model predictions and actual target values. Common loss functions for regression tasks include Mean Squared Error (MSE) or Huber loss, offering resilience to outliers. The model parameters, including weights and biases, are iteratively adjusted to minimize this loss using

gradient-based optimization algorithms such as Stochastic Gradient Descent (SGD) or its variants. The learning rate (α) is a critical hyperparameter influencing the size of parameter updates during optimization. A well-chosen learning rate ensures a balanced convergence without overshooting or converging too slowly.

The goal is to find the parameter vector θ that minimizes the objective function. In the context of Mean Squared Error (MSE) loss, the optimization problem can be written as:

$$\hat{\theta} = \underset{\theta}{\operatorname{argmin}}\ \mathrm{MSE}(\theta) = \underset{\theta}{\operatorname{argmin}}\ \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2$$

Here, θ represents the set of model parameters, including weights and biases, and MSE(θ) is the Mean Squared Error loss. N corresponds to the number of samples, y_i is the true output, and $\hat{y}_i$ is the predicted output.

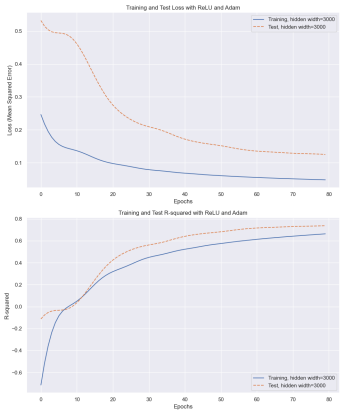
Roughly speaking, training a neural model requires a substantial amount of time. We drew inspiration from the recommended reference [4] to define a feedforward fully connected network with four hidden layers.

After several training phases, in which we varied the number of neurons per hidden layer, the learning rate, and the optimizer, the results obtained led us to the following choices:

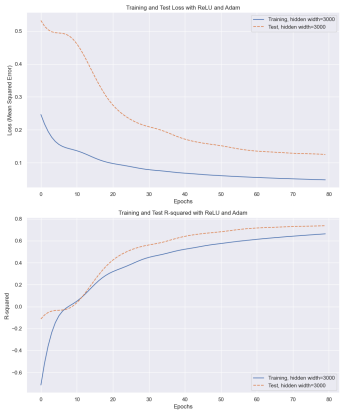
- 3000 units per hidden layer,
- 80 as a number of epochs;
- a fixed learning rate of 10^{-5},
- ReLU() as the activation function (ReLU(x) = max(x, 0)),
- Adam (Adaptive Momentum) as the optimizer.

Neural Network Model for Enhanced Prediction The neural network model is recognized as a powerful machine learning tool, excelling in capturing intricate non-linear relationships among data features. In our context, we encounter font comment investment features characterized by notably low correlation values. This inherent challenge has impeded the performance of previous machine learning algorithms in accurately predicting Realized Alpha values during the test phase. Using the neural network architecture defined above, the validation results are illustrated in Figure 7a, while the testing results are shown in Figure 7b.

As observed during training, the loss function converges to 0.05, while the R-squared converges to 0.75 on the test set and 0.25 after 80 iterations. As described earlier, the quality of a neural network model is primarily assessed based on the loss function on the test set. These results indicate clear overfitting on the validation set. Following the defined training process, the achieved results are significantly improved, with R-squared converging to 0.8 and the loss function around 0.1 after 80 iterations. Further improvements can be pursued by increasing the number of iterations and adjusting other parameters. However, it's important to note that such enhancements require a substantial amount of time.



(a) Validation Results of the Neural Network.



(b) Testing Results of the Neural Network.

Figure 7: Combined Results of the Neural Network.

iii. Model comparison

We summarize in the table below the performance of previous models in predicting Realized Alpha values during the test phase based on their R-squared scores (table 2).

Model	**R-squared**
Random Forest ($max\_depth = 20$)	0.503
XGBoost ($max\_depth = 1$)	0.545
Elastic Net ($\alpha = 0$)	0.55
Neural network	0.8

Table 2: Performance of Previous Models in Predicting Realized Alpha During Test Phase.

