Asset Pricing with Neural Networks: Significance Tests*

Hasan Fallahgoul† Monash University

Vincentius Franstianto‡ Monash University

Xin Lin§ Monash University

This draft: August 29, 2023

First draft: March 12, 2020

Abstract

This study proposes a novel hypothesis test for evaluating the statistical significance of input variables in multi-layer perceptron (MLP) regression models. Theoretical foundations are established through consistency results and estimation rate analysis using the sieves method. To validate the test's performance in complex and realistic settings, an extensive Monte Carlo simulation is conducted. Results of the simulation reveal that the test has a high power and low rate of false positives, making it a powerful tool for detecting true effects in data. The test is further applied to identify the most influential predictors of equity risk premiums, with results indicating that only a small number of characteristics have statistical significance and all macroeconomic predictors are insignificant at the 1% level. These findings are consistent across a variety of neural network architectures.

Keywords: Asset Pricing; Risk Premium; Neural Networks; Variable Significance Test

JEL classification: C1; C5.

*An earlier version of this paper has been circulated under the title "Towards Explaining Deep Learning: A Variable Significance Test for Multi-Layer Perceptrons". We very grateful to the Editor Torben G. Andersen, the Associate Editor, and the two Referees for their invaluable comments. Their feedback has played a pivotal role in effecting significant enhancements to both the content and presentation of our work. We also thank Daniele Bianchi, Daniel Buncic, Ivan Guo, Loriano Mancini, Juan-Pablo Ortega, Farshid Vahid, and seminar participants at UNSW, Monash University, 4th conference on "Non-traditional Data, Machine Learning, and Natural Language Processing in Macroeconomics. Monash Centre for Quantitative Finance and Investment Strategies has been supported by BNP Paribas.

†Hasan Fallahgoul, Monash University, School of Mathematics and Centre for Quantitative Finance and Investment Strategies, 9 Rainforest Walk, 3800 Victoria, Australia. E-mail: hasan.fallahgoul@monash.edu.

‡Vincentius Franstianto, Monash University, School of Mathematics and Centre for Quantitative Finance and Investment Strategies, 9 Rainforest Walk, 3800 Victoria, Australia. E-mail: vincentius.franstianto@monash.edu.

§Xin Lin, Monash University, School of Mathematics and Centre for Quantitative Finance and Investment Strategies, 9 Rainforest Walk, 3800 Victoria, Australia. E-mail: xin.lin@monash.edu.

1 Introduction

Why do average rates of return differ among securities? It is well-established that returns are compensation for bearing systematic risk. However, accurately measuring systematic risk can be a difficult task, as there are hundreds of potential factors and possible asset pricing puzzles that have been proposed in the literature.[1] A common approach used in empirical asset pricing is linear factor models, which are used to run asset pricing tests. However, it should be noted that these models are only approximations of the true data generating process. Despite this, literature often sticks to linear specifications, largely due to the difficulty of explaining non-linear models. In this paper, we aim to address this issue by exploring how to explain the significance of variables in a non-linear, heavily parameterized asset pricing model using deep neural networks.

Multi-layer neural networks are a powerful class of artificial intelligence algorithms that can estimate multi-variable target functions with high accuracy (Anthony and Bartlett, 2009; Goodfellow et al., 2016). However, their statistical properties are often not well understood due to their non-linearity and heavy parametrization. Some recent research has begun to explore these properties, focusing on the convergence rate of the regression function and the significance of individual covariates, e.g., Horel and Giesecke (2020) Schmidt-Hieber et al. (2020), Kohler and Langer (2021), Farrell et al. (2021), and Fabozzi et al. (2021). These are useful for assessing the performance of neural network models and understanding the factors that influence the response variable. This paper builds on this work by providing a framework for conducting variable significance tests in the context of neural networks.

The high estimation and prediction power of neural networks make them well-suited for explaining average returns and measuring risk premia in finance and economics. However, their lack of statistical inference has limited their use among those who value explainability in their models. The main goal of this paper is to facilitate the transition from linear models to neural networks by developing significance tests. This will help to overcome the explainability issue and make it easier to apply neural networks in these fields.

This paper contributes to the growing body of literature on the application of machine learning in finance and economics. In particular, we focus on the use of neural networks in asset pricing, and develop a variable significance test to address the interpretability challenges that arise when using these models. Our work builds on that of Horel and Giesecke (2020) (HG) and Gu et al. (2020) (GKX), who also investigate the use of machine learning methods for measuring asset risk premia. Our contribution is to provide a framework for measuring the

[1] This is noted by Cochrane (2011), and more recently by Harvey et al. (2016), McLean and Pontiff (2016), Hou et al. (2017), and Feng et al. (2019). For example, Harvey et al. (2016) is a cautionary note discussing the statistical tests and suggest a much higher t-ratio should be achieved before a new factor is accepted. There are excellent reviews on asset pricing such as Campbell (2000), Fama and French (2004), and Goyal (2012), among others.

statistical significance of covariates in models with severe non-linearity and heavy parametrization. This complements the findings of GKX and helps to overcome the interpretability issue in applying machine learning methods to asset pricing.

We extend the variable significance tests from HG to smooth, multi-layer perceptrons (MLPs) with fully connected hidden nodes. In contrast to the sigmoid single-layer feed-forward neural networks considered in HG, our approach can be applied to a wider range of neural network architectures. To do so, we first establish consistency results for deep neural networks and derive an estimation rate. Farrell et al. (2021) provide important results on inference for deep neural networks, including nonasymptotic bounds for nonparametric estimation using various neural network architectures. We adopt a similar approach to Farrell et al. (2021) to prove the consistency of our estimator $\hat{f}_n$ of the target function $f_\star$. Consistency is crucial because it ensures that $\hat{f}_n$ is approximately equal to $f_\star$ for large sample sizes n, allowing us to interpret statistical tests on the estimator as tests on $f_\star$. However, unlike Farrell et al. (2021), we only consider smooth MLPs with a fixed depth and a differentiable activation function that is not a polynomial. This choice allows us to construct the sieve space, which we use to derive the asymptotic distribution of the smooth MLP estimator in a random function sense using the theory of empirical processes (Theorem 2.11.23 in Van der Vaart and Wellner (1996) and proof of Theorem 1 in HG). Unlike Farrell et al. (2021), we also use a scale-sensitive measure, metric entropy, in our sieve analysis. More specifically, we make the assumption that the weight parameters are bounded, which is necessary to apply Theorem 14.5 from Anthony and Bartlett (2009) to bound the metric entropy integral (see Subsection A.2.2).[2]

To the best of our knowledge, upper bounds, consistency results, and mathematically-based variable significance tests for fixed-depth multi-layer neural networks have not been previously reported in the literature. HG is the first paper to develop a theoretically-based statistical test for assessing variable impact in regression using a one-layer sigmoid network as the regression function. However, in practice, multi-layer networks are generally preferred over one-layer networks or other regression functions because they tend to have superior approximative power.

Our extension of the significance tests from HG to multi-layer neural networks with smooth activation functions broadens the range of applications for these networks. It is worth noting that the issue of training neural networks with smooth activation functions is not as challenging as it is often perceived by practitioners. By using a linear transformation, we can modify functions such as tanh and sigmoid to have a larger range than $(0, 1)$ or $(-1, 1)$, while maintaining the same steepness as the original functions. This ensures that the behavior of the transformed

[2]It is worth noting that Farrell et al. (2021) derive an estimation rate for causal parameters. In their work, "rates" primarily refer to nonasymptotic bounds (see Farrell et al. (2021), page 8).

functions is similar to the behavior of the original functions when they are trained with SGD algorithms. The larger range is necessary to prevent the activation functions from saturating quickly during training, which can impede the ability of SGD algorithms to effectively train the neural network. As a result, we can effectively train smooth neural networks whose behavior during training is not significantly different from the behavior of ReLU networks.

Measuring asset risk premia is closely tied to the structure of an asset pricing model and is essentially a prediction problem, e.g., (Cochrane, 2009). Machine learning methods, particularly neural networks, are effective tools for making predictions. Previous research (GKX) has found that all machine learning methods improve predictability compared to traditional approaches, and that neural networks with a small number of hidden layers perform best. However, GKX also notes that interpreting the results of machine learning models, particularly neural networks, can be difficult. This paper aims to address this issue by providing a statistical tool for discovering the factors that are statistically significant for predicting asset risk premia.

1.1 Literature review

This paper contributes to two strands of literature: neural networks and asset pricing.

1.1.1 Contribution to the neural networks

This paper addresses a gap in the current literature by extending the gradient-based statistical test developed by HG to MLPs, a more widely used and powerful architecture for neural networks. HG's original test statistic and its asymptotic analysis using nonparametric techniques are only applicable to single-layer feedforward networks, which is inadequate for MLPs. Our work focuses on the fully connected feed-forward neural networks where the activation function is a general bounded, non-polynomial and infinitely differentiable (smooth) function, which is important given the popularity of deep neural networks. Studies have shown that adding just one more layer to a neural network can lead to a significant improvement in performance when compared to increasing the number of nodes (width) (Eldan and Shamir, 2016). And since the seminal work of Hinton et al. (2006), the machine learning community has shifted its focus towards the use of deeper and wider networks.

To highlight our contribution relative to HG and Farrell et al. (2021), we summarize the theoretical results of this paper in five steps: (i) consistency; (ii) estimation rate; (iii) the asymptotic distribution of the estimator, i.e., $\hat{f}_n$; (iv) the asymptotic distribution for the transformed of the estimator, i.e., $\mathcal{T}[\hat{f}_n]$, and; (v) the asymptotic distribution for the sample version of $\mathcal{T}[\hat{f}_n]$, i.e., $\rho_n^2(\hat{f}_n)$. HG skips (i) and (ii) since it works with shallow networks and uses consistency results and an estimation rate in Chen and Shen (1998) (Theorem 3.1 in HG). However, the estimation rate in Chen and Shen (1998) is only valid for shallow networks. This is one of the reasons that they restrict their analysis to a single-hidden layer network with a sigmoid activation function.

In this paper, we prove the consistency for the smooth MLPs with fixed-depth and derive the estimation rate. The activation function does not have to be sigmoid, but it can be any smooth, non-polynomial functions. Note that the width of the network, H_n, increases with the sample size, n. This is meant to make the bias of the estimation by the least square estimator decreases as n increases. To derive the empirical function of asymptotic distribution, we work with Theorem 2.11.23 in Van der Vaart and Wellner (1996) which requires the estimation rate. Then, we use Theorem 3.2.2 of Van der Vaart and Wellner (1996) to show the convergence of our test statistics, enabling us to conduct statistical testing with this statistics. Our results complement the finding of Farrell et al. (2021) as we work with smooth MLPs instead of ReLU. Note that similarly to classic sieve analysis and unlike Farrell et al. (2021), we use a scale sensitive measure, metric entropy. More specifically, we assume the weight parameter is bounded. Farrell et al. (2021) do not restrict their class of network architectures to have bounded weights for each node, which allows for a richer set of approximating possibilities. They use a localized analysis that derives bounds based on scale insensitive compexity measure (Farrell et al., 2021, Section A.2).

Our variable significant test has the similar spirit to the wide variety of different methods that have been recently proposed to tackle interpretability and explainability of machine learning algorithms (Štrumbelj and Kononenko, 2014; Bach et al., 2015; Datta et al., 2016; Ribeiro et al., 2016; Shrikumar et al., 2016; Lundberg and Lee, 2017). For example, Lundberg and Lee (2017) propose a unified framework for interpreting predictions, SHAP (Shapley Additive explanations). However, our statistical test is analogous to the t-test for linear regressions, as both of them have rigorous theoretical foundations and are designed to focus on assessing statistical significance of covariates.

1.1.2 Contribution to the asset pricing

Our paper contributes to the asset pricing and econometrics literature by building on several existing research areas. We are closely aligned with the literature devoted to identifying asset pricing factors and the vast econometrics literature that estimates factor models. Additionally, our paper relates closely to the recent literature on the high dimensionality of cross-sectional asset pricing models, e.g., Feng et al. (2019) and GKX, among others.

Several recent papers in the literature have applied neural networks to predict risk premiums. For example, GKX conduct a comprehensive analysis of machine learning methods for measuring asset risk premia and demonstrate that allowing for nonlinearities improves predictions. Bianchi et al. (2021) evaluate and compare a range of machine learning methods for bond return predictability and find that non-linear approaches can be effective for out-of-sample predictions of bond excess returns. Chen et al. (2023) apply deep neural networks to estimate an

asset pricing model by using the fundamental no-arbitrage condition as a criterion function.

An important distinction between our paper and existing literature (e.g., GKX and Chen et al. (2023)) is that we focus on the statistical evaluation of the importance of factors through estimating an entire reduced-form asset pricing model using MLPs. Our research extends the understanding from the existing literature by identifying specific predictors that are statistically significant.

In Feng et al. (2019), a new method is introduced to evaluate the contribution of new factors to asset pricing, by comparing them to a high-dimensional set of existing factors. The model selection approach accounts for potential mistakes in variable selection that can lead to bias, unlike traditional methods. When applied to a set of factors found in recent literature, the method finds that most factors are redundant, but a small number have a statistically significant explanatory power beyond the many factors previously proposed. Our research supports these findings as we also found that only a limited number of covariates out of about 800 are statistically significant in predicting asset risk premia. It is important to note that while Feng et al. (2019) uses the model selection approach to identify relevant factors, our approach is focused on identifying statistically significant predictors, assuming the model is already given.

The structure of the paper is as follows: In Section 2, we establish the regression framework and present the smooth MLPs that we study. The main theoretical contribution of the paper is outlined in Section 3, which covers the inferences based on these MLPs. Specifically, Subsection 3.1 discusses the probabilistic convergence of the smooth MLPs and the estimation rate of the regression target function. This convergence rate serves as a basis for constructing the statistical test in Subsection 3.2, which is presented along with its limiting distribution. In Section 4, we evaluate the theoretical results through simulations. In Section 5, we compare neural networks for measuring asset risk premiums through an empirical analysis. Finally, we conclude the paper in Section 6.

2 The setting

In this section, we discuss the regression framework, hypothesis test, and architecture of deep neural networks. Table A1 introduces the notation we use throughout.

Insert Table A1 about here

2.1 Regression framework and visual structure of target function

Fix a probability space $(\Omega, \mathcal{F}, P)$ and assume Y is related to covariates $\boldsymbol{X} \in \mathbb{X} := [-1,1]^d$ through the following regression model

$$Y = f_\star(\boldsymbol{X}) + \varepsilon. \tag{1}$$

Assumption 2.1. *Assume that the response variable Y whose absolute value is bounded above by an arbitrary positive parameter M_Y, covariate vector $\mathbf{X}$ is continuously distributed according to P, the joint law of both $\mathbf{X}$ and ε, which is absolutely continuous with respect to the Lebesgue measure Δ. The error variable ε satisfies the following assumptions: ε are independent of $\mathbf{X}$, $\mathbb{E}[\varepsilon] = 0$, $|\varepsilon| \leq 1$, and $\mathbb{E}[\varepsilon^2] = \sigma^2 < \infty$.*

Note that $f_\star$ is the true mean of the response variable Y. We assume

$$\|f_\star\|_\infty + \sum_{\substack{\vec{\alpha} \in (\mathbb{N} \cup \{0\})^d \\ \|\vec{\alpha}\|_1 = 1}} \|\nabla^{\vec{\alpha}} f_\star\|_\infty \leq 2M \tag{2}$$

where $\vec{\alpha}$ is a multi-index notation and M is an arbitrary large positive constant. We use classic vector notation to indicate a multi-index. Bold letters (small bold letters if they are deterministic, and capital bold letters if they are random) are used for vectors. The upper bounds of ε and $f_\star$ imply $|Y| < 2M + 1$.

The assumption of a bounded response variable, Y, in our regression model is not significantly more restrictive than the usual assumptions made in such models, which typically include compact support for the predictor variables, $\mathbf{X}$, boundedness of f_*, and finite moments for the error term, ε, see (Farrell et al., 2021, Assumption 1). In many cases, the assumption of bounded outcomes is necessary for the analysis to be meaningful, for example, when lagged outcomes are used as predictors. The use of continuously distributed covariates is typical in these types of analyses. From a theoretical perspective, when covariates only take on a few values, they can be treated as if they were continuous by considering their averages, which does not affect the rate of convergence. When covariates take on many discrete values, it may be more accurate to treat them as if they were continuous, which may result in slower convergence rates.

To ensure that the error term is bounded in our model, we can normalize the response values by dividing them by a large constant, M_Y. This will produce a new error term, $\frac{\varepsilon}{M_Y}$, that is bounded between -1 and 1. For example, in the regression model given by equation (1), we can write:

$$\left|\frac{Y}{M_Y}\right| = \left|\frac{f_\star(\mathbf{X}) + \varepsilon}{M_Y}\right| \leq \left|\frac{f_\star(\mathbf{X})}{M_Y}\right| + \left|\frac{\varepsilon}{M_Y}\right| \leq 1.$$

This equation tells us that the absolute value of the normalized response values is always less than or equal to 1. This implies that the absolute value of the normalized error term, $\left|\frac{\varepsilon}{M_Y}\right|$, must also be less than or equal to 1. In other words, the normalization process produces a new error term, $\frac{\varepsilon}{M_Y}$, that is bounded above by 1. This means that our assumption of a bounded error term is reasonable in this case due to the normalization by dividing Y by a large constant.

Assumption 2.2. *Assume target function $f_\star \in \mathcal{C}^{\lfloor d/2 \rfloor + 2}\left([-1,1]^d\right) \cap \mathbb{H}^{\lfloor d/2 \rfloor + 2, 2}\left([-1,1]^d\right)$, where $\mathbb{H}^{\lfloor d/2 \rfloor + 2, 2}\left([-1,1]^d\right)$ indicates the Sobolev space with weak differentiation order equal to $\lfloor d/2 \rfloor + 2$*

and the weak derivatives of the functions in this space is in $L^2\left([-1,1]^d\right)$.

Assumption 2.3. *Assume target function* $f_\star$ *have a binary tree visual structure as in Poggio et al. (2017) with certain depth and number of nodes.*

The notation $\mathbb{H}^{\lfloor d/2\rfloor+2,2}\left([-1,1]^d\right)$ refers to the Sobolev space of functions with weak differentiation order equal to $\lfloor d/2\rfloor+2$, where the weak derivatives of the functions in this space are in $L^2\left([-1,1]^d\right)$. It is important to note that the intersection of $\mathcal{C}^{\lfloor d/2\rfloor+2}\left([-1,1]^d\right)$ and $\mathbb{H}^{\lfloor d/2\rfloor+2,2}\left([-1,1]^d\right)$ is not dense in $\mathbb{H}^{\lfloor d/2\rfloor+2,2}\left([-1,1]^d\right)$. For example, a constant function on the real line belongs to $\mathcal{C}^{\lfloor d/2\rfloor+2}(\mathbb{R})$ but is not a member of $\mathbb{H}^{\lfloor d/2\rfloor+2,2}(\mathbb{R})$. Additional information about the Sobolev, Hölder, and Besov spaces can be found in Giné and Nickl (2021). In order to bound the integral of the entropy number of $\mathcal{F}$ (see Section 3.2 for the construction of $\mathcal{F}$), we require $f_\star$ to belong to the Sobolev space $\mathbb{H}^{\lfloor d/2\rfloor+2,2}\left([-1,1]^d\right)$ (see Corollary 4 of Nickl and Pötscher (2007)).

To emphasize that $f_\star$ must be a differentiable function, we assume that it belongs to the space $\mathcal{C}^{\lfloor d/2\rfloor+2}\left([-1,1]^d\right)$. Note that the differentiation order that we impose here is much higher than those in (2), where the order is just 1 (which can be seen from the restriction $\|\alpha\|_1 = 1$). This requirement is necessary because the differentiation order of $f_\star$ must be at least $\lfloor d/2\rfloor+2$ in order to use Corollary 4 of Nickl and Pötscher (2007) in our proof (see the bound on the metric entropy integral in Appendix D). Additionally, in order to use Theorem 2 of Poggio et al. (2017), we need Assumption 2.3 to hold.

The assumptions outlined in 2.2 and 2.3 are in line with commonly used asset pricing models that employ linear factor models, e.g., Fama (1970), Fama and French (1993), Fama and French (2015), and Cochrane (2009). Linear factor models are a popular choice in finance as they offer a useful tool for understanding the underlying structure of financial markets and the relationships between different assets. These models posit that the return of a portfolio or individual asset can be represented as a linear combination of a small number of underlying factors, which are often chosen for their economic interpretability, such as the market return, the value premium, and the size premium.

We next discuss a binary tree visual structure assumption.

A hierarchical binary tree is a tree in which every child node has exactly two parents (see Figure A1). The tree structure here is only a visual representation of the mathematical operations inside the target function $f_\star$. It is not related to the tree structure in computer science or machine learning. Hierarchical function composition has been used to understand why and when one should use deep networks instead of shallow networks, e.g., (Poggio et al., 2015, 2017).

We are using Assumption 2.3 to utilize Theorem 1 and 2 from Poggio et al. (2017). This

assumption is more general than it may appear, as many functions can be represented using a binary tree structure, such as linear combinations and basic one-variable functions like trigonometric, exponential, and logarithmic functions, e.g., (Poggio et al., 2017, Figure 1). We are not intending to exclude any specific functions with this assumption. In fact, a binary tree visual structure can approximate smooth functions well, at least on a compact domain. To represent a linear combination with an even number of variables, we can follow these steps: take the independent variables and place them at the base nodes, create pairs of these variables, take the weighted summation of each pair and place the result at a higher level, and repeat this process until all variables are summed. If the number of variables is odd, we can add another variable with a value of zero to the tree to make the number of variables even. Similarly, to represent a multiplication of multiple variables, we can replace the weighted summation with multiplication and follow the same steps. If the number of variables is odd, we can add a new variable with a value of 1 instead of 0. As a concrete example, function $\sin(X_1 + X_2) + X_3$ can be restructured into binary tree where depth $L_d = 3$ with four nodes in the bottom level, two nodes in the second level, and one node in the top level. The four nodes in the bottom represent X_1, X_2, X_3, and 0. For the nodes in the second level, it represents $\sin(X_1 + X_2)$ and $X_3 + 0 = X_3$. The top node represents the function $\sin(X_1 + X_2) + X_3$ itself. As a specific example, we provide a binary tree depiction of the function $\sin(X_1 + X_2 + X_3 + X_4 + X_5)$ below

Insert Figure A1 about here

The binary tree visual structure is only needed for the target function $f_\star$ not the least-square estimator $\hat{f}_n$. The least-square estimator is obtained by minimizing sum of squared errors (see Subsection 2.3 for the detailed explanation). In this paper, this estimator is in the form of deep neural networks. Note that, in Poggio et al. (2017), it is directly stated that "*the deep network does not need to have the same compositional architecture as the compositional function to be approximated. It is sufficient that the acyclic graph representing the structure of the function is a subgraph of the graph representing the structure of the deep network*".

We assume that the complexity (number of nodes in the binary tree visualization in target function), N_f, and depth, L_d of target function $f_\star$ is fixed. This assumption is not restrictive. For a function with a certain binary tree structure, we can always expand the depth. If we want to increase the depth of the binary tree by one, for instance, then we can do it by simply adding the output of the original tree with the output of the new function with the exact same binary tree structure. The new function has a separate, identical binary tree representation, but it simply takes zeros for all of its inputs and the mathematical operation in each node of non-input layers is single addition – the new function is thus equal to zero. The addition of these two output nodes means that the two binary trees are combined together to form a new binary

tree whose depth is $L'_d = L_d + 1$, and the output is the addition between the original function output and zero (from the new function), and thus preserving the original binary tree output. Therefore, what is really required is L_d to be sufficiently large. If it is then N_f will automatically be sufficiently large (see (Poggio et al., 2017)).

We need the complexity of deep networks to derive the estimation rate for deep neural networks (see Theorem 3.2.3 and its proof). Note that HG do not consider classes of functions with the hierarchical binary tree compositions, since they work with the single-layer feedforward neural network (shallow networks) and use the estimation rate results of Chen and Shen (1998).[3] However, the estimation rate in Chen and Shen (1998) is only valid for shallow networks (Chen and Shen, 1998, Equation (2.4)).

