Factorization Asset Pricing

February 12, 2022

Abstract

Conventional measurements of risk premiums are biased if the estimation models are potentially misspecified and unstable. Say, *factor interactions* is one of the crucial omitted specifications that standard models cannot involve. Motivated by this argument, we propose an interpretable factorization-based method to estimate the risk premium of factors in a linear asset pricing model (we call it Factorization Asset Pricing Model, FAPM), which is able to account for all interactions between factors using factorized parameters. We emphasize the critical importance of the factor interactions in measuring risk premiums. We show that our factorization approach can be identified as the best-performing method among current methodologies (including trees and neural networks, among other nonlinear models), even in a parsimonious linear framework. We also highlight that few factors input can predict well, while numerous factors set may generate negative effects due to adverse factor interactions. Remarkably, weak factors in standard models may play important roles in FAPM because their interactions with other factors can be significant.

Keywords: Factor Interactions, Tensor Factorization, Interpretable Machine Learning, Approximation Error, Factorization Asset Pricing Model, FinTech.

1 Introduction

In this article, we investigate the central importance of the factor interactions by introducing a generalized linear framework based on the factorization method. We do so in the context of estimating the risk premiums, thereby proposing a novel Factorization Asset Pricing Model (henceforth, FAPM).

Our primary contributions are threefold. First, we provide a novel benchmark for analyzing the factor interactions in a generalized linear framework. This benchmark model can be exploited to extend current widely adopted multifactor models in various disciplines. In this paper, we extend the standard arbitrage pricing theory (APT) (see Ross (1976)) by exploiting this factorization-based framework, thereby formalizing our Factorization Asset Pricing Model (FAPM). Note that this extension is very straightforward, in particular, we consider a generalized linear factor model that directly adds the components of factor interactions based on the factorization method. Therefore, the FAPM remains to be a linear model that can rule out the nonlinear effect explicitly. Remarkably, this linearity is beneficial for us to identify the fundamental mechanism of the factor interactions in isolation.

Second, we synthesize the traditional empirical asset pricing methods with interpretable machine learning, especially on factor interactions. Relative to the recent adoption of machine learning methods in empirical asset pricing studies, our FAPM can be identified as a more logically straightforward and natural extension to the conventional empirical asset pricing models. Most remarkably, we show that our FAPM can be the best-performing method among current methodologies, including trees and neural networks among other nonlinear models, even in a parsimonious linear framework. The outstanding performance of FAPM is summarized in two ways. The first is the highest out-of-sample predictive R^2 (over 2.5 times relative to the current frontier of risk premium measurement) relative to preceding literature. Second, and more importantly, we push the frontier of the current machine learning forecasts economic gains to investors using FAPM. A value-weighted long-short decile spread strategy that takes positions based on stock-level FAPM forecasts earns an annualized out-of-sample Sharpe ratio of 1.42, more than 2.3 times the performance of a leading regression-based strategy, and over 46.7%

better than neural network forecasts-based strategy from the literature.[1]

Third, to the best of our knowledge, we are the first paper that systematically and precisely analyzes the factor interactions in a dedicated generalized framework.[2] Our findings are novel and can be summarized in two aspects. First, we highlight that the weak factors in generalized linear models can be crucial in FAPM because the effect of their interactions with other factors can be quite significant. Second, we identify a counterintuitive result that the excessive factor loadings may diminish the performance of FAPM because some adverse factor interactions due to redundant factors may generate negative impacts on the model predictions.

To test the two aspects of former findings, we propose a novel FAPM-7 model against the all-factor FAPM. In contrast to conventional factor asset pricing models (e.g., Fama and French (1993), Jegadeesh and Titman (1993), and most recently Fama and French (2015)), the FAPM-7 model no longer picks the most significant factors in the generalized linear factor models; instead, we select the strongest seven factors and the weakest seven factors around 103 factors pool. Although this specification (the FAPM-7) may not be the best approach that we can provide, it is good to identify the interaction effect of the weak factors. For instance, we find that the performance of FAPM-7 is much better than either the only strongest 7-factor FAPM or the only weakest 7-factor FAPM. This result indicates that the weak factors can generate significant interactions in FAPM, thereby affecting the model predictions; however, this effect cannot be identified in the existing methods so far. Otherwise, as Figure 5 illustrates, we can observe that the FAPM performs worse after putting more than the first seven pair factors (seven strongest factors and seven weakest factors) into the model. The phenomenon that more factor loadings despite poor model performance can be recognized as the negative impact due to the adverse factor interactions. However, no current approach can identify these adverse factor interactions so far, including some nonlinear machine learning models, for example, the neural network models.

[1]Note that the performance of FAPM is much more robust than a set of nonlinear machine learning models in recent literature. We struggle to replicate the results of neural network forecasts (the best performance in current literature); however, unfortunately, we still cannot reach the best version of the neural network models even after numerous experiments so far.

[2]Gu, Kelly, and Xiu (2020) also try to analyze the factor interactions via neural network methods. However, as the "universal approximation" model, the neural network is neither dedicated to exploring factor interactions nor logically straightforward in the analysis.

At the very beginning, a conventional prediction of asset pricing models is that some risk factors should command the risk premium. However, most theoretical multifactor models assume that risk factors are independent and have no interaction effect with each other (see Ross (1976), Roll and Ross (1980), Ingersoll Jr (1984), and Huberman, Kandel, and Stambaugh (1987), among many others). As an important issue, factor interactions has been paid more and more attention by scholars in recent years, for example, Gu et al. (2020) emphasize the importance of the factor interactions and recognize that the conventional generalized linear models are comparatively poorly suited for capturing factor interactions.[3] Therefore, the problem is, while different approaches have been proposed to estimate risk premia, they are all affected by one common potential issue of model misspecifications: omitted factor interactions.

Omitted factor interactions arise in standard linear predictions of asset pricing models whenever the model used in the estimation cannot identify the high-order interactions among the risk factors. This is a fundamental concern when estimating current asset pricing theories, because theoretical models are parsimonious and usually suggest that the risk factors are independent, whereby neglecting the impact of the factor interactions. Namely, the standard generalized linear models are misspecified in approximating the scenario that incorporates factor interactions, which we also refer to as the approximation error.

While the possibility of model misspecification (or the approximation error), for example, omitted factor interactions, is known in the literature (e.g., Bai (2009), Moon and Weidner (2015), and Gu et al. (2020)), until now, no systematic and explicitly solution on factor interactions has been proposed so far. In particular, papers focus on the interactive fixed effect (e.g., Bai (2009)) typically add the multiplicative interacted term of individual

[3]Gu et al. (2020) highlight that the deep learning method may approximate the ambiguous functional forms in a good way. They also point out that the nonlinear approximation based on deep learning may also be related to factor interactions. However, unfortunately, the ambiguous theoretical mechanism for the deep learning models (black box) leads to an unclear interpretation of the factor interactions. Namely, it is really difficult to clearly isolate the factor interactions effect from the nonlinearity effect in the conventional deep learning models. Note that the deep learning models indeed present a composition effect with numerous unknowable effects. However, nobody can identify the specific factor interactions effect in the deep learning models. That is, the deep learning models are hopefully incorporating the factor interactions effect, but not for sure. By contrast, this paper builds up a linear framework, the Factorization Asset Pricing Model (FAPM), that clearly rules out the nonlinearity effect and only the factor interactions present. One advantage of our FAPM framework is its precise theoretical mechanism.

and time fixed effect into the standard linear model to rule out some interactive effects. However, the interactive fixed effect model is far from nesting all factor interactions as well as analyzing their economic significance. Otherwise, papers using the deep learning approach (e.g., Gu et al. (2020)) usually select a "universal approximation" model such as the neural network, thereby claiming that the model can entwine many telescoping layers of nonlinear predictor interactions. Unfortunately, the ambiguous theoretical mechanism for the deep learning models cannot clearly identify the mechanism of the factor interactions in their analysis. There is, however, no methodology guarantee that the specific effect of factor interactions and their economic mechanism can be precisely identified so far.

In contrast to current literature, we propose a general solution for the omitted factor interactions issue in generalized linear factor models, the FAPM. We introduce a novel factorization-based methodology that exploits the factorized parametrization of available test factors to nest all factor interactions correctly. We show that the FAPM depends on a linear number of parameters and can be computed in linear time, allowing direct optimization and storage of model parameters without storing any training data. Remarkably, FAPM is able to estimate factor interactions even in problems with huge data sparsity or omitted variable bias (see Giglio and Xiu (2021)).

1.1 Literature review

This paper sits at the confluence of several strands of literature, combining empirical asset pricing with high-order factor interactions analysis.

Our paper relates to the literature on factor empirical asset pricing models since the arbitrage pricing theory (APT) Ross (1976). Chamberlain and Rothschild (1983) provide an extension of this framework to approximate factor models. Connor and Korajczyk (1986), Connor and Korajczyk (1988), and Lehmann and Modest (1988) try to estimate and test in the APT setting by extracting principal components of returns. Most recently, Kozak, Nagel, and Santosh (2018) show principal components can capture sizable fraction of the cross-section of expected returns. Gu et al. (2020) use neural network models to capture the model nonlinearity. In contrast to these papers, we extend the standard linear factor models by incorporating factor interactions, thereby improving the model explanatory

power to the cross-sectional variation of expected returns.

Our paper is also related to the literature that has pointed out misspecification in estimating and testing linear factor models. Kleibergen (2009) argue that ignoring model misspecification and identification-failure leads to an overly positive assessment of the pricing performance of spurious, otherwise biased risk premiums estimates of true factors in the model (see Jagannathan and Wang (1998)), and even useless factors (see Kan and Zhang (1999)). Therefore, some inference methods have been used that are more reliable and robust to model misspecification (e.g., Shanken and Zhou (2007), Kleibergen (2009), Kan and Robotti (2009), Kan, Robotti, and Shanken (2013), and Gospodinov, Kan, and Robotti (2013)). Giglio and Xiu (2021) focus on the omitted variables bias and measurement error. We study and correct the biases due to omitted factor interactions and approximation error.

Note that our work indeed contributes to the existing investigations of the weak factors. Kan and Zhang (1999) first note that the estimation on risk premia from linear regression becomes distorted when a factor to which test assets have zero exposure is included in the model. Kleibergen (2009) highlights that standard estimation fails if the betas are relatively small. Bryzgalova (2015) suggests eliminating weak factors via a penalized two-pass regression, and Jegadeesh, Noh, Pukthuanthong, Roll, and Wang (2019) adopt instrumental variable estimator to correct the error-in-variables bias. Giglio, Xiu, and Zhang (2021) argue that the weak factor problem is fundamentally an issue of test asset selection, and Anatolyev and Mikusheva (2021) propose a four-split approach that addresses the issues of weak factors. Besides current literature, our paper focus on the interaction effect of the weak factors. We emphasize the importance of weak factors when considering the factor interaction effect, in particular, our approach shows that the interaction of these weak factors with other factors can be significant in improving the model predictions.

Our work extends the empirical asset pricing literature on machine learning adoption. Rapach, Strauss, and Zhou (2013) predict global equity market returns using lagged returns of all countries by adopting lasso regression. The neural network models have been widely used in early studies, for example, the derivatives prices forcast (e.g., Hutchinson, Lo, and Poggio (1994)). More recently, various machine learning methods have been used to investigate the cross-section of stock returns. Kelly, Pruitt, and Su (2019) use di-

mension reduction methods to estimate and test factor pricing models. Kozak, Nagel, and Santosh (2020) use shrinkage and selection methods to approximate a stochastic discount factor, and Freyberger, Neuhierl, and Weber (2020) approximate a nonlinear function for expected returns by adopting similar method. Kelly et al. (2019), Gu, Kelly, and Xiu (2021), and Feng, Giglio, and Xiu (2020) try to nest machine learning into equilibrium asset pricing. Harvey and Liu (2021) study the multiple comparisons problem using a bootstrap procedure. Gu et al. (2020) simultaneously explore a wide range of machine learning methods to study the behavior of expected stock returns. The focus of our paper is to identify the factor interactions and their impact on expected stock returns, with a particular emphasis on our factorization asset pricing model, among other methods.

Gu et al. (2020) highlight that the deep learning method may approximate the ambiguous functional forms in a good way. They also point out that the nonlinear approximation based on deep learning may also be related to factor interactions. However, unfortunately, the ambiguous theoretical mechanism for the deep learning models (black box) leads to an unclear interpretation of the factor interactions. Namely, it is really difficult to clearly isolate the factor interactions effect from the nonlinearity effect in the conventional deep learning models. Note that the deep learning models indeed present a composition effect with numerous unknowable effects. However, nobody can identify the specific factor interactions effect in the deep learning models. That is, the deep learning models are hopefully incorporating the factor interactions effect, but not for sure. By contrast, this paper builds up a linear framework, the Factorization Asset Pricing Model (FAPM), that clearly rules out the nonlinearity effect and only the factor interactions present. One advantage of our FAPM framework is its precise theoretical mechanism.

The rest of the paper is organized as follows. Section 2 discusses the methodology and the key approximation error in the standard risk premia predictors, the omitted factor interactions. Section 3 introduces our main results of estimation and discusses some of the related empirical evidence. Section 4 provides the results of factorization portfolios based on our FAPM, and Section 5 concludes.

2 Methodology

In this section, we are going to describe our model, Factorization Asset Pricing Model (FAPM). First, we formalized the problem of an asset's excess return, along with a brief introduction to the linear estimation approach. Then, we introduce our model, the FAPM, which aims to estimate the risk premiums using second-order factor interactions. We also present a comprehensive description of optimizing our model through an efficient and scalable fashion.

2.1 Problem Formalization and Linear Approach

We aim to estimate the risk premiums according to the observed factors of the asset. Following Gu et al. (2020), we define the asset's excess return for stock i on month $t+1$ as:

$$r_{i,t+1} = \mathbb{E}_t(r_{i,t+1}) + \epsilon_{i,t+1}, \qquad \text{(Excess Return)}$$

where $\epsilon_{i,t+1}$ is an unpredictable noisy, and $\mathbb{E}_t(r_{i,t+1}) = y(\boldsymbol{x}^{(i,t)})$ is the risk premiums estimation based on observed factors $\boldsymbol{x}^{(i,t)} \in \mathbb{R}^n$. Note that the stocks are indexed as $i = 1, \cdots, N_t$, and months are indexed as $t = 1, \cdots, T$. We will omit the superscript and represent the factors of an asset as $\boldsymbol{x}$ if there is no ambiguity.

Remark 2.1. *It is worth noting that the result of the function $\boldsymbol{y}(\cdot)$ does not depend on any stock or month. Namely, we use the same function estimator to estimate the asset's excess for stocks at different times. This function relies only on the factors of one period, without any information about other stocks at any time. $r_{i,t}$ represents the excess return of stock i at time t. In this model, output $r_{i,t+1}$ represents the predicted excess return in time $t+1$ of stock i. Also, if we talk about portfolios, the output can be our average predicted excess return of a certain portfolio in time $t+1$.*

To estimate the risk premiums for factor $\boldsymbol{x} \in \mathbb{R}^n$, an intuitive approach is to use linear model, such as Ordinary Least-Squares (OLS) regression Hutcheson (2011) or Support Vector Regression (SVR) Awad and Khanna (2015). The core idea is to model the result, the estimated risk premiums $\hat{y}(\boldsymbol{x})$ as a linear combination of input factors $\boldsymbol{x} \in \mathbb{R}^n$. That is,

$$\hat{y}(\boldsymbol{x}) = w_0 + \boldsymbol{w}^T\boldsymbol{x} = w_0 + \sum_{i=1}^{n} w_i x_i, \qquad \text{(Linear Model)}$$

where $w_0 \in \mathbb{R}, \boldsymbol{w} = (w_1, w_2, \ldots, w_n)^T \in \mathbb{R}^n$ are the parameters to be optimized.

However, the linear model assumes that the contributions of the input factors to the result are independent. The interactions between factors are ignored, which also plays an important role in risk premiums (see Gu et al. (2020)).

2.2 FAPM: Learning second-order factor interactions

The main idea of FAPM is to estimate the risk premiums using second-order factor interactions, i.e., the second-order term $x_i x_j$.

To begin with, we establish the naive Second-order Model.

$$\hat{y}(\boldsymbol{x}) = w_0 + \sum_{i=1}^{n} w_i x_i + \sum_{i=1}^{n} \sum_{j=i+1}^{n} B_{ij} x_i x_j. \qquad \text{(Second-order Model)}$$

Proposition 1. *In Second-order Model, the total number of parameters is* $O(n^2)$[4].