We first observe that these optimized models outperform the R-squared obtained with the linear regression model (0.518), although the differences are generally not substantial. The most effective among them is the neural network model, yielding an R-squared of 0.8. However, as previously mentioned, the neural network's quality is primarily assessed through the loss function, where a lower loss indicates a better model. Therefore, we could increase the number of iterations and adjust other parameters to achieve a higher R-squared and a loss function close to zero. This implies that with these models, predictions of Realized Alpha values on the test set will be highly accurate.

5. Variable importance and Marginal Relationships

Within the specialized domain of mutual fund management, it is paramount to discern the key variables that significantly impact alpha predictions. Employing sophisticated algorithms, predictive optimization is achieved by harnessing features tailored to the intricacies of mutual fund investments. The evaluation of variable importance is conducted through SHAP (SHapley Additive exPlanations), a powerful machine learning methodology. SHAP quantifies the marginal contribution of each variable by considering all conceivable combinations, thereby offering a comprehensive understanding of their respective impacts. This approach equips fund managers with invaluable insights, enabling well-informed decision-making in fund placements and the optimization of port-

folio performance [3]. The Shapley values assigned to each variable serve as interpretable metrics, reflecting the relative importance of each variable in shaping the model's predictions for a given instance (Figure 9).

This figure illustrates the significance of each characteristic, quantified by the average absolute SHAP (SHapley Additive exPlanations) value across all observations. The rankings of variable importance, depicted in histograms from top to bottom, are presented for ordinary least squares (OLS), elastic net, gradient boosting, and random forests. . In the comprehensive analysis of various machine learning models, the figure above reveals that `betasMkt` stands out as the most influential variable among all the predictors with a value of 12% for all the models, except XGradient Boost model where it has 8% as value. This finding underscores its critical role in the predictive capabilities of these models. Examining the linear regression model, it becomes evident that this model incorporates nearly all predictors, excluding only the variable `RMW`. Notably, the `betaMkt` factor emerges as the pivotal variable, exerting significant influence on predictions within the linear regression framework. In stark contrast, the XGBoost model takes a more selective approach by excluding several variables such as `RMW`, `SMB`, `MOM`, `Return`, and `HML`. Instead, it relies predominantly on variables like `betasMkt`, `Dividend.Yield`, and `betasCMA` to drive its predictive power. The model derived from ElasticNet exhibits a similarity to the linear regression model, further emphasizing the importance of `betasMkt` in prediction. This cohesive pattern across different models reinforces the significance of `betasMkt` as a key factor in predicting outcomes.

Moreover, the figure 13 displays pair grid with kernel density estimates in the upper triangle, histograms on the diagonal, and scatter plots in the lower triangle Scattered point clouds, particularly those representing regions of high probability density, are generally of minimal significance.

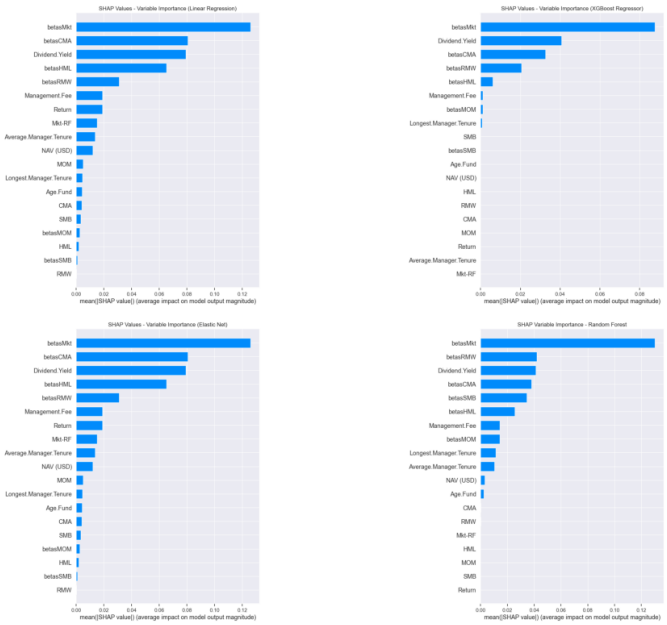


Figure 9: Variable importance for Random Forest, Linear Regression, Xgradient Boosting and Elastic Net models previous obtained.