2.2 The hypotheses testing

The objective of the proposed statistical testing is to test whether a chosen covariate, e.g., x_j, significantly influences Y through function $f_\star$. In line with HG, the proposed importance test is based on the expected value of the partial derivative square taken in the probability space $(\Omega, \mathcal{F}, P)$. We say that covariate x_j is not significant if the stated expected value is zero. This statement forms the null hypothesis of our statistical testing which is given by

$$\mathcal{H}_0 : \zeta_j = \mathbb{E}\left[\left(\frac{\partial f_\star(\boldsymbol{x})}{\partial x_j}\right)^2\right] = \int_{\mathbb{X}} \left(\frac{\partial f_\star(\boldsymbol{x})}{\partial x_j}\right)^2 dP(\boldsymbol{x}) = 0 \tag{3}$$

$$\mathcal{H}_1 : \zeta_j \neq 0.$$

The null hypothesis $\mathcal{H}_0$ is an extension of the classical variable significance tests $\mathcal{H}_0 : \beta_j = 0$ for linear regression with linear functions $f(\boldsymbol{x}) = \beta_0 + \sum_{i=1}^{d} \beta_i x_i$. Intuitively, the partial derivative $\frac{\partial f_\star(\boldsymbol{x})}{\partial x_j}$ assesses the influence of jth covariate on $f_\star$. If it is "large enough", then we can say that jth covariate is "significant" to the variation of $f_\star$ values. Otherwise, jth covariate is "not significant". The square of the partial derivative is taken to avoid compensation of positive and negatives values, ensure differentiability, and help discriminate between large and small values.

2.3 Deep neural networks

This section covers the mathematical foundation of deep neural networks. For a more in-depth understanding, we recommend consulting the works of Anthony and Bartlett (2009) and Goodfellow et al. (2016). Farrell et al. (2021) also offer a thorough discussion on deep neural networks and their relationship to nonparametric estimation.

In this work, we approximate the target function $f_\star$ using deep neural networks, which can be thought of as analogues to classical nonparametric techniques such as smoothing splines

[3] See r_n in Theorem 3.1 of HG.

and kernel regression. A key aspect of working with neural networks is deciding on the network architecture or class. This choice is similar to selecting the smoothing and tuning parameters for a smoothing spline in nonparametric techniques. A neural network has an input layer with d units, an output layer, and one or more hidden layers (a shallow network has one hidden layer, while a deep network has multiple hidden layers). Each hidden layer has one or more nodes, also known as units or neurons, and an activation function $\psi(.) : \mathbb{R} \to \mathbb{R}$ is applied to each hidden node. The neural network used in this paper is a feedforward neural network (FFN), which means that there are no connections between units that form a cycle and no connections between hidden nodes within the same hidden layer. Additionally, all hidden layers in the FFN have the same number of nodes, making it an example of a Multi-Layer Perceptron (MLP). In this study, we always use MLPs with equal numbers of hidden nodes in each hidden layer. Throughout the following discussions, we use the terms MLP and deep neural network (DNN) interchangeably to refer to the class of neural networks under consideration.

Figure 1 provides an illustration of an MLP architecture. The network consists of $d = 2$ input units, corresponding to covariates $\boldsymbol{x} \in \mathbb{R}^d$, one output unit for outcome y. Hidden units are grouped in a sequence of $L_d = 2$ layers where a node is in layer $l = 1, 2, ..., L_d$ if it has a predecessor in layer $l - 1$ and no predecessor in any layer $l' \geq l$. L_d is referred by the depth of network. The number of units in a hidden layer is called the width of layer, denoted by $H_l = H_n$. Note that we index H with the sample size to show the dependency of H with n. The activation function $\psi(.) : \mathbb{R} \to \mathbb{R}$ is applied to all hidden nodes. For the MLP architecture, all hidden layers have the same number of nodes, i.e., H_n. Thereby, the complexity of the network is $U = \mathcal{O}(H_n L_d)$. As a note on exposition, to be consistent with Poggio et al. (2017), if f denotes the neural networks architecture, we would refer to the complexity of f by N_f.

Insert Figure 1 about here

Let L_d and H_n represent the number of hidden layers and nodes in each layer, respectively, where n is the sample size. The depth of the MLP is indexed by the input size d to show that it is not dependent on n, but may depend on d.

We now construct a class of MLPs with depth L_d and H_n hidden nodes per layer, which is defined by

$$\mathcal{F}_{DNN,H_n} = \{z_{L_d+1,1}(\boldsymbol{x}) \mid \boldsymbol{x} \in \mathbb{X}\}$$

where $z_{l,j}(\boldsymbol{x})$ is the output of the j^{th} node of the layer l in the neural network with input $\boldsymbol{x}$, $l = 0$ or $l = L_d + 1$ correspond to the input and output layers, respectively, and $1 \leq l \leq L_d$ correspond to the l^{th} hidden layer. Also, $j \in \{1, 2, \ldots, H_{n,l}\}$, where $H_{n,l}$ is the number of nodes in the l^{th} layer, $H_{n,0} = d$, $H_{n,L_d+1} = 1$. Note that all hidden layers have H_n nodes. For $1 \leq l \leq$

L_d, the formula $z_{l,j}(\boldsymbol{x})$ is

$$z_{l,j}(\boldsymbol{x}) = \psi\left(\sum_{k=1}^{H_{n,l-1}} \gamma_{l,j,k} \cdot z_{l-1,k}(\boldsymbol{x}) + \gamma_{l,j,0}\right)$$

where $z_{0,k}(\boldsymbol{x}) = x_k$, the k^{th} element of $\boldsymbol{x}$, $\gamma_{u,j,k}$ and $\gamma_{u,j,0}$ denote the weight and bias parameters of the node $u = (l, j)$, and ψ is an activation function and defined as

$$\psi : \mathbb{R} \to [-b, b]$$

where $b > 1$ and $\psi \in \mathcal{C}^\infty$ is smooth and non-polynomial function.

When $l = L_d + 1$, function $z_{L_d+1,1}(\boldsymbol{x})$ is

$$z_{L_d+1,1}(\boldsymbol{x}) = \sum_{k=1}^{H_{n,L_d}} \gamma_{L_d+1,1,k} \cdot z_{L_d,k}(\boldsymbol{x}) + \gamma_{L_d+1,1,0}. \tag{4}$$

From (4), it is clear that the activation function is not applied in the output node. We set $j = 1$ for the output layer as there is only one node. The output of a neural network can be thought of as a type of sieve estimation where the basis functions are flexibly learned from the data. Equation (4) can then be rewritten as

$$\begin{aligned} f_{DNN,H_n}(\boldsymbol{x}) &= \sum_{k=1}^{H_{L_d}} \gamma_{L_d+1,1,k} \cdot z_{L_d,k}(\boldsymbol{x}) + \gamma_{L_d+1,1,0} \\ &= \boldsymbol{\gamma}_{L_d+1} \cdot \boldsymbol{z}_{L_d}(\boldsymbol{x}) + \gamma_{L_d+1,0}. \end{aligned} \tag{5}$$

We assume that $\gamma_{u,j,k}$ is bounded with the ℓ^1 upper bound $\|\gamma_{u,j,k}\|_1 \leq Mb$, where M is the same arbitrary positive parameter as in (2). The function class $\mathcal{F}_{DNN,H_n}$ can thus be seen as the set of all bounded-weight, deep neural network functions f_{DNN,H_n} whose mathematical form is described in (4) (or equivalently (5)).

Remark 2.3.1. *The assumption of the boundedness for weight parameters is crucial, since we use (Anthony and Bartlett, 2009, Theorem 14.5) for bounding metric entropy integral in Subsection A.2.2. HG have also the same assumption of weights, see (Horel and Giesecke, 2020, Equation (11)). Note that Farrell et al. (2021) do not restrict their class of network architectures to have bounded weights for each node, which allows for a richer set of approximating possibilities. They use the arbitrary upper bound on the neural network class from which they take their estimators (Farrell et al., 2021, Equation 2.3.), and this leads to a more flexible bounds on the weight parameters. However, since we work with smooth activation function, not ReLU activation function, we need (Anthony and Bartlett, 2009, Theorem 14.5) that requires bounded weight parameter assumption. Therefore, we cannot employ Farrell et al. (2021)'s approach.*

We use tanh as the activation function in training, e.g., $\psi \equiv \tanh$. We prefer tanh to other common non-polynomial activation function such as sigmoid or negative sigmoid, because it has a more expansive function range $(-1, 1)$ instead of $(0, 1)$ and $(-1, 0)$ for sigmoid or negative sigmoid, respectively. Theoretically, this larger range gives tanh a better performance

in training when compared to them. Also, to work with (Poggio et al., 2017, Theorem 2), the infinite differentiability of the neural network w.r.t. each of its weight parameter is required. Hence, we cannot use the neural networks with non-differentiable activation functions such as ReLU (the activation function that is used in Farrell et al. (2021)) or its variants such as Leaky/Parametric ReLU – $LReLU(x) = \alpha x + (1-\alpha)\max(0,x)$, and Exponential Linear (see Clevert et al. (2015)).

All in all, for a user-chosen architecture, encompassing the choice of $\psi(.)$, H_n, L_d, and the graph structure, the least square estimator belonging to $\mathcal{F}_{DNN,H_n}$ is defined by $\hat{f}_n$ and obtained from

$$\hat{f}_n = \underset{f \in \mathcal{F}_{DNN,H_n}}{\arg\min} \sum_{i=1}^{n} \ell\left(f, \boldsymbol{x}_i, y_i\right) \tag{6}$$

where $(\boldsymbol{x}_i, y_i)$ is the samples from n observations of $(\boldsymbol{X}, Y)$, and

$$\ell\left(f, \boldsymbol{x}_i, y_i\right) := \frac{1}{2}\left(y_i - f(\boldsymbol{x}_i)\right)^2 = \frac{1}{2}\left(f_\star(\boldsymbol{x}_i) + \varepsilon_i - f(\boldsymbol{x}_i)\right)^2. \tag{7}$$

The $\ell(.)$ function is referred to as the loss or cost function.

3 Inferences for smooth MLPs

In this section, we: (i) provide the consistency of our least square estimator, i.e., $\hat{f}_n$; (ii) derive the distribution of the test statistic under the null hypothesis using delta method, and; (iii) discuss a discretization approach to compute the asymptotic distribution of the test statistic. The consistency of the least square estimator estimation is guaranteed by an $o_P(1)$ upper bound of both $\|\cdot\|_{L^2(P)}$ and the semi-norm

$$\rho_n(g(\boldsymbol{x})) := \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(g(\boldsymbol{x_i})\right)^2}$$

where $\boldsymbol{x_i}$, $i = 1, \ldots, n$ are empirical samples of $\boldsymbol{x}$. Readers who are interested in the proofs may refer to Appendixes.

3.1 Consistency of smooth MLPs

To conduct hypotheses testing described in Subsection 2.2, we first show that our least square estimator $\hat{f}_n$ is a consistent estimator of $f_\star$. The consistency plays an important role. Because it guarantees that the estimator $\hat{f}_n$ is approximately equal to $f_\star$ for large n. Thus the statistical test conducted on the estimator can be seen as the test on $f_\star$. Unlike Farrell et al. (2021), we only work with smooth MLPs. The main reason for this choice is that we need the parallel containment of smooth MLPs to construct the sieve space.[4] We use the sieve space to derive the asymptotic distribution of the smooth MLP estimator in a random function sense using the theory of empirical processes (Van der Vaart and Wellner, 1996, Theorem 2.11.23) and (Horel

[4]The term "parallel containment" means any two smooth MLPs S_1 and S_2 with the same depth L_d can be combined together to form a new, single smooth MLP S_3. The weights of hidden nodes in S_3 that connect the hidden nodes from S_1 to S_2 (or vice versa) are zero.

and Giesecke, 2020, Theorem 1).

Before stating our next assumption, we provide a discussion about an approximation error of the DNN under the estimation of $\|\cdot\|_\infty$. Define the $\|\cdot\|_\infty$-minimizer of $\mathcal{F}_{DNN,H_n}$ and related $\|\cdot\|_\infty$ distance to $f_\star$ by

$$f_n := \underset{f\in\mathcal{F}_{DNN,H_n}}{\arg\min} \|f - f_\star\|_\infty \tag{8}$$
$$\epsilon_n := \|f_n - f_\star\|_\infty$$

where ϵ_n represents the approximation error of $f_\star$ by f_n and is directly related to the upper bound rate of Theorem 3.1.1. It is also obvious that f_n is *different* from $\hat{f}_n$, since the former indicates the optimal $\|\cdot\|_\infty$-approximator and latter is the least square approximator.

Note that ϵ_n involves the the upper bound in the consistency Theorem 3.1.1. Farrell et al. (2021) prove the upper bound of ϵ_n by using (Yarotsky, 2017, Theorem 1). Since (Yarotsky, 2017, Theorem 1) is only valid for ReLU networks, we cannot use it. Instead, we use similar results from Poggio et al. (2017) for neural networks with smooth, non-polynomial activation function. To this ends, we need the following assumption.

Assumption 3.1. *The weight restrictions of $\gamma_{u,j,k}$, the weight parameters of $\mathcal{F}_{DNN,H_n}$, does not affect the usage of Theorem 1 and Theorem 2 of Poggio et al. (2017).*

Assumption 3.1 is not as strict as it might seem. Upper and lower bounds of the weights of neural networks inside $\mathcal{F}_{DNN,H_n}$ depend on parameter M, where it can be made arbitrary large. For sufficiently large M, we can construct $\mathcal{F}_{DNN,H_n}$ such that has $\|\cdot\|_\infty$-approximator as described in (Poggio et al., 2017, Theorem 2).

Under Assumptions 2.1, 2.2, 2.3, and 3.1, we obtain the following result.

Theorem 3.1.1. (Consistency and Upper Bound of Estimation) *Let $\hat{f}_n$ be the least square estimator defined by (6), for a loss function obeying (7), with width H_n and depth L_d. Then with probability at least $1 - e^{-p}$, for n large enough,*

$$\|\hat{f}_n - f_\star\|_{L^2(P)} \leq U(C', \epsilon_n) := C'\left(\sqrt{\frac{H_n^2 L_d^2 \ln(H_n L_d n)}{n}} + \sqrt{\frac{\ln(\ln(n)) + p}{n}} + \epsilon_n\right)$$

where the order of $p = p(n)$ must be less than $H_n^2 L_d^2 \ln(H_n L_d n)$ and $C' > 0$ independent of n but might depend on d, M, and other fixed constants. Furthermore, the same upper bound applies for $\rho_n((\hat{f}_n - f_\star)(\boldsymbol{x}))$.

Several aspects of this result warrant discussion. Theorem 3.1.1 is an extension of (Farrell et al., 2021, Theorem 2) in the context of smooth neural networks. Similar to (Farrell et al., 2021, Theorem 2), this theorem gives a flexibility in controlling the upper bound $U(C', \epsilon_n)$ by using H_n – the number of hidden nodes in each of hidden layers in

the network belonging to $\mathcal{F}_{DNN,H_n}$. We can also adjust values of p to get the required convergence rate, as long as $p = p(n) = o(H_n^2 L_d^2 \ln(H_n L_d n))$. Note that this implies $U(C', \epsilon_n) = \mathcal{O}\left(n^{-1}\left(H_n^2 L_d^2 \ln(H_n L_d n) + \ln(\ln(n))\right)\right) \to 0$ as $n \to \infty$.

Recall that $f_\star$ is a function with a binary tree visual representation that has continuous partial derivatives (order one is enough here). By Theorem 2 of Poggio et al. (2017), we have that $N_{f_n} = \mathcal{O}(d\epsilon_n^{-2})$, where N_{f_n} is the number of hidden nodes that exist in f_n. This theorem (Poggio et al., 2017, Theorem 2) enables us to choose ϵ_n. This in turn allows us to choose the number of nodes in f_n, which is needed to approximate $f_\star$ with approximation error at least ϵ_n. The fact $f_n \in \mathcal{F}_{DNN,H_n}$ implies that f_n is a neural network with L_d hidden layers and H_n hidden nodes in each of hidden layer. Hence, we have $N_{f_n} = \mathcal{O}(H_n L_d)$. Finally, since $N_{f_n} = \mathcal{O}(H_n L_d)$ and $N_{f_n} = \mathcal{O}(d\epsilon_n^{-2})$, we can deduce that

$$H_n = O\left(\frac{d}{L_d}\epsilon_n^{-2}\right).$$

Take ϵ_n such that

$$o(1) = \epsilon_n = \omega(n^{-0.25}) \tag{9}$$

with $a_n = \omega(b_n)$ indicates $\lim_{n\to\infty} \frac{a_n}{b_n} = \infty$. Statement (9) together with the upper bound from Theorem 3.1.1 give us

$$o(1) = U(C', \epsilon_n) = \omega(n^{-0.25}). \tag{10}$$

Note that we need both ϵ_n and $U(C', \epsilon_n)$ to be of order $\omega(n^{-0.25})$ to prove Theorem 3.2.3.[5]

Remark 3.1.1. *Our methodology is closely aligned with the strategies pursued by Farrell et al. (2021) and Horel and Giesecke (2020), both addressing similar challenges. Notably, these approaches do not explicitly account for time dependency, as emphasized in Assumption 1 by Farrell Farrell et al. (2021). Additionally, it is essential to highlight that the methods introduced by Chen and Shen (1998) are designed for handling weakly dependent data over time, and similarly, the techniques outlined by Chen and White (1999) operate within this context. However, these methods predominantly center around shallow, sigmoid-based networks, unlike our approach that employs deep,* tanh*-based networks.*

Chen and Shen (1998) present a comprehensive theoretical framework that addresses the convergence rate of sieve extremum estimates and establishes the root-n asymptotic normality of "plug-in" sieve estimates for smooth functionals in the context of time series stationary β-mixing observations. Under a set of sufficient conditions akin to those delineated by Shen and Wong (1994) for i.i.d. data, the authors attain parallel convergence rates for β-mixing data when compared to the rates for i.i.d. data.

The rate result for β-mixing observations is established through a two-step process. First, a rigorous uniform exponential probability inequality is established for empirical processes indexed by a wide-ranging class of functions. This inequality serves as the foundation for establishing stochastic equicon-

[5] Any power of n^a where $-0.25 < a < 0$ that satisfies equation (9) can be chosen such that ϵ_n is of order n^a. One can take $\epsilon_n = n^{-0.17}$, for example, to get one specific sequence of ϵ_n. See Appendix D for details.

tinuity with the desired rate. This, in turn, facilitates the extension of Shen (1997)'s result on root-n asymptotic normality for i.i.d. data to encompass β-mixing data.

In our sieve analysis, we employ a localized approach to bounding empirical processes. This approach is grounded in scale-sensitive metrics, specifically metric entropy (see localization analysis in Section A). Our methodology is influenced by the work of Farrell et al. (2021), which draws from concepts outlined by Koltchinskii and Panchenko (2000), Bartlett et al. (2005), and the related work of Koltchinskii (2006).

Intuitively, Rademacher complexity gauges the flexibility of the function class in predicting random signs. By calculating the expectation of Rademacher complexity conditioned on the data, we derive the empirical Rademacher complexity. This expectation, taken over both the data and the draws, gives rise to the Rademacher complexity.

3.2 Asymptotic distribution of test statistic

Before deriving the asymptotic distribution of the test statistic, we define a working function space with its sieve space. Specifically, we construct a function space and a sieve space inside it such that is related to $\mathcal{F}_{DNN,H_n}$. We need a function space which is both convex and $\|\cdot\|_{L^2(P)}$-compact. The convexity is required for the proof of Theorem 3.2.2, where we use (Römisch, 2004, Theorem 2).

Suppose that $\mathcal{F}_{DNN}$ is a smooth neural network space which is composed of all fixed-depth smooth MLPs with depth L_d, finite numbers of nodes per layer and finite weight assignments, equal number of hidden nodes for each of hidden layers, and ψ as its activation function. Note that the number of hidden nodes per layer has no upper bound in this case. We assume that the weights of the computation nodes of $\mathcal{F}_{DNN}$ is bounded above with the same upper bound as $\mathcal{F}_{DNN,H_n}$ under the same norm. This ensures $\mathcal{F}_{DNN,H_n} \subseteq \mathcal{F}_{DNN}$.

Remark 3.2.1. *It is possible to find a neural network v_1' in the set $\mathcal{F}_{DNN,H_n}$ and a natural number N_1 such that, for any $n \geq N_1$, the neural network v is equivalent to v_1'. This is true because: (i) the set of smooth MLPs is parallel and contained within $\mathcal{F}_{DNN}$, and; (ii) the weight parameters of nodes that do not exist in the smaller MLP are zero. Additionally, it is clear that the set $\mathcal{F}_{DNN}$ is closed under addition. To see this, consider two neural networks S_1 and S_2 that belong to $\mathcal{F}DNN$. If we combine these networks into a third network $S3$ using the following rules: (i) the weight parameters of S_3 that connect the input layer or hidden layers from S_1 to S_2 are zero, except for the output layer; (ii) the weight parameters of S_3 that connect the input layer or hidden layers from S_2 to S_1 are zero for the output layer; (iii) $S_3 = S_1 + S_2$. Then it follows that S_3 also belongs to $\mathcal{F}_{DNN}$, because $\mathcal{F}_{DNN}$ is composed of neural networks with an equal number of hidden layers.*

We next define the function space $\mathcal{F}$ from which we construct our sieve space. Take the equivalence class $[f_\star]$ such that $\forall f \in [f_\star]$, $\exists t > 0$ such that $f = tf_\star$. Then, we construct a function set $[f_\star] + \mathcal{F}_{DNN} := \{c_1 f_\star + c_2 v \mid v \in \mathcal{F}_{DNN}, c_1, c_2 \in \mathbb{R}\}$. It is clear that $[f_\star] + \mathcal{F}_{DNN}$ is

convex. To have $\|\cdot\|_{L^2(P)}$-compactness, we define

$$\mathcal{F} := \left\{ f \in [f_\star] + \mathcal{F}_{DNN} : \|f\|_{\mathbb{H}^{\lfloor d/2 \rfloor + 2,2}} \leq B, \|f\|_{L^2(P)} \leq M_{11} \right\}. \tag{11}$$

The $\|\cdot\|_{L^2(P)}$-compactness of $\mathcal{F}$ is guaranteed by the following facts: (i) $\mathcal{F}$ is $\|\cdot\|_{L^2(P)}$-closed,$\mathcal{F} \subseteq \overline{\mathcal{F}} := \{\mathcal{C}^1(\mathbb{X}) \mid \|f\|_{\mathbb{H}^{\lfloor d/2 \rfloor + 2,2}} \leq B\}$ and $\overline{\mathcal{F}}$ is proven to be $\|\cdot\|_{L^2(P)}$-compact in Freyberger and Masten (2015). In fact, we design $\mathcal{F}$ in this way to make sure that it is $\|\cdot\|_{L^2(P)}$ compact. Because we use (Kim et al., 1990, Lemma 2.6.) in Appendix B.2, whose one of its requirements is the compactness of the index space of the related Gaussian process, where the index space itself is $\mathcal{F}$.[6]

We now define the sieve space, which is given by

$$\mathcal{F}_n := \left\{ t_1 v_1' + t_2 v_2' \in \mathcal{F} \mid v_1', v_2' \in \mathcal{F}_{DNN,H_n}, t_1, t_2 \in \mathbb{R} \text{ s.t. } \sup_{\mathcal{V}_1 \in v_1'} \|t_1 \cdot w_1(\mathcal{V}_1)\|_1, \right. \tag{12}$$

$$\left. \sup_{\mathcal{V}_2 \in v_2'} \|t_2 \cdot w_2(\mathcal{V}_2)\|_1 \leq M_{22} b \right\}$$

where w_1 and w_2 are weights of computation nodes $\mathcal{V}_1$ and $\mathcal{V}_2$, respectively. The fact that $\mathcal{F}_{DNN}$ is closed with respect to the addition allows the existence of two arbitrary elements of $\mathcal{F}_{DNN,H_n}$, v_1' and v_2', whose linear combination belongs to $\mathcal{F}$.