Proof. The parameters needs to be optimized are $w_0 \in \mathbb{R}, \boldsymbol{w} = (w_1, w_2, \ldots, w_n) \in \mathbb{R}^n, \boldsymbol{B} = [B_{ij}]_{1 \le i < j \le n}$. The size of each part of parameters are $1, n, \frac{n(n-1)}{2}$. The total size of parameters is $\frac{n(n-1)}{2} + n + 1 = O(n^2)$. □

Note that the second-order weight parameter $\boldsymbol{B}$ is extremely high-dimensional, with $\frac{n(n-1))}{2}$ parameters. This situation brings difficulties to the conventional learning method. First, the more parameters a model has, the more training data it requires to make the model well-generalize. Otherwise, the learned model will easily be overfitting (see Mohri, Rostamizadeh, and Talwalkar (2018)). Second, the training data has to cover all the factor interactions to optimize the whole parameter space. This is not realistic in many cases, since the second-order factor interactions are usually sparse (see Rendle (2010)).

To reduce the dimension of the second-order weight parameter $\boldsymbol{B}$, we factorize it by a low-rank matrix $\boldsymbol{V} = (\boldsymbol{v}_1, \boldsymbol{v}_2, \ldots, \boldsymbol{v}_n) \in \mathbb{R}^{k \times n}$, i.e.,

$$\boldsymbol{B} := \boldsymbol{V}^T \boldsymbol{V}, \qquad \text{(Factorization)}$$

[4]Here $O(\cdot)$ is a function of the time complexity of the algorithm. That means that the algorithm takes approximately $\cdot$ calculations

where $k << n$ is a hyperparameter, and $\boldsymbol{v}_i \in \mathbb{R}^k$ is the i-th column vector for the matrix $\boldsymbol{V}$. Based on the factorization, we get the proposed model, **F**actorization **A**sset **P**ricing **M**odel (FAPM):

$$\hat{y}(\boldsymbol{x}) := w_0 + \sum_{i=1}^{n} w_i x_i + \sum_{i=1}^{n} \sum_{j=i+1}^{n} \langle \boldsymbol{v}_i, \boldsymbol{v}_j \rangle x_i x_j, \tag{FAPM}$$

where $\langle \cdot, \cdot \rangle$ is the dot product.

Proposition 2. *By making $\boldsymbol{B} = \boldsymbol{V}^T \boldsymbol{V}$, we reduce the parameter space from $O(n^2)$ to $O(nk)$.*

Proof. The parameters needs to be optimized are $w_0 \in \mathbb{R}, \boldsymbol{w} = (w_1, w_2, \ldots, w_n) \in \mathbb{R}^n, \boldsymbol{V} = (v_1, v_2, \ldots, v_n) \in \mathbb{R}^{n \times k}$. The size of each part of parameters are $1, n, nk$. The total size of parameters is $nk + n + 1 = O(nk)$. □

Now we are going to better understand the factorization $\boldsymbol{B} = \boldsymbol{V}^T \boldsymbol{V}$. For any positive definite matrix $\boldsymbol{B}$, there exists a matrix $\boldsymbol{V}$ such that $\boldsymbol{B} = \boldsymbol{V}^T \boldsymbol{V}$ if k is sufficiently large. Therefore, by adjust k, FAPM can sufficiently express the interaction matrix $\boldsymbol{B}$. We restrict k to restrict the expressiveness of FAPM in order to better generalize under sparse factor interactions.

The second-order factor interaction part in Equation (FAPM) can be computed in $O(nk)$ as following:

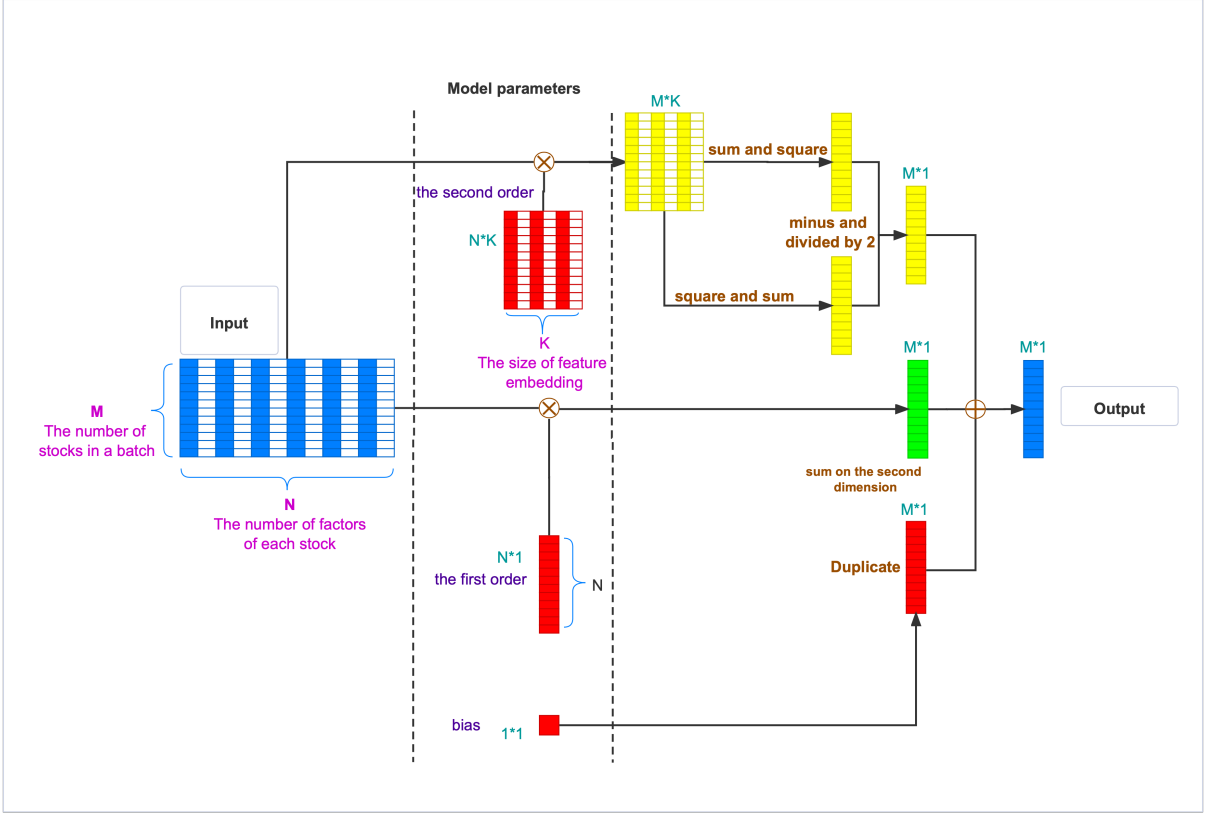
$$\begin{aligned}
\sum_{i=1}^{n} \sum_{j=i+1}^{n} \langle \boldsymbol{v}_i, \boldsymbol{v}_j \rangle x_i x_j =& \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \langle \boldsymbol{v}_i, \boldsymbol{v}_j \rangle x_i x_j - \frac{1}{2} \sum_{i=1}^{n} \langle \boldsymbol{v}_i, \boldsymbol{v}_i \rangle x_i x_i \\
=& \frac{1}{2} \left(\sum_{i=1}^{n} \sum_{j=1}^{n} \sum_{f=1}^{k} (x_i v_{i,f})(x_j v_{j,f}) - \sum_{i=1}^{n} \sum_{f=1}^{k} v_{i,f}^2 x_i^2 \right) \\
=& \frac{1}{2} \sum_{f=1}^{k} \left[(\sum_{i=1}^{n} v_{i,f} x_i)(\sum_{j=1}^{n} v_{j,f} x_j) - \sum_{i=1}^{n} v_{i,f}^2 x_i^2 \right] \\
=& \frac{1}{2} \sum_{f=1}^{k} \left[(\sum_{i=1}^{n} v_{i,f} x_i)^2 - \sum_{i=1}^{n} v_{i,f}^2 x_i^2 \right].
\end{aligned} \tag{1}$$

Based on Equation (1), we can compute FAPM model efficiently with computation

complexity $O(nk)$:

$$\hat{y}(\boldsymbol{x}) := w_0 + \sum_{i=1}^{n} w_i x_i + \frac{1}{2} \sum_{f=1}^{k} \left[(\sum_{i=1}^{n} v_{i,f} x_i)^2 - \sum_{i=1}^{n} v_{i,f}^2 x_i^2 \right]. \qquad \text{(FAPM:}O(nk)\text{)}$$

Figure 1: The implementation of the algorithm on a batch.
The input is an $M \times N$ matrix, where M is the number of stocks in a batch and N is the number factors of each stock. The output is an M-dimensional vector, the predicted risk premiums for each stock. The yellow, green and red m-dimensional vectors before the output are the result second-order term (see Equation (1)), linear term and bias term in Equation (FAPM:$O(nk)$).



[Insert Algorithm 1 Here]

Figure 1 and Algorithm 1 present the model's predictive processing for each input, specifically the implementation of the above equation.

Moreover, we establish a simple example to illustrate our approach.

Example 1. *Assume that we have a simple asset pricing problem. For each stock, we define the factor below and detailedly in figure 2:*

[Insert Figure 2 Here][5]

Assume that we have a simple asset pricing problem (see Figure 2). Each stock contains information of permno, time, continuous factors, and domains (discrete factors). For each domain, the $0/1$ *value indicates that whether this stock belongs to the corresponding domain (*1 *for positive and* 0 *for negative). We set the continuous factors and domains as the factors we use in our task. Assuming that we set* $k = 3$*. As a result, the number of parameters of* $\boldsymbol{V}$ *is* $8 \times 3 = 24$*, while the number of different second-order interactions are* $\frac{8\times(8-1)}{2} = 28$*.*

2.3 Optimization

The gradients of the FAPM with respect to its parameters$(w_0, \boldsymbol{w}, \boldsymbol{V})$ are:

$$\frac{\partial}{\partial\theta}\hat{y}(\boldsymbol{x}) = \begin{cases} 1, & for\ \theta = w_0, \\ x_i, & for\ \theta = w_i, \\ x_i \sum_{j=1}^{n} v_{j,f} x_j - v_{i,f} x_i^2, & for\ \theta = v_{i,f}. \end{cases} \tag{2}$$

As a matter of result, the model parameters$(w_0, \boldsymbol{W}, \boldsymbol{V})$ of FAPM can be optimized efficiently by gradient descent methods such as stochastic gradient descent (SGD) Bottou (2012) for a variety of differentiable loss functions.

Denote $\hat{r}_{i,t+1} = \hat{y}(\boldsymbol{x}_{i,t})$ and $r_{i,t+1}$ as the estimated risk premiums and ground truth for asset $\boldsymbol{x}_{i,t+1}$, respectively. To evaluate the performance of the predicted result in training,

[5]This figure shows a simple example of asset pricing problem. The first part is a batch of stock factors. And the second part shows how to generate the second-order relationships.

we define in-sample R^2_{is} as

$$R^2_{is}(r, \hat{r}) = 1 - \frac{\sum_{(i,t)\in\tau_4}(r_{i,t+1} - \hat{r}_{i,t+1})^2}{\sum_{(i,t)\in\tau_4} r^2_{i,t+1}}, \qquad \text{(FAPM:}R^2_{is}\text{)}$$

where τ_4 is the set of all the training samples, i.e., the assets that appeared in the training. Since R^2_{is} is differentiable with respect to the estimate result $\hat{r}_{i,t+1} = \hat{y}(\boldsymbol{x}_{i,t})$. Therefore, we can also use it as the loss function to optimize the parameters of FAPMthrough gradient descent.

Similarly, we define out-of-sample R^2_{oos} as

$$R^2_{oos}(r, \hat{r}) = 1 - \frac{\sum_{(i,t)\in\tau_3}(r_{i,t+1} - \hat{r}_{i,t+1})^2}{\sum_{(i,t)\in\tau_3} r^2_{i,t+1}}, \qquad \text{(FAPM:}R^2_{oos}\text{)}$$

where τ_3 is the test set, whose data never enter into model training. R^2_{oos} pools prediction errors across firms and over time into a grand panel-level assessment of each model. It can be used to evaluate the generalization ability of the learned model.

Remark 2.2. *For the loss function, it is common to consider the $\mathcal{L}_2$ regularization Girosi, Jones, and Poggio (1995). It can be used to constraint the parameters of the model to avoid overfitting. However, it doesn't work well in our experiments. It only works when the coefficient of regularization is extremely large, with a significant increase of training time. Moreover, we found that by only applying the R^2_{is} loss (defined in Equation (FAPM:R^2_{is})) we can obtain an excellent model which outperforms all the previous works. As a result, we only set R^2_{is} as loss function.*

2.4 Framework

[Insert Figure 3 Here][6]

[Insert Algorithm 2 Here]

[6]This figure illustrates the general framework of FAPM algorithm operation. The model and data set are initialized first. When the model and data set are in place, the training begins. We trained the epoch wheel each time, and from the results of these multiple trainings, selected the model that performed best on the validation set. Perform test set predictions on selected models, save and evaluate the results. By rolling adjustment of training set, verification set and test set according to year, the final test set prediction result of year was obtained.

The figure 3 and algorithm 2 illustrate the general framework of FAPM algorithm operation. The model and data set are initialized first. The second-order parameters of the model are initialized randomly according to the normal distribution, while the first-order parameters are initialized according to the weights of the linear model. Such treatment is in line with the demand, because FAPM degenerates into a simple linear regression model without considering the second-order relationship. We only need a model that is stronger than the linear model. If the FAPM is inferior to the linear model, there is no need for us to study. In other words, we can build a second-order relationship based on a first-order linear model. Then when training, we selected the model that performed best on the validation set. Perform test set predictions on selected models, save and evaluate the results. By rolling adjustment of the training set, verification set, and test set according to year, we can obtain the yearly prediction result for the test set.

3 Empirical Evidence

3.1 Dataset and Implementation

Following Gu et al. (2020), we obtain monthly total individual equity returns from CRSP for all firms listed in the NYSE, AMEX, and NASDAQ. The data starts in March 1957 (the start date of the S&P 500) and ends in December 2017. As a result, we have 31,924 stocks in 60 years. After that, we calculate individual excess returns from the Treasury-bill rate to proxy for the risk-free rate. In addition, we build 94 factors (61 of which are updated annually, 13 are updated quarterly, and 20 are updated monthly) based on a cross-section of stock returns literature. Besides, there are 74 industry dummies corresponding to the first two digits of Standard Industrial Classification (SIC) codes included.

We divide the whole dataset into the initial training set (the first 18 years, 1957-1974), the initial validation set (the following 12 years, 1975-1986), and the out-of-sample testing (the last 30 years, 1987-2016). To evaluate our algorithm, we adopt a "rolling training and testing" approach. That is, we run the algorithm, including the training and testing, for 30 times with different training, validation and test set. At the beginning, the test set is the data in 1987, the training set is the data from 1957 to 1974, and the validation set is the data from 1975 to 1986. After we finish the training and testing procedure for the first setting,

we extend the training set as the data from 1957 to 1975, set the validation set as the data from 1976 to 1987, and adopt the data in 1988 as the test set. Repeat the process above, and we will obtain the resulted R^2_{oos} for each year in the 30 years (1987-2016). The final result is computed by averaging the result in each testing year. It's worth noting that our calculation for R_{oos} here is very straightforward. In the process of model rolling training, we will calculate the R_{oos} of the year according to the annual predicted sequence and actual sequence. At the end of model training, we spliced the 30-year predicted sequence with the 30-year actual sequence to calculate the overall R_{oos}.

Implementation. Figure 3 and Algorithm 2 illustrate the general framework of our algorithm. Firstly, we set the training, validation and testing data via the "rolling training and testing" approach described above. Then, we initialize $\boldsymbol{V}$ and w (the second-order and linear parameters) randomly using normal distribution. During training, we select the model that performs best on the validation set with respect to R^2_{oos}. After we trained the model, we evaluate it by computing R^2_{oos} on the test set.

3.2 The cross-section of individual stocks

Here we present the empirical results of our FAPM model. We compare our model with all the comparison method in Gu et al. (2020): OLS with all covariates, OLS-3 (which preselects size, book-to-market, and momentum as the only covariates), PLS, PCR, elastic net (ENet), generalized linear model with group lasso (GLM), random forest (RF), gradient boosted regression trees (GBRT), and neural network architectures, proposed in Gu et al. (2020) with one to five layers (NN1,...,NN5). The experiment results are shown in Table 1 and Figure 4.

[Insert Table 1 Here][7]

[Insert Figure 4 Here][8]

[7] In this table, we report monthly R^2_{oos} for the entire panel of stocks using OLS with all factors (OLS), OLS using only size, book-to-market, and momentum (OLS-3), PLS, PCR, elastic net (ENet), generalize linear model (GLM), random forest (RF), gradient boosted regression trees (GBRT), neural networks with 1 to 5 layers (NN1–NN5), FAPM and FAPM-7. "+H" indicates the use of Huber loss instead of the l2 loss.We also report these R^2_{oos} within sub samples that include only the top-1,000 stocks or bottom-1,000 stocks by market value.