6. Machine Learning Portfolios

The use of deciles in a study is generally associated with the analysis of data distribution. Deciles divide a sorted set of data into ten equal parts, each representing 10% of the sample. This division allows for the examination of value distribution and the identification of trends or specific characteristics within different parts of the distribution.

In the context of an in-depth analysis of characteristics of mutual funds, the use of deciles could be employed to assess how certain measures or specific characteristics are distributed among the funds. We will focus on interpreting the predicted results, which are valuable for mutual fund managers. This information is useful for fund managers to understand potential negotiations that may be initiated for selling and potentially investing elsewhere, where positive accumulated alphas can lead to better performance.

Our dataset is slghtly altered and we obtain the figure plots the time series of cumulative out-of-sample portfolio realized alphas of the excess returns net of all costs of the top-decile fund portfolios. In the depicted figure, we observe a noteworthy trend in the cumulative alpha curve, specifically for decile 0 (illustrated by the blue curve). This curve consistently decreases and takes on increasingly negative values. Such a decline indicates a potential underperformance, prompting the manager of mutual funds to consider initiating sales negotiations. This proactive approach aims to anticipate and mitigate potential losses associated with the declining alpha.

Furthermore, examining other deciles, such as 6 or 8, reveals a contrasting pattern. These deciles exhibit more favorable cumulative positive alphas. For a fund manager, this suggests an opportunity for potential reinvestment, aiming to capitalize on these positive trends and achieve better gains. Assessing the cumulative alpha across various deciles provides valuable insights for strategic decision-making, enabling the manager to navigate market dynamics effectively.

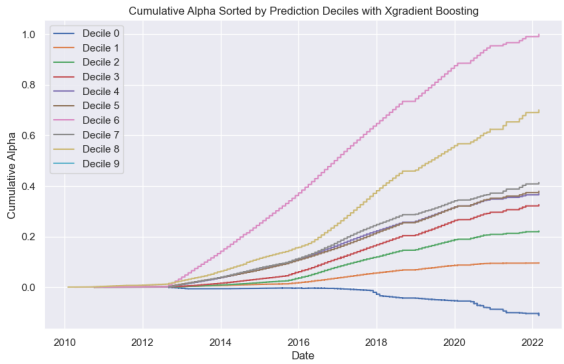


Figure 10: Cumulative Alpha Sorted by Prediction Deciles using XGradient Boosting.

7. Conclusion

In this project, we find a common thread in the emphasis on employing advanced machine-learning methods to address the challenge of achieving positive net alpha in mutual-fund investments. Both texts advocate for a departure from traditional methodologies, with the first text providing context to the debate and highlighting economic mechanisms, while the second text delves into a multifaceted approach, incorporating sentiment and momentum factors. Together, these findings

suggest a comprehensive toolset for portfolio managers, integrating sophisticated predictions, variable importance analysis, and tradable portfolio construction to detect positive alpha and make well-informed investment decisions.

8. Additional Figures and tables

Algorithms in Details in "Empirical Asset Pricing via Machine Learning"

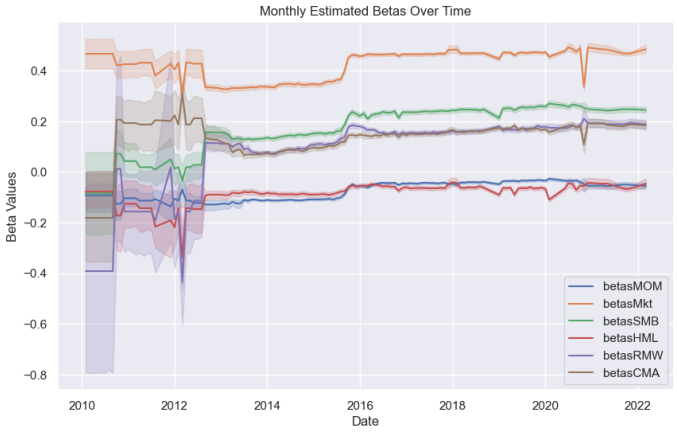


Figure 11: Graphs of beta features over time.