Remark 3.2.2. *Note that M_{11}, M_{22}, b, and B in (11) and (12) can be chosen arbitrarily. Although $\hat{f}_n \in \mathcal{F}_{DNN,H_n}$, we can construct $\mathcal{F}$ and $\mathcal{F}_n$ that satisfy $\hat{f}_n \in \mathcal{F}_n \subseteq \mathcal{F}$, $\forall n$ by adjusting the arbitrary parameters M, M_{11}, M_{22}, b, and B. However, it is required that $M_{11} \ll M$, $M_{22} \ll M$ and $M_{11} \ll M_{22}$. The first two inequalities are needed to make sure that we can use M as the largest arbitrary positive constant. The third inequality is to ensure the denseness of $\mathcal{F}_n$. We can do this since their values are taken after we have $f_\star$. As these parameters can be set arbitrarily large, the realistic algorithm such as unbounded SGD-based algorithm can be used to train the neural network that yields a least-square estimator belonging to $\mathcal{F}_n$.*

Remark 3.2.1, the way f_n is defined in (8), and the requirement $M_{11} \ll M_{22}$ from Remark 3.2.2 show that every $t_1 f_\star + t_2 v \in \mathcal{F}$ has a related converging sequence of functions $t_1 f_n + t_2 v \in \mathcal{F}_n$ that satisfies $\|(t_1 f_\star + t_2 v) - (t_1 f_n + t_2 v)\|_\infty = \|(t_1 f_\star - t_1 f_n)\|_\infty \leq t_1 \epsilon_n$. Therefore, the sequence of functions $t_1' f_n + t_2 v$ is the guarantor of the denseness of $\mathcal{F}_n$ in $\mathcal{F}$, which justifies the usage of $\mathcal{F}_n$ as a sieve space of $\mathcal{F}$.

We are now ready to state the limiting distribution of $\hat{f}_n$.

Theorem 3.2.1. (Asymptotic distribution of $\hat{f}_n$). *Suppose that $H_n^2 L_d^2 \ln(H_n L_d n)$, $\ln(\ln(n)) + p(n)$, and ϵ_n from Theorem 3.1.1 are all $o(n)$. Then,*

$$U(C', \epsilon_n)^{-1}(\hat{f}_n - f_\star) \xrightarrow{d} h^{(max)} \text{ in } (\mathcal{F}, \|\cdot\|_{L^2(P)})$$

[6]Please see the proof in Appendix B.2 for more details about this lemma and how it is applied.

where $h^{(max)}$ *is the unique arg max of the Gaussian process* $\{\mathbb{G}_f : f \in \mathcal{F}\}$ *with* $\mathbb{E}[\mathbb{G}_f] = 0$ *and* $\text{Cov}(\mathbb{G}_{f_1}, \mathbb{G}_{f_2}) = 4\sigma^2 \mathbb{E}_{\boldsymbol{X}}[f_1(\boldsymbol{X}) f_2(\boldsymbol{X})]$.

Since Theorem 3.2.1 provides the asymptotic distribution of the weighted version of $\hat{f}_n - f_\star$, it enables the construction of test statistic involving $f_\star$. However, since statistic $U(C', \epsilon_n)^{-1}(\hat{f}_n - f_\star)$ does not involve any relation to covariates x_j, it cannot provide any assessment to variable significance tests. Thus, we need to develop a statistic that involve the covariates and does not have explicit expression of $f_\star$. Following HG and (3), the test statistic under consideration is $\mathcal{T}_j[\hat{f}_n] = \mathcal{T}[\hat{f}_n]$, where $\mathcal{T}$ is the functional

$$\mathcal{T}[f_*] = \int_{\mathbb{X}} \left(\frac{\partial f_*(\boldsymbol{x})}{\partial x_j} \right)^2 dP(\boldsymbol{x}). \tag{13}$$

The statistic $\mathcal{T}_j[\hat{f}_n]$ is the quantification of the significance of x_j to the values of $\hat{f}_n$, because it takes the square of the partial derivatives of $\hat{f}_n$ w.r.t. x_j. Since $\hat{f}_n$ is the consistent estimator of f_*, $\mathcal{T}_j[\hat{f}_n]$ can also be seen as the "representative" of $\mathcal{T}_j[f_*]$. Note that $\mathcal{T}_j[f_*]$ cannot be directly calculated because the exact functional form of f_* is unknown. Next, we derive the asymptotic distribution of $\mathcal{T}_j[\hat{f}_n]$.

Theorem 3.2.2. (Asymptotic distribution of $\mathcal{T}_j[\hat{f}_n]$). *Under the conditions of Theorem 3.2.1 and null hypothesis (3), we obtain*

$$U(C', \epsilon_n)^{-2} \mathcal{T}_j[\hat{f}_n] \xrightarrow{d} \mathcal{T}_j[h^{(max)}]$$

and $\mathcal{T}_j[h^{(max)}]$ *is defined similar to (13).*

Note that the randomness of $\mathcal{T}$ comes from the members of $\mathcal{F}$ themselves. Since we work with samples, and $\rho_n^2 \left(\frac{\partial \hat{f}_n}{\partial x_j}(\boldsymbol{X}) \right) = \frac{1}{n} \sum_{i=1}^{n} \left(\frac{\partial \hat{f}_n(\boldsymbol{x_i})}{\partial x_j} \right)^2$ can be seen as empirical functional of $\mathcal{T}_j[\hat{f}_n]$. Thus, it is natural to work with $\rho_n^2 \left(\frac{\partial \hat{f}_n}{\partial x_j}(\boldsymbol{X}) \right)$.

We close this section by providing the asymptotic distribution of $\rho_n^2 \left(\frac{\partial \hat{f}_n}{\partial x_j}(\boldsymbol{X}) \right)$.

Theorem 3.2.3. (Asymptotic distribution of $\rho_n^2 \left(\frac{\partial \hat{f}_n}{\partial x_j}(\boldsymbol{X}) \right)$). *Suppose that all hypotheses of Theorem 3.2.2 are satisfied. Then, we have*

$$U(C', \epsilon_n)^{-2} \rho_n^2 \left(\frac{\partial \hat{f}_n}{\partial x_j}(\boldsymbol{X}) \right) \xrightarrow{d} \mathcal{T}_j[h^{(max)}].$$

3.3 Computing the asymptotic distribution

In this subsection, we discuss a discretization approach to compute the asymptotic distribution of test statistic $U(C', \epsilon_n)^{-2} \rho_n^2 \left(\frac{\partial \hat{f}_n}{\partial x_j}(\boldsymbol{X}) \right)$, i.e., $\mathcal{T}_j[h^{(max)}]$. A discrete approximation of the Gaussian process $\mathbb{G}$ paths can be obtained by sampling a multivariate normal distribution

with: (i) dimension C, where C is the cardinality of ζ-cover $\{f_1, \ldots, f_C\}$ of $\mathcal{F}$; (ii) mean vector $\mathbf{0}$, and; (iii) a covariance matrix whose elements in l^{th} row and k^{th} column are given by $c_{l,k} = \mathbb{E}((f_l f_k)(\boldsymbol{X})), \forall l, k \in \{1, \ldots, C\}$. Because $\forall a > 0$, $\mathbb{G}(h^{(max)}(\phi), \phi) > \mathbb{G}(f(\phi), \phi) \iff a \cdot \mathbb{G}(h^{(max)}(\phi), \phi) > a \cdot \mathbb{G}(f(\phi), \phi)$, the constant $4\sigma^2$ in the covariance matrix can be ignored in the implementation. Note that since computing the exact ζ-cover of $\mathcal{F}$ for a specific value of ζ is challenging, we replace the explicit ζ-cover with an approximate cover.

The discretization approach can be summarized into the following three steps:

Step 1. Approximating ζ-cover of $\mathcal{F}$:

Sample m neural networks, $(f_1, f_2, \cdots, f_m)$, either with the same architecture as the trained model or other types of neural network that can serve as the "correct approximator",[7] by sampling its weight parameters from a Glorot normal distribution – a truncated normal distribution centered at 0 with standard deviation $\sqrt{2/(d+1)}$. This collection of neural network is the approximate ζ-cover of $\mathcal{F}$.

Step 2. Simulate a sample from a multivariate normal and find $\hat{h}^{(max)}$:

Take a sample from a multivariate normal of dimension m which has mean vector $\mathbf{0}$ and covariance matrix with elements in l^{th} row and k^{th} column are approximated by $\hat{c}_{l,k} = \frac{1}{n}\sum_{i=1}^{n}(f_l f_k)(\boldsymbol{x}_i)$ – the empirical covariance or variance of the function $f_l f_k$.

- The index of the maximum value of this multivariate normal corresponds to the one-layer network that serves as an approximate arg max of $\mathbb{G}$, and denote it by $\hat{h}^{(max)}$. This is true as the arg max $h^{(max)}$ of $\mathbb{G}$ satisfies $\mathbb{G}(h^{(max)}(\phi), \phi) > \mathbb{G}(f(\phi), \phi), \forall \phi \in \Omega$, $\forall f \in \mathcal{F}$.

Step 3. Generate one approximate sample of $\mathcal{T}_j[h^{(max)}]$:

Generate one approximate sample of $\mathcal{T}_j[h^{(max)}]$ by taking $\rho_n^2 \left(\frac{\partial \hat{h}^{(max)}}{\partial x_j}(\boldsymbol{X}) \right)$.

Repeating these three steps n_p times gives $\hat{h}_1^{(max)}, \ldots, \hat{h}_{n_p}^{(max)}$ to generate n_p samples of $\mathcal{T}_j[h^{(max)}]$. After simulating n_p samples of the asymptotic distribution $\mathcal{T}_j[h^{(max)}]$, the testing is conducted at $100\alpha\%$ significance level by comparing the values of the test statistics $U(C', \epsilon_n)^{-2}\rho_n^2 \left(\frac{\partial \hat{f}_n}{\partial x_j}(\boldsymbol{X}) \right)$ to the $100(1-\alpha)\%$ quantile of n_p samples of $\mathcal{T}_j[h^{(max)}]$, denote by $q_{(1-\alpha)}^{\mathcal{T}_j[h^{(max)}]}$.

Next, the following standard procedure is performed:

- If $U(C', \epsilon_n)^{-2}\rho_n^2 \left(\frac{\partial \hat{f}_n}{\partial x_j}(\boldsymbol{X}) \right) > q_{(1-\alpha)}^{\mathcal{T}_j[h^{(max)}]}$, then we reject $\mathcal{H}_0$ in favor of $\mathcal{H}_1$ at $100\alpha\%$ level.

[7] Being the "correct approximator" means the bias of the approximation of each member of the related function space decreases as the number of hidden nodes in each of the neural networks composing the approximate ζ-cover is increased.

- Otherwise, we fail to reject $\mathcal{H}_0$.

Note that for an iteration of the discretization, one member of the the $\xi-$cover is used as the argmax of the Gaussian process to derive the following sample

$$\rho_n^2\left(\frac{\partial \hat{h}^{(max)}}{\partial x_j}(\boldsymbol{X})\right) = \frac{1}{n}\sum_{i=1}^{n}\left(\frac{\partial \hat{h}^{(max)}(\boldsymbol{x_i})}{\partial x_j}\right)^2 \tag{14}$$

where $\boldsymbol{X}$ is the training sample. Given a sample test, as we consider n_p iteration, we derive n_p samples per predictor/covariate using (14). For each predictor, we have (possibly) n_p different samples which is used to derive the p-values by calculating sample quantiles. Hence, in each iteration $\hat{h}^{(max)}$ is the same across all predictors, however, the sensitivity of it respect to each predictor can be different through the partial derivative in (14).

The discretization of the asymptotic distribution has three main adjustable parameters: (i) the architecture (the number of hidden layers and nodes) of the neural networks used for approximating ζ-cover of $\mathcal{F}$; (ii) the dimension of the Gaussian process, m, and; (iii) the number of samples for the asymptotic distribution, n_p. Note that the values of m, n_p and the number of hidden layers and nodes of the approximating neural network must be "in tandem" with each other. Increasing n_p without changing the the architecture and/or m can give redundant samples of $\mathcal{T}_j\left[h^{(max)}\right]$ whose values are not close to its true values. Increasing the network's hidden layer and/or hidden node numbers without increasing n_p can result in unsatisfactory variation of the samples of $\mathcal{T}_j\left[h^{(max)}\right]$, and we have the same problem if m is increased while the other parameters are fixed. To ensure that these parameters are proportionally adjusted, we consider a combination of these parameters.

For additional clarification in simpler terms, you can refer to Section F in the Online Appendix.

4 Monte Carlo simulations

In this section, we describe the procedure for generating the data in the Monte Carlo study. We start by explaining the steps involved in data generation. We then detail the calculation of test statistics and the estimation of the quantiles of the asymptotic distribution. Finally, we assess the ability of the proposed significance test in identifying true covariates.

4.1 The data generating process

To effectively evaluate the proposed significance test in a more realistic setting, the simulated data should closely resemble real-world financial data, such as stock returns and characteristics. To achieve this, we adopt a similar approach to GKX by simulating data in a scenario that includes: (i) a high-dimensional set of covariates relative to the sample size; (ii) covariates that exhibit strong temporal dependencies and high correlation, and; (iii) a relatively low signal-to-noise ratio. These conditions provide a suitable framework to evaluate the performance of the

proposed test under realistic scenarios, in terms of its size and power.

We simulate the firm i's characteristic j at time t, $c_{i,j,t}$, as follows:

$$\bar{c}_{i,j,t} = \varrho_j \bar{c}_{i,j,t-1} + \delta_{i,j,t}, \;\; i = 1,2,..,N, \;\; j = 1,2,...,P_c, \; t = 1,2,...,T$$
$$c_{i,j,t} = \frac{2}{N+1} CSrank(\bar{c}_{i,j,t}) - 1$$

where N, P_c, and T are the number of firms, characteristics, and time periods, respectively. The parameter ϱ_j is drawn from a uniform distribution on the interval $[0.9, 1]$ and $\delta_{i,j,t}$ is drawn from a normal distribution with mean 0 and standard deviation $\sqrt{1-\varrho_j^2}$. The *CSrank* is a cross-sectional ranking function that ranks each characteristic across all firms at a given time period. We then simulate a common factor at time t, m_t, as follows:

$$m_t = 0.95 m_{t-1} + \mu_t$$

where $\mu_t \sim \mathcal{N}(0, 1 - 0.95^2)$.

We finally simulate excess returns $r_{i,t+1}$ for firm i at time t from the following latent three-factor model:

$$r_{i,t+1} = \varphi_\star(\boldsymbol{x_{i,t}}) + \boldsymbol{\eta_{i,t}} \boldsymbol{\nu_{t+1}} + \varsigma_{i,t+1}, \text{ where } \boldsymbol{x_{i,t}} = [1, m_t] \otimes \boldsymbol{c_{i,t}}$$
$$\varphi_\star(\boldsymbol{x_{i,t}}) = 0.02(c_{i,1,t} + c_{i,2,t} + c_{i,3,t} \times m_t)$$

where $\boldsymbol{c_{i,t}} = [c_{i,1,t}, c_{i,2,t}, ..., c_{i,P_c,t}]$ is a vector of firm i's characteristics at time t, $\boldsymbol{x_{i,t}} = [c_{i,1,t}, c_{i,2,t}, ..., c_{i,P_c,t}, c_{i,1,t} \times m_t, c_{i,2,t} \times m_t, ..., c_{i,P_c,t} \times m_t]$ is a vector of covariates for firm i at time t, $\boldsymbol{\eta_{i,t}} = [c_{i,1,t}, c_{i,2,t}, c_{i,3,t}]$ is a vector of three firm i's characteristics at time t, $\boldsymbol{\nu_{t+1}} = [\nu_{1,t+1}, \nu_{2,t+1}, \nu_{3,t+1}]'$ is a vector of factors at time t and it follows the distribution $\mathcal{N}(0, 0.005^2 \times \mathbb{1}_3)$, and $\varsigma_{i,t+1} \sim t_5(0, 0.05^2)$ is the idiosyncratic error for firm i at time t.[8]

In order to investigate how the number of covariates and sample size affect the performance of the significance test, we consider two different values for P_c, $P_c = 50$ and $P_c = 100$, and three different pairs of (N, T), which are (200,180), (200,120), and (100,180), resulting in a total of 36,000, 24,000, and 18,000 observations, respectively. We use a sample splitting approach, where for each Monte Carlo repetition, we divide the data into three sets: a training set, a validation set, and a testing set, each consisting of a third of the total number of months. We simulate covariates and excess returns for each value of P_c and each pair of (N, T) for 100 Monte Carlo repetitions.

4.2 Fitting Neural Networks

Determining the optimal architecture for a neural network can be a time-consuming and computationally intensive task, requiring the evaluation of numerous potential configurations

[8]We incorporate dependence of the error term $e_{i,t+1}$ on certain covariates in the simulation to evaluate the performance of the proposed significance test in a more realistic scenario, where the assumption of independence between errors and covariates may not hold. To test the performance under the assumption of independence, one can set $\boldsymbol{\nu_{t+1}}$ to 0 in the simulation, which may lead to improved performance of the proposed significance test.

through cross-validation. However, for our purpose of performing a significance test using a neural network with a satisfactory level of predictive performance, it is not necessary to exhaustively search for the best possible architecture.

We consider five Neural Network (NN) models with varying architectures. The number of hidden layers is increased from 1 to 5 and the number of neurons in each hidden layer is decreased following the pyramid rule (Masters, 1993). For example, NN1 has a single hidden layer with 32 nodes, NN3 has three hidden layers with 32, 16, and 8 neurons respectively, and NN5 has five hidden layers with 32, 16, 8, 4, and 2 neurons respectively. For each NN model, the number of neurons in the input layer is the same as the number of covariates and the output layer has only one neuron. The activation function in each hidden layer is hyperbolic tangent (tanh) and the activation function in the output layer is a linear function.

We adopt a strategy to fit NNs that includes the use of the Adam optimizer and four regularization techniques to avoid overfitting: l_1 penalty, early stopping, batch normalization, and ensembles. To ensure stability and reduce variance, we use an ensemble approach where we initialize 10 models with the same NN architecture but from different random seeds, and fit these models to the same training and validation sample. We then construct the final predictions by averaging the predictions from these 10 fitted models (Dietterich, 2000). The learning rate for the Adam optimizer is set to 0.001, and the batch size is set to 10,000 for all NN models. The l_1 penalty is tuned within the range [0.0001, 0.003], the number of epochs is in the range [350, 900], and the patience range is determined by the value of the number of epochs.

We measure the predictive accuracy of the neural network models using in-sample (IS) and out-of-sample (OOS) R^2 metrics. For each Monte Carlo repetition, we calculate the in-sample R^2 as:

$$R^2_{IS} = 1 - \frac{\sum_{(i,t)\in\Lambda_1}(r_{i,t+1} - \hat{r}_{i,t+1})^2}{\sum_{(i,t)\in\Lambda_1}(r_{i,t+1} - \bar{r})^2}$$

where $\hat{r}_{i,t+1}$ is the predicted excess return for firm i at time $t+1$. Λ_1 represents the training sample, and $\bar{r}$ is the mean of excess return in the training sample. Similarly, the out-of-sample R^2 is calculated as:

$$R^2_{OOS} = 1 - \frac{\sum_{(i,t)\in\Lambda_3}(r_{i,t+1} - \hat{r}_{i,t+1})^2}{\sum_{(i,t)\in\Lambda_3}(r_{i,t+1} - \bar{r})^2}$$

where Λ_3 represents the testing sample. We then average the 100 R^2_{IS}s and the 100 R^2_{OOS}s to measure the the overall IS and OOS predictive performance, respectively.

In Table 1, we present the overall IS and OOS R^2 for each NN model and the Oracle model (a pooled linear regression on the true covariates) when $P_c = 50$ and $P_c = 100$. The results indicate that: (i) as the number of hidden layers in the NN models increases, the OOS R^2 decreases, indicating that shallow NNs perform better than deep NNs;[9] (ii) the IS R^2 is higher when

[9] As reported by GKX, shallow NNs with fewer hidden layers tend to perform better in terms of predictive perfor-

$P_c = 100$ and the OOS R^2 is higher when $P_c = 50$ for NN models. This suggests that the IS R^2 increases with P_c and the OOS R^2 decreases with P_c, and; (iii) for the $(N = 200, T = 180)$ case, all the NN models have a higher IS R^2 than the Oracle model, while for $(N = 100, T = 180)$ and $(N = 200, T = 120)$, only a few NN models have a higher IS R^2 than the Oracle model. These results are consistent with a smaller OOS R^2 and a higher IS R^2 in the Oracle model compared to GKX.

Insert Table 1 about here

4.3 Variable significance tests

In this section, we first present the summary statistics of the test statistics for several covariates, obtained from the analysis of five neural networks. We then provide a graphical representation of the simulated test statistic samples and the estimated quantiles, $\mathcal{T}_j[h^{(max)}]$. Lastly, we present the results of our significance test at different levels of statistical significance, under various simulation and discretization settings.

4.3.1 Test statistic

We compute test statistic for covariate x_j as the mean of squared partial derivatives of the fitted model $\hat{\varphi}_n$ to that covariate within the training sample Λ_1

$$\rho_n^2\left(\frac{\partial \hat{\varphi}_n}{\partial x_j}(\boldsymbol{X})\right) = \frac{1}{n}\sum_{(i,t)\in\Lambda_1}\left(\frac{\partial \hat{\varphi}_n(\boldsymbol{x_{i,t}})}{\partial x_j}\right)^2$$

where n is the number of observations in the training sample.

Tables A2 and A3 present the summary statistics of the scaled test statistics for various covariates for five different neural networks, in the simulation scenario with $(N = 200, T = 180)$ and varying levels of P_c, where P_c is 50 and 100, respectively. As expected, the most influential covariates are found to be $c_{i,1,t}$, $c_{i,2,t}$, and $c_{i,3,t} \times m_t$, while the second most impactful are $c_{i,1,t} \times m_t$, $c_{i,2,t} \times m_t$, and $c_{i,3,t}$, which are related to the true covariates and contain redundant information for excess return prediction. On the other hand, the remaining $2P_c - 6$ covariates, which do not carry any useful information for forecasting excess returns, have test statistics close to zero. This result holds true across all five NN models and is consistent for the two other simulation scenarios of $(N = 200, T = 120)$ and $(N = 100, T = 180)$.

4.3.2 Quantile estimation

We use the discretization method outlined in Section 3.3 to first generate 1,000 samples of $\mathcal{T}_j[h^{(max)}]$ for each of the 100 Monte Carlo repetitions, given the simulation parameters (N, T)

mance compared to deeper NNs. However, this result is based on a specific set of hyperparameter values and a limited number of architectures considered. Further tuning and exploring different architectures could potentially yield different results, but this is beyond the scope of our current study.

and discretization parameters $(m, \#layers, n_p)$. Each simulation of $\mathcal{T}_j[h^{(max)}]$ for all covariates takes about 10 to 15 minutes on a standard computer, with the time varying depending on the value of P_c (which is either 50 or 100). To reduce the computing time, we can run the simulations in parallel. For example, we utilized 400 CPUs from cluster machines, which reduced the overall simulation time to 2 to 3 days for a given set of simulation and discretization parameters.

4.3.3 Aggregated results of significance tests

To get a comprehensive insight into the dynamics of the asymptotic distribution and evaluate the proposed test, we first conduct the significance test for each covariate by averaging the test statistics and combining the $\mathcal{T}_j[h^{(max)}]$ samples from all Monte Carlo repetitions. It is possible to generate a different set of neural networks for each Monte Carlo repetition at each iteration of the discretization process. However, in this paper, we generate the same set of neural networks for each Monte Carlo repetition. For instance, at iteration it, we generate the same m neural networks, $(f_1^{it}, f_2^{it}, ..., f_m^{it})$, across all Monte Carlo repetitions. Since all Monte Carlo repetitions are generated from the same data generating process, and given a large enough sample size, the empirical covariance $\hat{c}_{l,k} = \frac{1}{n}\sum_{i=1}^{n}(f_l^{it} f_k^{it})(\boldsymbol{x}_i)$ is highly consistent across repetitions. As a result, the simulated $\hat{h}_{it}^{(max)}$ values are also consistent across repetitions, as is the case for other iterations. This consistency enables us to average the test statistics for all Monte Carlo repetitions and perform the significance test on the combined sample of the $\mathcal{T}_j[h^{(max)}]$.

Figure 2 displays the results of the significance test when the simulation parameters are set to $(N = 200, T = 180)$ and the discretization parameters are set to $(m = 500, one\ layer, n_p = 1,000)$ and $P_c = 100$. The distribution of the aggregated $\mathcal{T}_j[h^{(max)}]$ samples shows a heavy right tail that is similar to the χ^2 distribution for all covariates. In line with the findings in Table A3, true covariates exhibit significantly higher test statistics compared to the other covariates in all five models. The true covariates also exhibit higher test statistics in NN1, NN2, and NN3 compared to NN4 and NN5, possibly because shallow neural networks perform better in prediction than deep neural networks. For all five NN models, the test statistics of true covariates are above the 99% quantile line, while the test statistics of other covariates are below the 90% quantile line, indicating that true covariates are significant at the 1% level, while other covariates are not significant even at the 10% level. These results suggest that, regardless of the neural network architecture, the proposed significance test is able to effectively distinguish true covariates from a large number of covariates, e.g., (or "factor zoo", that may include strongly correlated variables and multiple sources of noise.