[8] The figure of Table 1

We can see from the results that,

- It is difficult for OLS-3+H to reach a positive R^2_{oos}, and its performance on top-1,000 stocks is extremely inferior.
- For all the comparison methods (without FAPM), NN4 gets the best performance in all the settings. But in general, the gap between the generalized linear model and NN4 is not large. Linear models have their own advantages.
- FAPM, as a novel generalized linear model, embodies this advantage. Compared with all the baseline algorithms, FAPM-7 reaches the best R^2_{oos} of 1.01%.
- For the top-1,000 stocks, the R^2_{oos} increased from 0.67% to 1.28% (FAPM-7), far outperforming the other models.
- For the Bottom-1000 stocks, the $R^2_{oos}(0.81\%)$ of FAPM-7 outperforms all the comparison algorithms as well.

Ideally, For FAPM, The more factors it has, the better performance it could reach. However, during practice, we find that training with too many factors, especially the redundant ones, may cause adverse factor interactions that are not conducive to prediction. The underlying reason is that, the empirical data is usually noisy, thus all the models, including FAPM, are easily overfitting. Inspired by this, we consider selecting several representative factors for our model to train. We do so by selecting factors via a "one strong factor with one weak factor" approach. That is, we set a specific number m and then select M factors with the best performance and m factors with the worst performance to train FAPM. As m increase, R^2_{is} will fluctuate. The results are shown in Table 2 and Figure 5.

[Insert Table 2 Here][9]

Table 2 shows the performance of FAPM when we train the model via the top-m and bottom-m (for $m \leq 7$) important factors. The importance of factor i is defined as the first-order parameter w_i of OLS model with all factors. We can see that The top-1 factor, $ntis$, reaches the importance of 57%. This is the reason why a simple pair of factors can make the predictive R^2 up to 0.48%, and when $2 \leq m \leq 3$, the result is close to the case

[9]The importance correspond to the top-7 and bottom-7 factors chosen for prediction. factor importance each line is normalized to sum to one. The figure shows the standard R^2 (all the factors) and the R^2_{is} of only m pairs factors using "one strong factor with one weak factor".

when $m = 1$. However, when $m \geq 4$, the performance of FAPM significantly improve (from 0.43% to 0.80%). The performance even outperforms the FAPM model using all the factors. The reason is that, as we will discuss later, there is a strong interaction between the factor *bm* and *ntis*.

[Insert Figure 5 Here][10]

In Figure 5, we further present the performance of FAPM using top-m and bottom-m (For $m \leq 20$) importance factors. We can observe that

- When $M = 7$, the R^2_{oos} reaches the optimum value 1.01%. Notice that the selected factors here are mostly macro factors.
- When $M = 15$, the R^2 reaches 0.75%, which is the second maximum point.
- When $M > 15$, The relationship between R^2_{oos} and m is no longer significant.

We define FAPM-m as FAPM using the top-m and bottom-m factors in training. In Table 1, we can see the R^2_{oos} of FAPM-7 significantly outperforms all the other algorithms, including NN4.

When $m = 7$, since all factors are macro factors, we try to add individual stock factors to enhance interpretation, such as SIC2. At each feeding of a pair of factors, a 75-dimensional SIC2 factor (one-hot treatment) is added. Follow the training method described above, we get a set of R^2_{oos}.

[Insert Table 3 Here][11]

[Insert Figure 6 Here][12]

Figure 6 and Table 3 show the comparison of overall R^2_{oos} in the final test set before and after the addition of SIC2 factors. In a word, the addition of domains has a negative influence on predictions, though SIC2 increases explanatory ability. When $M = 1$, addition of

[10]This figure reports the in-sample R^2 of different pairs of factors. The blue line reports the total R^2_{is} of m-pairs factors, where m is x-axis. The blue line reports the R^2 of the FAPM with all factors. The vertical axis is R^2 of monthly returns.

[11]In this table, we report the R^2_{oos} of different pairs(1,2,$\cdots$,7) of factors with and without the discrete factor SIC2.

[12]This figure reports the R^2_{oos} of the FAPM with m-pair factors, with and without factor SIC2.

SIC2 improves the R^2 from 0.48% to 0.51%. When $M = 7$, addition of SIC2 decreases the R^2 from 1.01% to 0.87%. After adding SIC2, with the increase of training parameters, the training effect gradually changed from improvement to equivalent, and then decrement. Although theoretically, the addition of SIC2 should strengthen the interpretation of the model, and the training effect will be better. Still, from the results of a few factors, the effect may be better without the addition of SIC2. As for the follow-up, when there are enough factors, adding SIC2 is obviously beneficial to model training and convergence, but we do not consider it at present. We can see that the interpretation of few factors is enhanced after the addition of SIC2, and the effect of the model is improved. In the process of increasing factors, this effect is not obvious and will even weaken the model's prediction ability.

Figure 7 and 8 show year-by-year in-sample R^2 of model with and without SIC2. From the training results, after SIC2 is added, the influence of the increase of factors in the initial training period would be decreased. In contrast, at the end of the training period (2009-2020), the training effect of each model fluctuated violently and was uneven. And if we don't add the factor SIC2, all the models follow a trend generated by the data set itself. For example, they all perform well in years 1 and 22, while in year 23 have a terrible prediction. This may indicate that SIC2 has not correlated sufficiently with the returns in recent years. However, the relationship between SIC2 and return was not weak in previous years, which may lead to some bad first or second-order relationships with enormous weights, thus affecting the forecast results.

[Insert Figure 7 Here][13]

[Insert Figure 8 Here][14]

Figure 9 shows the annual differences between each model and standard model (all factors) R^2. That is, we subtracted the annual prediction R^2 of the full-factor model from the annual prediction R^2 of each model with a few exceptions. Almost all models consistently outperformed standard models in year-to-year forecasting. The green line repre-

[13]We report different models' in-sample R^2 of each training year on train set. Those models are different m-pair FAPM s with SIC2 factor

[14]We report different models' in-sample R^2 of each training year on train set. Those models are different m-pair FAPM s without SIC2 factor

sents the annual training R^2 when $M = 7$, which is significantly higher than other models. And the improvement even reaches 0.05% in the year 2003. This may further prove the validity of the "strong factor and weak factor" approach.

[Insert Figure 9 Here][15]

3.3 Factor importance

One advantage of FAPM is that we can precisely extract the parameters of the factors we need. These parameters represent the extent to which this factor affects the dependent factor. As we mentioned in the figure 1, we divided the model parameters into three parts. The first-order parameter is actually the independent importance of each factor, while the second-order parameter can calculate the importance of interaction between factors. Using these parameters, we can also calculate the T-statistics of each factor and so on, to evaluate the importance of the factor. Here, we simply show the 12 factors with the highest absolute value of coefficients. We reduce the coefficients of these factors to 1 in order to reflect the impact of different factors on model prediction intuitively. Obviously, such a reduction does not affect the mutual ranking of factors.

Figure 10 shows the 12 largest coefficients, ranked from largest to smallest in absolute value. They are: Short-term reversal($mom1m$), Industry momentum($indmom$), Recent maximum return($maxret$), Risk measures constitute the third influential group($retvol$), Real estate holdings($realestate$), Earnings volatility($roavol$), Log market equity($mve1$), Asset growth(agr), Revenue surprise($rsup$), Share turnover($turn$), stock Momentum($mom12m$), and Bid-ask spread($baspread$). Noting that here we only report the individual factors, and Short-term reversal($mom1m$) has 48% importance of all factors, which makes sense. From this, we can see the first-order importance of factors, and it's democratic, drawing predictive information from a broader set of characteristics.

[Insert Figure 10 Here][16]

[15] We report the difference between the predicted R^2 of different models and the full-factor predicted R^2 by year

[16] factor importance for the top-12 most influential factors in each model. factor importance is an average over all training samples. factor importance within the model is normalized to sum to one.

Indeed, since most macro factors have a greater impact on returns than individual stock factors, their importance is, of course, far greater than individual stock factors, so we separate macro factors for alternative discussions. The Figure 11 and Table 4 show the R^2-based importance measure for each macroeconomic predictor (again normalized to sum to one within a model). In Gu et al. (2020), book-to-market ratio (bm) is the most important macroeconomic factor trained by FAPM. Therefore, according to the model in this paper, the book-to-market ratio(bm) is still a significant macro factor. We can see that the importance of bm reaches 57.2%, far larger than 27% in NN4. And default spread(dfy) also plays a crucial role with the importance of 34%, next to 42% in the ENet model. Regarding the importance of other macro factors, the results of FAPM's model are more similar to those of linear and generalized linear models. This is mainly reflected in the high importance of the default spread(dfy) factor and the low importance of dividend-price ratio(dp), earnings-price ratio(ep), and other factors. So the $ntis$ and tbl factors are much lower than nonlinear model NN4. Many accounting characteristics have low importance because they are not available at the monthly frequency.

[Insert Table 4 Here][17]

[Insert Figure 11 Here][18]

Next, we explore interaction between factors. As we mentioned in equation (Factorization), we can get the second-order coefficient matrix of the factors. Table 5 illustrates the 20 most important interactions which are showed in Figure 12 and 13. The strongest second-order relationship exists between $ntis$ and bm, which reaches 37%. Gu et al. (2020) also reports the high interaction between $ntis$ and bm factors, although using an absolutely different approach. Meanwhile, among all the factors, the interaction between macroeconomic factors is obviously stronger than that between a macroeconomic factor and an individual stock factor. The latter is also obviously stronger than the internal interaction between individual stock factors. Interactions between individual stock factors are not large, and the macroeconomic factors play the chief characteristic. That

[17] factor importance for eight macroeconomic factors in each model. factor importance is an average over all training samples. factor importance within each model is normalized to sum to one.

[18] The figure of Table 4

interaction makes economic sense. For example, the strong second-order relationship between *bm* and *ntis* indicates that the size effect is more pronounced when low aggregate valuations (*bm* is high). When equity issuance (*ntis*) is low, the low volatility anomaly is powerful in high valuation and issuance environments. As for interactions between individual factors, Cash flow volatility($stdcf$), Earnings volatility(*roavol*) and Corporate investment(*cinvest*) perform strongest.

[Insert Table 5 Here][19]

[Insert Figure 12 Here][20]

[Insert Figure 13 Here][21]

Through Table 5, we also find that *ntis* has a better performance in interactions than any other macroeconomic factors. And also, Cash flow volatility($stdcf$) performs best among individual factors. We suspect some factors might be better at maintaining second-order relationships, even though first-order relationships are less meaningful. These factors make it easier to establish many effective second-order relationships, so much so that all of them are involved. This is an evaluation of the importance of metrics on another level. We think roughly that the second-order significance of a factor is the sum of all its related second-order relationships. To prove that, we compute the average interactions of each factor and report in Table 6 and Figure 14. Apparently, *ntis*-related interactions accounted for 36.6% of all interactions, the highest of all. This means that the level of *ntis* to some extent, affects the impact of other factors on asset returns. The sensitivity of

[19]In this table, we rank all the interaction of the total 95 individual and 8 macroeconomic factors and choose some on the top. Then we normalize all the interaction in order to easily report the comparison. The first row of the table reports the interactions between macroeconomic factors, and the second row reports the interactions between an individual factor and another.

[20]In this figure, we report the interaction between macroeconomic factors and between macro and individual factors. We rank all those interactions and choose the top-12. All the interactions are normalized to 1. Actually, we report the interaction of $bm - ntis$, $ntis - dp$, $ntis - ep$, $ntis - dfy$, $bm - dp$, $bm - ep$, $ntis - tms$, $bm - dfy$, $ntis - sgr$, $ntis - tbl$, $ntis - cash$, $ntis - stdcf$.

[21]In this figure, we report the interactions between individual factors. In fact, we only choose the top-6 and normalize them to 1 to report.

many factors to R^2 is affected by the size of *ntis* factor.

[Insert Table 6 Here][22]

[Insert Figure 14 Here][23]

4 Factorization Portfolios

So far, we have analyzed the predictability of individual stock earnings. FAPM has an excellent performance in predicting individual stocks. Next, we compare the results of FAPM and other models in portfolio return prediction to reflect the superiority of FAPM. The analysis of the investment portfolio has vital practical significance.

[Insert Figure 15 Here][24]

First, portfolio forecasting provides additional indirect evaluation of the model. By establishing new sample data based on the original data set, the robustness of the model is guaranteed.

Second, in real life, portfolios are more common than individual stocks. A good portfolio usually makes the returns much more robust. We evaluate the predictive performance of models by studying value-weighted portfolios, providing economic significance for models in the most valuable (and essential) assets.

Third, the distribution of portfolio returns is sensitive to the dependence of stock returns, so a good stock return prediction model cannot guarantee the accurate prediction of the portfolio level. Bottom-up portfolio forecasting enables us to evaluate the model's ability to translate its asset forecasting into a broader and more complex investment environment.

[22] In this table, we also deal with the interactions between all the 103 factor. Consider the absolute value of the coefficient as the interaction. Then we compute the average of all the interactions of each factor. Sum the average returns to one, and choose the top-20.

[23] This figure reports the average interaction of each factor using pie. The 20 most significant factors are *ntis*, *bm*, *dp*, *ep*, *dfy*, *securedind*, *tms*, *stdcf*, *cash*, *tbl*, *roeq*, *roaq*, *stdacc*, *roavol*, *cinvest*, *aeavol*, *rsup*, *nincr*, *ear*, *chtx*. Their corresponding colors are shown in the legend. Gray represents other factors.

[24] This figure shows how to generate portfolios from dataset

4.1 Prespecified portfolios

We build bottom-up forecasts by aggregating individual stock return predictions into portfolios. This bottom-up approach works for any target portfolio whose weights are known a priori. The portfolio return forecast is constructor as

$$\hat{r}_{t_1}^{p} = \sum_{i=1}^{n} w_{i,t+1}^{p} \hat{r}_{i,t+1}, \tag{3}$$

where p is a portfolio and is denoted as $w_{i,t+1}^{p}$ for stock i. $\hat{r}_{i,t+1}$ is a model-based out-of-sample forecast for stock i.

In this part, we compare the performance of FAPM , partial generalized linear model (OLS-3+H,PLS,PCR,ENet+H,GLM+H,RF,GBRT+H) and neural network (NN4) in portfolio prediction. Refer to Gu et al. (2020) for detailed definitions of these models. For each model, we made bottom-up predictions for the 30 best-known portfolios in the previous empirical studies. These portfolios include S&P 500, the Fama-French size, value, profitability, investment, and momentum factor portfolios (SMB, HML, RMW, CMA, and UMD, respectively), and subcomponents of these Fama-French portfolios, including six size and value portfolios, six size and investment portfolios, six size and profitability portfolios, and six size and momentum portfolios. The subcomponent portfolios are long only, and SMB, HML, RMW, CMA, and UMD are zero-net-investment long-short portfolios. We create the portfolios ourselves using CRSP market equity value weights in all cases. According to the current data, we adjust positions at time t and adjust the portfolio at time $t+1$. Although different from the approach of S&P 500 index and the characteristic-based Fama-French portfolios(Fama and French (2021)), we can clearly track the weight of each portfolio and adjust it flexibly. The figure 15 and algorithm 3 shows how to generate portfolios. The purple and red lines mean splitting the dataset and ranking them by the factor in the rectangle. The blue and pink lines mean combining them with small and big groups. The green lines refer to especially calculating for common factor portfolios.

[Insert Table 7 Here][25]

[25]In this table, we report the out-of-sample predictive R^2s for thirty portfolios using OLS with size, book-to-market, momentum, OLS-3, PLS, PCR, elastic net (ENet), generalized linear model with group lasso (GLM), random forest (RF), gradient boosted regression trees (GBRT), neural networks (NN4), FAPM, FAPM-7. "+H" indicates the use of Huber loss instead of the l_2 loss. The six portfolios in panel A are the

Table 7 reports the monthly R^2_{oos} over our 30-year testing set. Actually, we calculate the predicted index and the actual index for all the stocks in each period, and then use the index sequence to compute R^2. Mostly, generalized linear models are poor predictors of the CMA portfolio returns. The FAPM-7 can improve the out-of-sample prediction R^2 from 0.99% to 1.89%, which is a significant improvement. It is worth mentioning that the R^2_{oos} of neural network(NN4) in Gu et al. (2020) can reach 1.84%, while it is only 1.36% in this paper. It means that the neural network may not be robust enough relative to the FAPM . Figure 16 reports the comparison among those 10 models. Obviously, the generalized linear model is difficult to give a good prediction of the portfolio, with negative R^2 appearing on many indices. In particular, OLS-3 has a much lower fitting effect than other models. The performance of the neural network (NN4) is also unsatisfactory. Although it performed well in the portfolio "Big Market value", it did not achieve the expected effect in general. However, FAPM and FAPM-7 model always maintains a high R^2. Especially in the common Factor portfolios, the FAPM-7 has a surprising advantage.