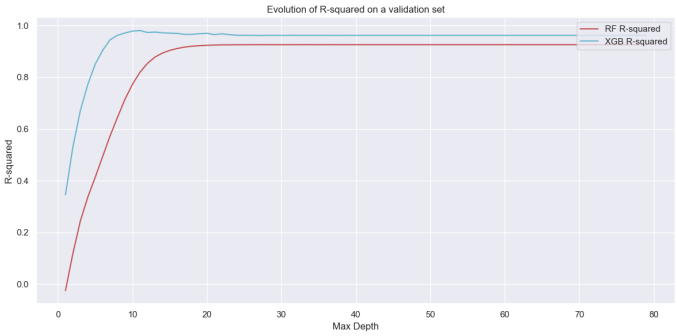


Figure 12: Evolution of R-squared on a validation set (Random Forest and XGradient Boosting.

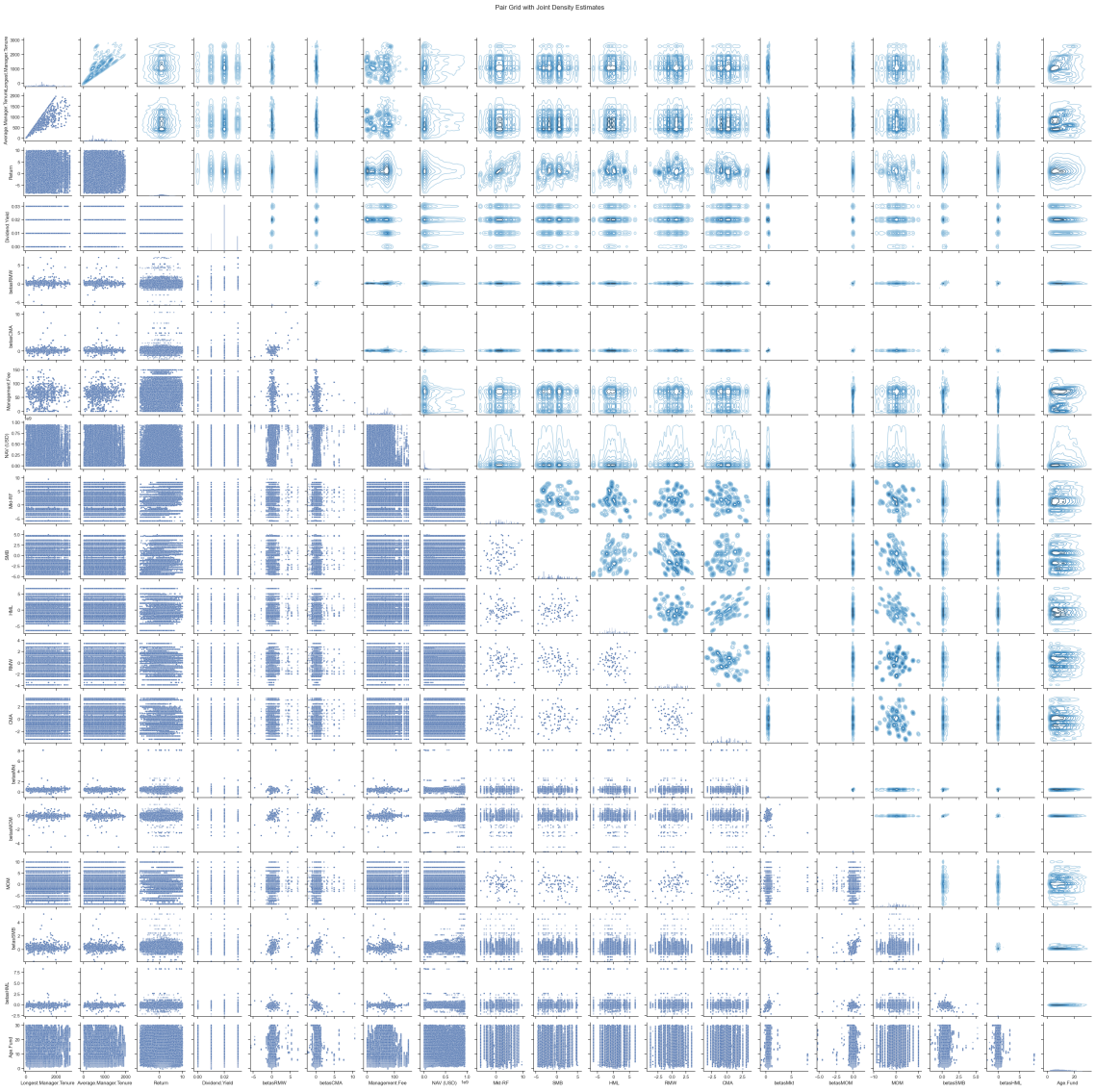


Figure 13: Pair Grid with Joint Density Estimates.