Insert Figure 2 about here

4.3.4 Significance test results from individual Monte Carlo runs

We conduct 100 significance tests for each covariate by performing the significance test for each Monte Carlo repetition individually. The variable significance frequency for each covariate is calculated as the proportion of times that the covariate is identified as significant in the 100 significance tests. Table 2 presents the variable significance frequencies at three levels of significance, 10%, 5%, and 1%, across five different discretization settings. Panel A shows the results for the discretization choice of $(m = 500, one\ layer, n_p = 1,000)$. At the 10% significance level, true covariates are identified as significant in 76%-88% of the tests, while only 17%-26% of tests identify other covariates as significant. At the 5% level, the variable significance frequency decreases slightly to 69%-82% for true covariates and 12%-20% for other covariates. Even at the 1% level, the variable significance frequency remains above 58% for true covariates and below 13% for other covariates. Although the variable significance frequency decreases as the significance level decreases, the significance test still performs well: at any of the three significance levels, the variable significance frequency is consistently higher than 58% for true covariates and lower than 27% for other covariates. Panels B and C present the variable significance frequencies for the discretization choice of $(m = 300, one\ layer, n_p = 1,000)$ and $(m = 500, one\ layer, n_p = 2,000)$, respectively. The results are similar to those in Panel A, indicating that the value of m or n_p has little impact on the performance of the significance test.

Insert Table 2 about here

The results of the variable significance frequency for the discretization choices $(m = 500, two\ layers, n_p = 1,000)$ and $(m = 500, three\ layers, n_p = 1,000)$ are shown in Panels D and E, respectively. While true covariates still have a higher variable significance frequency than other covariates, there are some differences from the results in the previous three panels. Firstly, the overall variable significance frequency decreases. For example, when $P_c = 100$, the variable significance frequency of $c_{i,1,t}$ at 10% level drops from above 78% in Panel A to below 73% in both Panel D and E. Secondly, in contrast to the previous three panels, the variable significance frequency of true covariates does not consistently stay above 50%. For instance, when $P_c = 100$, the variable significance frequency of $c_{i,3,t} \times m_t$ at the 1% level is less than 50% in NN4 and NN5 in both Panel D and E. This suggests that the performance of the significance test deteriorates when using neural networks with two or three hidden layers for the discretization. This might be due to the fact that using more hidden layers for approximating the ζ-cover of $\mathcal{F}$ leads to a larger set of weights to estimate, resulting in increased model uncertainty and thus greater variation in the $\mathcal{T}_j[h^{(max)}]$ samples.

In addition to assessing the statistical significance of the covariates at specified significance levels, we also calculate their p-values based on the $\rho_n^2\left(\frac{\partial \hat{\varphi}_n}{\partial x_j}(\boldsymbol{X})\right)$ and the 1,000 $\mathcal{T}_j[h^{(max)}]$ sam-

ples from each Monte Carlo repetition. As a result, each covariate has a total of 100 p-values. Figure 3 displays the distribution of the 100 p-values for the simulation choice $(N = 200, T = 180)$ and the discretization choice $(m = 500, one\ layer, n_p = 1,000)$. Panel (a) shows the p-value distribution for $P_c = 50$. For all five neural network models, the median p-values of the true covariates are smaller than 0.003, while the median p-values of the other covariates are larger than 0.3.[10] This indicates that at least half of the significance tests indicate that the true covariates are highly statistically significant, for instance, at the 0.3% level, and that the other covariates are not statistically significant even at the most relaxed level, the 10% level. This trend also holds in panel (b) with $P_c = 100$, suggesting that increasing the number of covariates does not diminish the strength of the significance test.

Insert Figure 3 about here

4.3.5 Robustness and the multiple comparisons

In this section, we examine the robustness of the proposed test to variations in the parameters of the Monte Carlo simulation. Additionally, we examine the impact of multiple comparison adjustments, such as the Bonferroni correction, on the conclusions drawn about various variables.

The results of Figure A2, for the simulation choice $(N = 200, T = 120)$ and $(N = 100, T = 180)$, are in line with those of Figure 3 (a). The median p-values of true covariates indicate that at least half of the significance tests show that they are statistically significant (at least at the 10% level), while the other covariates are not. However, there is a noticeable trend that the median p-values of true covariates decrease as the sample size increases. For instance, when comparing NN5 in Panel (b) and Panel (a), the median p-values for true covariates drop from between 0.053 and 0.084 to between 0.019 and 0.047. This reduction in p-values is more pronounced in Figure 3 (a), where the median p-values for true covariates are 0.000. These results suggest that the power of the significance test increases with sample size, as a larger sample size allows for a better estimation of the covariance matrix and a more accurate approximation of the $\zeta-$cover of $\mathcal{F}$, leading to improved performance of the significance test.

Figure A3 presents the results of the significance test for four different discretization choices: $(m = 500, one\ layer, n_p = 2,000)$, $(m = 300, one\ layer, n_p = 1,000)$, $(m = 500, two\ layers, n_p = 1,000)$ and $(m = 500, three\ layers, n_p = 1,000)$ in Panel (a), (b), (c), and (d), respectively. The results are consistent with those in Figure 3 (b), with at least half of the significance tests indicating that true covariates are statistically significant (at a 10% level or lower) and the other covariates being not statistically significant. The median p-values for

[10] The median p-values of the remaining covariates have all been rounded to 1.000, so they are not presented here.

true covariates remain largely unchanged with changes in the values of m or n_p, but tend to increase when the neural network used in discretization has more hidden layers. For instance, in NN4 and NN5, the median p-values for $c_{i,3,t} \times m_t$ are 0.001 in Panel (a) and (b), 0.018 in Panel (c), and around 0.03 in Panel (d). These results are consistent with Table 2, which suggests that the performance of the significance test is not greatly impacted by the values of m or n_p, and that the best results are achieved with a neural network with one hidden layer for discretization.

We assess the robustness of our proposed test under multiple comparison scenarios. To account for potential multiple comparison problems, we use the conservative Bonferroni correction. The results of this correction are shown in Table A4, which reports the variable selection frequencies after the correction has been applied, for the simulation choice $(N = 200, T = 180)$ and $P_c = 100$.

Compared to the results shown in Table 2, the variable selection frequency decreases in all panels after the correction has been applied. This is expected, as the correction is designed to account for multiple comparisons, and the significance level is adjusted to be very strict, at the 0.0025% level. Nevertheless, the significance test still performs well for NN1, NN2, and NN3, as the variable selection frequency for true covariates remains above 50% in most cases.

In conclusion, the results of the multiple comparison correction validate the power of the significance test in identifying true covariates in shallow neural networks.

5 An empirical analysis of U.S. equities

In this section, we put the proposed variable significance test to use by applying it to a comprehensive set of factors, or predictors, that have been previously considered in the literature (Green et al., 2017; Kelly et al., 2019; Freyberger et al., 2020; Gu et al., 2020). Our analysis begins with a examination of the out-of-sample performance of multi-layer perceptrons (MLPs) in assessing risk premia. We then use the proposed test to identify the predictors that are statistically significant.

5.1 Data

We obtained monthly total individual equity returns from the Center for Research in Security Prices (CRSP) for firms listed on the NYSE, AMEX, and NASDAQ. The sample period covers 53 years, from January 1, 1967 to December 31, 2019. This start date is similar to the one used in the study by Chen et al. (2023).

To measure risk premia, we used the Treasury-bill rate as an approximation of the risk-free rate and calculated the monthly excess returns.[11] We divided the data into three sections: 20 years of training data (1967-1986), 5 years of validation data (1987-1991), and 28 years of out-of-

[11]The monthly Treasury-bill rate is from Amit Goyal's Web site.

sample testing data (1992-2019). Our sample includes 26,300 stocks, with an average of around 4,400 stocks per month.

To build a collection of stock-level predictive characteristics, we followed the literature on cross-section of stock returns, such as Green et al. (2017) and Kelly et al. (2019), among others. We initially constructed 94 stock-level predictive characteristics from variables in Compustat and CRSP, as described in detail in Table A5.[12] However, not all stocks had complete information for all characteristics in each month, so we only included stocks with all characteristic information available in a given month.[13] This resulted in dropping 16 characteristics and reducing our sample to 4,826 stocks with an average of around 814 stocks per month. For each characteristic in a given month, we cross-sectionally rank the values of that characteristic and map these ranks into the $[-0.5, 0.5]$ interval similarly to Kelly et al. (2019).

In addition to stock-level characteristics, we also constructed 8 macroeconomic predictors based on Welch and Goyal (2008), including the dividend-price ratio (dp), earnings-price ratio (ep), book-to-market ratio (bm), net equity expansion (ntis), Treasury-bill rate (tbl), term spread (tms), default spread (dfy), and stock variance (svar). We transformed these macroeconomic predictors to remove non-stationarity, as described in McCracken and Ng (2016), and normalized them using the mean and standard deviation calculated from the training data. The transformations and normalizations are described in detail in Table A6.

As an illustration, we show the time series (top panel), transformation (middel panel), and normalization (bottom panel) of three macroeconomic predictors in Figure A4. A visual inspection reveals that neither of time series are stationary while applying the standard transformations solve non-stationarity in these predictors. Note that neither the firm characteristics nor the macroeconomic predictors have to be independent and identically distributed (i.i.d.), as we have not imposed this assumption for the predictors.

5.2 Equity premium prediction: the overarching model

The equity premium, or excess return, is calculated as the difference between the equity returns on a given asset i at time $t+1$, represented as $r_{i,t+1}$, and the risk-free rate at time $t+1$, represented as r^f_{t+1}. The equation is expressed as:

$$\text{Equity Premium(t)} = r^e_{i,t+1} = r_{i,t+1} - r^f_{t+1}.$$

Excess returns are modeled as:

$$r^e_{i,t+1} = \mathbb{E}_t[r^e_{i,t+1}] + \varepsilon_{i,t+1} = f_\star(\boldsymbol{x}_{i,t}) + \varepsilon_{i,t+1}$$

where $f_\star$ is a multivariate function that depends on the feature variables $\boldsymbol{x}_{i,t}$, and $\varepsilon_{i,t+1}$ repre-

[12] We construct these characteristics based on Green et al. (2017). We adapt the SAS code available from Jeremiah Green's Web site and extend the sample to December 31, 2019.

[13] To predict the return at month $t+1$, we calculate the monthly characteristics at the end of month t, assuming that monthly accounting data is available at that time. We calculate the annually characteristics at the end of month $t-6$, assuming that the annually data is released at least six months before the end of month t.

sents the error term. The goal is to approximate the $f_\star$ function using a multi-layer perceptron (MLP) that maximizes the out-of-sample explanatory power for the realized excess returns.

To predict the excess return of stock i in the next month $t+1$, two stock-level covariates are constructed that incorporate the interaction between stock characteristics and macroeconomic predictors. The first covariate is represented as:

$$\boldsymbol{x}_{i,t} = cm_{i,t} = [c_{i,t}, m_t]$$

where $c_{i,t}$ is a matrix of P_c characteristics for each stock i, and m_t is a vector of P_m macroeconomic predictors. The second covariate is represented as:

$$\boldsymbol{x}_{i,t} = z_{i,t} = c_{i,t} \otimes [1, m_t]$$

where $z_{i,t}$ is a vector of $P = P_c P_m$ features for predicting individual stock returns. The total number of covariates for $cm_{i,t}$ and $z_{i,t}$ are 86 and 702, respectively. The proposed method can be applied to other neural network architectures similarly to prior research (Bianchi et al., 2021). For simplicity, the two sets of predictors will be referred to as CpM (characteristics plus macroeconomics) and CtM (characteristics times macroeconomics) in the remainder of the discussion.

It is worth noting that prior research has shown that neural networks are among the best performing algorithms for predicting equity risk premiums. The aim of the present analysis is to complement these findings by identifying the statistically significant covariates and "opening the black-box". Unlike prior research, the proposed method trains all models only once for the entire test sample, rather than recursively refitting models (Bianchi et al., 2021; Gu et al., 2020) each month or year, due to the high computational cost of training neural networks. However, if one were to use the recursively refitting methodology, separate tests would need to be conducted for each month or year, as the proposed test can be applied to a trained ML

5.3 Performance evaluation

The performance of equity premium forecasts is evaluated using the explained variation in individual asset returns, denoted as EV, which is calculated as follows:

$$EV = 1 - \frac{\frac{1}{T}\sum_{t=1}^{T}\frac{1}{N_t}\sum_{i=1}^{N_t}(r^e_{i,t+1} - \hat{r}^e_{i,t+1})^2}{\frac{1}{T}\sum_{t=1}^{T}\frac{1}{N_t}\sum_{i=1}^{N_t}(r^e_{i,t+1})^2}$$

where T is the number of months in the sample and N_t is the number of available assets in month t. $\hat{r}^e_{i,t+1}$ is the predicted excess return for asset i at time $t+1$. This prediction is the average of the predicted excess returns from nine independently trained models with the same neural network architecture but different initializations, i.e., $\hat{r}^e_{i,t+1} = \hat{f}_n(\boldsymbol{x_{i,t}}) = \frac{1}{9}\sum_{k=1}^{9}\hat{f}_n{}^k(\boldsymbol{x_{i,t}})$. The formula and nine assesmbles are based on the methodology from Chen et al. (2023).

In the calculation of the EV metric, the denominator is the sum of squared excess returns

without demeaning, which uses the naive zero forecast. This approach was adopted to counteract the potential for noisy estimation of historical mean stock returns. By benchmarking the EV against a naive forecast, the bar for acceptable prediction performance is not artificially lowered. Additionally, similar to Chen et al. (2023), other performance evaluation metrics such as the Sharp ratio and cross-sectional R^2 were calculated, but are not reported in this paper due to space constraints. These metrics are available upon request.

5.4 Model uncertainty

Table 3 reports monthly out-of-sample prediction performance in terms of EV measure for both CpM and CtM predictors for three sets of assets. We compare five different models for each universe of predictors, CpM and CtM, totalling 10 models.

Insert Table 3 about here

Table 3 highlights two key insights. Firstly, all the models surpass the performance of the naive zero forecast in terms of the EV metric. The minimum monthly EV is around 9.9% (for the CtM predictors) while the maximum is approximately 12.2% (for the CpM predictors). This outcome aligns with the results reported in GKX and is expected. It confirms the superiority of the neural network models compared to the naive zero forecast in predicting excess returns. The neural network's ability to capture complex and non-linear relationships among the predictors is a significant advantage over the simplistic naive zero forecast.

Secondly, the shallow network on the CpM predictors effectively captures interaction effects among predictors. Increasing the complexity of the model, for instance by adding more nodes or layers to the MLPs, or by incorporating more predictors, can result in a more complex model. As seen in Table 3, the shallowest network (NN1) on the CpM predictors performs comparably to the deepest network (NN5) on the CtM predictors. Additionally, NN models on the CpM predictors outperform their counterparts (with the same architecture) on the CtM, indicating that the neural network architecture effectively captures interaction effects among predictors.

The second and third rows of Table 3 break out predictability for large stocks (the top-100 stocks by market equity each month) and small stocks (the bottom-100 stocks each month). This is based on the full estimated model (using all stocks in training), but focuses on fits of trained models among the two subsamples. Similarly to GKX, neural networks are successful among large stocks, with EV ranging from 23.1% to 27.8%.

Table 3 provides a quantitative comparison of models' predictive performance, while Table 4 offers the statistical significance of differences among models at the monthly frequency. It reports Diebold-Mariano test statistics for pairwise comparisons of a column model versus a row model.[14] Similar to GKX, the null hypothesis is that there is no difference between models.

[14]Detailed information about the Diebold-Mariano test and its modification can be found in Section 1.8 of GKX.

Positive numbers indicate the column model outperforms the row model.

Insert Table 4 about here

The results from Table 3 indicate that deeper neural networks outperform shallower ones. For instance, the NN5 model shows better performance compared to NN1 to NN4 for both sets of predictors. It's important to note that the study follows the approach of previous research, such as Kelly et al. (2019), Freyberger et al. (2020), and Chen et al. (2023), in that it only includes stocks with complete characteristic information for a given month and does not replace missing values with cross-sectional medians or means. Additionally, the study deviates from the rolling or expanding window approach and instead uses a single trained model to make predictions for the entire out-of-sample testing period, similar to Chen et al. (2023).

5.5 Which predictors matter?

In this section, we focus on our primary objective of identifying statistically significant predictors. To ensure robust results, we follow the approach of GKX by also reporting LOOM. To rank the covariates in terms of their influence, we calculate their values within each model and then average the rankings across the five NN models to obtain an overall ranking for each metric.

The hypothesis test involves two important components: the test statistic, also referred to as the t-statistic, and the p-value. The t-statistic measures the sensitivity of the model to a particular predictor and provides insight into its significance, while the p-value indicates the statistical significance of the t-statistic

Following Theorem 3.2.3, the t-statistic is given by

$$\rho_n^2\left(\frac{\partial \hat{f}_n}{\partial x_j}(\boldsymbol{X})\right) = \frac{1}{n}\sum_{i=1}^{n}\left(\frac{\partial \hat{f}_n(\boldsymbol{x_i})}{\partial x_j}\right)^2$$

where x_j is the jth covariate, $\boldsymbol{x}_i$ is the ith sample in the training set, and n is the number of training samples, $\hat{f}_n$ is the trained NN model. Consistent with the theoretical findings, Theorems 3.1.1 and 3.2.3, we scale all t-statistics within a model with the same scaling number, however, we use different scaling factor for each NN model.

To obtain a p-value, we employ the discretization method described in Subsection 3.3. The method involves setting $m = 100$, $n_p = 10,000$, and randomly selecting 1,000 stocks from the entire training sample in each iteration. The selection of only 1,000 stocks is done to reduce computational cost. However, we also evaluated the results on the entire sample and with different selection methods, such as randomly selecting 10 stocks from each percentile based on a stock characteristic, such as market value. The results were consistent and comparable to those obtained from randomly selecting 1,000 stocks from the entire training sample.

As a demonstration, Figure A5 shows the approximation of asymptotic distributions for two predictors drawn from the CpM universe. The top and bottom left (right) panels of Figure

A5 display the probability density function (pdf) and cumulative distribution function (cdf), respectively, of the approximated asymptotic distribution of the 1-month momentum (R&D increase) predictor. The figure highlights several key observations. First, the asymptotic distributions of the two predictors are distinct, as expected based on (14) and its accompanying discussion. Second, it is evident that the 1-month momentum predictor is significant at the 5% level (its scale t-statistic exceeds the 95% quantile), while the R&D increase predictor is not significant (its scaled test statistic is below the 95% quantile). Additionally, Figure A6 shows the approximated asymptotic distributions for two predictors from the CtM universe. A comparison of the asymptotic distributions of the CpM and CtM predictors reveals that the pdfs of CpM predictors have higher kurtosis and smaller variance. Furthermore, all asymptotic distributions exhibit positive skewness, which is consistent with the one-tailed hypothesis.[15]

Tables 5 and A7 present the results of the significance tests for all stock characteristics and NN models, where the universes of predictors are the CpM and CtM, respectively. The characteristics are sorted based on their t-statistic values within each model, and an overall influence in terms of test statistic is calculated by averaging the predictors' rankings across the five NN models. The scaled test statistics for all models are listed in front of each predictor, and their corresponding p-values are shown underneath. Red color is used for predictors with a p-value less than 10%, while green color is used for predictors with a p-value greater than 10%. The darkness of the red or green color indicates the level of significance.[16].

Insert Tables 5 about here

Table 5 provides several significant insights. Firstly, the top-5 most influential predictors, based on their test statistic values, are 1-month momentum (mom1m), volatility of liquidity (std−turn), share turnover (turn), dollar trading volume (dolvol), and change in 6-month momentum (chmom). On the other hand, the bottom-5 predictors are depreciation (depr), industry-adjusted change in profit margin (chpmia), industry-adjusted change in capital expenditures (pchcapx−ia), R&D increase (rd), and industry sales concentration (herf). Secondly, it is worth to note that only a limited number of characteristics are statistically significant at 99% level across all NN models, with only a few of the 78 characteristics being significant. Lastly, it is worth mentioning that all macroeconomic predictors are statistically insignificant, although their results have not been reported in the interest of space.

Comparing Tables 5 and A7 reveals that the relative importance of characteristics is almost identical across the two universes of predictors. Given the strong out-of-sample performance

[15]Tables A10, A11, and A13 present the 95% quantile of 78 characteristics for all models on the CpM, CtM, and top-34 common interactions that exist in each NN model's top 100 interactions, respectively.
[16]For additional information regarding the variable significance test, all p-values are reported. Alternatively, in line with conventional asset pricing literature, one can use *, **, and *** to denote 10%, 5%, and 1% significant levels.

of the NN models on both CpM and CtM predictors, it can be concluded that the NN models effectively capture non-linear interactions among predictors in predicting individual stock excess returns, which is consistent across all NN models.

Tables A8 and A9 provide the variable significant test results for the top-34 and bottom-48 common interaction predictors, respectively, of NN models with the universe of predictors being CtM. An inspection of these tables shows two key findings. Firstly, the most influential common interaction predictors are based on the product of a characteristic with either the dividend-price ratio (dp) or earning-price ratio (ep). Secondly, in contrast to the CpM universe, there are more statistically significant predictors at the 99% level for the CtM universe, which is not surprising given the higher number of predictors (702) in the case of CtM.

The empirical study section concludes with a robustness check of the variable significant test by considering the leave-one-out-metric (LOOM). This involved setting the values of a given predictor to zero to generate a new training sample, and calculating the LOOM as the absolute difference between the expected value based on all covariates and the expected value based on all covariates except the predictor being tested. The results of the LOOM were generally positive and a large reduction indicated the large influence of the predictor. Figures showed the overall rankings of predictors based on the sum of their ranks across all models, with the most influential predictors on the top and the least influential on the bottom. The comparison between test statistic and LOOM showed that the models agreed on the most influential predictors.[17]

6 Conclusion

This study has proposed a novel variable significance test for evaluating the statistical significance of input variables in multi-layer perceptron (MLP) regression models. By providing a robust and reliable method for variable selection, this test can help practitioners and researchers better understand the underlying relationships in their data and make more informed decisions based on their findings.

The theoretical foundations of the test were established through consistency results and estimation rate analysis using the sieves method. This provides a solid theoretical basis for the test's performance and ensures that the test's results are both accurate and reliable. Additionally, an extensive Monte Carlo simulation was conducted to validate the test's performance in complex and realistic settings. The simulation results showed that the test has a high power and low rate of false positives, making it a powerful tool for detecting true effects in data.

The test was also applied to identify the most influential predictors of equity risk premiums, with results indicating that only a small number of characteristics have statistical significance

[17] For further insight on the LOOM, refer to Figures A8 to A11.

and all macroeconomic predictors are insignificant at the 1% level. This highlights the importance of a robust variable selection method for identifying true effects in data and can have important implications for practitioners and researchers working in the field of finance.

The study opens up exciting and unexplored avenues for further research. One possibility is to extend the variable significance test to other classes of neural networks such as long short-term memory (LSTM) networks, which have been shown to be useful for financial applications, such as capturing the time-variation of the stochastic discount factor with a recurrent LSTM network (Chen et al., 2023). To do so, one might leverage on Chen and Shen (1998) and Chen and White (1999) on sieves for dependent data. Furthermore, this study focused on evaluating the statistical significance of covariates, given a trained model, by fixing the training sample. It would be interesting to explore the time dimension of significant predictors and their financial implication by considering different training samples (Gu et al., 2020; Bianchi et al., 2021). Additionally, by assuming neural networks as nonparametric estimators, it would be valuable to explore model misspecification tests in future research.

Table 1: **Predictive R^2 for neural networks (NNs) and Simulation Choices.** This table reports the average in-sample (IS) R^2s and the average out-of-sample (OOS) R^2 of 100 Monte Carlo repetitions for five architectures of neural networks (NN1...NN5) and Oracle model, when $P_c = 50$ and $P_c = 100$. From NN1 to NN5, the number of hidden layers increases from 1 to 5 and the number of neurons in each hidden layer decreases from 32 to 2 according to the pyramid rule. Oracle model is a pooled ordinary least square (OLS) regression with the regressors being the true covariates. We consider three pairs of the number of firms N and the number of months T in simulation: $(N = 200, T = 180)$, $(N = 200, T = 120)$, and $(N = 100, T = 180)$, and report the results in Panel A, B, and C, respectively.