[Insert Figure 16 Here][26]

Compared with other famous portfolio predictors, we can find the advantage of FAPM. In the survey, Welch and Goyal (2007), nearly all the macroeconomic return predictor factors failed to produce a positive R^2_{oos}. Kelly and Pruitt (2013) find that PLS's R^2_{oos} can reach 1%, though their forecast is not based on the bottom-up approach. Even Cochrane (2007), the most well-studied portfolio predictors, just produce an in-sample predictive R^2 around 1%, which is lower than that we find in FAPM.

Next, we compare neural network (NN4) and FAPM in detail. First, positive R^2 is obtained for all portfolio predictions of both, which is in line with the conclusion of Gu et al. (2020). But only in a few portfolios, such as HML, did NN4 outperform FAPM by 0.59%. For most portfolio forecasts, FAPM performance is significantly better than NN4 results. In some subcomponents of factor portfolios, the difference between them can reach 1.2%. And on average, FAPM was almost twice as good as NN4.

S&P 500 indices and the Fama-French SMB, HML, CMA, RMW, and UMD factors. The twenty-four portfolios in panel B are 3×2 size double-sorted portfolios used in the construction of the Fama-French value, investment, profitability, and momentum factors.

[26]The figure of Table 7

Campbell and Thompson (2007) indicates that a minimal change in R^2 can cause a massive shift in utility gains for a mean-variance investor. They define the Sharpe ratio (SR^*) earned by an active investor exploiting predictive information improves over the Sharpe ratio (SR) made by a by-and-hold investor according to

$$SR^* = \sqrt{\frac{SR^2 + R^2}{1 - R^2}}, \tag{4}$$

Where R^2 stands for the performance for the predictor. When the predictive information is more valuable, we can gain a more significant improvement in SR.

We first calculate the full-time Sharpe ratio of each portfolio earned by a by-and-hold investor. Then we translate the predictive R^2_{oos} and the SR calculated into an improvement in annualized Sharpe ratio, $SR^* - SR$, for an investor exploiting machine learning predictions for portfolio timing. For example, the buy-and-hold Sharpe ratio of the S&P500, which is 0.51 in the 30-year out-of-sample period, improved to 0.54 by a market-timer exploiting forecasts from the FAPM . For characteristic-based portfolios, FAPM learning methods improve Sharpe ratios by anywhere from a few percentage points to over 27 percentage points.

[Insert Table 8 Here][27]

Table 8 reports the annualized Sharpe ratio gains (relative to a buy-and-hold strategy) for timing strategies based on machine learning forecasts. "-" means the predicted $R^2 < 0$. In our results, the strongest and most consistent trading strategies are mostly based on nonlinear models (without FAPM). They all get improvement in all portfolios. Remarkably, in FAPM-7, the HML index is improved 27%, and the CMA index of FAPM is 45% higher than a buy-and-hold position, which is significantly better than that reports in Gu et al. (2020). Figure 17 shows that in detail. Clearly, in a subcomponent of factor portfolios, FAPM-7 performs much better than other models, especially in the "small market value" group.

[Insert Figure 17 Here][28]

[27]This table documents improvement in annualized Sharpe ratio $SR^* - SR$. We compute the SR^* by weighting the portfolios based on a market timing strategy (see Campbell and Thompson (2007)).

[28]The figure of Table 8

4.2 Machine learning portfolios

Since the discussion above uses only R^2, not all the forecast information, we will discuss none of the portfolios above. We tried to construct a new portfolio to make predictions directly using machine learning approach. At the end of each month, we forecast the information for the next month, dividing the forecast into ten fractions from the smallest to the largest, and using market weight to restructure the ten portfolios. We construct a zero-net portfolio, buying the group with the best forecast(decile 10) and selling the group with the worst prognosis (decile 1).

Table 9 reports the results. For each machine learning prediction, the actual gains are monotonously increased. The grouping of each model is of practical significance. In the table, all data except SR are percentages. We noted that the SR of the portfolio constructed according to the OLS-3 model can reach 0.61, which is consistent with Gu et al. (2020). However, the SR of the neural network (NN4), which performed well in the previous test, was only 0.80, which was like that of the PCR-model in Gu et al. (2020), not quite in line with the author's expectation. To explain this problem, in fact, the training of neural networks may have different degrees of fitting in different parts of the data. The prediction results of the neural network in this paper may not fit well at the beginning and end, resulting in the final SR no longer reaching the previous desirable outcome. Neural network(NN4) is less stable in such a structured portfolio. Among the models compared in this paper, FAPM-7 still has outstanding performance, with its corresponding SR reaching 1.42, nearly two times higher than the best model performance in the current stage.

[Insert Table 9 Here][29]

[29]In this table, we report the performance of prediction-sorted portfolios over the 30-year out-of-sample testing period. All stocks are sorted into deciles based on their predicted returns for the next month. Columns "Pred", "Avg", "SD", and "SR" provide the predicted monthly returns for each decile, the average realized monthly returns, their standard deviations, and Sharpe ratios, respectively. All portfolios are value weighted.

5 Conclusion

We propose a generalized linear framework based on the factorization method to investigate factor interactions. We adopt this novel approach to estimate the risk premium of observable factors in a linear asset pricing model, thereby proposing a Factorization Asset Pricing Model (the FAPM). Our methodology relies on a simple generalized linear factor model plus the components of factor interactions based on the factorization method. In a linear framework, the FAPM can correct the omitted factor interactions problem in cases where the data sets are sparse. In this case, the risk premiums for observable factors are estimated more accurately, and the model performance is the best around the currently existing methods.

Our FAPM can be viewed as an extension of the conventional factor asset pricing models, including the Capital Asset Pricing Model (the CAPM) and Arbitrage Pricing Theory (the APT). In particular, it can be thought of as the first formalized model that incorporates factor interactions. It can also be thought of as the benchmark to evaluate the dedicated model for identifying the factor interactions. The main advantage of our FAPM is that it provides a linear, specific, and systematic way to tackle the concern that the model predicted by theory is misspecified because of omitted factor interactions. Rather than relying on arbitrarily chosen interactive fixed effects and ”universal approximations’. It also explicitly takes into account the possibility of approximation error in any linear factor models with observed factors.

References

Anatolyev, Stanislav, and Anna Mikusheva, 2021, Factor models with many assets: strong factors, weak factors, and the two-pass procedure, *Journal of Econometrics* .

Awad, Mariette, and Rahul Khanna, 2015, Support vector regression, in *Efficient learning machines*, 67–80 (Springer).

Bai, Jushan, 2009, Panel data models with interactive fixed effects, *Econometrica* 77, 1229–1279.

Bottou, Léon, 2012, Stochastic gradient descent tricks, in *Neural networks: Tricks of the trade*, 421–436 (Springer).

Bryzgalova, Svetlana, 2015, Spurious factors in linear asset pricing models, *LSE manuscript* 1.

Campbell, John Y., and Samuel B. Thompson, 2007, Predicting Excess Stock Returns Out of Sample: Can Anything Beat the Historical Average?, *The Review of Financial Studies* 21, 1509–1531.

Chamberlain, Gary, and Michael Rothschild, 1983, Arbitrage, factor structure, and mean-variance analysis on large asset markets, *Econometrica* 51, 1281–1304.

Cochrane, John H., 2007, The Dog That Did Not Bark: A Defense of Return Predictability, *The Review of Financial Studies* 21, 1533–1575.

Connor, Gregory, and Robert A Korajczyk, 1986, Performance measurement with the arbitrage pricing theory: A new framework for analysis, *Journal of Financial Economics* 15, 373–394.

Connor, Gregory, and Robert A Korajczyk, 1988, Risk and return in an equilibrium apt: Application of a new test methodology, *Journal of Financial Economics* 21, 255–289.

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

Fama, Eugene F, and Kenneth R French, 2015, A five-factor asset pricing model, *Journal of Financial Economics* 116, 1–22.

Fama, Eugene F., and Kenneth R. French, 2021, *Common Risk Factors in the Returns on Stocks and Bonds*, 392–449 (University of Chicago Press).

Feng, Guanhao, Stefano Giglio, and Dacheng Xiu, 2020, Taming the factor zoo: A test of new factors, *Journal of Finance* 75, 1327–1370.

Freyberger, Joachim, Andreas Neuhierl, and Michael Weber, 2020, Dissecting characteristics nonparametrically, *Review of Financial Studies* 33, 2326–2377.

Giglio, Stefano, and Dacheng Xiu, 2021, Asset pricing with omitted factors, *Journal of Political Economy* 129, 1947–1990.

Giglio, Stefano, Dacheng Xiu, and Dake Zhang, 2021, Test assets and weak factors, *Chicago Booth Research Paper Forthcoming* .

Girosi, Federico, Michael Jones, and Tomaso Poggio, 1995, Regularization theory and neural networks architectures, *Neural computation* 7, 219–269.

Gospodinov, Nikolay, Raymond Kan, and Cesare Robotti, 2013, Chi-squared tests for evaluation and comparison of asset pricing models, *Journal of Econometrics* 173, 108–125.

Gu, Shihao, Bryan Kelly, and Dacheng Xiu, 2020, Empirical Asset Pricing via Machine Learning, *The Review of Financial Studies* 33, 2223–2273.

Gu, Shihao, Bryan Kelly, and Dacheng Xiu, 2021, Autoencoder asset pricing models, *Journal of Econometrics* 222, 429–450.

Harvey, Campbell R, and Yan Liu, 2021, Lucky factors, *Journal of Financial Economics* .

Huberman, Gur, Shmuel Kandel, and Robert F Stambaugh, 1987, Mimicking portfolios and exact arbitrage pricing, *Journal of Finance* 42, 1–9.

Hutcheson, Graeme D, 2011, Ordinary least-squares regression, *L. Moutinho and GD Hutcheson, The SAGE dictionary of quantitative management research* 224–228.

Hutchinson, James M, Andrew W Lo, and Tomaso Poggio, 1994, A nonparametric approach to pricing and hedging derivative securities via learning networks, *Journal of Finance* 49, 851–889.

Ingersoll Jr, Jonathan E, 1984, Some results in the theory of arbitrage pricing, *Journal of Finance* 39, 1021–1039.

Jagannathan, Ravi, and Zhenyu Wang, 1998, An asymptotic theory for estimating beta-pricing models using cross-sectional regression, *Journal of Finance* 53, 1285–1309.

Jegadeesh, Narasimhan, Joonki Noh, Kuntara Pukthuanthong, Richard Roll, and Junbo Wang, 2019, Empirical tests of asset pricing models with individual assets: Resolving the errors-in-variables bias in risk premium estimation, *Journal of Financial Economics* 133, 273–298.

Jegadeesh, Narasimhan, and Sheridan Titman, 1993, Returns to buying winners and selling losers: Implications for stock market efficiency, *Journal of finance* 48, 65–91.

Kan, Raymond, and Cesare Robotti, 2009, Model comparison using the hansen-jagannathan distance, *The Review of Financial Studies* 22, 3449–3490.

Kan, Raymond, Cesare Robotti, and Jay Shanken, 2013, Pricing model performance and the two-pass cross-sectional regression methodology, *Journal of Finance* 68, 2617–2649.

Kan, Raymond, and Chu Zhang, 1999, Two-pass tests of asset pricing models with useless factors, *Journal of Finance* 54, 203–235.

Kelly, Bryan, and Seth Pruitt, 2013, Market expectations in the cross-section of present values, *The Journal of Finance* 68, 1721–1756.

Kelly, Bryan T, Seth Pruitt, and Yinan Su, 2019, Characteristics are covariances: A unified model of risk and return, *Journal of Financial Economics* 134, 501–524.

Kleibergen, Frank, 2009, Tests of risk premia in linear factor models, *Journal of Econometrics* 149, 149–173.

Kozak, Serhiy, Stefan Nagel, and Shrihari Santosh, 2018, Interpreting factor models, *Journal of Finance* 73, 1183–1223.

Kozak, Serhiy, Stefan Nagel, and Shrihari Santosh, 2020, Shrinking the cross-section, *Journal of Financial Economics* 135, 271–292.

Lehmann, Bruce N, and David M Modest, 1988, The empirical foundations of the arbitrage pricing theory, *Journal of Financial Economics* 21, 213–254.

Mohri, Mehryar, Afshin Rostamizadeh, and Ameet Talwalkar, 2018, *Foundations of machine learning* (MIT press).

Moon, Hyungsik Roger, and Martin Weidner, 2015, Linear regression for panel with unknown number of factors as interactive fixed effects, *Econometrica* 83, 1543–1579.

Rapach, David E, Jack K Strauss, and Guofu Zhou, 2013, International stock return predictability: what is the role of the united states?, *Journal of Finance* 68, 1633–1662.

Rendle, Steffen, 2010, Factorization machines, in *2010 IEEE International conference on data mining*, 995–1000, IEEE.

Roll, Richard, and Stephen A Ross, 1980, An empirical investigation of the arbitrage pricing theory, *Journal of Finance* 35, 1073–1103.

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

Shanken, Jay, and Guofu Zhou, 2007, Estimating and testing beta pricing models: Alternative methods and their performance in simulations, *Journal of Financial Economics* 84, 40–86.

Welch, Ivo, and Amit Goyal, 2007, A Comprehensive Look at The Empirical Performance of Equity Premium Prediction, *The Review of Financial Studies* 21, 1455–1508.

Table 1: **Monthly out-of-sample stock-level prediction performance(percentage R^2_{oos})**

	OLS+H	OLS-3+H	PLS	PCR	ENet+H	GLM+H	RF	GBRT+H	NN1	NN2	NN3	NN4	NN5	FAPM	FAPM-7
All	-3.46	0.16	0.27	0.26	0.11	0.19	0.33	0.34	0.33	0.39	0.4	0.39	0.36	0.47	1.01
Top 1000	-11.28	0.31	-0.14	0.06	0.25	0.14	0.63	0.52	0.49	0.62	0.7	0.67	0.64	0.73	1.28
Bottom 1000	-1.3	0.17	0.42	0.34	0.2	0.3	0.35	0.32	0.38	0.46	0.45	0.47	0.42	0.42	0.81

Note: In this table, we report monthly R^2_{oos} for the entire panel of stocks using OLS with all factors (OLS), OLS using only size, book-to-market, and momentum (OLS-3), PLS, PCR, elastic net (ENet), generalize linear model (GLM), random forest (RF), gradient boosted regression trees (GBRT), neural networks with 1 to 5 layers (NN1–NN5), FAPM and FAPM-7. "+H" indicates the use of Huber loss instead of the l2 loss. We also report these R^2_{oos} within subsamples that include only the top-1,000 stocks or bottom-1,000 stocks by market value.

Table 2: **Importance of the top and bottom 7 factors**

m	1	2	3	4	5	6	7
Top-7	ntis(57.18)	svar(39.00)	dp(1.35)	bm(0.87)	dfy(0.46)	ep(0.48)	tms(0.40)
Bottom-7	ear(2.49e-5)	mom36m(2.57e-5)	herf(4.70e-5)	salerec(17.30e-5)	rd_sale(20.21e-5)	std_turn(20.86e-5)	opperprof(28.76e-5)
R^2_{oos}(percentage)	0.48	0.43	0.43	0.80	0.80	0.88	1.01

Note: The importance correspond to the top-7 and bottom-7 factors chosen for prediction. Factor importance each line is normalized to sum to one. The table shows the standard R^2 (all the factors) and the R^2_{is} of only m pairs factors using "one strong factor with one weak factor" style.

Table 3: R^2_{oos} **with and without factor SIC2**

	1	2	3	4	5	6	7
With sic2	0.51	0.56	0.52	0.72	0.64	0.78	0.87
Without sic2	0.48	0.42	0.44	0.80	0.80	0.88	1.01

Note: In this table, we report the R^2_{oos} of different pairs(1,2,$\cdots$,7) of factors with and without the discrete factor SIC2.

Table 4: **Factor importance for macroeconomic predictors**

	PLS	PCR	ENet+H	GLM+H	RF	GBRT+H	NN1	NN2	NN3	NN4	NN5	FAPM
dp	12.52	14.12	2.49	4.54	5.80	6.05	15.57	17.58	14.84	13.95	13.15	1.46
ep	12.25	13.52	3.27	7.37	6.27	2.85	8.86	8.09	7.34	6.54	6.47	1.81
bm	14.21	14.83	33.95	43.46	10.94	12.49	28.57	27.18	27.92	26.95	27.90	57.20
ntis	11.25	9.10	1.30	4.89	13.02	13.79	18.37	19.26	20.15	19.59	18.68	0.19
tbl	14.02	15.29	13.29	7.90	11.98	19.49	17.18	16.40	17.76	20.99	21.06	0.48
tms	11.35	10.66	0.31	5.87	16.81	15.27	10.79	10.59	10.91	10.38	10.33	2.15
dfy	17.17	15.68	42.13	24.10	24.37	22.93	0.09	0.06	0.06	0.04	0.12	34.39
svar	7.22	6.80	3.26	1.87	10.82	7.13	0.57	0.83	1.02	1.57	2.29	2.30

Note: This table presents the factor importance for eight macroeconomic factors in each model. Note that factor importance is an average overall training sample, and the factor importance within each model is normalized to sum to one.