Variables	**Mean**	**Median**	***Standard Deviation***
Longest.Manager.Tenure	1178	1133	595.2844
Average.Manager.Tenure	780.9	744	398.7158
Return	1.062	1.060	3.0613
Dividend.Yield	0.01879	0.02000	0.0074
Management.Fee	52.36	60.00	30.8570
NAV.(USD).	145061426	45012028	212430600
MOM	0.1795	0.1900	3.4885
Mkt.RF	1.797	1.810	3.1426
SMB	-0.5549	-1.0700	2.4668
HML	-0.1385	-0.4600	2.4197
RMW	0.1568	0.4100	1.4907
CMA	-0.1508	-0.1100	1.5209
RF	0.06354	0.02000	0.0706
Age.Fund	11.4007	10.5000	6.2723
Realized Alpha	**0.2954**	**0.3151**	**0.4594**
betasMOM	-0.05574	-0.06926	0.1515
betasMkt	0.4467	0.4688	0.2203
betasSMB	0.2255	0.1593	0.2297
betasHML	-0.065065	-0.063025	0.2373
betasRMW	0.1551	0.1194	0.2685
betasCMA	0.14964	0.13675	0.2718

Table 3: Summary statistics including standard deviation for each variable.

OLS Regression Results

Dep. Variable:	alpha	R-squared:	0.518
Model:	OLS	Adj. R-squared:	0.518
Method:	Least Squares	F-statistic:	1.058e+04
Date:	Mon, 01 Jan 2024	Prob (F-statistic):	0.00
Time:	20:23:30	Log-Likelihood:	-48854.
No. Observations:	177033	AIC:	9.775e+04
Df Residuals:	177014	BIC:	9.794e+04
Df Model:	18		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	0.2954	0.001	389.749	0.000	0.294	0.297
Longest.Manager.Tenure	0.0066	0.001	5.909	0.000	0.004	0.009
Average.Manager.Tenure	0.0160	0.001	14.336	0.000	0.014	0.018
Return	0.0248	0.001	28.062	0.000	0.023	0.027
Dividend.Yield	-0.0957	0.001	-107.841	0.000	-0.097	-0.094
betasRMW	0.0544	0.001	52.305	0.000	0.052	0.056
betasCMA	-0.1354	0.001	-129.817	0.000	-0.137	-0.133
Management.Fee	0.0233	0.001	28.313	0.000	0.022	0.025
NAV (USD)	0.0169	0.001	21.935	0.000	0.015	0.018
Mkt-RF	-0.0189	0.001	-20.712	0.000	-0.021	-0.017
SMB	-0.0038	0.001	-3.802	0.000	-0.006	-0.002
HML	-0.0023	0.001	-2.424	0.015	-0.004	-0.000
RMW	-0.0003	0.001	-0.323	0.747	-0.002	0.001
CMA	0.0048	0.001	5.257	0.000	0.003	0.007
betasMkt	-0.1901	0.001	-179.144	0.000	-0.192	-0.188
betasMOM	0.0048	0.001	5.278	0.000	0.003	0.007
MOM	-0.0064	0.001	-6.411	0.000	-0.008	-0.004
betasSMB	-0.0012	0.001	-1.200	0.230	-0.003	0.001
betasHML	-0.1196	0.001	-132.291	0.000	-0.121	-0.118

Omnibus:	173271.333	Durbin-Watson:	1.978
Prob(Omnibus):	0.000	Jarque-Bera (JB):	157315742.026
Skew:	-3.817	Prob(JB):	0.00
Kurtosis:	148.838	Cond. No.	2.87

Figure 14: Summary of the linear regression model on the learning set.