Parameter	PC=50		PC=100	
R^2(%)	IS	OOS	IS	OOS
Panel A: N=200, T=180				
NN1	6.85	3.87	7.14	3.85
NN2	6.83	3.77	7.14	3.71
NN3	6.77	3.67	7.23	3.61
NN4	6.85	3.65	7.19	3.56
NN5	6.97	3.63	7.01	3.42
Oracle	6.43	4.86	6.43	4.86
Panel B: N=200, T=120				
NN1	6.66	2.80	6.92	2.74
NN2	6.51	2.52	6.70	2.43
NN3	6.15	2.35	6.29	2.14
NN4	6.37	2.20	6.59	1.92
NN5	6.17	2.18	6.46	1.70
Oracle	6.53	4.15	6.53	4.15
Panel C: N=100, T=180				
NN1	6.79	3.76	7.10	3.66
NN2	6.70	3.54	6.93	3.24
NN3	6.92	3.19	6.57	2.77
NN4	6.49	3.15	6.54	2.64
NN5	6.17	3.02	6.21	2.41
Oracle	6.44	4.84	6.44	4.84

Table 2: **Variable Significance Frequencies for neural networks (NNs) and Discretization Choices.** This table reports the variable selection frequencies for 6 covariates for five neural networks (NN1...NN5) for $P_c = 50$ and $P_c = 100$, when the simulation choice is $(N = 200, T = 180)$. The variable significance frequency of a covariate is calculated as the proportion of significance test results showing that the covariate is significant at the [10%, 5%, 1%] level, out of 100 Monte Carlo repetitions. $c_{i,1,t}$, $c_{i,2,t}$, and $c_{i,3,t} \times m_t$ are true covariates. $c_{i,1,t} \times m_t$, $c_{i,2,t} \times m_t$, and $c_{i,3,t}$ are covariates that are correlated to one of the three true covariates. From NN1 to NN5, the number of hidden layers increases from 1 to 5 and the number of neurons in each hidden layer decreases from 32 to 2 according to the pyramid rule. We consider five discretization choices: $(m = 500, one\ layer, n_p = 1,000)$, $(m = 300, one\ layer, n_p = 1,000)$, $(m = 500, one\ layer, n_p = 2,000)$, $(m = 500, two\ layers, n_p = 1,000)$, and $(m = 500, three\ layers, n_p = 1,000)$. Their results are reported in Panel A, B, C, D, and E, respectively.

Panel A: $m = 500, one\ layer, n_p = 1,000$							
Parameter	Method	$c_{i,1,t}$	$c_{i,2,t}$	$c_{i,3,t} \times m_t$	$c_{i,1,t} \times m_t$	$c_{i,2,t} \times m_t$	$c_{i,3,t}$
$P_c = 50$	NN1	[80, 73, 70]	[81, 80, 65]	[76, 72, 64]	[21, 19, 8]	[17, 12, 7]	[18, 12, 7]
	NN2	[81, 74, 67]	[81, 77, 63]	[78, 70, 61]	[20, 15, 7]	[17, 12, 7]	[17, 13, 7]
	NN3	[82, 76, 67]	[82, 79, 62]	[79, 72, 59]	[20, 14, 8]	[19, 14, 6]	[19, 12, 6]
	NN4	[85, 79, 74]	[83, 79, 68]	[83, 75, 67]	[24, 20, 10]	[26, 18, 12]	[22, 14, 8]
	NN5	[88, 82, 75]	[84, 81, 70]	[85, 79, 70]	[26, 20, 10]	[25, 21, 12]	[20, 18, 9]
$P_c = 100$	NN1	[85, 80, 72]	[84, 81, 75]	[79, 76, 68]	[24, 20, 10]	[21, 15, 9]	[22, 17, 9]
	NN2	[82, 77, 67]	[83, 79, 70]	[77, 74, 65]	[23, 19, 9]	[22, 14, 7]	[19, 16, 8]
	NN3	[87, 79, 72]	[83, 81, 73]	[83, 76, 67]	[25, 20, 8]	[25, 18, 10]	[21, 17, 12]
	NN4	[79, 74, 66]	[79, 76, 64]	[79, 69, 65]	[24, 18, 7]	[21, 14, 5]	[20, 15, 12]
	NN5	[81, 76, 69]	[80, 70, 64]	[79, 70, 66]	[22, 15, 8]	[24, 14, 5]	[19, 17, 10]
Panel B: $m = 300, one\ layer, n_p = 1,000$							
Parameter	Method	$c_{i,1,t}$	$c_{i,2,t}$	$c_{i,3,t} \times m_t$	$c_{i,1,t} \times m_t$	$c_{i,2,t} \times m_t$	$c_{i,3,t}$
$P_c = 50$	NN1	[81, 74, 69]	[83, 79, 64]	[76, 72, 64]	[21, 18, 8]	[17, 13, 6]	[17, 12, 7]
	NN2	[81, 76, 67]	[82, 77, 62]	[76, 70, 62]	[21, 16, 7]	[18, 12, 7]	[16, 12, 7]
	NN3	[82, 76, 67]	[82, 75, 62]	[79, 74, 57]	[20, 14, 8]	[19, 14, 7]	[18, 12, 6]
	NN4	[85, 78, 74]	[83, 78, 67]	[82, 75, 67]	[23, 19, 9]	[26, 20, 9]	[20, 14, 8]
	NN5	[89, 82, 74]	[85, 81, 70]	[86, 79, 70]	[26, 20, 9]	[25, 22, 11]	[20, 18, 9]
$P_c = 100$	NN1	[87, 79, 72]	[84, 81, 75]	[80, 76, 66]	[24, 21, 8]	[20, 14, 9]	[22, 17, 8]
	NN2	[83, 76, 66]	[83, 79, 71]	[77, 74, 65]	[23, 19, 9]	[22, 14, 7]	[20, 15, 8]
	NN3	[89, 81, 72]	[84, 80, 73]	[84, 81, 65]	[24, 20, 8]	[25, 17, 10]	[21, 16, 10]
	NN4	[80, 75, 64]	[79, 76, 65]	[79, 74, 64]	[23, 20, 7]	[22, 14, 6]	[19, 13, 9]
	NN5	[81, 76, 68]	[81, 71, 66]	[79, 75, 62]	[24, 14, 9]	[24, 13, 5]	[17, 15, 9]

Table 2 Continue.

Panel C: $m = 500, one\ layer, n_p = 2,000$							
Parameter	Method	$c_{i,1,t}$	$c_{i,2,t}$	$c_{i,3,t} \times m_t$	$c_{i,1,t} \times m_t$	$c_{i,2,t} \times m_t$	$c_{i,3,t}$
$P_c = 50$	NN1	[81, 73, 70]	[82, 80, 67]	[76, 72, 64]	[21, 19, 10]	[17, 12, 7]	[18, 12, 7]
	NN2	[81, 75, 66]	[81, 77, 63]	[76, 70, 61]	[20, 15, 7]	[18, 12, 7]	[17, 13, 7]
	NN3	[82, 76, 67]	[82, 77, 63]	[79, 72, 57]	[20, 14, 8]	[19, 14, 6]	[19, 12, 6]
	NN4	[85, 78, 74]	[83, 78, 70]	[82, 75, 67]	[23, 20, 10]	[26, 18, 12]	[21, 14, 10]
	NN5	[88, 82, 74]	[84, 81, 72]	[85, 78, 70]	[25, 20, 11]	[25, 21, 12]	[20, 18, 11]
$P_c = 100$	NN1	[86, 79, 72]	[84, 81, 74]	[80, 76, 67]	[24, 21, 10]	[21, 14, 9]	[22, 17, 9]
	NN2	[82, 76, 67]	[83, 79, 70]	[77, 74, 66]	[23, 20, 9]	[22, 14, 7]	[20, 15, 9]
	NN3	[87, 79, 73]	[83, 81, 72]	[83, 81, 67]	[25, 20, 8]	[25, 17, 9]	[21, 16, 12]
	NN4	[79, 75, 66]	[79, 76, 63]	[79, 71, 65]	[23, 18, 8]	[22, 14, 5]	[20, 14, 11]
	NN5	[81, 75, 70]	[80, 70, 64]	[79, 71, 65]	[22, 15, 9]	[24, 13, 5]	[19, 16, 10]
Panel D: $m = 500, two\ layers, n_p = 1,000$							
Parameter	Method	$c_{i,1,t}$	$c_{i,2,t}$	$c_{i,3,t} \times m_t$	$c_{i,1,t} \times m_t$	$c_{i,2,t} \times m_t$	$c_{i,3,t}$
$P_c = 50$	NN1	[73, 71, 69]	[81, 81, 75]	[73, 72, 64]	[21, 18, 8]	[16, 9, 5]	[12, 10, 7]
	NN2	[72, 69, 64]	[81, 78, 71]	[74, 71, 63]	[19, 15, 7]	[13, 9, 7]	[12, 10, 6]
	NN3	[73, 70, 67]	[80, 80, 67]	[77, 73, 62]	[18, 13, 8]	[18, 10, 6]	[12, 10, 6]
	NN4	[78, 75, 70]	[83, 82, 77]	[79, 76, 68]	[23, 19, 9]	[21, 13, 9]	[14, 13, 7]
	NN5	[79, 75, 72]	[83, 83, 79]	[80, 78, 72]	[23, 20, 9]	[25, 15, 9]	[18, 15, 8]
$P_c = 100$	NN1	[72, 68, 65]	[79, 77, 69]	[70, 66, 51]	[16, 11, 4]	[13, 11, 6]	[10, 8, 3]
	NN2	[65, 64, 61]	[76, 73, 60]	[65, 65, 48]	[14, 9, 4]	[13, 9, 4]	[9, 8, 5]
	NN3	[69, 66, 61]	[78, 72, 65]	[67, 65, 46]	[14, 10, 6]	[13, 11, 8]	[12, 10, 5]
	NN4	[61, 59, 54]	[69, 66, 61]	[65, 62, 42]	[8, 8, 5]	[10, 8, 4]	[12, 8, 5]
	NN5	[59, 56, 53]	[68, 67, 56]	[68, 61, 40]	[9, 9, 6]	[11, 8, 5]	[11, 8, 4]
Panel E: $m = 500, three\ layers, n_p = 1,000$							
Parameter	Method	$c_{i,1,t}$	$c_{i,2,t}$	$c_{i,3,t} \times m_t$	$c_{i,1,t} \times m_t$	$c_{i,2,t} \times m_t$	$c_{i,3,t}$
$P_c = 50$	NN1	[73, 72, 63]	[81, 79, 73]	[75, 72, 64]	[21, 16, 12]	[13, 9, 9]	[16, 14, 10]
	NN2	[74, 73, 61]	[78, 76, 66]	[74, 70, 64]	[17, 12, 11]	[13, 12, 11]	[16, 14, 11]
	NN3	[76, 72, 62]	[80, 78, 64]	[76, 73, 62]	[15, 10, 9]	[13, 11, 10]	[17, 13, 11]
	NN4	[79, 77, 66]	[81, 78, 73]	[80, 77, 69]	[20, 15, 12]	[19, 13, 12]	[18, 17, 13]
	NN5	[81, 78, 70]	[83, 81, 77]	[81, 79, 72]	[21, 14, 13]	[22, 17, 13]	[19, 19, 16]
$P_c = 100$	NN1	[72, 71, 61]	[78, 74, 66]	[74, 65, 59]	[16, 12, 10]	[13, 8, 4]	[15, 10, 8]
	NN2	[68, 65, 60]	[75, 69, 61]	[70, 65, 55]	[14, 10, 9]	[11, 4, 2]	[15, 11, 7]
	NN3	[72, 69, 57]	[75, 71, 65]	[77, 65, 55]	[14, 10, 8]	[13, 8, 6]	[16, 12, 8]
	NN4	[66, 61, 52]	[67, 63, 59]	[69, 61, 49]	[8, 8, 8]	[10, 4, 2]	[13, 12, 8]
	NN5	[70, 57, 46]	[67, 63, 56]	[69, 61, 44]	[9, 9, 9]	[10, 5, 2]	[14, 11, 8]

Table 3: Monthly out-of-sample stock-level prediction performance (EV).

There are two sets of predictors: CpM and CtM. CpM: characteristics plus macroeconomics. CtM: characteristics times macroeconomics. EV is given by: $EV = 1 - \frac{\sum_{t=1}^{T} \frac{1}{N_t} \sum_{i=1}^{N_t} (\hat{\epsilon}_{i,t+1})^2}{\sum_{t=1}^{T} \frac{1}{N_t} \sum_{i=1}^{N_t} (r^e_{i,t+1})^2}$ where T is the number of months in the sample, N_t is the number of available stocks in month t, $r^e_{t+1,i}$ is the excess return of stock i at month $t+1$, and $\hat{\epsilon}_{i,t+1} = \left(I_{N_t} - \hat{\beta}_t(\hat{\beta}_t^\top \hat{\beta}_t)^{-1}\hat{\beta}_t^\top\right) r^e_{i,t+1}$ where $\hat{\beta}_t$ is the output of a trained neural network. We also report these EV within subsamples that include only the top-100 stocks or bottom-100 stocks by market value (mve). The lower panel provides a visual comparison of the EV statistics in the table. NNi: a multi-layer perceptron with i-hidden layer.

	CpM					CtM				
	NN1	NN2	NN3	NN4	NN5	NN1	NN2	NN3	NN4	NN5
All	0.115	0.121	0.114	0.117	0.122	0.095	0.099	0.099	0.107	0.108
Top 100	0.259	0.278	0.272	0.269	0.273	0.231	0.237	0.236	0.255	0.261
Bottom 100	0.076	0.077	0.076	0.077	0.077	0.062	0.064	0.066	0.069	0.068

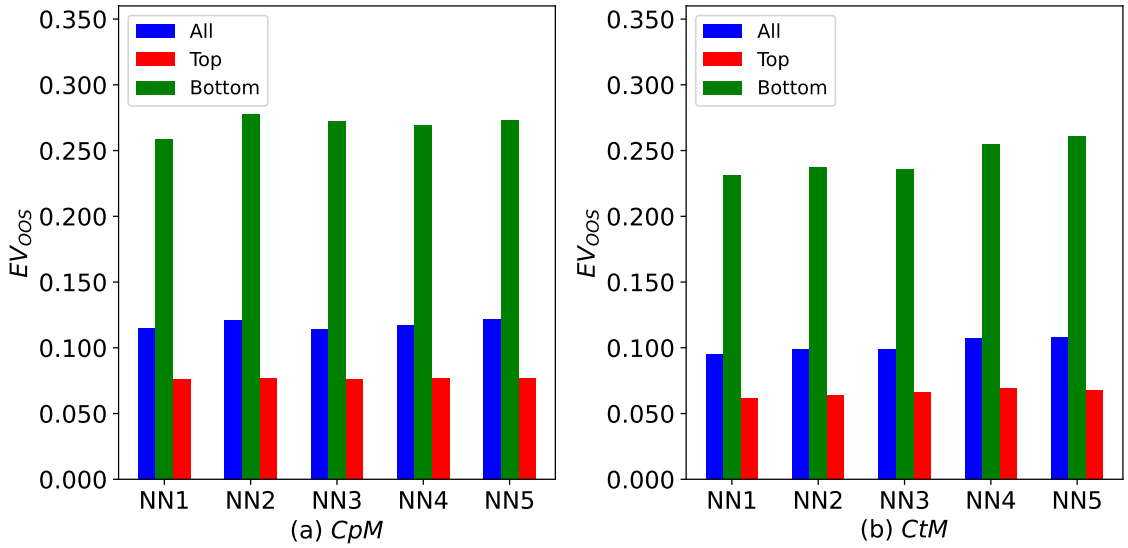


Table 4: Comparison of monthly out-of-sample prediction using Diebold-Mariano tests.

This table reports pairwise Diebold-Mariano test statistics comparing the out-of-sample stock-level prediction performance among five models. Positive numbers indicate the column model outperforms the row model. Superscripts ***, ** and * indicate statistical significance at the 1%, 5% and 10% level, respectively. NNi: a multi-layer perceptron with i-hidden layer.

	CpM				CtM			
	NN2	NN3	NN4	NN5	NN2	NN3	NN4	NN5
NN1	2.472***	-0.123	0.641	2.096**	3.009***	5.315***	4.553***	4.090***
NN2		-1.399	-1.972	0.878		-0.244	2.592***	2.288**
NN3			0.563	1.983**			3.429***	3.308***
NN4				2.287**				0.605

Table 5: Variable significant test results of characteristics where the universe of predictors is CpM.

Predictors are sorted based on their t-stat values within each model, and averaging the predictors rankings among the five NN models to obtain an overall influence in terms of t-stat. In front of each predictor, the scaled t-stats for each model are reported, while their corresponding p-values are underneath of them. We use the red (green) color for predictors that their p-value is less (larger) than 10%. The darkness of red (green) color represents its significant (insignificant) level. See Table A5 for the definitions of stock characteristics. NN*i*: a multi-layer perceptron with *i*-hidden layer.

Predictor	NN1	NN2	NN3	NN4	NN5	Predictor	NN1	NN2	NN3	NN4	NN5	Predictor	NN1	NN2	NN3	NN4	NN5
mom1m	0.287	0.169	0.211	0.187	0.093	invest	0.079	0.077	0.077	0.069	0.052	chcsho	0.073	0.075	0.073	0.066	0.051
	0.000	0.009	0.005	0.008	0.024		0.089	0.089	0.087	0.090	0.090		0.101	0.101	0.094	0.093	0.094
std_turn	0.209	0.126	0.133	0.122	0.065	agr	0.084	0.077	0.074	0.071	0.051	divo	0.077	0.075	0.073	0.064	0.051
	0.004	0.027	0.020	0.021	0.055		0.076	0.090	0.089	0.080	0.091		0.097	0.100	0.103	0.100	0.098
turn	0.146	0.112	0.110	0.123	0.063	cfp	0.080	0.077	0.080	0.068	0.051	mve_ia	0.075	0.074	0.072	0.064	0.052
	0.014	0.034	0.035	0.019	0.058		0.091	0.091	0.083	0.089	0.095		0.099	0.101	0.099	0.098	0.088
dolvol	0.245	0.117	0.126	0.079	0.057	lev	0.078	0.081	0.074	0.067	0.053	ps	0.075	0.076	0.073	0.064	0.050
	0.001	0.033	0.025	0.064	0.074		0.090	0.085	0.092	0.087	0.086		0.100	0.092	0.099	0.099	0.094
chmom	0.101	0.085	0.093	0.083	0.055	pchsale_pchinvt	0.077	0.077	0.076	0.068	0.054	grltnoa	0.073	0.075	0.074	0.064	0.050
	0.046	0.076	0.053	0.056	0.081		0.093	0.091	0.087	0.086	0.085		0.103	0.100	0.090	0.100	0.103
divi	0.099	0.087	0.087	0.077	0.057	pchquick	0.078	0.076	0.076	0.070	0.052	chempia	0.074	0.075	0.072	0.067	0.049
	0.049	0.072	0.066	0.073	0.076		0.094	0.095	0.089	0.081	0.088		0.101	0.106	0.098	0.091	0.102
indmom	0.091	0.083	0.090	0.084	0.057	bm_ia	0.083	0.079	0.078	0.066	0.050	age	0.075	0.074	0.071	0.064	0.051
	0.064	0.078	0.059	0.052	0.073		0.076	0.095	0.080	0.092	0.099		0.096	0.097	0.096	0.101	0.091
ill	0.101	0.092	0.084	0.072	0.053	retvol	0.085	0.081	0.076	0.063	0.051	pchgm_pchsale	0.075	0.073	0.072	0.064	0.050
	0.050	0.059	0.067	0.078	0.089		0.074	0.083	0.092	0.108	0.093		0.100	0.102	0.102	0.099	0.097
betasq	0.126	0.191	0.078	0.067	0.057	maxret	0.082	0.078	0.077	0.066	0.051	tang	0.075	0.073	0.073	0.064	0.049
	0.025	0.006	0.081	0.089	0.075		0.087	0.089	0.088	0.095	0.093		0.094	0.098	0.095	0.098	0.102
gma	0.099	0.081	0.080	0.075	0.054	egr	0.083	0.076	0.077	0.067	0.050	mom36m	0.073	0.073	0.072	0.065	0.049
	0.051	0.086	0.083	0.071	0.087		0.080	0.087	0.085	0.098	0.092		0.096	0.105	0.099	0.099	0.097
bm	0.096	0.086	0.083	0.070	0.053	chatoia	0.080	0.077	0.076	0.066	0.051	grcapx	0.072	0.075	0.072	0.063	0.050
	0.057	0.073	0.073	0.079	0.087		0.090	0.093	0.086	0.094	0.092		0.099	0.098	0.103	0.104	0.099
sp	0.083	0.079	0.090	0.078	0.054	acc	0.078	0.078	0.072	0.068	0.053	cfp_ia	0.071	0.074	0.072	0.064	0.049
	0.077	0.093	0.054	0.061	0.088		0.087	0.096	0.099	0.087	0.088		0.111	0.100	0.102	0.098	0.102
operprof	0.113	0.083	0.085	0.082	0.050	cashpr	0.077	0.075	0.076	0.068	0.053	saleinv	0.075	0.074	0.072	0.063	0.049
	0.038	0.077	0.065	0.059	0.100		0.092	0.094	0.088	0.087	0.087		0.097	0.100	0.096	0.104	0.101
cashdebt	0.086	0.080	0.077	0.072	0.054	currat	0.081	0.077	0.075	0.068	0.050	pchsale_pchxsga	0.075	0.073	0.071	0.064	0.049
	0.069	0.091	0.086	0.078	0.081		0.084	0.091	0.088	0.085	0.096		0.093	0.108	0.098	0.101	0.100
BETA	0.128	0.200	0.077	0.066	0.054	pchcurrat	0.077	0.076	0.076	0.066	0.053	salerec	0.070	0.073	0.071	0.063	0.050
	0.028	0.005	0.086	0.098	0.081		0.091	0.094	0.087	0.094	0.087		0.117	0.101	0.101	0.105	0.102
sgr	0.092	0.080	0.079	0.071	0.053	dy	0.077	0.076	0.075	0.067	0.052	pricedelay	0.068	0.072	0.070	0.066	0.049
	0.060	0.089	0.084	0.077	0.088		0.090	0.091	0.094	0.093	0.089		0.117	0.107	0.108	0.096	0.101

Table 5 (cont.)