Table 5: **Interaction between factors**

Macroeconomic factor	bm-ntis	ntis-dp	ntis-ep	ntis-dfy	bm-dp	bm-ep	ntis-tms	bm-dfy	ntis-tbl
Interaction	0.37	0.13	0.11	0.08	0.06	0.05	0.04	0.04	0.03
Individual factor	ntis-sgr	ntis-cash	ntis-stdcf	stdacc-stdcf	cinvest-roabol	divi-stdcf	chtx-rsup	securedind-roavol	divi-cinvest
Interaction	0.310	0.279	0.271	0.032	0.025	0.024	0.022	0.019	0.018

Note: In this table, we rank all the interaction of the total 95 individual and 8 macroeconomic factors and choose some on the top. Then we normalize all the interaction in order to easily report the comparison. The first row of the table reports the interactions between macroeconomic factors, and the second row reports the interactions between an individual factor and another factor.

Table 6: **Top-20 Average interactions of each factor**

factor	Av	—	factor	Av	—	factor	Av	—	factor	Av
nits	0.066	—	bm	0.032	—	dp	0.011	—	ep	0.009
dfy	0.008	—	securedind	0.004	—	tms	0.003	—	stdcf	0.003
cash	0.003	—	tbl	0.002	—	roeq	0.0026	—	roaq	0.0025
stdacc	0.002	—	roavol	0.0021	—	cinvest	0.0021	—	aeavol	0.0021
rsup	0.0021	—	nincr	0.0020	—	ear	0.0019	—	chtx	0.001829332
others	0.053									

Note: In this table, we deal with the interactions between all the 103 factor as well. Consider the absolute value of the coefficient as the interaction. Then we compute the average of all the interactions of each factor. Sum the average returns to one, and choose the top-20.

Table 7: **Monthly portfolio-level out-of-sample predictive R^2**

	OLS-3 +H	PLS	PCR	ENet +H	GLM +H	RF	GBRT +H	NN4	FAPM	FAPM-7
A. Commom factor portfolios										
S&P500	-0.02	-0.42	-0.83	0.05	0.37	1.03	1.25	0.17	0.48	0.53
SMB	0.57	1.87	0.34	1.33	2.08	0.49	0.44	0.53	1.36	1.33
HML	0.78	0.56	0.75	0.53	0.92	0.77	0.02	1.27	0.68	1.52
RMW	-0.29	0.89	-0.46	-1.03	0.12	-0.82	-1.21	0.64	0.44	1.38
CMA	0.27	-0.54	-0.15	-0.67	0.99	-0.06	-0.93	0.68	0.74	1.89
UMD	-0.63	-0.75	-0.35	0.37	-0.10	-0.25	-0.04	0.59	1.18	0.50
B. Subcomponents of factor portfolios										
Big value	0.31	0.06	-0.13	0.59	0.32	1.01	1.03	0.88	0.35	0.52
Big growth	0.38	-1.17	-1.47	0.20	0.19	1.01	0.89	0.89	1.07	0.88
Big neutral	0.37	-0.09	-1.16	0.38	0.73	1.23	0.78	0.97	0.82	0.64
Small value	-0.07	0.73	0.35	0.37	0.72	0.54	0.73	0.11	0.84	1.09
Small growth	0.10	0.29	-0.27	-0.13	-0.29	0.62	0.84	0.39	0.34	1.48
Small neutral	0.08	0.28	0.27	0.39	0.15	0.94	0.36	0.51	1.47	1.86
Big conservative	0.53	-0.28	-0.57	1.19	0.53	1.05	0.53	1.05	0.49	0.69
Big aggressive	0.33	-0.38	-1.09	0.35	0.57	1.38	1.20	0.85	0.87	0.95
Big neutral	0.47	-1.38	-1.36	0.73	0.48	1.03	0.76	0.99	0.27	0.29
Small conservative	0.31	1.17	0.66	-0.01	0.45	0.83	0.66	0.64	1.85	1.93
Small aggressive	0.01	0.48	0.05	-0.18	0.06	0.73	1.46	0.03	0.28	0.90
Small neutral	0.07	0.28	0.38	0.22	0.37	0.78	-0.05	0.42	1.17	1.70
Big robust	0.42	-0.78	-1.36	0.76	0.55	1.04	0.33	0.93	0.47	0.29
Big weak	0.31	0.98	0.46	0.76	0.89	1.23	0.96	0.80	1.00	1.08
Big neutral	-1.02	-0.86	0.61	0.53	0.70	0.99	0.91	0.99	0.31	0.54
Small robust	0.04	0.54	0.29	-0.48	-0.14	-0.52	0.39	0.26	1.54	1.66
Small weak	0.01	0.82	0.49	-0.43	0.54	1.25	1.12	0.36	1.55	1.77
Small neutral	0.06	0.18	-0.38	-0.29	-0.38	0.54	-0.19	0.25	0.36	1.07
Big up	0.32	-0.18	-1.02	0.46	0.87	0.92	0.79	0.76	0.18	0.37
Big down	0.27	-1.53	-1.75	0.54	-0.34	1.07	0.61	0.70	1.13	1.09
Big medium	0.49	-1.31	-1.84	0.68	-0.19	1.34	1.70	0.99	0.14	0.21
Small up	0.01	0.88	0.76	-0.17	0.15	0.52	-0.13	0.23	1.03	1.54
Small down	-0.01	0.27	-0.38	0.65	-0.19	1.37	1.34	0.33	1.16	1.32
Small medium	0.06	0.28	0.32	0.59	0.28	1.02	0.86	0.28	0.68	1.47

Note: In this table, we report the out-of-sample predictive R^2s for thirty portfolios using OLS with size, book-to-market, momentum, OLS-3, PLS, PCR, elastic net (ENet), generalized linear model with group lasso (GLM), random forest (RF), gradient boosted regression trees (GBRT), neural networks (NN4), FAPM, FAPM-7. "+H" indicates the use of Huber loss instead of the $\mathcal{L}_2$ loss. The six portfolios in panel A are the S&P 500 index and the Fama-French SMB, HML, CMA, RMW, and UMD factors. The twenty-four portfolios in panel B are 3×2 size double-sorted portfolios used in the construction of the Fama-French value, investment, profitability, and momentum factors.

Table 8: **Marketing timing Sharpe ratio gains**

	OLS-3 +H	PLS	PCR	ENet +H	GLM +H	RF	GBRT +H	NN4	FAPM	FAPM-7
A.Commom factor portfolios										
S&P500	-	-	-	0.00	0.02	0.13	0.19	0.00	0.03	0.03
SMB	0.03	0.36	0.01	0.18	0.44	0.02	0.02	0.03	0.19	0.18
HML	0.07	0.04	0.07	0.03	0.10	0.07	0.00	0.19	0.05	0.27
RMW	-	0.09	-	-	0.00	-	-	0.04	0.02	0.21
CMA	0.01	-	-	-	0.13	-	-	0.06	0.07	0.45
UMD	-	-	-	-	-	-	-	0.05	0.18	0.03
B.Subcomponents of factor portfolios										
Big value	0.01	0.00	-	0.04	0.01	0.12	0.12	0.09	0.01	0.03
Big growth	0.02	-	-	0.00	0.00	0.11	0.08	0.08	0.12	0.08
Big neutral	0.02	-	-	0.02	0.06	0.17	0.07	0.11	0.08	0.05
Small value	-	0.11	0.02	0.03	0.10	0.06	0.11	0.00	0.14	0.23
Small growth	0.00	0.02	-	0.00	0.02	0.11	0.19	0.04	0.03	0.55
Small neutral	0.00	0.01	0.01	0.02	0.00	0.09	0.01	0.03	0.23	0.37
Big conservative	0.03	-	-	0.17	0.03	0.13	0.03	0.13	0.03	0.06
Big aggressive	0.01	-	-	0.01	0.03	0.20	0.15	0.08	0.08	0.10
Big neutral	0.03	-	-	0.06	0.03	0.13	0.07	0.12	0.01	0.01
Small conservative	0.01	0.15	0.05	-	0.02	0.07	0.05	0.04	0.37	0.40
Small aggressive	0.00	0.07	0.00	-	0.00	0.16	0.57	0.00	0.02	0.23
Small neutral	0.00	0.01	0.01	0.00	0.01	0.06	0.00	0.02	0.14	0.30
Big robust	0.02	-	-	0.07	0.04	0.13	0.01	0.10	0.03	0.01
Big weak	0.01	0.10	0.02	0.06	0.08	0.15	0.09	0.06	0.10	0.12
Big neutral	-	-	0.04	0.03	0.06	0.11	0.09	0.11	0.01	0.03
Small robust	0.00	0.03	0.01	0.02	-	-	0.02	0.01	0.24	0.28
Small weak	0.00	0.13	0.05	0.04	0.06	0.31	0.25	0.03	0.46	0.59
Small neutral	0.00	0.00	0.02	0.01	-	0.04	0.00	0.01	0.02	0.14
Big up	0.01	0.00	-	0.02	0.09	0.10	0.07	0.07	0.00	0.02
Big down	0.01	-	-	0.03	-	0.12	0.04	0.05	0.13	0.12
Big medium	0.03	-	-	0.06	-	0.22	0.35	0.12	0.00	0.01
Small up	0.00	0.10	0.07	-	0.00	0.03	-	0.01	0.13	0.30
Small down	-	0.01	-	0.05	-	0.22	0.21	0.01	0.16	0.20
Small medium	0.00	0.03	0.03	0.11	0.03	0.32	0.23	0.03	0.15	0.61

Note: This table documents improvement in annualized Sharpe ratio $SR^* - SR$. We compute the SR^* by weighting the portfolios based on a market timing strategy (see Campbell and Thompson (2007)).

Table 9: **Performance of the machine learning portfolios**

	OLS-3+H				NN4			
	Pred	Avg	SD	SR	Pred	Avg	SD	SR
Low(L)	0	0.4	5.9	0.24	-0.48	0.02	5.79	0.01
2	0.17	0.58	4.65	0.43	-0.08	0.13	5.32	0.08
3	0.35	0.6	4.43	0.47	0.24	0.49	4.85	0.35
4	0.49	0.71	4.32	0.57	0.32	0.51	4.57	0.38
5	0.62	0.79	4.57	0.6	0.57	0.74	4.44	0.58
6	0.75	0.92	5.03	0.63	0.79	0.77	4.53	0.59
7	0.88	0.85	5.18	0.57	0.98	0.79	4.62	0.59
8	1.02	0.86	5.29	0.56	1.13	0.86	4.69	0.63
9	1.21	1.18	5.47	0.75	1.37	1.04	4.58	0.78
High(H)	1.51	1.34	5.88	0.79	1.69	1.16	5.65	0.71
H-L	1.67	0.94	5.33	0.61	2.17	1.14	4.91	0.80
	FAPM-7				FAPM			
	Pred	Avg	SD	SR	Pred	Avg	SD	SR
Low(L)	-0.06	-0.08	3.15	-0.13	-0.18	-0.63	7.17	-0.09
2	0.31	0.15	3.08	0.01	0.59	0.13	6.16	0.33
3	0.49	0.46	3.05	0.07	0.86	0.22	5.14	0.57
4	0.64	0.55	3.02	0.12	1.37	0.24	4.13	1.14
5	0.78	0.79	2.98	0.17	1.46	0.35	4.12	1.22
6	0.96	0.83	2.96	0.23	1.66	0.48	4.11	139
7	1.19	1.12	2.91	0.32	1.78	0.59	4.11	0.99
8	1.47	1.38	2.84	0.43	1.82	0.83	5.11	1.23
9	1.72	1.84	2.81	0.52	1.87	1.12	6.11	1.05
High(H)	2.02	2.01	2.76	0.64	1.45	7.24	6.09	0.70
H-L	2.08	2.09	2.45	1.42	2.09	1.63	4.28	1.15

Note: In this table, we report the performance of prediction-sorted portfolios over the 30-year out-of-sample testing period. All stocks are sorted into deciles based on their predicted returns for the next month. Columns "Pred", "Avg", "SD", and "SR" provide the predicted monthly returns for each decile, the average realized monthly returns, their standard deviations, and Sharpe ratios, respectively. All portfolios are value weighted.

Figure 2: A simple example
This figure shows a simple example of asset pricing problem. The first part is a batch of stock factors. And the second part shows how to generate the second-order relationships.

Figure 3: Algorithm Framework

This figure presents the general framework of FAPM algorithm operation. The model and data set are initialized first. When the model and data set are in place, the training begins. We trained the epoch wheel each time, and from the results of these multiple pieces of training, selected the model that performed best on the validation set. Perform test set predictions on selected models, save and evaluate the results. By rolling adjustment of the training set, verification set, and test set according to year, the final test set prediction result of the year was obtained.

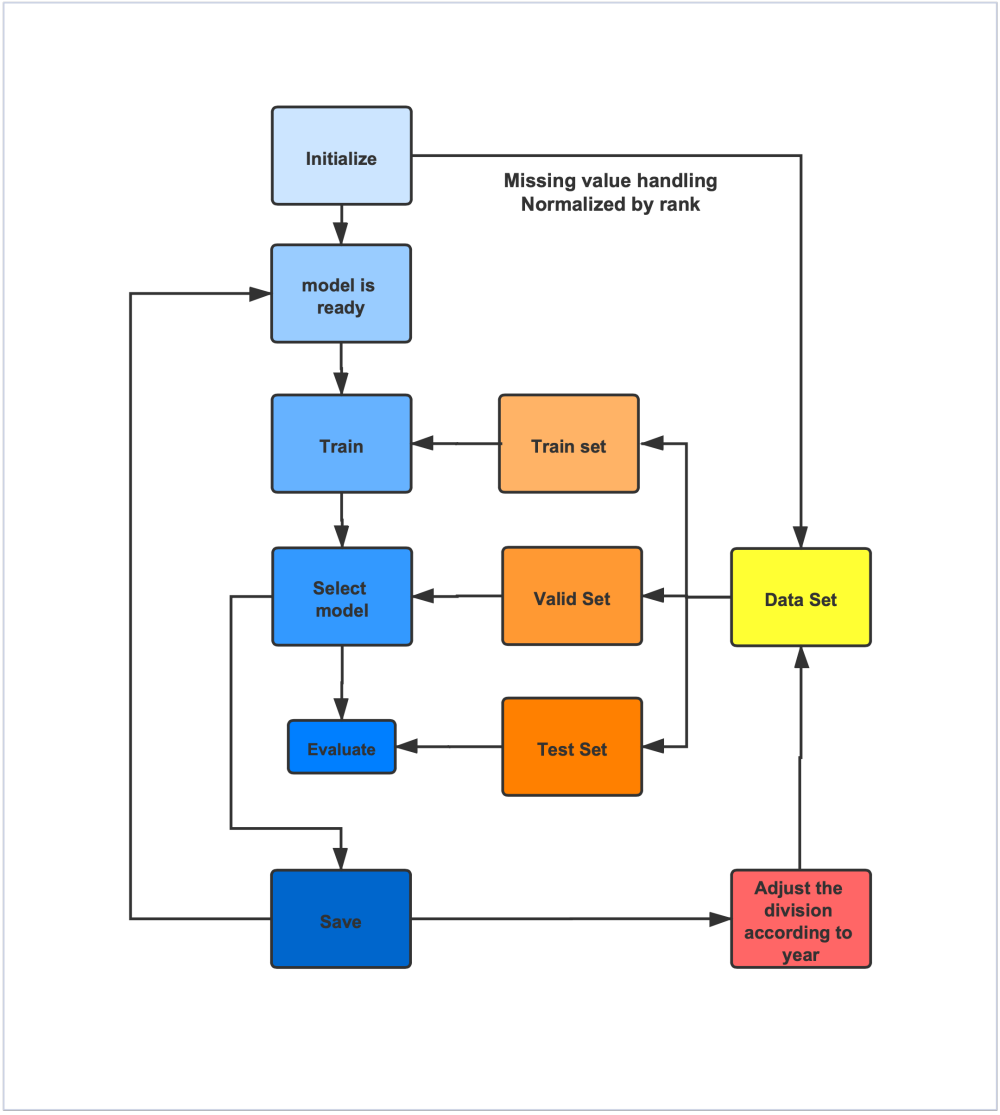


Figure 4: Monthly out-of-sample stock-level prediction performance(percentage R^2_{oos})
This figure report monthly R^2_{oos} for the entire panel of stocks using OLS with all factors (OLS), OLS using only size, book-to-market, and momentum (OLS-3), PLS, PCR, elastic net (ENet), generalize linear model (GLM), random forest (RF), gradient boosted regression trees (GBRT), neural networks with 1 to 5 layers (NN1–NN5), FAPM, and FAPM-7. "+H" indicates the use of Huber loss instead of the $\mathcal{L}_2$ loss. We also report the R^2_{oos} within subsamples that include only the top-1,000 stocks(black) or bottom-1,000 stocks(blue) by market value.