Variables	**Coefficients**
Longest.Manager.Tenure	7.062229×10^{-5}
Average.Manager.Tenure	8.967989×10^{-5}
Return	9.689885×10^{-3}
Dividend.Yield	**4.973391**
betasRMW	-1.099056×10^{-1}
betasCMA	-6.874698×10^{-1}
Management.Fee	1.317951×10^{-3}
NAV (USD)	1.692204×10^{-10}
Mkt-RF	2.784492×10^{-4}
SMB	-2.611486×10^{-3}
HML	1.879932×10^{-3}
RMW	-7.084877×10^{-4}
CMA	1.152368×10^{-3}
betasMkt	-1.072794×10^{-1}
betasMOM	1.139361×10^{-1}
MOM	3.043971×10^{-3}
betasSMB	2.335541×10^{-1}
betasHML	-6.954410×10^{-1}
Age.Fund	5.534389×10^{-3}

Table 4: Estimated coefficients for each predictor according to the optimal value of $\lambda = 10^{-4}$ obtained through cross-validation.

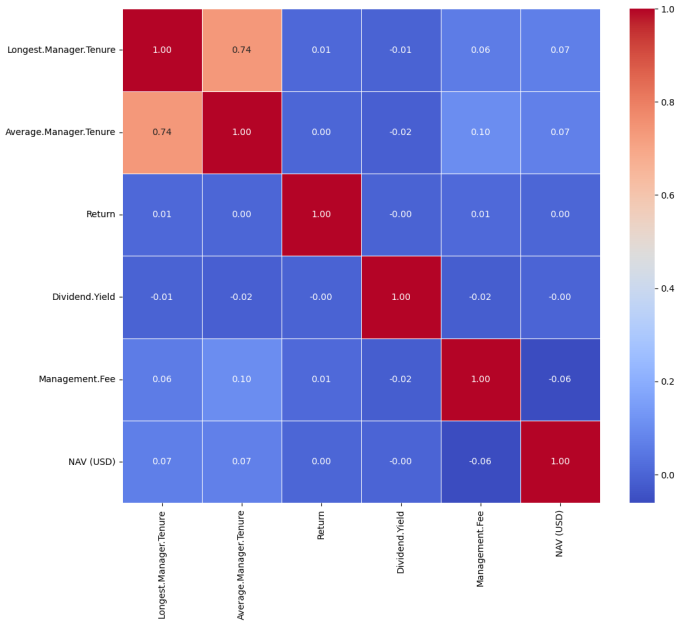


Figure 15: Pearson's correlation matrix based on the initial Morningstar dataset.