Predictor	NN1	NN2	NN3	NN4	NN5	Predictor	NN1	NN2	NN3	NN4	NN5	Predictor	NN1	NN2	NN3	NN4	NN5
mve	0.092	0.079	0.077	0.068	0.055	convind	0.079	0.075	0.077	0.070	0.050	pchsale_pchrect	0.074	0.072	0.071	0.064	0.049
	0.063	0.087	0.091	0.087	0.079		0.090	0.098	0.086	0.085	0.097		0.100	0.103	0.103	0.104	0.097
roic	0.084	0.078	0.078	0.074	0.053	chinv	0.079	0.078	0.074	0.064	0.052	absacc	0.070	0.072	0.070	0.064	0.050
	0.078	0.090	0.085	0.074	0.086		0.084	0.088	0.094	0.102	0.089		0.113	0.103	0.109	0.099	0.096
mom12m	0.080	0.080	0.080	0.069	0.054	rd_mve	0.077	0.076	0.074	0.069	0.051	pchdepr	0.069	0.072	0.071	0.062	0.050
	0.083	0.084	0.080	0.086	0.086		0.092	0.091	0.094	0.084	0.096		0.106	0.107	0.101	0.109	0.097
sin	0.078	0.081	0.081	0.067	0.055	pctacc	0.079	0.076	0.075	0.066	0.051	securedind	0.069	0.073	0.072	0.063	0.049
	0.090	0.080	0.076	0.088	0.076		0.093	0.093	0.092	0.094	0.092		0.118	0.100	0.099	0.107	0.102
pchsaleinv	0.083	0.080	0.079	0.067	0.055	salecash	0.077	0.076	0.075	0.066	0.051	tb	0.069	0.073	0.070	0.064	0.049
	0.082	0.081	0.081	0.094	0.080		0.091	0.094	0.093	0.090	0.090		0.116	0.098	0.107	0.100	0.101
lgr	0.084	0.078	0.080	0.073	0.051	baspread	0.076	0.077	0.075	0.064	0.051	depr	0.070	0.072	0.071	0.063	0.049
	0.071	0.087	0.083	0.075	0.092		0.091	0.095	0.096	0.106	0.095		0.115	0.108	0.100	0.107	0.104
std_dolvol	0.090	0.078	0.080	0.069	0.051	rd_sale	0.075	0.076	0.077	0.066	0.051	chpmia	0.072	0.071	0.069	0.064	0.048
	0.065	0.091	0.080	0.083	0.092		0.097	0.093	0.082	0.097	0.093		0.106	0.115	0.109	0.100	0.104
mom6m	0.085	0.079	0.080	0.070	0.050	ep	0.076	0.075	0.072	0.065	0.053	pchcapx_ia	0.070	0.073	0.071	0.063	0.049
	0.074	0.086	0.080	0.086	0.091		0.100	0.095	0.097	0.095	0.087		0.114	0.101	0.101	0.105	0.104
idiovol	0.080	0.077	0.080	0.070	0.053	hire	0.076	0.078	0.073	0.067	0.049	rd	0.069	0.072	0.071	0.062	0.049
	0.086	0.093	0.082	0.082	0.084		0.093	0.091	0.102	0.095	0.098		0.114	0.104	0.105	0.111	0.101
quick	0.082	0.079	0.076	0.072	0.052	orgcap	0.077	0.076	0.073	0.066	0.049	herf	0.071	0.070	0.070	0.061	0.048
	0.084	0.091	0.090	0.077	0.091		0.092	0.092	0.095	0.095	0.103		0.108	0.116	0.105	0.109	0.105
scaling factor	206.450	106.355	167.857	188.426	156.299												

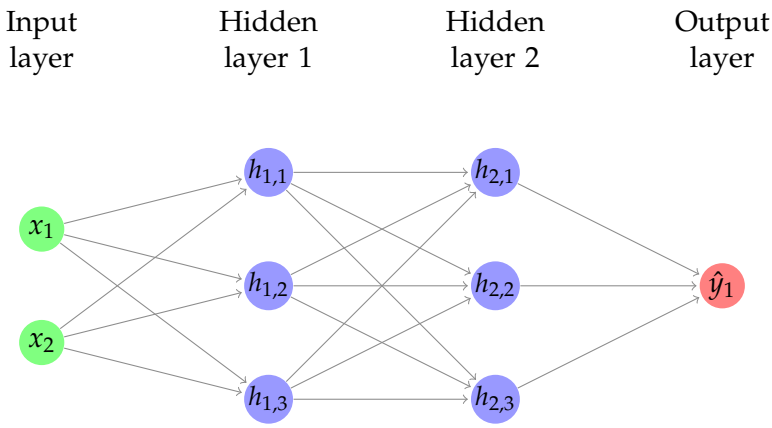


Figure 1: Architecture of a fully connected feed-forward network with two hidden layers in form of a multi-layer perceptron (MLP).

An example of a feed-forward network (FFN) in form of a multi-layer perceptron (MLP) that is used as an estimator in this project. This MLP has two hidden layers with two inputs, three nodes in each hidden layer, and a single output node. Under the mathematical formulation in Subsection 2.3, this means $L_d = 2$, $H_{n,0} = 2$, $H_{n,1} = H_{n,2} = 3$, and $H_{n,3} = 1$. The neural network is called FFN because it is a directed graph in which the arrows only point forward to the graph in the next layer. The reason it is named an MLP is each hidden node is connected to all nodes in the nodes of the adjacent layers, but not others. Mathematically, a hidden node in a hidden layer acts as a node where linear aggregation operation is performed. This operation takes input from all nodes in the previous layer. Then, the hidden node applies an activation function to the linear aggregation and send it as an input to next layer. The final output node does not apply any activation function. Using the framework of Section 2, this output per node corresponds to the function $z_{u,j}(\boldsymbol{x})$, with ψ as the activation function. This paper specifically focus on smooth activation function, but other function such as ReLU can also be applied. Note that this can be seen as a deciding process for the activation of the hidden node if we look at the MLP as a model of a biological neural network. If the activation yields positive value, for example, then the hidden node is activated and it sends that activating signal to the next forward-connected neural nodes.

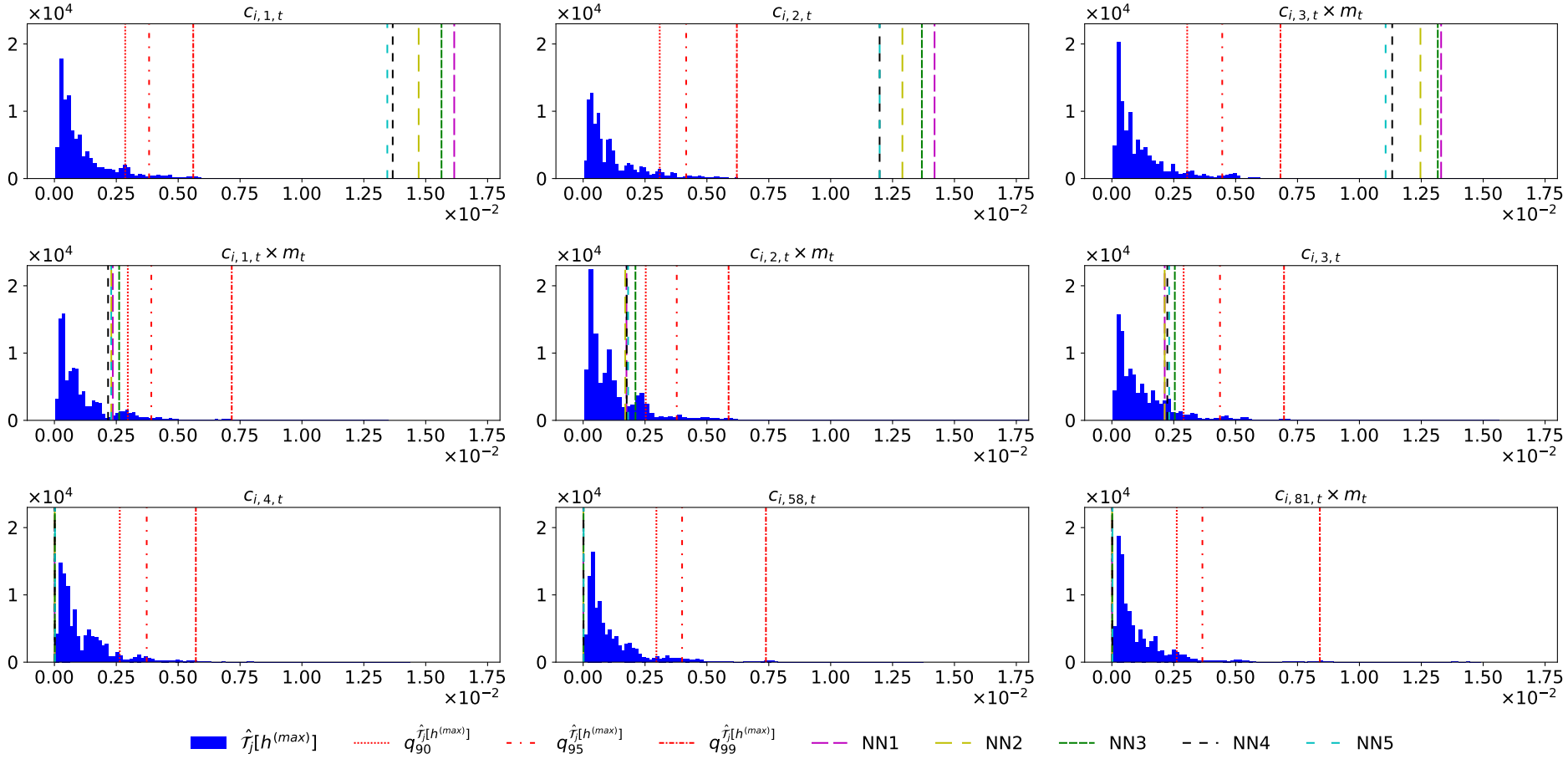
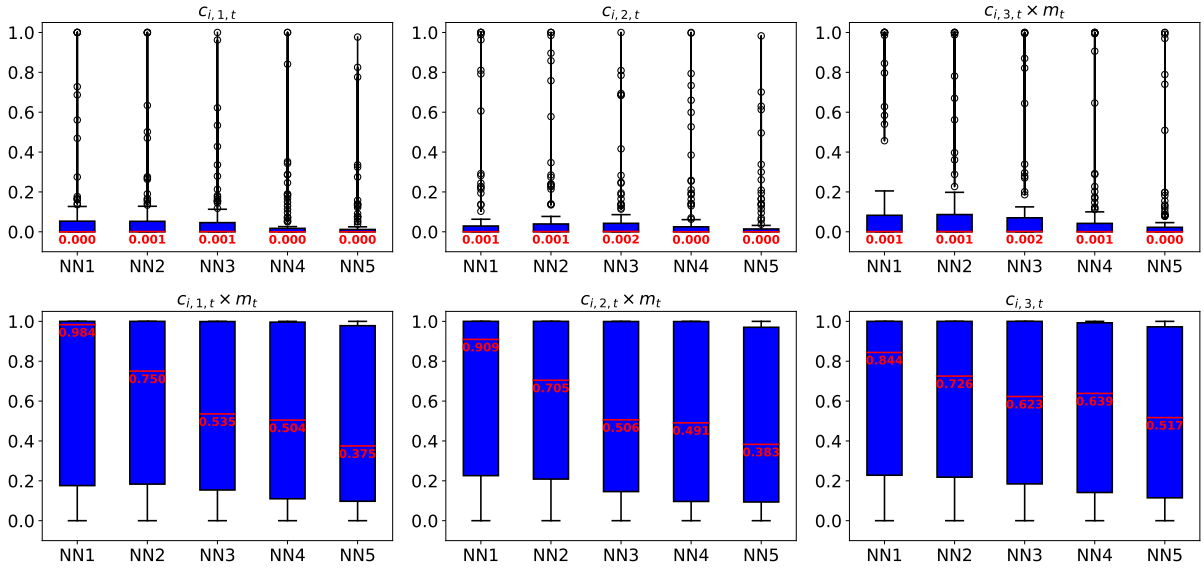
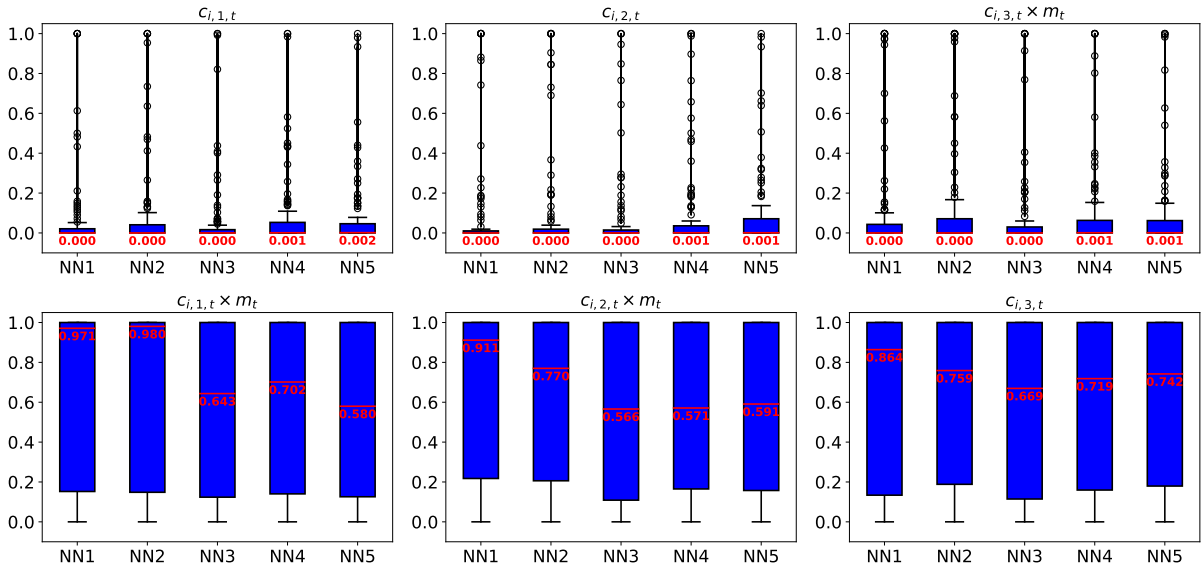


Figure 2: Distribution of $\mathcal{T}_j[h^{(max)}]$ samples and the test statistics for covariates. This figure shows the distributions of 100,000 $\mathcal{T}_j[h^{(max)}]$ samples and the averaged test statistics for 9 covariates for five neural networks, when the simulation choice is $(N = 200, T = 180)$, the discretization choice is $(m = 500, one\ layer, n_p = 1,000)$, and $P_c = 100$. The 100,000 $\mathcal{T}_j[h^{(max)}]$ samples include the 1,000 $\mathcal{T}_j[h^{(max)}]$ samples from each of the 100 Monte Carlo repetitions. The averaged test statistic is the average of the scaled test statistics from the 100 Monte Carlo repetitions. The blue part shows the distribution of $\mathcal{T}_j[h^{(max)}]$ samples. The magenta, yellow, green, black, and cyan dashed lines represent the average test statistics for NN1, NN2, NN3, NN4, and NN5, respectively. The red dotted, dashdotted, and densely dashdotted lines represent the 90%, 95%, and 99% quantiles of the 100,000 $\mathcal{T}_j[h^{(max)}]$ samples, respectively. $c_{i,1,t}$, $c_{i,2,t}$, and $c_{i,3,t} \times m_t$ are the true covariates. $c_{i,1,t} \times m_t$, $c_{i,2,t} \times m_t$, and $c_{i,3,t}$ are covariates that are correlated to one of the true covariates. $c_{i,4,t}$, $c_{i,58,t}$, and $c_{i,81,t} \times m_t$ are noise. The test statistics are scaled by $(n^{-0.215})^{-2}$, where $n = N \times T/3$.



(a) $P_c = 50$



(b) $P_c = 100$

Figure 3: Distribution of p-values for covariates. This figure presents the distribution of 100 p-values for 6 covariates for five neural networks, when the simulation choice is $(N = 200, T = 180)$ and the discretization choice is $(m = 500, one\ layer, n_p = 1{,}000)$. Panel (a) and (b) show the distribution for $P_c = 50$ and $P_c = 100$, respectively. $c_{i,1,t}$, $c_{i,2,t}$, and $c_{i,3,t} \times m_t$ are the true covariates. $c_{i,1,t} \times m_t$, $c_{i,2,t} \times m_t$, and $c_{i,3,t}$ are covariates that are correlated to one of the true covariates. For each Monte Carlo repetition, we compute the p-value for each covariate based on its scaled test statistic and the 1,000 $\mathcal{T}_j[h^{(max)}]$ samples. Thus, a total of 100 Monte Carlo repetitions lead to a total of 100 p-values for each covariate. The red line marks the median level of the 100 p-values. The number underneath the red line is the median p-value, rounded to three-digits. From NN1 to NN5, the number of hidden layers increases from 1 to 5 and the number of neurons in each hidden layer decreases from 32 to 2 according to the pyramid rule. The test statistics are scaled by $(n^{-0.245})^{-2}$ in Panel (a) and $(n^{-0.215})^{-2}$ in Panel (b), where $n = N \times T/3$.

References

Adler, R. J. (1990). An introduction to continuity, extrema, and related topics for general gaussian processes. *Lecture Notes-Monograph Series* 12: i–155.

Anthony, M. and Bartlett, P. L. (2009). *Neural network learning: Theoretical foundations*. cambridge university press.

Bach, S., Binder, A., Montavon, G., Klauschen, F., Müller, K.-R. and Samek, W. (2015). On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *PloS one* 10: e0130140.

Bartlett, P. L., Bousquet, O., Mendelson, S. et al. (2005). Local rademacher complexities. *Annals of Statistics* 33: 1497–1537.

Bianchi, D., Büchner, M. and Tamoni, A. (2021). Bond risk premiums with machine learning. *Review of Financial Studies* 34: 1046–1089.

Campbell, J. Y. (2000). Asset pricing at the millennium. *Journal of Finance* 55: 1515–1567.

Chen, L., Pelger, M. and Zhu, J. (2023). Deep learning in asset pricing. *Management Science, forthcoming* .

Chen, X. and Shen, X. (1998). Sieve extremum estimates for weakly dependent data. *Econometrica* : 289–314.

Chen, X. and White, H. (1999). Improved rates and asymptotic normality for nonparametric neural network estimators. *IEEE Transactions on Information Theory* 45: 682–691.

Clevert, D.-A., Unterthiner, T. and Hochreiter, S. (2015). Fast and accurate deep network learning by exponential linear units (elus). *arXiv preprint arXiv:1511.07289* .

Cochrane, J. H. (2009). *Asset pricing*. Princeton university press.

Cochrane, J. H. (2011). Presidential address: Discount rates. *Journal of Fnance* 66: 1047–1108.

Datta, A., Sen, S. and Zick, Y. (2016). Algorithmic transparency via quantitative input influence: Theory and experiments with learning systems. In *2016 IEEE symposium on security and privacy (SP)*. IEEE, 598–617.

Dietterich, T. G. (2000). Ensemble methods in machine learning. In *International workshop on multiple classifier systems*. Springer, 1–15.

Eldan, R. and Shamir, O. (2016). The power of depth for feedforward neural networks. In *in Vitaly Feldman, Alexander Rakhlin, and Ohad Shamir, eds., 29th Annual Conference on Learning Theory, volume 49 of Proceedings of Machine Learning Research*, 907–940.

Fabozzi, F., Fallahgoul, H., Franstianto, V. and Loeper, G. (2021). Towards explaining deep learning: Asymptotic properties of relu ffn sieve estimators. *Available at SSRN* .

Fama, E. F. (1970). Efficient capital markets: A review of theory and empirical work. *Journal of Finance* 25: 383–417.

Fama, E. F. and French, K. R. (1993). Common risk factors in the returns on stocks and bonds. *Journal of Financial Economics* .

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

Fama, E. F. and French, K. R. (2015). A five-factor asset pricing model. *Journal of Financial Economics* 116: 1–22.

Farrell, M. H., Liang, T. and Misra, S. (2021). Deep neural networks for estimation and inference. *Econometrica* 89: 181–213.

Feng, G., Giglio, S. and Xiu, D. (2019). Taming the factor zoo: A test of new factors. *Journal of Finance, forthcoming* .

Freyberger, J. and Masten, M. (2015). Compactness of infinite dimensional parameter spaces. Tech. rep., cemmap working paper.

Freyberger, J., Neuhierl, A. and Weber, M. (2020). Dissecting characteristics nonparametrically. *Review of Financial Studies* 33: 2326–2377.

Giné, E. and Nickl, R. (2021). *Mathematical foundations of infinite-dimensional statistical models*. Cambridge university press.

Goodfellow, I., Bengio, Y. and Courville, A. (2016). *Deep learning*. MIT press.

Goyal, A. (2012). Empirical cross-sectional asset pricing: a survey. *Financial Markets and Portfolio Management* 26: 3–38.

Green, J., Hand, J. R. and Zhang, X. F. (2017). The characteristics that provide independent information about average us monthly stock returns. *Review of Financial Studies* 30: 4389–4436.

Gu, S., Kelly, B. and Xiu, D. (2020). Empirical asset pricing via machine learning. *Review of Financial Studies* 33: 2223–2273.

Harvey, C. R., Liu, Y. and Zhu, H. (2016). ... and the cross-section of expected returns. *Review of Financial Studies* 29: 5–68.

Hinton, G. E., Osindero, S. and Teh, Y.-W. (2006). A fast learning algorithm for deep belief nets. *Neural computation* 18: 1527–1554.

Horel, E. and Giesecke, K. (2020). Significance tests for neural networks. *Journal of Machine Learning Research* 21: 1–29.

Hou, K., Xue, C. and Zhang, L. (2017). Replicating anomalies. Tech. rep., National Bureau of Economic Research.

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

Kim, J., Pollard, D. et al. (1990). Cube root asymptotics. *Annals of Statistics* 18: 191–219.

Kohler, M. and Langer, S. (2021). On the rate of convergence of fully connected deep neural network regression estimates. *Annals of Statistics* 49: 2231–2249.

Koltchinskii, V. (2006). Local rademacher complexities and oracle inequalities in risk minimization. *Annals of Statistics* 34: 2593–2656.

Koltchinskii, V. and Panchenko, D. (2000). Rademacher processes and bounding the risk of function learning. In *High dimensional probability II*. Springer, 443–457.

Lundberg, S. and Lee, S.-I. (2017). A unified approach to interpreting model predictions. *arXiv preprint arXiv:1705.07874* .

Masters, T. (1993). *Practical neural network recipes in C++*. Academic Press Professional, Inc.

McCracken, M. W. and Ng, S. (2016). Fred-md: A monthly database for macroeconomic research. *Journal of Business & Economic Statistics* 34: 574–589.

McLean, R. D. and Pontiff, J. (2016). Does academic research destroy stock return predictability? *Journal of Finance* 71: 5–32.

Nickl, R. and Pötscher, B. M. (2007). Bracketing metric entropy rates and empirical central limit theorems for function classes of besov-and sobolev-type. *Journal of Theoretical Probability* 20: 177–199.

Poggio, T., Anselmi, F. and Rosasco, L. (2015). I-theory on depth vs width: hierarchical function composition. Tech. rep., Center for Brains, Minds and Machines (CBMM).

Poggio, T., Mhaskar, H., Rosasco, L., Miranda, B. and Liao, Q. (2017). Why and when can deep-but not shallow-networks avoid the curse of dimensionality: a review. *International Journal of Automation and Computing* 14: 503–519.

Ribeiro, M. T., Singh, S. and Guestrin, C. (2016). " Why should i trust you?" Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, 1135–1144.

Römisch, W. (2004). Delta method, infinite dimensional, extended entry, encyclopedia of statistical sciences.

Schmidt-Hieber, J. et al. (2020). Nonparametric regression using deep neural networks with relu activation function. *Annals of Statistics* 48: 1875–1897.

Shen, X. (1997). On methods of sieves and penalization. *Annals of Statistics* 25: 2555–2591.

Shen, X. and Wong, W. H. (1994). Convergence rate of sieve estimates. *Annals of Statistics* : 580–615.

Shrikumar, A., Greenside, P., Shcherbina, A. and Kundaje, A. (2016). Not just a black box: Learning important features through propagating activation differences. *arXiv preprint arXiv:1605.01713* .

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I. and Salakhutdinov, R. (2014). Dropout: a simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research* 15: 1929–1958.

Štrumbelj, E. and Kononenko, I. (2014). Explaining prediction models and individual predictions with feature contributions. *Knowledge and information systems* 41: 647–665.

Vaart, A. W. Van der and Wellner, J. A. (1996). *Weak convergence and empirical processes: with applications to statistics*. Springer.

Welch, I. and Goyal, A. (2008). A comprehensive look at the empirical performance of equity premium prediction. *Review of Financial Studies* 21: 1455–1508.

Yarotsky, D. (2017). Error bounds for approximations with deep relu networks. *Neural Networks* 94: 103–114.

Online Appendix to:

Asset Pricing with Neural Networks: Significance Tests

Hasan Fallahgoul Vincentius Franstianto Xin Lin

A Proof of Theorem 3.1.1

The proof follows the proving strategies from Farrell et al. (2021). Let $\mathbb{E}_n$ be the empirical mean operator. That is, for i.i.d. draws of random variables $T_i, \ldots, T_n$, $\mathbb{E}_n(T) := \frac{1}{n}\sum_{i=1}^{n} T_i$. Note that if T_i are positive i.i.d. random variables, then we have $\mathbb{E}_n(T) = \rho_n^2\left(\sqrt{T}\right)$.