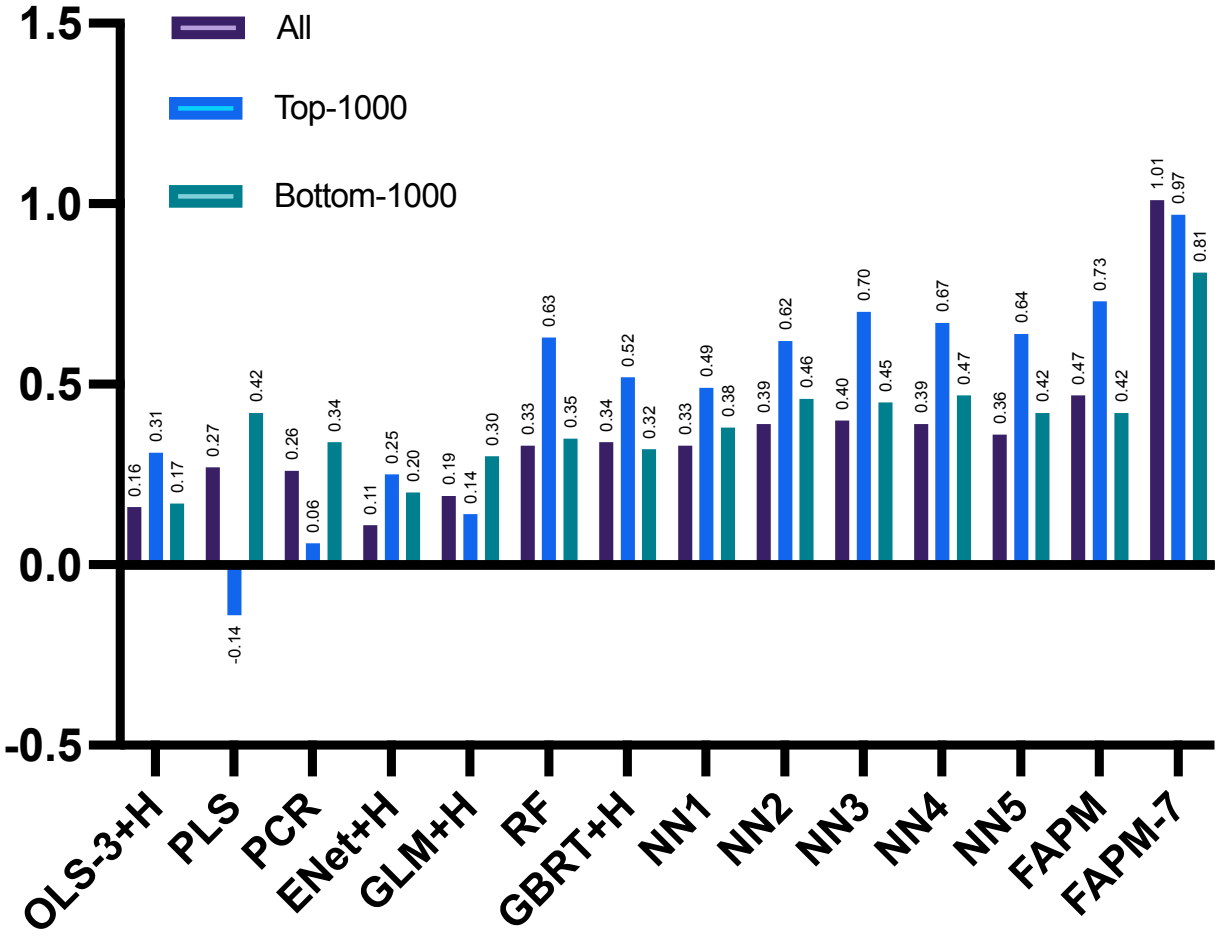


Figure 5: R^2_{is} of m Pairs factors and all features

This figure reports the in-sample R^2 of different pairs of factors. The blue line reports the total R^2_{is} of m-pairs factors where m is x-axis. The blue line reports the R^2 of the FAPM with all factors. The vertical axis is R^2 of monthly returns.

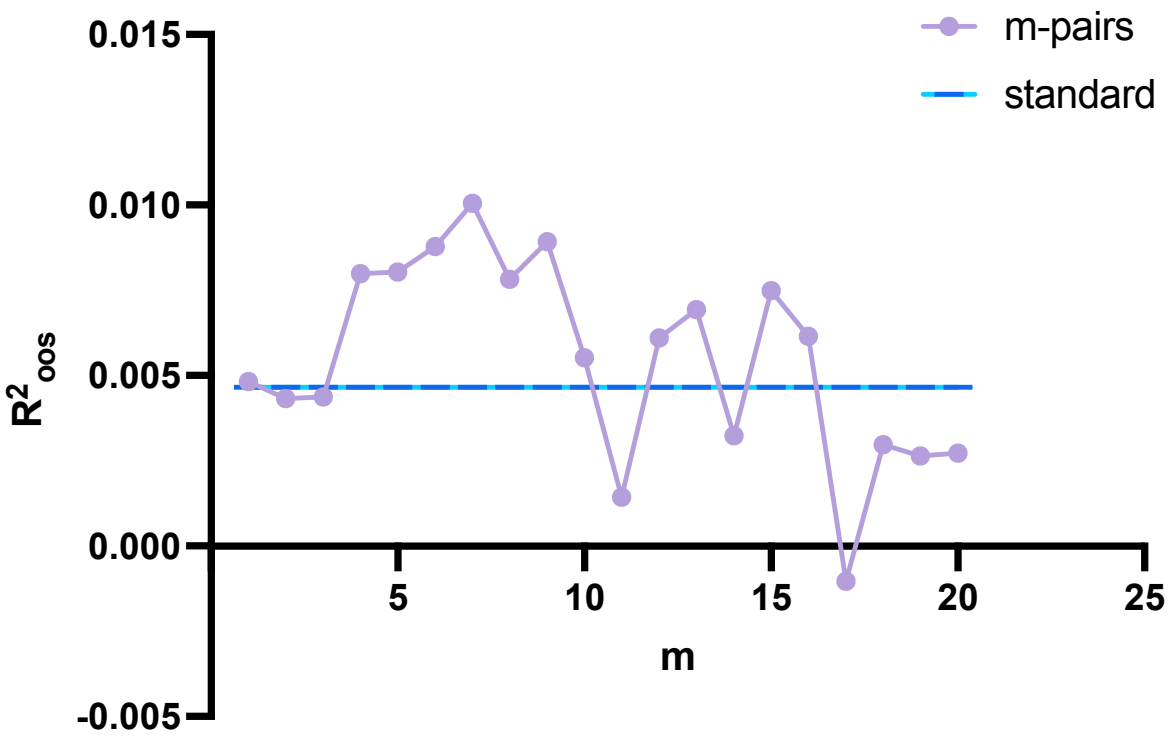


Figure 6: R^2 with and without SIC2

This figure reports the R^2_{oos} of the FAPM model with m-pair factors, with and without SIC2 factor.

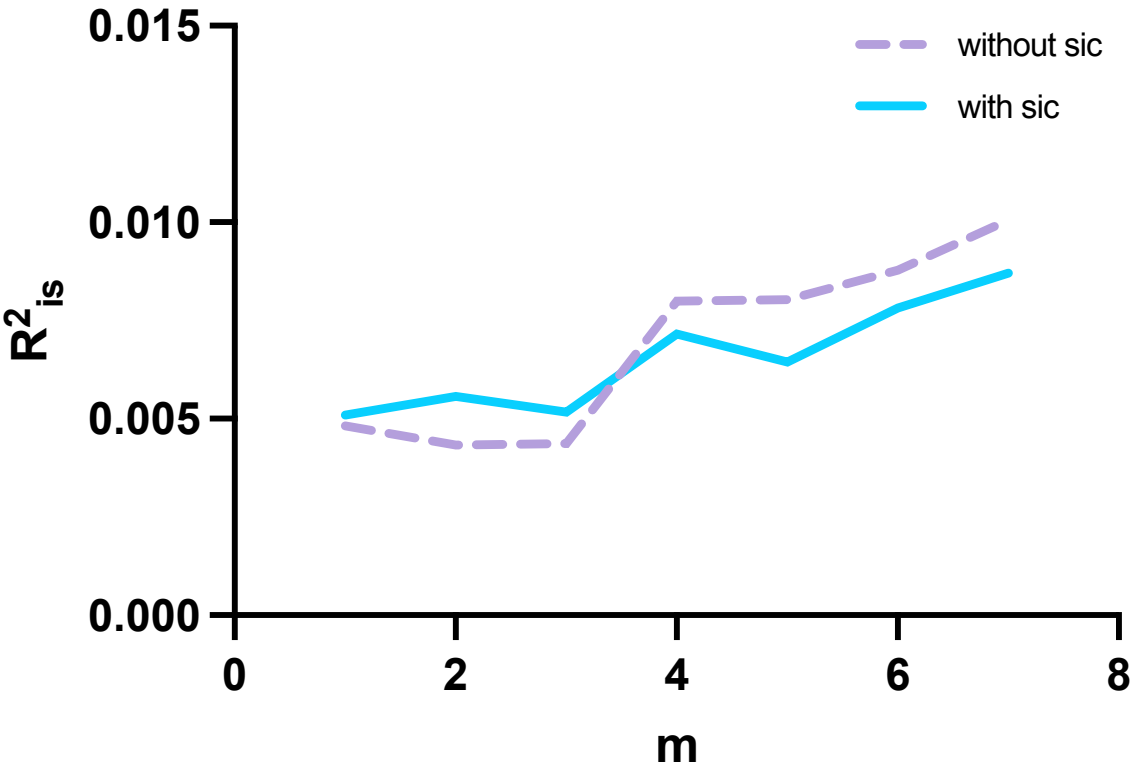


Figure 7: R^2_{is} with SIC2 by year

We report different models' in-sample R^2 of each training year on the train set. Those models are different m-pair FAPM models with SIC2 factor.

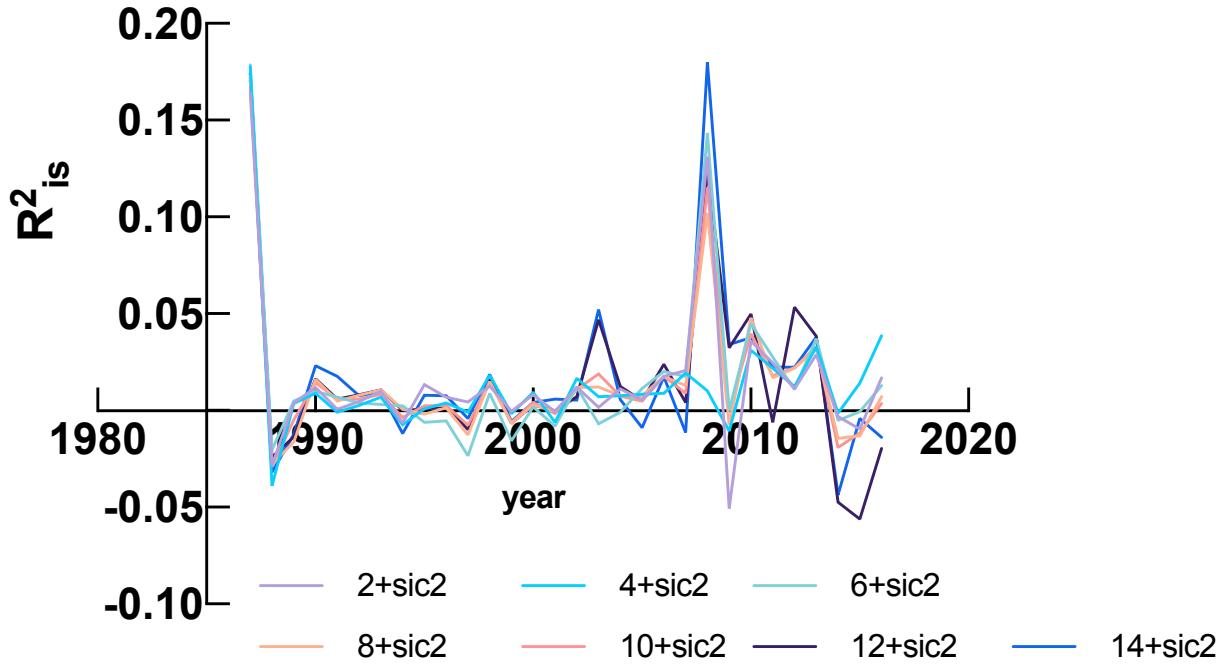


Figure 8: R^2_{is} without sic by year

We report different models' in-sample R^2 of each training year on the train set. Those models are different m-pair FAPM models without SIC2 factor.

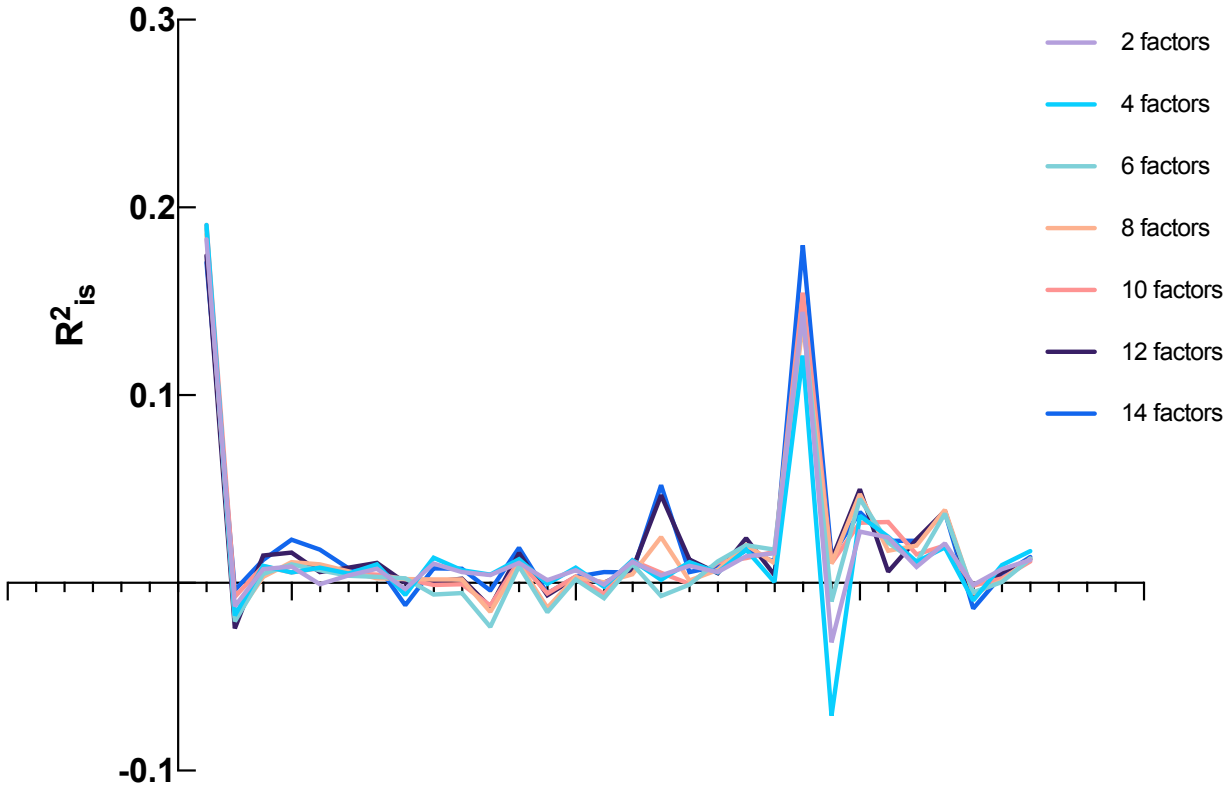


Figure 9: R^2_{is} of m-pairs factors on the basis of all-factors

We report the difference between the predicted R^2 of different models and the full-factor predicted R^2 by year.