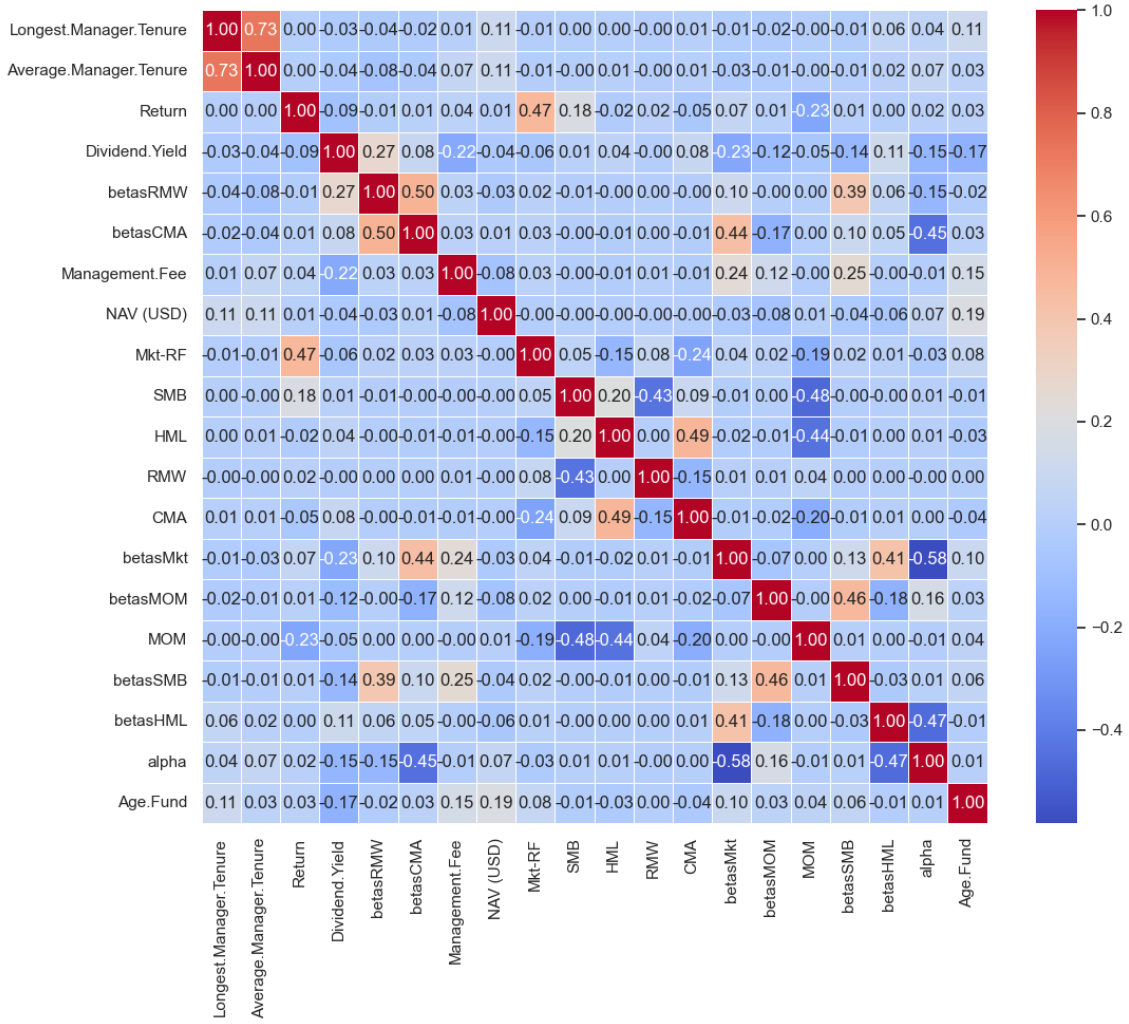


Figure 16: Pearson's correlation matrix based on the tuned dataset.