A.1 Main Decomposition and Bias Term

Because of the obvious fact $\mathbb{E}_n(\ell(\hat{f}_n, \mathbf{Z})) \leq \mathbb{E}_n(\ell(f_n, \mathbf{Z}))$ and Lemma 9 from Farrell et al. (2021), we have

$$\begin{aligned}
\frac{1}{2}\|\hat{f}_n - f_\star\|^2_{L^2(P)} &\leq \mathbb{E}(\ell(\hat{f}_n, \mathbf{Z})) - \mathbb{E}(\ell(f_\star, \mathbf{Z})) \\
&\leq \mathbb{E}(\ell(\hat{f}_n, \mathbf{Z})) - \mathbb{E}(\ell(f_\star, \mathbf{Z})) + \mathbb{E}_n(\ell(f_n, \mathbf{Z})) - \mathbb{E}_n(\ell(\hat{f}_n, \mathbf{Z})) \\
&\leq (\mathbb{E} - \mathbb{E}_n)\left(\ell(\hat{f}_n, \mathbf{Z}) - \ell(f_\star, \mathbf{Z})\right) + \mathbb{E}_n\left(\ell(f_n, \mathbf{Z}) - \ell(f_\star, \mathbf{Z})\right)
\end{aligned} \tag{A.1}$$

The Subsection A.2 will treat the empirical process term $(\mathbb{E} - \mathbb{E}_n)\left(\ell(\hat{f}_n, \mathbf{Z}) - \ell(f_\star, \mathbf{Z})\right)$. For $\mathbb{E}_n\left(\ell(\hat{f}_n, \mathbf{Z}) - \ell(f_\star, \mathbf{Z})\right)$, Bernstein's inequality (see the Additional Theorems secion), Lemma 10 of Farrell et al. (2021) to get the following with probability at least $1 - 2e^{-\gamma}$

$$\begin{aligned}
\mathbb{E}_n\left(\ell(f_n, \mathbf{Z}) - \ell(f_\star, \mathbf{Z})\right) &\leq \mathbb{E}\left(\ell(f_n, \mathbf{Z}) - \ell(f_\star, \mathbf{Z})\right) + \sqrt{\frac{2M^2\|f_n - f_\star\|_\infty \gamma}{n}} + \frac{14M^2\gamma}{3n} \\
&\leq 2\mathbb{E}\left(\|f_n - f_\star\|^2_\infty\right) + \sqrt{\frac{2M^2\|f_n - f_\star\|_\infty \gamma}{n}} + \frac{14M^2\gamma}{3n} \\
&\leq 2\epsilon_n^2 + \epsilon_n\sqrt{\frac{2M^2\gamma}{n}} + \frac{14M^2\gamma}{3n}
\end{aligned} \tag{A.2}$$

We need to have $\gamma = \gamma(n) = o(n)$ for fulfilling the Bernstein's inequality requirement for γ. Note that for the context of the usage of the Bernstein's inequality, we have $d_i = \left(\ell(\hat{f}_n, \mathbf{Z}_i) - \ell(f_\star, \mathbf{Z}_i)\right) - \mathbb{E}\left(\ell(\hat{f}_n, \mathbf{Z}) - \ell(f_\star, \mathbf{Z})\right)$. The term $\sqrt{\frac{2M^2\|f_n - f_\star\|_\infty \gamma}{n}}$ comes from the term $\frac{\sqrt{2A_2\gamma}}{n}\left[1 + \frac{A_4\gamma}{6A_2^2}\right]$ in the original inequality statement that has been divided by n. As $\frac{A_4}{A_2}$ is of $\mathcal{O}(1)$, $A_2 = \mathcal{O}(n)$, and $\gamma = o(n)$, we then have $\left[1 + \frac{A_4\gamma}{6A_2^2}\right] \leq \sqrt{M}$ for sufficiently large M. Also, as $\frac{\sqrt{2A_2\gamma}}{n}$ is of $\mathcal{O}\left(\frac{1}{\sqrt{n}}\right)$, the statement $\frac{\sqrt{2A_2\gamma}}{n} \leq \sqrt{\frac{2M\|f_n - f_\star\|_\infty \gamma}{n}}$ holds if M is taken sufficiently large. The term $\frac{14M^2\gamma}{3n}$ is derived from the term $\frac{A_3\gamma}{3A_2 n}$ of the original inequality statement (after division by n). It is obvious that from the facts that M can be made arbitrarily large and $\frac{A_3}{A_2} = \mathcal{O}(1)$, we have $\frac{A_3\gamma}{3A_2 n} \leq \frac{14M^2\gamma}{3n}$. This is how the first inequality in (A.2) is derived. Lemma 10 of Farrell et al. (2021) is used in the second inequality. Once the empirical process term is controlled, the two bounds will be brought back together in Subsubsection A.2.4.

A.2 Localized Analysis

Given i.i.d. Rademacher draws $\xi_i = \pm 1$, with equal probability independent of the data the random variable $R_n\mathcal{F}$ for a function class $\mathcal{F}$ is defined as

$$R_n\mathcal{F} := \sup_{f\in\mathcal{F}} \frac{1}{n}\sum_{i=1}^{n} \xi_i f(\boldsymbol{x_i}),$$

and we write $\mathbb{E}(R_n\mathcal{F})$ and $\mathbb{E}_\xi(R_n\mathcal{F})$ for denoting the Rademacher complexity and empirical Rademacher complexity, respectively. In the case of the former, the expectation is taken over both data and ξ_i, whereas in the latter it is taken only on ξ_i conditioned on the data.

A.2.1 Step I: Quadratic Processes

In this section, it is going to be shown that $\rho_n(f - f_\star)$ is at most twice the $\|\cdot\|_{L^2(P)}$ bound, for certain function f outside a certain critical radius. This fact will be used later on.

Consider the quadratic process $\rho_n^2(f - f_\star) - \|f - f_\star\|_{L^2(P)}^2$. Let $g = (f - f_\star)^2$ be a function restricted to a radius $\|f - f_\star\|_{L^2(P)} \leq r$. This function g has bounded variance as

$$\mathrm{Var}(g) \leq \mathbb{E}(g^2) \leq \mathbb{E}((f - f_\star)^4) \leq 4M^2r^2.$$

Because $\|f_\star\|_\infty + \sum_{\substack{\vec{\alpha}\in(\mathbb{N}\cup\{0\})^d \\ \|\vec{\alpha}\|_1=1}} \|\nabla^{\vec{\alpha}} f_\star\|_\infty \leq 2M$, we have $|f - f_\star| \leq 2M$ and $|f + f_\star| \leq 2M$. As $g = (f - f_\star)(f + f_\star)$, it is clear that $|g| \leq 4M^2$. Using Theorem 2.1. from Bartlett et al. (2005), Lemma 2 from Farrell et al. (2021), and the fact that $|f - f_\star| \leq 2M$, we can deduce that the following holds with probability at least $1 - e^{-\gamma}$, for any $f \in \mathcal{F}$ with $\|f - f_\star\|_{L^2(P)} \leq r$,

$$\begin{aligned}
\rho_n^2(f - f_\star) - \mathbb{E}((f - f_\star)^2) &\leq 3\mathbb{E}\left(R_n\left\{g = (f - f_\star)^2 : f \in \mathcal{F}, \|f - f_\star\|_{L^2(P)} \leq r\right\}\right) \\
&\quad + 2Mr\sqrt{\frac{2\gamma}{n}} + \frac{16M^2\gamma}{3n} \\
&\leq 6M\mathbb{E}\left(R_n\left\{f - f_\star : f \in \mathcal{F}, \|f - f_\star\|_{L^2(P)} \leq r\right\}\right) \\
&\quad + 2Mr\sqrt{\frac{2\gamma}{n}} + \frac{16M^2\gamma}{3n}.
\end{aligned} \tag{A.3}$$

Suppose the radius r is taken in such way so that it satisfies both

$$r^2 \geq 6M\mathbb{E}\left(R_n\left\{f - f_\star : f \in \mathcal{F}, \|f - f_\star\|_{L^2(P)} \leq r\right\}\right), \tag{A.4}$$

and

$$r^2 \geq \frac{8M^2\gamma}{n} \tag{A.5}$$

Then from statement (A.3) we have

$$\begin{aligned}\rho_n^2(f-f_\star) &\leq \mathbb{E}\left((f-f_\star)^2\right) + r^2 + 2Mr\sqrt{\frac{2\gamma}{n}} + \frac{16M^2\gamma}{3n} \\ &\leq 2r^2 + 2Mr\sqrt{\frac{2\gamma}{n}} + \frac{16M^2\gamma}{3n} \\ &\leq (2r)^2.\end{aligned} \tag{A.6}$$

This means that for r above the "critical radius" (that is, r that satisfies both (A.4) and (A.5). It is used in Subsubsection A.2.3), the semi-norm $\rho_n(\cdot)$ is at most twice $\|\cdot\|_{L^2(P)}$ with probability at least $1-e^{-\gamma}$.

A.2.2 Step II: One Step Improvement

We are going to show that for a given bound on $\|\hat{f}_n - f\|_{L^2(P)}$ we can use this bound as information to obtain a "tighter" bound. Suppose that $\exists r_0 > 0$ such that $\|\hat{f}_n - f_\star\|_{L^2(P)}$, and, by the statement (A.6), $\rho_n(\hat{f}_n - f_\star) \leq 2r_0$. Given $\|f_\star\|_\infty + \sum_{\substack{\vec{\alpha}\in(\mathbb{N}\cup\{0\})^d \\ \|\vec{\alpha}\|_1=1}} \|\nabla^{\vec{\alpha}} f_\star\|_\infty \leq 2M$ and the fact that both $\rho_n(\cdot)$ and $\|\cdot\|_{L^2(P)}$ are bounded above by $\|\cdot\|_\infty$, we can take $r_0 = 2M$. By applying Theorem 2.1. from Bartlett et al. (2005) on

$$\mathcal{G} := \left\{ g = \ell(f,\boldsymbol{z}) - \ell(f_\star,\boldsymbol{z}) : f \in \mathcal{F}_{DNN,H_n}, \|f-f_\star\|_{L^2(P)} \leq 2r_0 \right\},$$

we find that with probability at least $1-2e^{-\gamma}$, the empirical process term $(\mathbb{E}-\mathbb{E}_n)\left(\ell(\hat{f}_n,\boldsymbol{Z}) - \ell(f_\star,\boldsymbol{Z})\right)$ from statement (A.1) is bounded above in this way

$$\begin{aligned}(\mathbb{E}-\mathbb{E}_n)\left(\ell(\hat{f}_n,\boldsymbol{Z}) - \ell(f_\star,\boldsymbol{Z})\right) &\leq 6\mathbb{E}_\xi(R_n\mathcal{G}) + \sqrt{\frac{2\mathrm{Var}\,(g)\,\gamma}{n}} + \frac{23\sup_{g\in\mathcal{G}}|g|\,\gamma}{3n} \\ &\leq 6\mathbb{E}_\xi(R_n\mathcal{G}) + \sqrt{\frac{2M^2r_0\gamma}{n}} + \frac{46M^2\gamma}{3n}\end{aligned} \tag{A.7}$$

where the middle term in the RHS of (A.7) is due to the following variance calculation with Lemma 10 from Farrell et al. (2021)

$$\mathrm{Var}(g) \leq \mathbb{E}(g^2) = \mathbb{E}((\ell(f,\boldsymbol{z}) - \ell(f_\star,\boldsymbol{z}))^2) \leq M^2\mathbb{E}((f-f_\star)^2) \leq M^2r_0^2$$

and the final inequality term of the RHS of (A.7) follows because the right side of Lemma 10 from Farrell et al. (2021) is bounded by M^2. Here the facts that Theorem 2.1. from Bartlett et al. (2005) is variance dependent, and that variance depends on the radius r_0, are important. It is this property which enables a sharpening of the rate with step-by-step reductions in the variance bound, as in Subsection A.2.4.

For dealing with the empirical Rademacher complexity term, note that the statement (A.7), the fact that the semi-norm $\rho_n(\cdot)$ is at most twice of $\|\cdot\|_{L^2(P)}$ with probability at least $1-e^{-\gamma}$ for r above the critical radius, Lemma 2 and 3 from Farrell et al. (2021) yield

$$\mathbb{E}_\xi(R_n\mathcal{G}) = \mathbb{E}_\xi\left(R_n\left\{g : g = \ell(f,\boldsymbol{z}) - \ell(f_\star,\boldsymbol{z}), f\in\mathcal{F}_{DNN,H_n}, \|f-f_\star\|_{L^2(P)} \leq r_0\right\}\right)$$

$$\leq M \cdot \mathbb{E}_\xi \left(R_n \left\{ f - f_\star : f \in \mathcal{F}_{DNN,H_n}, \|f - f_\star\|_{L^2(P)} \leq r_0 \right\} \right)$$

$$\leq M \cdot \mathbb{E}_\xi \left(R_n \left\{ f - f_\star : f \in \mathcal{F}_{DNN,H_n}, \rho_n(f - f_\star) \leq 2r_0 \right\} \right)$$

$$\leq M \cdot \inf_{0<\alpha<2r_0} \left\{ 4\alpha + \frac{12}{\sqrt{n}} \int_\alpha^{2r_0} \sqrt{\log(\mathcal{N}(\delta, \mathcal{F}_{DNN,H_n}|_{x_1,\ldots,x_n}, \rho_n(\cdot)))} d\delta \right\}$$

$$\leq M \cdot \inf_{0<\alpha<2r_0} \left\{ 4\alpha + \frac{12}{\sqrt{n}} \int_\alpha^{2r_0} \sqrt{\log(\mathcal{N}(\delta, \mathcal{F}_{DNN,H_n}|_{x_1,\ldots,x_n}, \|\cdot\|_{\ell_\infty}))} d\delta \right\}.$$

From Theorem 14.5 of Anthony and Bartlett (2009) and the way $\mathcal{F}_{DNN,H_n}$ is constructed, we have the following for sufficiently large n,

$$\inf_{0<\alpha<2r_0} \left\{ 4\alpha + \frac{12}{\sqrt{n}} \int_\alpha^{2r_0} \sqrt{\log(\mathcal{N}(\delta, \mathcal{F}_{DNN,H_n}|_{x_1,\ldots,x_n}, \|\cdot\|_{\ell_\infty}))} d\delta \right\}$$

$$\leq \inf_{0<\alpha<2r_0} \left\{ 4\alpha + \frac{12\sqrt{W_n}}{\sqrt{n}} \int_\alpha^{2r_0} \sqrt{\log\left(\frac{4en(MH_nb)W_n(MH_nb)^{L_d+1}}{\delta(MH_nb-1)} \right)} d\delta \right\}$$

$$\leq \inf_{0<\alpha<2r_0} \left\{ 4\alpha + \frac{12\sqrt{W_n}}{\sqrt{n}} \int_\alpha^{2r_0} \sqrt{\log\left(\frac{4en(MH_nb)W_n(MH_nb)^{2L_d}}{\delta(MH_nb-1)} \cdot \frac{4r_0}{\delta} \right)} d\delta \right\}$$

$$\leq \inf_{0<\alpha<2r_0} \left\{ 4\alpha + \frac{12\sqrt{W_nL_d}}{\sqrt{n}} \int_\alpha^{2r_0} \sqrt{\log\left(\frac{4en(MH_nb)W_n(MH_nb)^2}{\delta r_0(MH_nb-1)} \right) + \log\left(\frac{4r_0}{\delta} \right)} d\delta \right\}$$

$$\leq 8r_0 + \inf_{0<\alpha<2r_0} \left\{ \frac{12\sqrt{W_nL_d}}{\sqrt{n}} \int_\alpha^{2r_0} \sqrt{\log\left(\frac{4en(MH_nb)W_n(MH_nb)^2}{\delta r_0(MH_nb-1)} \right) + \log\left(\frac{4r_0}{\delta} \right)} d\delta \right\}$$

$$\leq 8r_0 + \inf_{0<\alpha<2r_0} \left\{ \frac{12\sqrt{W_nL_d}}{\sqrt{n}} \int_\alpha^{2r_0} \sqrt{\log\left(\frac{4en(MH_nb)W_n(MH_nb)^2}{r_0(MH_nb-1)} \right) + \log(4r_0) + \log\left(\frac{1}{\delta^2} \right)} d\delta \right\}$$

$$\leq 8r_0\sqrt{\frac{W_nL_d}{n}} + 48\sqrt{2}\sqrt{\frac{W_nL_d}{n}} \sqrt{\log\left(\frac{2eM^2b^2}{r_0} \right) + \log(nW_nH_n^2)}$$

$$\leq 8r_0\sqrt{\frac{W_nL_d}{n}} + 48\sqrt{2}\sqrt{\frac{W_nL_d}{n}} \sqrt{\log\left(\frac{2eM^2b^2}{r_0} \right) + 3\log(nW_nH_n)}.$$

where the second-last inequality comes from the fact that to get the infimum value of the second term in the RHS of the third-last inequality, we can take α as close to $2r_0$ as possible. If n is taken large enough to satisfy $r_0 > \frac{1}{nW_nH_n}$ and $(nW_nH_n)^2 > 2eM^2b^2$, then we get

$$\mathbb{E}_\xi \left(R_n \left\{ f - f_\star : f \in \mathcal{F}_{DNN,H_n}, \rho_n(f - f_\star) \leq 2r_0 \right\} \right)$$

$$\leq \inf_{0<\alpha<2r_0}\left\{4\alpha + \frac{12}{\sqrt{n}}\int_{\alpha}^{2r_0}\sqrt{\log(\mathcal{N}(\delta,\mathcal{F}_{DNN,H_n}|_{x_1,\ldots,x_n},\|\cdot\|_{\ell_\infty}))}d\delta\right\}$$

$$\leq 104\sqrt{3}r_0\sqrt{\frac{W_nL_d\log(W_nL_dn)}{n}}.$$

This leads to $\mathbb{E}_{\xi}(R_n\mathcal{G}) \leq 104\sqrt{3}Mr_0\sqrt{\frac{W_nL_d\log(W_nH_nn)}{n}}$. Take positive K such that $K \leq 104\sqrt{3}M$. Then, applying the last inequality to statement (A.7), the empirical process term satisfies the following inequality with the probability at least $1-2e^{-\gamma}$

$$(\mathbb{E}-\mathbb{E}_n)\left(\ell(\hat{f}_n,\boldsymbol{z})-\ell(f_\star,\boldsymbol{z})\right) \leq Kr_0\sqrt{\frac{W_nL_d\log(W_nH_nn)}{n}} + r_0\sqrt{\frac{2M^2\gamma}{n}} + \frac{46M^2\gamma}{3n} \quad \text{(A.8)}$$

We go back to the main decomposition (A.1) now. By plugging in inequalities (A.8) and (A.2) into (A.1), the following hold with probability at least $1-3e^{-\gamma}$,

$$\begin{aligned}
&\frac{1}{2}\|\hat{f}_n - f_\star\|^2_{L^2(P)} \\
&\leq Kr_0\sqrt{\frac{W_nL_d\log(W_nH_nn)}{n}} + r_0\sqrt{\frac{2M^2\gamma}{n}} + \frac{46M^2\gamma}{3n} + 2\epsilon_n^2 + \epsilon_n\sqrt{\frac{2M^2\gamma}{n}} + \frac{14M^2\gamma}{3n} \quad \text{(A.9)}\\
&\leq r_0\left(K\sqrt{\frac{W_nL_d\log(W_nH_nn)}{n}} + \sqrt{\frac{2M^2\gamma}{n}}\right) + 2\epsilon_n^2 + \epsilon_n\sqrt{\frac{2M^2\gamma}{n}} + \frac{20M^2\gamma}{n}.
\end{aligned}$$

The last inequality above shows that if we take $\gamma = \gamma(n)$ to be $\omega(1) = \gamma(n) = o(n)$, the upper bound of $\|\hat{f}_n - f_\star\|_{L^2(P)}$ decreases as n increases. Hence, whenever $\epsilon_n \ll r_0$ and $\sqrt{\frac{W_nL_d\log(W_nH_nn)}{n}} \ll r_0$, the knowledge $\|\hat{f}_n - f_\star\|_{L^2(P)} \leq r_0$, $r_0 = 2M$, and (A.9) imply (with high probability that becomes higher for larger values of n) that $\exists r_1 \ll r_0$ such that $\|\hat{f}_n - f_\star\|_{L^2(P)} \leq r_1$. Therefore one can recursively improve the bound r to a fixed point/radius r_*, which describes the fundamental difficulty of the problem treated in the next subsubsection.

A.2.3 Step III: Critical Radius

Define the critical radius r_* to be the largest fixed point

$$r_* := \inf\left\{r>0 : 6M\mathbb{E}\left(R_n\left\{f-f_\star : f\in\mathcal{F}_{DNN,H_n}\|f-f_\star\|_{L^2(P)}\leq u\right\}\right) < u^2, \forall u\geq r\right\}.$$

By construction the radius u here obeys the statement (A.4), and thus so does $2r_*$.

Denote the event E (depending on the data) to be

$$E = \left\{\rho_n(f-f_\star)\leq 4r_*, \textit{ for all } f\in\mathcal{F} \textit{ and } \|f-f_\star\|_{L^2(P)}\leq 2r_*\right\},$$

where $\mathbb{1}_E$ is the indicator the event E holds. We know from the logic of Subsubsection A.2.1, specifically using the statement (A.6), $\mathbb{P}(\mathbb{1}_E = 1) \geq 1-e^{-\gamma}$ provided that $r_* \geq \sqrt{8}M\sqrt{\frac{\gamma(n)}{n}}$ and $\gamma = \gamma(n)$. The lower bound $\sqrt{8}M\sqrt{\frac{\gamma(n)}{n}}$ for r_* is needed to satisfy (A.5).

To derive the upper bound of r_*, note that from the definition of r_*, the law of double expectation, the fact that $\mathbb{1}_E$ is measurable with respect to the data, and the inequality (A.6), we

have

$$
\begin{aligned}
r_*^2 &\leq 6M\mathbb{E}\left(R_n\left\{f-f_\star : f\in\mathcal{F}_{DNN,H_n}, \|f-f_\star\|_{L^2(P)}\leq r_*\right\}\right)\\
&\leq 6M\mathbb{E}\left(\mathbb{E}_\xi\left(R_n\left\{f-f_\star : f\in\mathcal{F}_{DNN,H_n}, \|f-f_\star\|_{L^2(P)}\leq 2r_*\right\}\right)\right)\\
&\leq 6M\mathbb{E}\left(\mathbb{E}_\xi\left(R_n\left\{f-f_\star : f\in\mathcal{F}_{DNN,H_n}, \|f-f_\star\|_{L^2(P)}\leq 2r_*\right\}\mathbb{1}_E\right)\right)\\
&\quad + 6M\mathbb{E}\left(\mathbb{E}_\xi\left(R_n\left\{f-f_\star : f\in\mathcal{F}_{DNN,H_n}, \|f-f_\star\|_{L^2(P)}\leq 2r_*\right\}\right)(1-\mathbb{1}_E)\right)\\
&\leq 6M\mathbb{E}\left(\mathbb{E}_\xi\left(R_n\left\{f-f_\star : f\in\mathcal{F}_{DNN,H_n}, \rho_n(f-f_\star)\leq 4r_*\right\}\right)\right)\\
&\quad + 6M\mathbb{E}\left(\mathbb{E}_\xi\left(R_n\left\{f-f_\star : f\in\mathcal{F}_{DNN,H_n}, \rho_n(f-f_\star)\leq 4r_*\right\}\right)(1-\mathbb{1}_E)\right).
\end{aligned}
$$

Then, we can follow the same steps as in Subsubsection A.2.2, the upper bounds of each of the two empirical Rademacher complexity terms in the last inequality can be derived and yield $r_*^2 \leq 2MKr_*\sqrt{\frac{W_nL_d\log(W_nH_nn)}{n}}\left(1+\frac{1}{n}\right)$ and thus we get

$$
\sqrt{8}M\sqrt{\frac{\gamma(n)}{n}} \leq r_* \leq 3MK\sqrt{\frac{W_nL_d\log(W_nH_nn)}{n}}. \tag{A.10}
$$

Note that the bounding of r_* in (A.10) is meant to be the smallest radius that allows the satisfaction of both (A.4) and (A.5). The lower bound of r_* can be seen as the upper bound with the smallest order for both the $L^2(P)$ and ρ_n distances between $\hat{f}_n$ and $f_\star$. Note also that we must take $\gamma(n)$ such that its complexity order must be at most $W_nL_d\log(W_nH_nn)$ to make (A.10) valid.