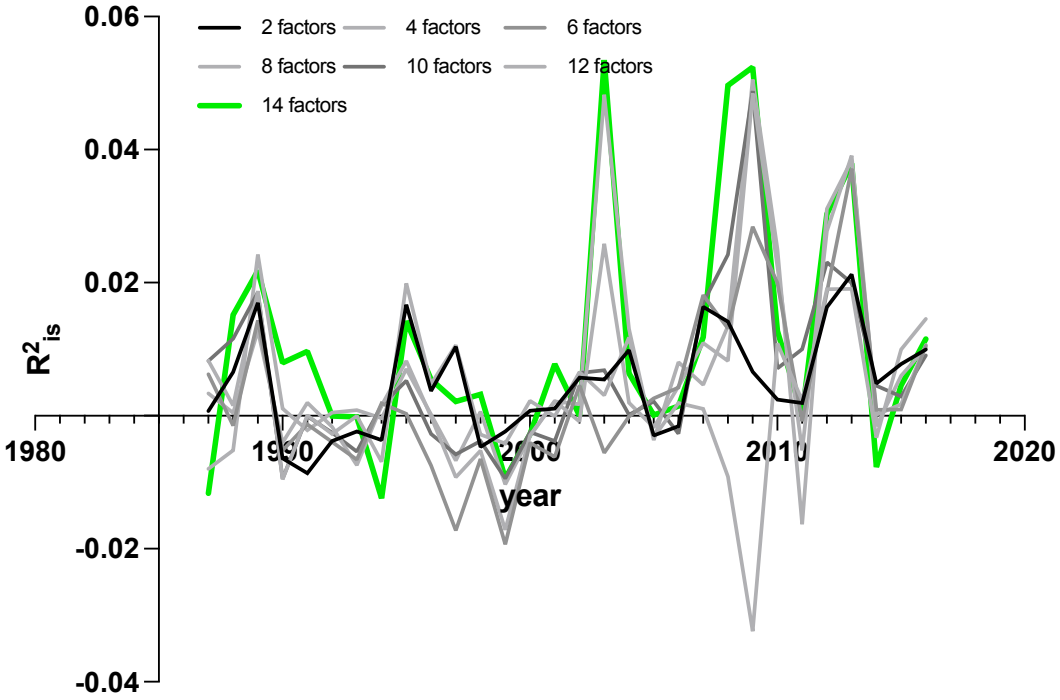


Figure 10: Factor importance

Factor importance for the top-12 most influential factors in each model. Note that factor importance is an average overall training sample, and the factor importance within the model is normalized to sum to one.

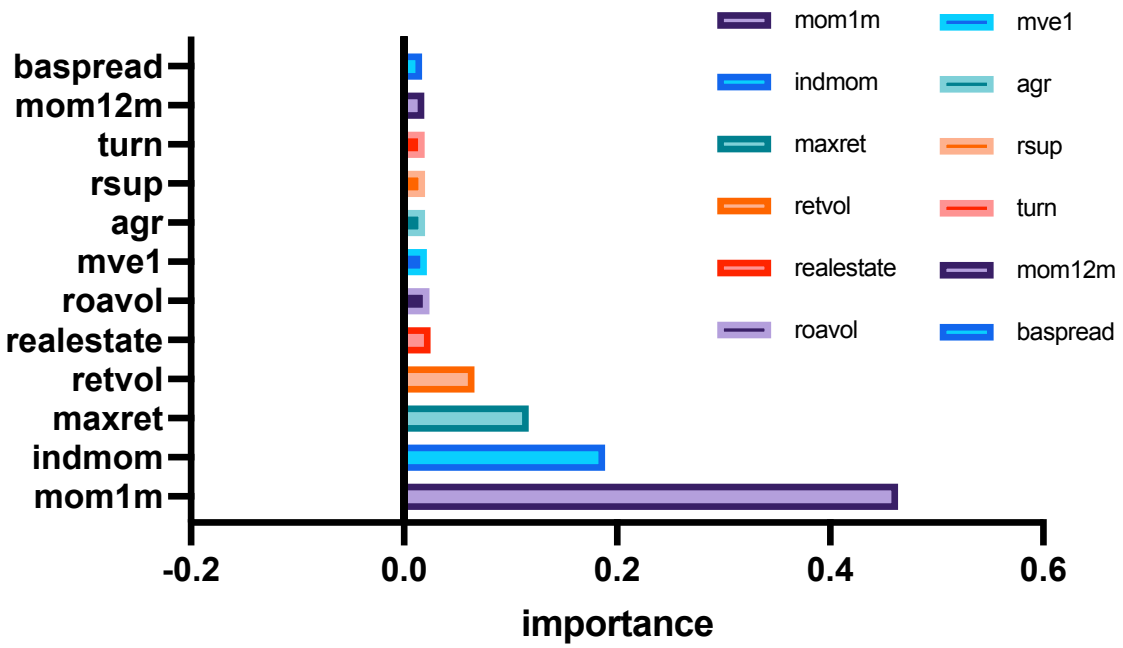


Figure 11: Macroeconomic factor importance

Factor importance for eight macroeconomic factors in each model. Note that the factor importance is an average over all training samples, and the factor importance within each model is normalized to sum to one.

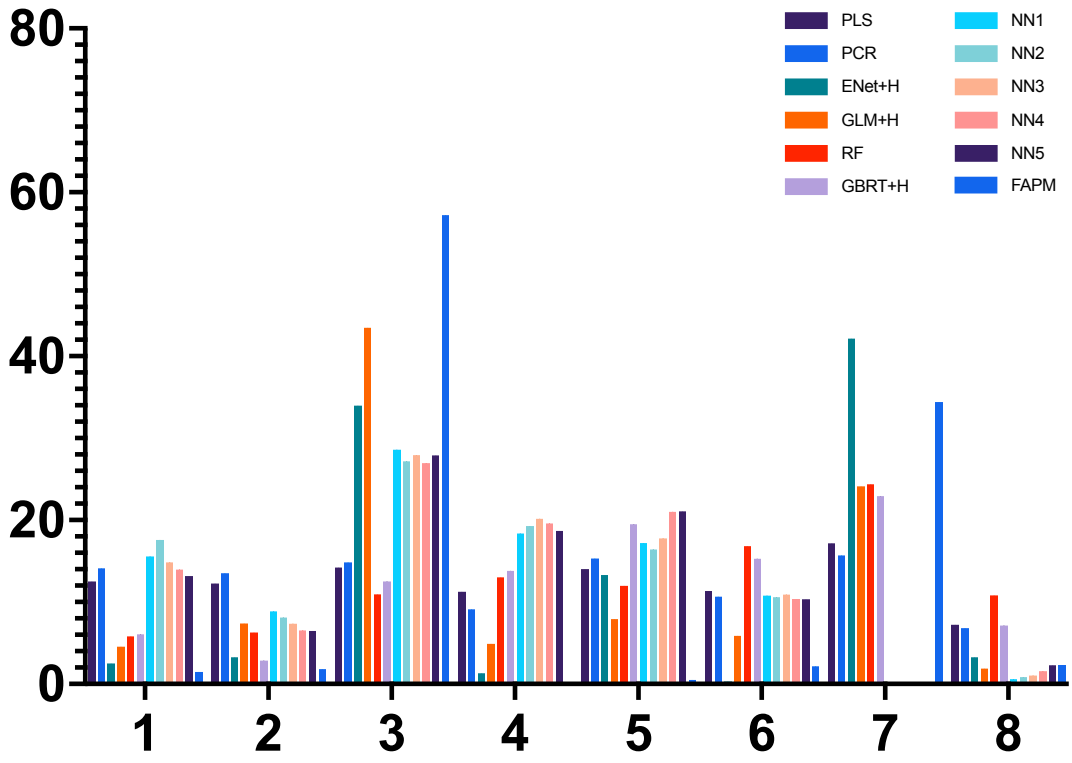


Figure 12: Interaction between Macroeconomic factors

In this figure, we report the interaction between macroeconomic factors and between macro and individual factors. We rank all those interactions and keep the top-12 factor interactions. All the interactions are normalized to 1. Eventually, we report the interaction of $bm - ntis$, $ntis - dp$, $ntis - ep$, $ntis - dfy$, $bm - dp$, $bm - ep$, $ntis - tms$, $bm - dfy$, $ntis - sgr$, $ntis - tbl$, $ntis - cash$, and $ntis - stdcf$.

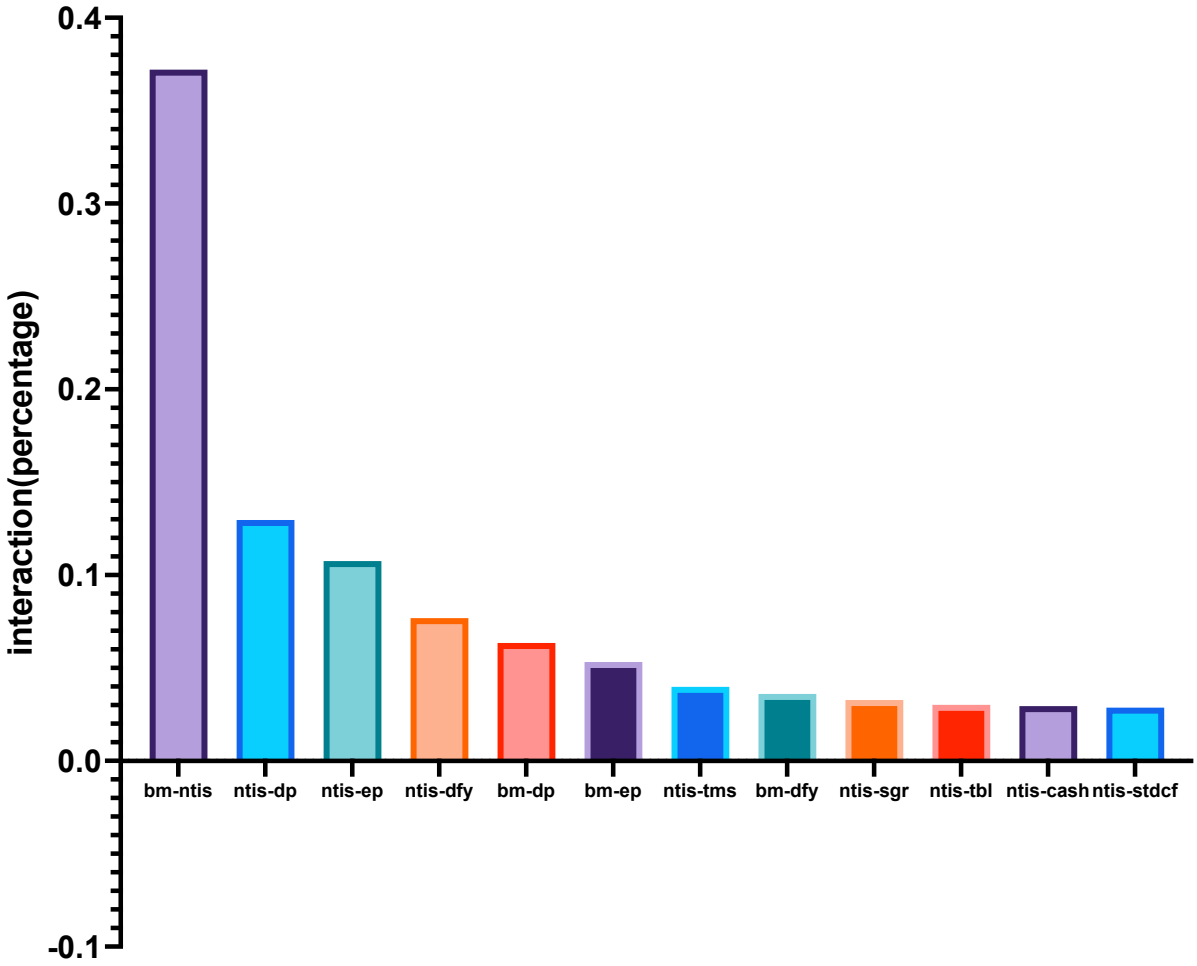


Figure 13: Interaction between Individual factors

In this figure, we report the interactions between individual factors. In fact, we only choose the top-6 factor interactions and normalize them to 1 to report.

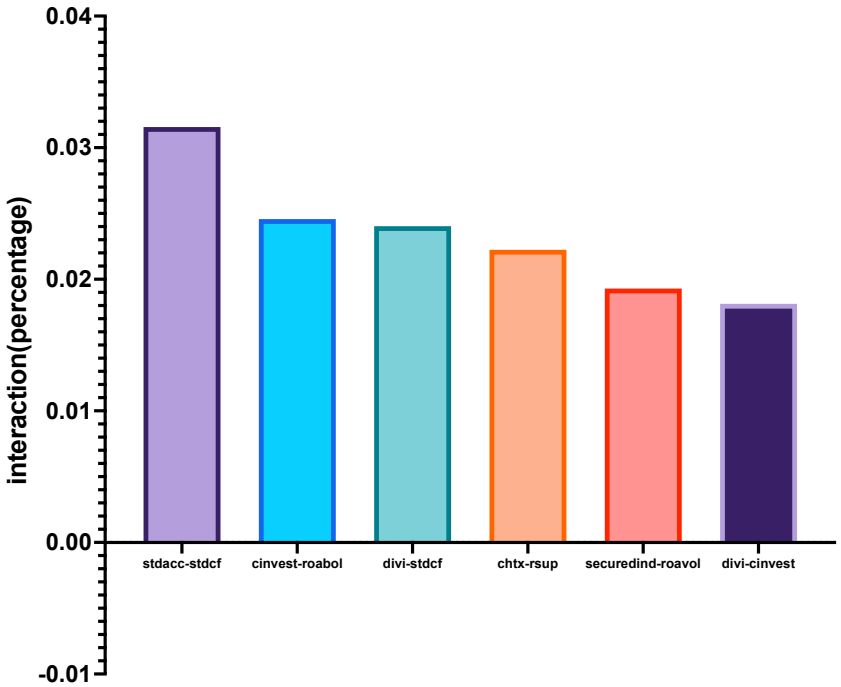


Figure 14: Average interaction of each factor

This figure reports the average ineraction of each factor using pie. The 20 most significant factors are *ntis*, *bm*, *dp*, *ep*, *dfy*, *securedind*, *tms*, *stdcf*, *cash*, *tbl*, *roeq*, *roaq*, *stdacc*, *roavol*, *cinvest*, *aeavol*, *rsup*, *nincr*, *ear*, and *chtx*. Their corresponding colors are shown in the legend. Gray represents other factors.

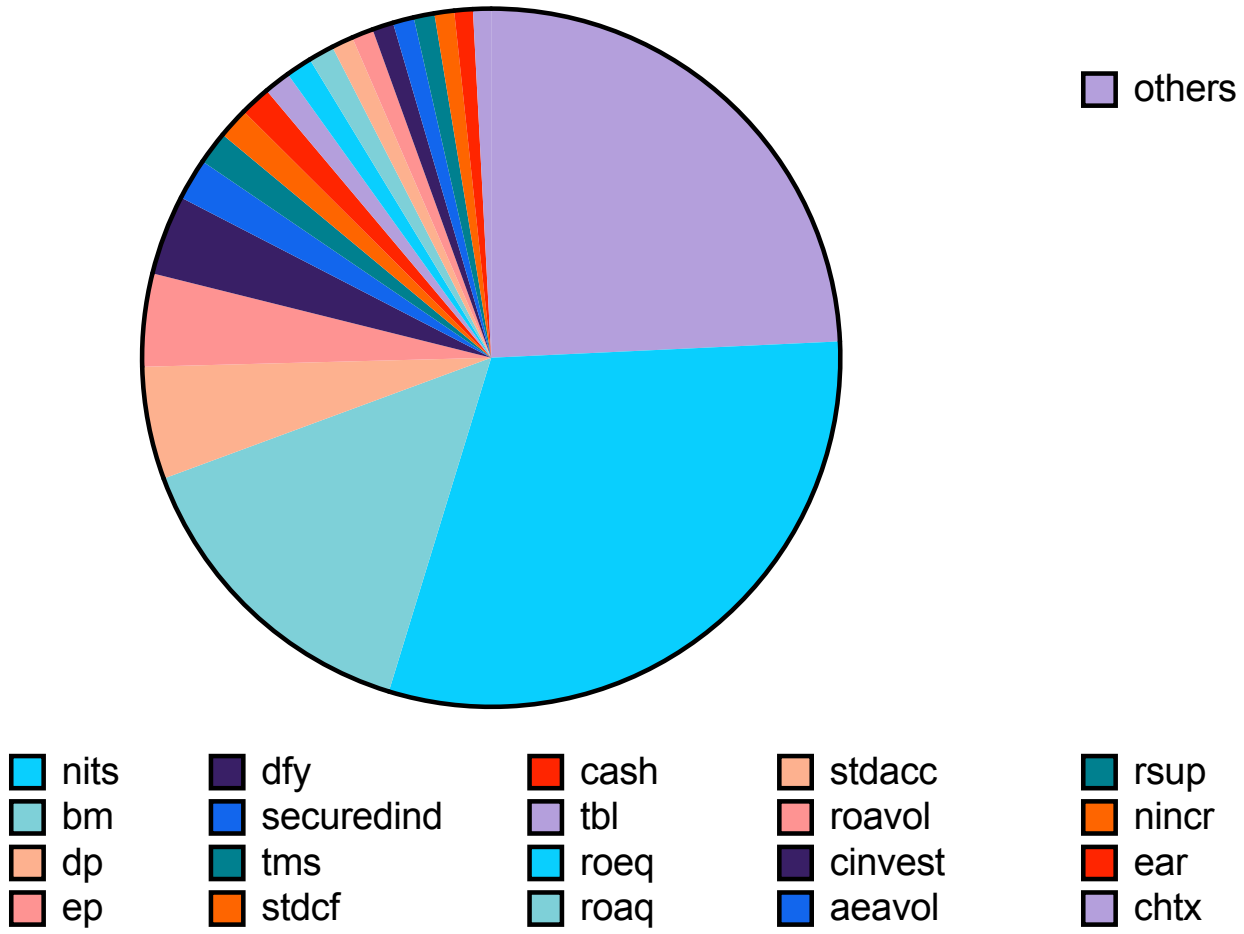


Figure 15: Portfolio calculation process

This figure shows how to generate portfolios from a given dataset.