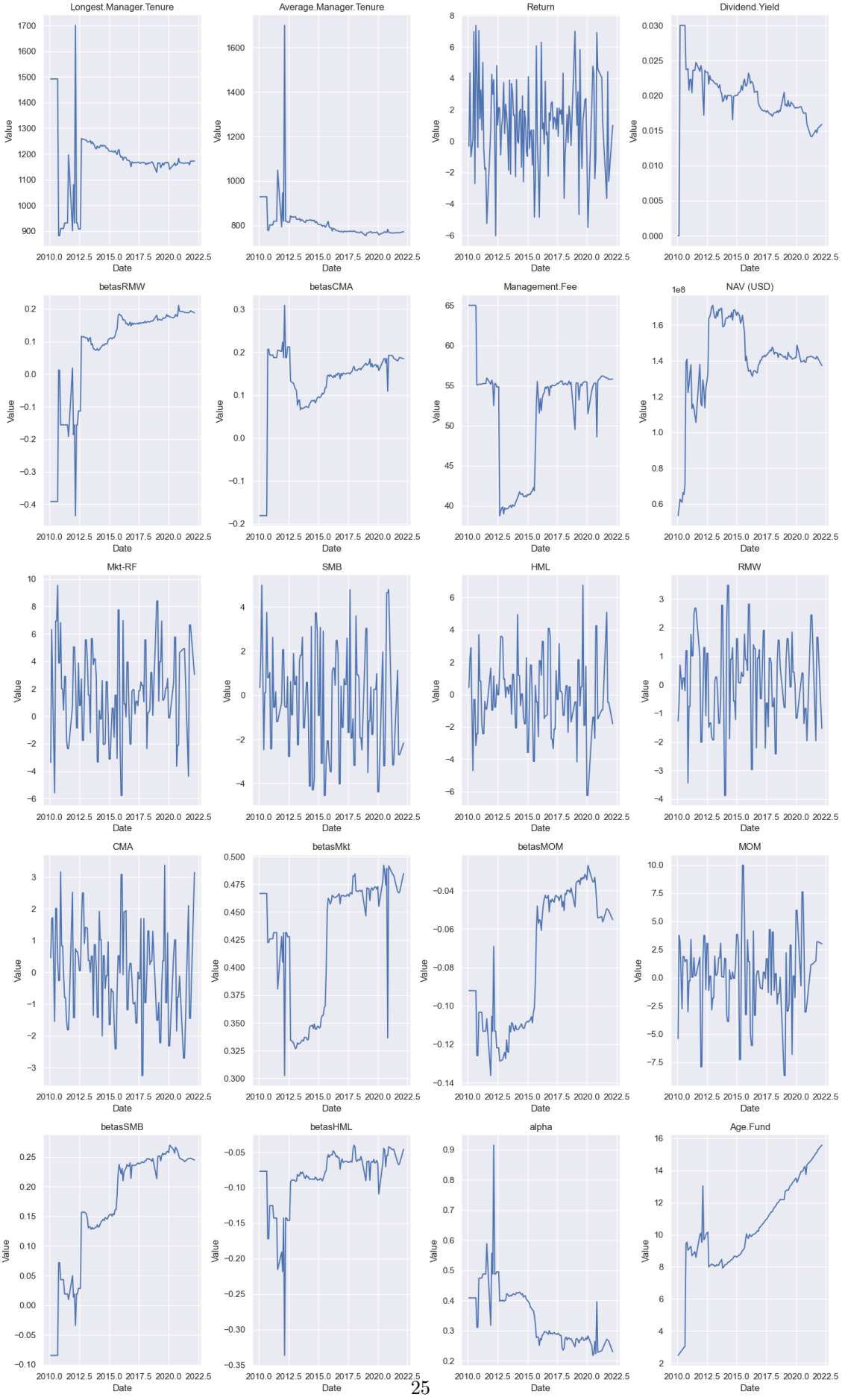


Figure 17: Some features over time.

Variable Name	Description
Date	Month date of reporting
Fund.Name	Name of the fund
Ticker	Ticker symbol of the fund
Global.Broad.Category	Global broad category of the fund
Global.Category	Global category of the fund
US.Category	US category of the fund
Firm.Name	Name of the firm associated with the fund
Inception.Date	Date when the fund was started
Longest.Manager.Tenure	Monthly longest tenure of a manager in the fund
Average.Manager.Tenure	Monthly average tenure of managers in the fund
Return	Monthly return value for the fund
Dividend.Yield	Monthly dividend yield of the fund mutual relative to its share
Management.Fee	Monthly management fee associated with the fund
NAV (USD)	Monthly net Asset Value in USD
MOM	Monthly momentum factor
Mkt-RF	Monthly market risk factor
SMB	Monthly size risk factor
HML	Monthly value risk factor
RMW	Monthly operating profitability risk factor
CMA	Monthly investment in conservative assets risk factor
RF	Monthly risk-free rate
Age.Fund	Monthly age of an investment
Realized alpha	Monthly realized alpha calculated using Equation (2)
betasMOM	Monthly beta for the momentum factor
betasMkt	Monthly beta for the market risk factor
betasSMB	Monthly beta for the size risk factor
betasHML	Monthly beta for the value risk factor
betasRMW	Monthly beta for the operating profitability risk factor
betasCMA	Monthly beta for the investment in conservative assets risk factor

Table 5: Characteristics of the mutual funds and their descriptions.

Bibliography

[1] Shihao Gu, Bryan Kelly, and Dacheng Xiu. Empirical asset pricing via machine learning. The Review of Financial Studies, 33(5):2223–2273, 2020.

[2] Thomas G Dietterich. Ensemble methods in machine learning. In International workshop on multiple classifier systems, pages 1–15. Springer, 2000.

[3] Victor DeMiguel Javier Gil-Bazo Francisco and J Nogales André AP Santos. Machine learning and fund characteristics help to select mutual funds with positive alpha. 2022.

[4] Ron Kaniel, Zihan Lin, Markus Pelger, and Stijn Van Nieuwerburgh. Machine-learning the skill of mutual fund managers. Journal of Financial Economics, 150(1):94–138, 2023.