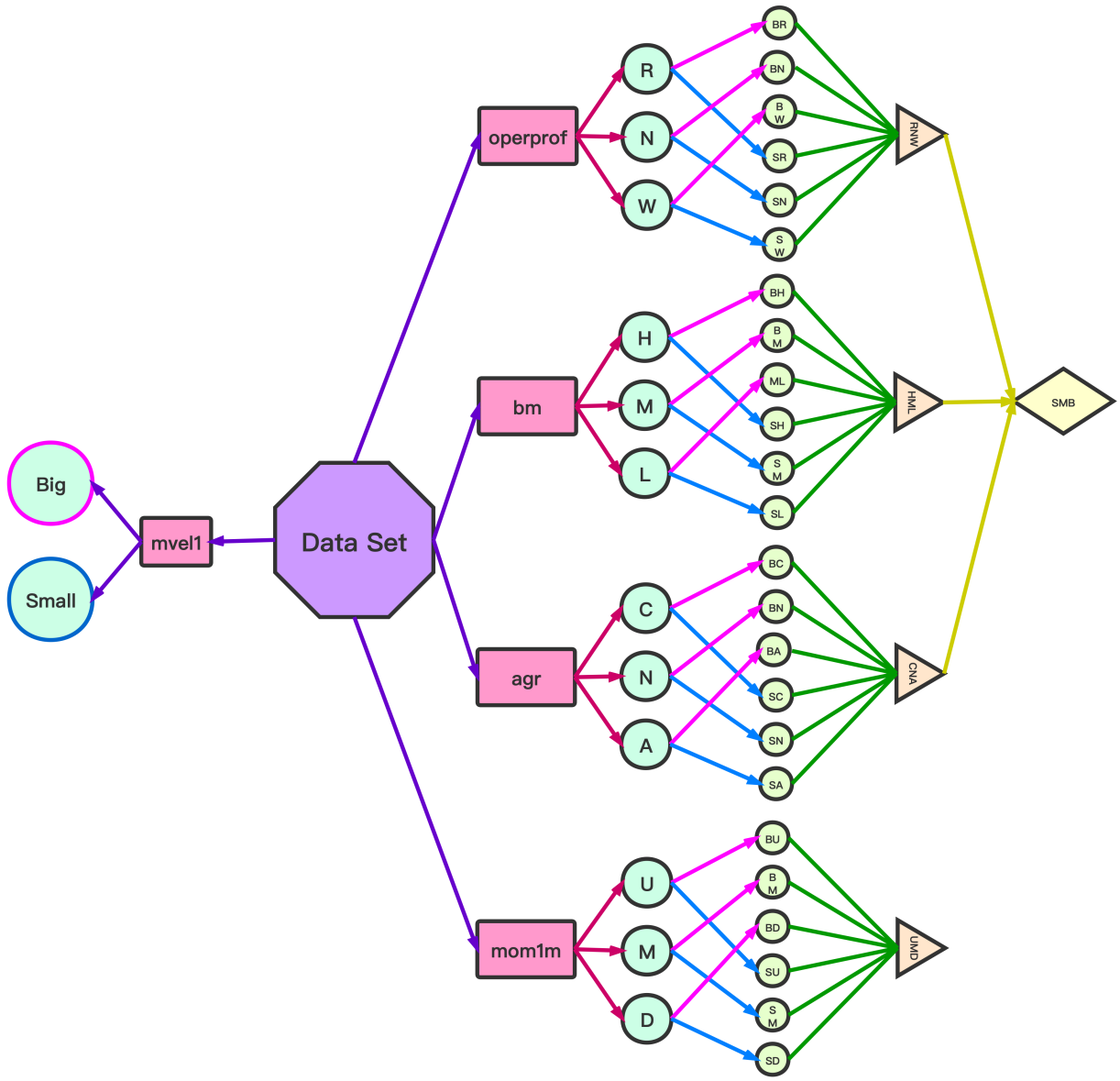


Figure 16: Monthly portfolio-level out-of-sample predictive R^2

In this figure, we report the out-of-sample predictive R^2 for thirty portfolios using OLS with size, book-to-market, momentum, OLS-3, PLS, PCR, elastic net (ENet), generalized linear model with group lasso (GLM), random forest (RF), gradient boosted regression trees (GBRT), neural networks (NN4), FAPM, and FAPM-7. "+H" indicates the use of Huber loss instead of the $\mathcal{L}_2$ loss.

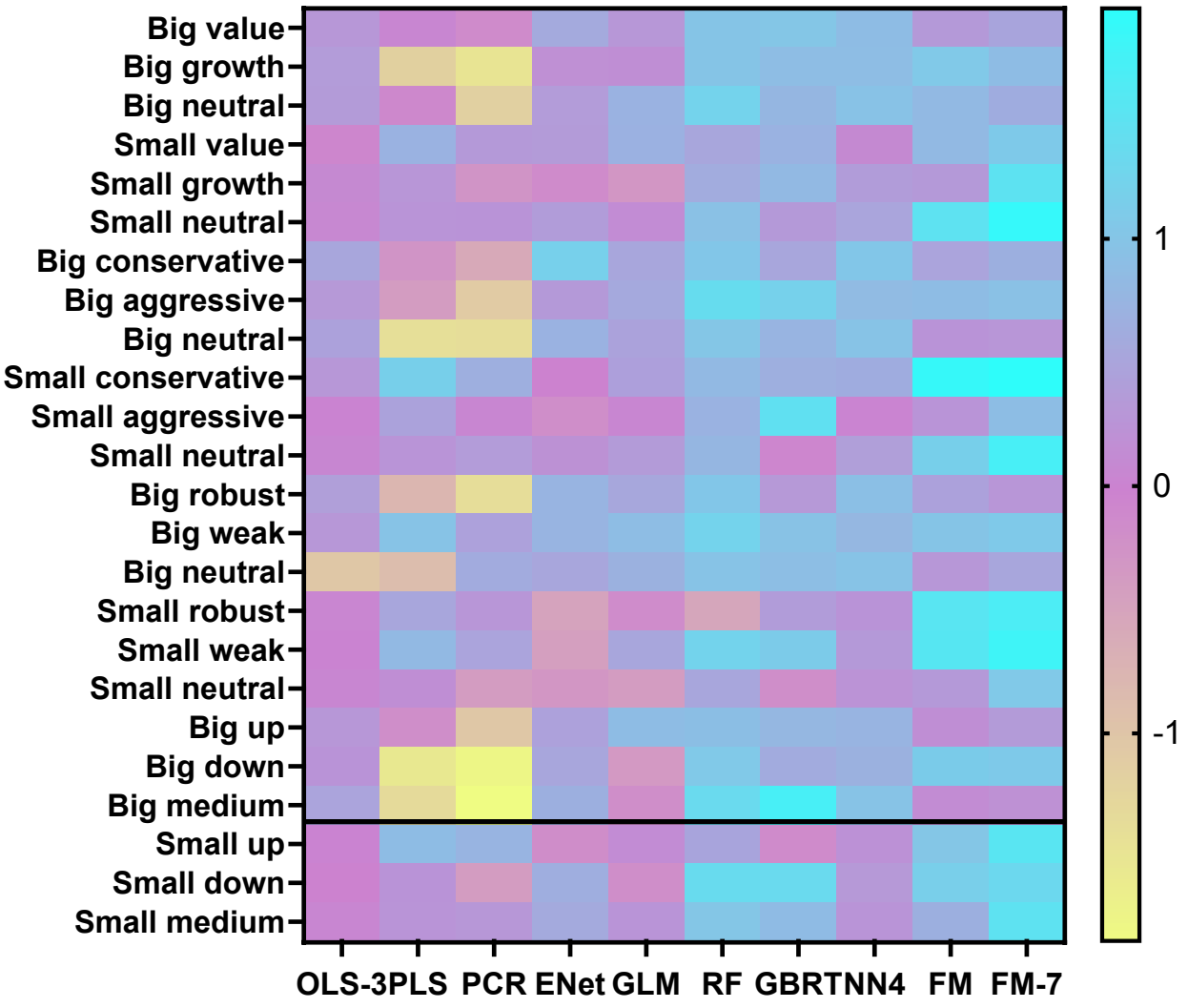
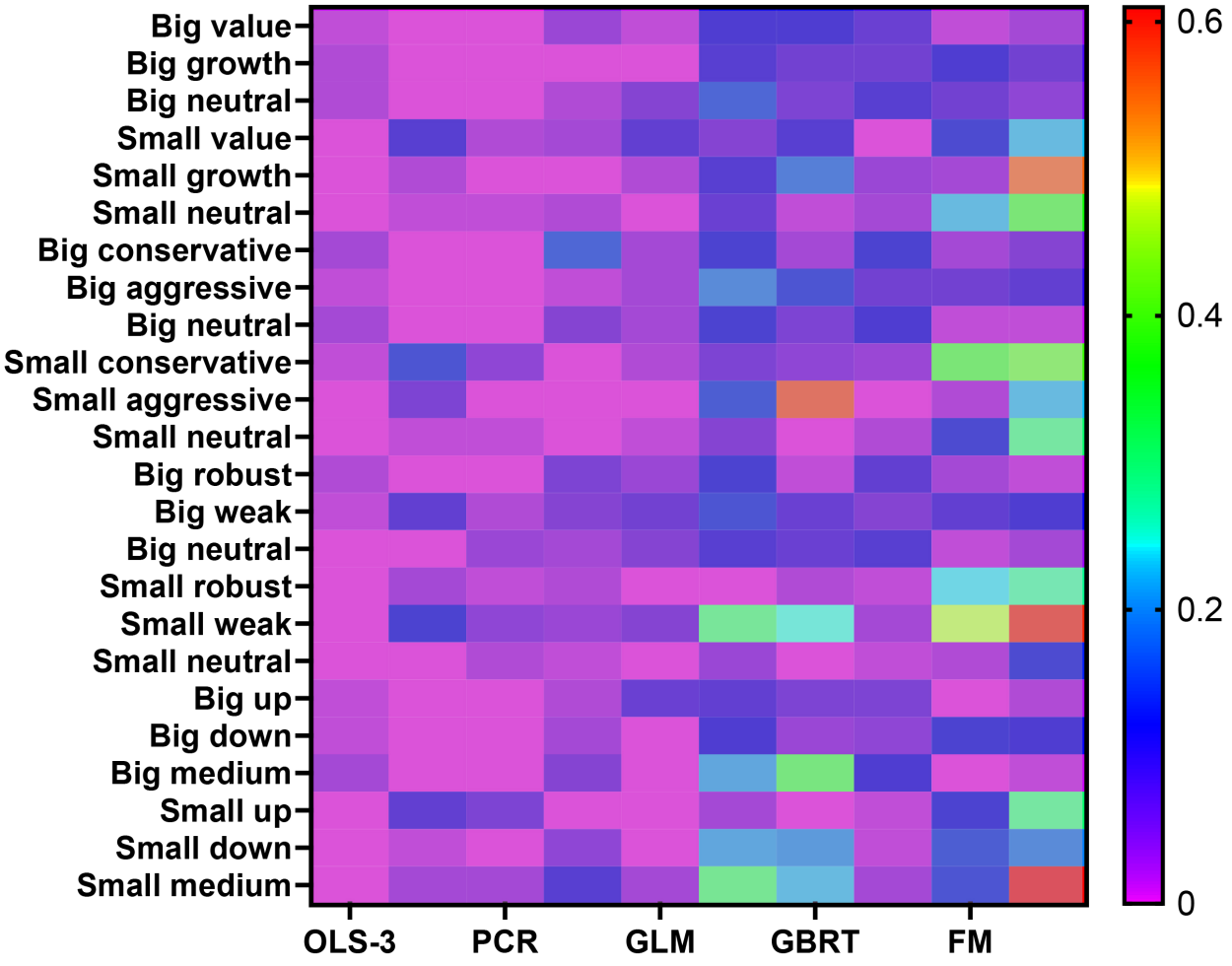


Figure 17: Marketing timing Sharpe ratio gains

This figure illustrates the improvement in annualized Sharpe ratio $SR^* - SR$. We compute the SR^* by weighting the portfolios based on a market timing strategy (see Campbell and Thompson (2007)).



Algorithm 1 Predict risk premiums via FAPM.

Input: $\boldsymbol{X} = (\boldsymbol{x}^{[1]}, \boldsymbol{x}^{[2]}, \ldots, \boldsymbol{x}^{[M]})^T \in \mathbb{R}^{M \times N}$: Asset data set.
Output: $\hat{\boldsymbol{r}} = (\hat{r}^{[1]}, \hat{r}^{[M]}, \ldots, \hat{r}^{[M]})^T \in \mathbb{R}^M$: The predicted risk premiums.

function PREDICT(X)
  for $i = 0 \to M$ **do**
    Get $\hat{y}(\boldsymbol{x}^{[i]})$ by Equation FAPM:$O(nk)$
    $\hat{r}^{[i]} \leftarrow \hat{y}(\boldsymbol{x}^{[i]})$
  end for
  $\hat{\boldsymbol{r}} \leftarrow (\hat{r}^{[1]}, \hat{r}^{[M]}, \ldots, \hat{r}^{[M]})^T$
  return $\hat{\boldsymbol{r}}$
end function

Algorithm 2 FAPM Train and Test: this algorithm shows how to train and test FAPM in experiments

Input: $\boldsymbol{X}, \boldsymbol{Y}$: Asset data set and the corresponding ground truth (the real asset price) from 1957 to 2016. $I > 0$: The number of iterations in training.
Output: R^2_{oos}: The evaluation metric.

function TRAIN AND TEST($\boldsymbol{X}, \boldsymbol{Y}$)
  for $i = 1 \to 30$ **do**
    $\boldsymbol{X}_{\text{train}}, \boldsymbol{Y}_{\text{train}} \leftarrow \boldsymbol{X}_{[1957,1974+i-1]}, \boldsymbol{Y}_{[1957,1974+i-1]}$
    $\boldsymbol{X}_{\text{valid}}, \boldsymbol{Y}_{\text{valid}} \leftarrow \boldsymbol{X}_{[1975+i-1,1986+i-1]}, \boldsymbol{Y}_{[1975+i-1,1986+i-1]}$
    $\boldsymbol{X}_{\text{test}}, \boldsymbol{Y}_{\text{test}} \leftarrow \boldsymbol{X}_{1987+i-1}, \boldsymbol{Y}_{1987+i-1}$
    $R^2_{\text{max}}, \text{index} \leftarrow 0, 0$
    for $j = 1 \to I$ **do**
      optimize the model parameters via gradient descent on R^2_{is} using $\boldsymbol{X}_{\text{train}}, \boldsymbol{Y}_{\text{train}}$
      compute R^2_{valid} using $\boldsymbol{X}_{\text{valid}}, \boldsymbol{Y}_{\text{valid}}$
      if $R^2_{\text{valid}} > R^2_{\text{max}}$ **then**
        $R^2_{\text{max}}, \text{index} \leftarrow R^2_{\text{valid}}, j$
      end if
    end for
    select the model in the index-th iteration.
    compute $R^2_{oos}[i]$ using $\boldsymbol{X}_{\text{test}}, \boldsymbol{Y}_{\text{test}}$
  end for
  $R^2_{oos} \leftarrow$ average value of $R^2_{oos}[1], R^2_{oos}[2], \ldots, R^2_{oos}[30]$
  return R^2_{oos}
end function

Algorithm 3 Portfolio: this algorithm shows how to calculate the portfolios and their subcomponent.

Input: Some certain columns of dataset, $bm, operprof, agr, mom1m, mvel1$, and we merge them to a subset $subdata$. The actual vector of returns RET, and the predicted returns Y

Output: The R^2 of the actual portfolio returns and the predicted portfolio returns.

function PORTFOLIO($subdata, RET, Y$)
  for $year = 1987 \rightarrow 2017$ **do**
    Get the $subdata$ of $year$
    $label$ ←Rank the columns $bm, operprof, agr, mom1m$. Then for each column, label $subdata$ into three parts.
    $subcomponent$ ←According to label and get 30 portfolios
    $sub_1 \leftarrow Cal\_portfolio\_return(mve1, RET)$
    $sub_2 \leftarrow Cal\_portfolio\_return(mve1, Y)$
    $common_1, common_2$ ←Calculate the common portfolios' returns using sub_1, sub_2
  end for
  R^2 ←Calculated by the sequence of $sub_1, sub_2, common_1, common_2$
end function