A.2.4 Step IV: Localization

Divide $\mathcal{F}_{DNN,H_n}$ into spherical segments of increasing radius by the following ball intersections

$$
B_{L^2(P)}(f_\star,\bar{r}), B_{L^2(P)}(f_\star,2\bar{r})\setminus B_{L^2(P)}(f_\star,\bar{r}), \ldots, B_{L^2(P)}\left(f_\star,2^l\bar{r}\right)\setminus B_{L^2(P)}\left(f_\star,2^{l-1}\bar{r}\right),
$$

where $l \leq \log_2\left(\frac{2M}{\sqrt{\log(n)/n}}\right)$. Suppose $\bar{r} > r_*$. Then, for each shell, Subsubsection A.2.1 implies that with probability at least $1-2le^{\gamma}$, for a suitable choice of j,

$$
\|f-f_\star\|_{L^2(P)} \leq 2^j\bar{r} \implies \rho_n(f-f_\star)\leq 2^{j+1}\bar{r}.
$$

Suppose that for some $j\leq l$,

$$
\hat{f}_n \in B_{L^2(P)}\left(f_\star,2^j\bar{r}\right)\setminus B_{L^2(P)}\left(f_\star,2^{j-1}\bar{r}\right).
$$

Then, applying Subsubsection A.2.2 logic (note again that the variance dependence of Theorem 2.1. of Bartlett et al. (2005) is crucial, as it enables accuracy sharpening by reducing r), especially the statement (A.9), the following holds with probability at least $1-3e^{-\gamma}$,

$$
\begin{aligned}
\|\hat{f}_n-f_\star\|_{L^2(P)}^2 &\leq 2\left(2^j\bar{r}\left(K\sqrt{\frac{W_nL_d\log(W_nH_nn)}{n}}+\sqrt{\frac{2M^2\gamma}{n}}\right)+\epsilon_n^2+\epsilon_n\sqrt{\frac{2M^2\gamma}{n}}+\frac{20M^2\gamma}{n}\right)\\
&\leq 2^{2j-2}\bar{r},
\end{aligned}
$$

if the following two conditions are satisfied

$$2\left(K\sqrt{\frac{W_nL_d\log(W_nH_nn)}{n}}+\sqrt{\frac{2M^2\gamma}{n}}\right)\leq\frac{1}{8}2^j\bar{r}$$

$$2\left(\epsilon_n^2+\epsilon_n\sqrt{\frac{2M^2\gamma}{n}}+\frac{20M^2\gamma}{n}\right)\leq\frac{1}{8}2^j\bar{r}^2.$$

These two hold for any j if we choose

$$\bar{r}=16\left(K\sqrt{\frac{W_nL_d\log(W_nH_nn)}{n}}+\sqrt{\frac{2M^2\gamma}{n}}\right)+\left(\sqrt{32}\epsilon_n+\sqrt{\frac{416M^2\gamma}{n}}\right)+r_*. \quad \text{(A.11)}$$

It is clear that $\bar{r}$ already satisfies both (A.4) and (A.5). Therefore with probability larger than $1-5le^{-\gamma}$, we can perform step-by-step argument for each subsequent spherical segment starting from the biggest segment to combine the results in Subsubsections A.2.1 and A.2.2 to get

$$\begin{aligned}
&\|\hat{f}_n-f_\star\|_{L^2(P)}\leq 2^l\bar{r}\ \textit{and}\ \rho_n(\hat{f}_n-f_\star)\leq 2^{l+1}\bar{r}\\
&\quad\textit{implies}\ \|\hat{f}_n-f_\star\|_{L^2(P)}\leq 2^{l-1}\bar{r}\ \textit{and}\ \rho_n(\hat{f}_n-f_\star)\leq 2^l\bar{r}\\
&\quad\quad\dots\\
&\quad\quad\quad\textit{implies}\ \|\hat{f}_n-f_\star\|_{L^2(P)}\leq \bar{r}\ \textit{and}\ \rho_n(\hat{f}_n-f_\star)\leq 2\bar{r},
\end{aligned}$$

and note that the "and" parts are proven by using Subsubsection A.2.1 and the implication parts are proven by using Subsubsection A.2.2. Thus with probability larger $1-5le^{-\gamma}$,

$$\|\hat{f}_n-f_\star\|_{L^2(P)}\leq\bar{r}, \quad \text{(A.12)}$$

$$\rho_n(\hat{f}_n-f_\star)\leq 2\bar{r}. \quad \text{(A.13)}$$

Hence, by choosing $\gamma=\log(5l)+\gamma'$ and using both (A.12) and (A.13) above, the equality (A.11), and the upper bound of r_* in the statement (A.10), the following is true with probability larger than $1-e^{-\gamma'}$ (the expression $1-e^{-\gamma'}$ is obtained after plugging in $l=\log_2\left(\frac{2M}{\sqrt{\log(n)/n}}\right)$ to the expression $1-5le^{-\gamma}$) and some constant $C_1>0$,

$$\begin{aligned}
\bar{r}&\leq 16\left(K\sqrt{\frac{W_nL_d\log(W_nH_nn)}{n}}+\sqrt{\frac{2M^2C_1(\log(\log(n))+\gamma')}{n}}\right)\\
&\quad+\left(\sqrt{32}\epsilon_n+\sqrt{\frac{416M^2C_1(\log(\log(n))+\gamma')}{n}}\right)+r_*\\
&\leq C'\left(\sqrt{\frac{W_nL_d\log(W_nH_nn)}{n}}+\sqrt{\frac{\log(\log(n))+\gamma'}{n}}+\epsilon_n\right)
\end{aligned}$$

for some absolute constant $C'>0$. We can thus conclude by taking $p=\gamma'$ that $\exists C''>0$ such that we have this inequality with probability at least $1-e^{-p}$,

$$\|\hat{f}_n-f_\star\|_{L^2(P)},\rho_n(\hat{f}_n-f_\star)\leq C''\left(\sqrt{\frac{W_nL_d\log(W_nH_nn)}{n}}+\sqrt{\frac{\log(\log(n))+p}{n}}+\epsilon_n\right)$$

where the order of $p=p(n)$ must be less than $W_nL_d\log(W_nH_nn)$ to validate (A.10). As the

order of W_n is less than $\mathcal{O}\left(H_n^2 L_d\right)$, we can simply replace W_n with H_n^2 in the term above. This completes the proof. □

B Proof of Theorem 3.2.1

The proof follows the logic of the proof of Theorem 3.1. in HG. The key difference here is that we have already obtained the estimation rate, i.e., $U(C', \epsilon_n)$. Thereby, the proof proceeds in two steps. First, by justifying all conditions of Theorem 2.11.23 in Van der Vaart and Wellner (1996), we show that a rescaled and shifted version of the empirical criterion function converges in distribution to a Gaussian process. Second, we apply the argmax continuous mapping theorem, i.e., Theorem 3.2.2. in Van der Vaart and Wellner (1996).

B.1 Gaussian Process as Limit

In this subsection, the main goal is that to prove all hypotheses of Theorem 2.11.23 from Van der Vaart and Wellner (1996) are satisfied. We construct a random sequence $h_n = U(C', \epsilon_n)^{-1}(\hat{f}_n - f_\star)$, where $f_\star \in \mathcal{F}$ and $\hat{f}_n \in \mathcal{F}_n$, where $\mathcal{F}$ and $\mathcal{F}_n$ are defined in (11) and (12), respectively. Note that Remark 3.2.1 guarantees that both Theorem 3.1.1 and the ϵ_n-order (9) hold for every $f_\star \in \mathcal{F}_{DNN}$ being estimated by $\hat{f}_n \in \mathcal{F}_n$. The expression of $U(C', \epsilon_n)$ remains the same with different constant C', as we have $\mathcal{F}_n \subseteq \mathcal{F}_{DNN,2H_n}$ instead of $\mathcal{F}_{DNN,H_n}$.

From the least square estimator $\ell(f, \boldsymbol{z}_i) = -\frac{1}{2}(y_i - f(\boldsymbol{x}_i))^2$, define the function $\ell_f(\mathbf{X}, \varepsilon) = 2(f - f_\star)(\mathbf{X})\varepsilon - (f_\star - f)^2(\mathbf{X})$ by using (1). Consider the set $\left\{U(C', \epsilon_n)^{-1}(\ell_{f_\star + U(C', \epsilon_n)h} - \ell_{f_\star}) : h \in K\right\}$, where K is an arbitrary $\|\cdot\|_{L^2(P)}$-compact subset of $\mathcal{F}$, which implies $\|\cdot\|_{L^2(P)}$-total boundedness.

Define also the criterion function

$$e_{n,h} := U(C', \epsilon_n)^{-1}(\ell_{f_\star + U(C', \epsilon_n)h} - \ell_{f_\star}) = 2h(\mathbf{X})\varepsilon - U(C', \epsilon_n)h^2(\mathbf{X}),$$

and the related function class

$$\mathcal{E}_n := \{e_{n,h} : h \in K\}. \tag{B.1}$$

The envelope functions E_n of $\mathcal{E}_n$ satisfy $\forall h \in K$, $e_{n,h}(\mathbf{X}, \varepsilon) \leq E_n(\mathbf{X}, \varepsilon)$. We now have to construct appropriate envelope functions. By Lemma 2 of Chen and Shen (1998), $\|h\|_\infty \lesssim \|h\|_{L^2(P)}^{2/(2+d)}$. By $\|\cdot\|_{L^2(P)}$-compactness of K, as a compact set in a metric space is always bounded, $\exists 1 < M_2 < \infty$ such that $\forall h \in K$, $\|h\|_{L^2(P)} < M_2$. This leads to

$$|e_{n,h}| \leq 2|\varepsilon||h(\boldsymbol{x})| + U(C', \epsilon_n)|h(\boldsymbol{x})|^2 \leq 2|\varepsilon|M_2 + U(C', \epsilon_n)M_2^2,$$

and thus we can choose the following function as the envelope

$$E_n(\mathbf{X}, \varepsilon) = 2|\varepsilon|M_2 + 2U(C', \epsilon_n)M_2^2,$$

where the constant 2 is added to help simplifying the metric entropy calculation.

From the way E_n is defined and the fact $\mathbb{E}(\varepsilon) = \sigma^2 < \infty$ (bounded second moment), we

have

$$\mathbb{E}(E_n^2(\mathbf{X},\varepsilon)) = O(1), \text{ and } \mathbb{E}(E_n^2 \mathbb{1}_{\{E_n > \eta\sqrt{n}\}}) \to 0, \forall \eta > 0. \tag{B.2}$$

Note also that

$$\begin{aligned}(e_{n,h_1} - e_{n,h_2})^2 &= \left[\left(2\varepsilon - U(C',\epsilon_n)(h_1+h_2)\right)(h_1-h_2)\right]^2 \\ &\leq \left[\left|2|\varepsilon| + 2U(C',\epsilon_n)M_2\right|(h_1-h_2)\right]^2\end{aligned}$$

where the last inequality comes from the simple fact that $|2\varepsilon - U(C',\epsilon_n)(h_1+h_2)| \leq |2|\varepsilon| + 2U(C',\epsilon_n)M_2|$. Therefore, by taking the expectation on both sides, we obtain

$$\mathbb{E}((e_{n,h_1} - e_{n,h_2})^2) \lesssim \mathbb{E}((h_1-h_2)^2),$$

which leads to

$$\sup_{\|h_1-h_2\|_{L^2(P)}<\delta_n} \mathbb{E}((e_{n,h_1} - e_{n,h_2})^2) \leq O(\delta_n) \to 0, \forall \delta_n \downarrow 0. \tag{B.3}$$

(B.2) and (B.3) lead to the satisfaction of the first three conditions of Theorem 2.11.23. of Van der Vaart and Wellner (1996). The second thing we need to prove is

$$\int_0^{\delta_n} \sqrt{\log(\mathcal{N}(z\|E_n\|_{L^2(P)}, \mathcal{E}_n, L^2(P)))}dz \to 0, \forall \delta_n \downarrow 0. \tag{B.4}$$

First, it is going to be shown that the $\|\cdot\|_{L^2(P)}$ metric entropy associated with $\mathcal{E}_n$ is bounded with respect to the $\|\cdot\|_\infty$ metric entropy of $\mathcal{F}$. By simple arithmetic,

$$\begin{aligned}&|e_{n,h_1} - e_{n,h_2}| = |h_1 - h_2|\left|2\varepsilon - U(C',\epsilon_n)(h_1+h_2)\right|, \\ \implies &|e_{n,h_1} - e_{n,h_2}| \leq |h_1 - h_2|\left|2|\varepsilon| + 2U(C',\epsilon_n)M_2\right|, \\ \implies &|e_{n,h_1} - e_{n,h_2}| \leq \|h_1 - h_2\|_\infty E_n(\boldsymbol{x},\varepsilon),\end{aligned}$$

and the last inequality implies that there exists a constant $c' > 0$ such that

$$\mathcal{N}(z\|E_n\|_{L^2(P)}, \mathcal{E}_n, L^2(P)) \leq \mathcal{N}\left(c'z, \mathcal{F}, \|\cdot\|_\infty\right). \tag{B.5}$$

Because $\mathcal{F}$ is a subset of the weighted Sobolev space $\mathbb{H}^{\lfloor d/2\rfloor+2,2}(\mathbb{R}^d, \langle x\rangle^3) \mid \mathbb{X}$ being restricted to $\mathbb{X}$, the first statement of Corollary 4 from Nickl and Pötscher (2007) gives us

$$\log\left(\mathcal{N}\left(c'z, \mathcal{F}, \|\cdot\|_\infty\right)\right) \lesssim \left(\frac{1}{c'z}\right)^{\frac{d}{\lfloor 0.5d\rfloor+2}} \tag{B.6}$$

and both (B.5) and (B.6) together give

$$\int_0^1 \sqrt{\log\left(\mathcal{N}(z\|E_n\|_{L^2(P)}, \mathcal{E}_n, L^2(P))\right)}dz < \infty, \forall n$$

which means that (B.4) is satisfied.

The last hypothesis that has to be proven is the pointwise convergence of the covariance functions $Pe_{n,s}e_{n,t} - Pe_{n,s}Pe_{n,t}$ on $K \times K$. Note that for any $s, t \in K$

$$Pe_{n,s}e_{n,t} = \mathbb{E}\left(\left(2s(\mathbf{X})\varepsilon - U(C',\epsilon_n)s^2(\mathbf{X})\right)\left(2t(\mathbf{X})\varepsilon - U(C',\epsilon_n)t^2(\mathbf{X})\right)\right) \xrightarrow{n\to\infty} 4\sigma^2\mathbb{E}\left(s(\mathbf{X})t(\mathbf{X})\right)$$

$$Pe_{n,s}Pe_{n,t} = \mathbb{E}\left(\left(2s(\boldsymbol{X})\varepsilon - U(C',\epsilon_n)s^2(\boldsymbol{X})\right)\right)\mathbb{E}\left(\left(2t(\boldsymbol{X})\varepsilon - U(C',\epsilon_n)t^2(\boldsymbol{X})\right)\right) \xrightarrow{n\to\infty} 0$$

and hence

$$Pe_{n,s}e_{n,t} - Pe_{n,s}Pe_{n,t} \xrightarrow{n\to\infty} 4\sigma^2\mathbb{E}\left(s(\boldsymbol{X})t(\boldsymbol{X})\right) \text{ pointwise.} \tag{B.7}$$

From the fact that $\mathcal{E}_n$ in (B.1) is indexed by an $\|\cdot\|_{L^2(P)}$-totally bounded function space $K \subseteq \mathcal{F}$, (B.2), (B.3), (B.4) and (B.7), we deduce using Theorem 2.11.23 from Van der Vaart and Wellner (1996) that the sequence of empirical processes $\{\mathbb{G}_n(e_{n,h}) : h \in K\}$, where $\mathbb{G}_n(e_{n,h}) \coloneqq \frac{1}{\sqrt{n}}\sum_{i=1}^{n}(e_{n,h}(\boldsymbol{X}_i) - Pe_{n,h})$, is asymptotically tight in $\ell^\infty(K)$ and converges in distribution to a tight mean zero Gaussian process $\mathbb{G}$ with covariance $4\sigma^2\mathbb{E}\left(s(\boldsymbol{X})t(\boldsymbol{X})\right)$.

B.2 Sample Path Upper Semi Continuity and a Unique Maximum of Gaussian Process

We are going to show that two required conditions from Theorem 3.2.2 of Van der Vaart and Wellner (1996) hold; every sample path of $h \mapsto \mathbb{G}(h)$ is upper semi continuous and $\{\mathbb{G}(f) : f \in K \subseteq \mathcal{F}\}$ has a unique maximum h^*, which is a tight random map.

To prove the first condition, the key tool is Corollary 4.15 from Adler (1990), a centered Gaussian process $\mathbb{G}$ is continuous if; its domain (the index space) K is totally bounded with respect to $d_{\mathbb{G}}(s,t) \coloneqq \sqrt{\mathbb{E}\left((\mathbb{G}(s) - \mathbb{G}(t))^2\right)}$, and;

$$\int_0^\infty \sqrt{\log\left(\mathcal{N}(z, K, d_{\mathbb{G}})\right)}dz < \infty. \tag{B.8}$$

First of all, note that

$$\begin{aligned}
d_{\mathbb{G}}(s,t) &= \sqrt{\mathbb{E}\left((\mathbb{G}(s) - \mathbb{G}(t))^2\right)} \\
&= \sqrt{\mathbb{E}\left(\mathbb{G}^2(s)\right) + \mathbb{E}\left(\mathbb{G}^2(t)\right) - 2\mathbb{E}\left(\mathbb{G}(s)\mathbb{G}(t)\right)} \\
&= \sqrt{4\sigma^2\mathbb{E}\left(s^2\right) + 4\sigma^2\mathbb{E}\left(t^2\right) - 8\sigma^2\mathbb{E}\left(st\right)} \\
&= 2\sigma\|s - t\|_{L^2(P)},
\end{aligned}$$

and thus $\|\cdot\|_{L^2(P)}$-compactness of K implies it also has $d_{\mathbb{G}}$-total-boundedness. We are left with (B.8). By the fact that $K \subseteq \mathcal{F}$, the way $\mathcal{F}$ is defined in (11), and Corollary 4 of Nickl and Pötscher (2007), we have

$$\log\left(\mathcal{N}\left(z, \mathcal{F}, \|\cdot\|_{L^2(P)}\right)\right) \lesssim \left(\frac{1}{z}\right)^\alpha, \text{ for } \alpha < 2,$$

which together with the fact that $d_{\mathbb{G}}(s,t) = 2\sigma\|s - t\|_{L^2(P)}$ gives (B.8). This leads to the continuity (and hence upper semi continuity) of $\mathbb{G}$.

To prove the arg max uniqueness of $\mathbb{G}$, the main tool is Lemma 2.6 from Kim et al. (1990), which states that sufficient conditions of the unique arg max h^* of a Gaussian process $t \mapsto \mathbb{G}(t)$ are; sample path continuity; index space compactness, and; $\forall s \neq t, \text{Var}\left(\mathbb{G}(s) - \mathbb{G}(t)\right) \neq 0$.

The continuity of $\mathbb{G}$ is already proven in the previous subsection. The compactness of

K is obvious. Note that the continuity of $\mathbb{G}$ over the compact index space K also implies the existence of its arg max $h^{(max)}$. For the last requirement, note that $\text{Var}\left(\mathbb{G}(s) - \mathbb{G}(t)\right) = 4\sigma^2\mathbb{E}\left(s^2 + t^2\right) - 8\sigma^2\mathbb{E}\left(st\right) = 4\sigma^2\mathbb{E}\left((s-t)^2\right) \neq 0$. We thus have the uniqueness of $h^{(max)}$.

It is clear that $e_{n,h}$ uniformly converges to $2h(\mathbf{X})\varepsilon$ uniformly for every compact $K \subseteq \mathcal{F}$, as we have $U(C', \epsilon_n) = o(1)$. In page 286 of Van der Vaart and Wellner (1996), it is stated that the uniform convergence of the criterion function $e_{n,h}$ in compacta is a sufficient condition for uniform tightness of h_n. Now, to fully apply Theorem 3.2.2. of Van der Vaart and Wellner (1996), we need to prove that $\mathbb{G}_n(e_{n,h_n}) \geq \sup_{h\in\mathcal{F}} \mathbb{G}_n(e_{n,h})$. Now, note that from the fact $(e_{n,h_1} - e_{n,h_2})^2 \leq \left[|2\varepsilon + 2U(C', \epsilon_n)M_2|\,(h_1 - h_2)\right]^2$, $\forall h_1, h_2$ and (B.3), we can deduce that

$$\frac{U(C', \epsilon_n)^{-1}}{\sqrt{n}} \sum_{i=1}^{n} \left(\ell_{\hat{f}_n}(\mathbf{X}_i, \varepsilon_i) - \mathbb{E}\left[\ell_{\hat{f}_n}(\mathbf{X}, \varepsilon)\right]\right) \geq$$

$$\sup_{h\in\mathcal{F}} \frac{U(C', \epsilon_n)^{-1}}{\sqrt{n}} \sum_{i=1}^{n} \left(\ell_{f_\star + \frac{h}{r_n}}(\mathbf{X}_i, \varepsilon_i) - \mathbb{E}\left[\ell_{f_\star + \frac{h}{r_n}}(\mathbf{X}, \varepsilon)\right]\right) - o_P(1)$$

for $U(C', \epsilon_n)^{-1} = o(\sqrt{n})$ (which holds following the fact that $o(1) = U(C', \epsilon_n) = \omega(n^{-0.25})$, which is an implication of (9)). As it has been discussed in Horel and Giesecke (2020), the condition above is equivalent to $\mathbb{G}_n\left(e_{n,h_n}\right) \geq \sup_{h\in\mathcal{F}} \mathbb{G}_n\left(e_{n,h}\right) - o_P(1)$. Hence, we can use Theorem 3.2.2. of Van der Vaart and Wellner (1996) to deduce Theorem 3.2.1 above.

C Proof of Theorem 3.2.2

To prove this theorem, both Theorem 3.2.1 and Theorem 2. from Römisch (2004) are utilized. The statistical test functional $\mathcal{T} : \mathcal{F} \to \underline{\mathcal{F}}$ can be viewed as a mapping between two metrizable function spaces. Hence, we can compute the second-order Hadamard directional derivative of $\mathcal{T}$ at $u_\star$ in the direction ψ tangential to $\mathcal{F}$, which has the following form

$$\mathcal{T}''_{u_\star}[\psi] = \lim_{t\downarrow 0, n\to\infty} \frac{\mathcal{T}[u_\star + t\psi_n] - \mathcal{T}[u_\star] - t\mathcal{T}'_{u_\star}[\psi_n]}{\frac{1}{2}t^2},$$

where $\mathcal{T}'_{u_\star}[\psi]$ indicates the first-order Hadamard directional derivative with the same direction point and tangential space, written as

$$\mathcal{T}'_{u_\star}[\psi] = \lim_{t\downarrow 0, n\to\infty} \frac{\mathcal{T}[u_\star + t\psi_n] - \mathcal{T}[u_\star]}{t},$$

and both derivatives are defined for every sequence $\psi_n \to \psi$ and $t > 0$ satisfying $u_\star + t\psi_n \in \mathcal{F}$. By the simple limit calculation, we have

$$\mathcal{T}'_{u_\star}[\psi] = 2\int_{\mathbb{X}} \left(\frac{\partial u_\star}{\partial x_j} \cdot \frac{\partial \psi}{\partial x_j}\right)(\mathbf{X}) dP(\mathbf{X})$$

and replacing ψ with ψ_n and plugging the explicit form of $\mathcal{T}'_{u_\star}[\psi_n]$ to the limit expression of $\mathcal{T}''_{u_\star}[\psi]$ give $\mathcal{T}''_{u_\star}[\psi] = 2\mathcal{T}[\psi]$.

We now return to our estimation context. By Theorem 3.2.1 and the simple fact

$U(C', \epsilon_n)^{-1} \to \infty$, we can use Theorem 2. from Römisch (2004) to get

$$U(C', \epsilon_n)^{-2} \left(\mathcal{T}[\hat{f}_n] - \mathcal{T}[f_\star] - \mathcal{T}'_{f_\star}[\hat{f}_n - f_\star] \right) \xrightarrow{d} \mathcal{T}[h^{(max)}].$$

Note that the randomness in the context of $\mathcal{F}$ estimation comes from members of $\mathcal{F}$ themselves, as different choices of $\mathcal{F}$ members definitely lead to different mapping results by $\mathcal{T}$. Under the null hypothesis (3), $\mathcal{T}[f_\star] = 0$. As the continuous mapping theorem and Theorem 3.2.1 give $\mathcal{T}'_{f_\star}[\hat{f}_n - f_\star] \xrightarrow{p} 0$, the conclusion of Theorem 3.2.2 is proved by the Slutsky's Theorem. □

D Proof of Theorem 3.2.3

First of all, define $\partial_{2,j}(f) := \left(\frac{\partial f}{\partial x_j} \right)^2$ (note that $\mathbb{E}\left(\partial_{2,j}(f)\right) = \mathcal{T}[f]$). Write

$$U(C', \epsilon_n)^{-2} \left(\rho_n^2 \left(\partial_{2,j}(\hat{f}_n) \right) - \mathbb{E}\left(\partial_{2,j}(f_\star)\right) \right) =$$

$$U(C', \epsilon_n)^{-2} \rho_n^2 \left(\partial_{2,j}(\hat{f}_n) \right) =$$

$$U(C', \epsilon_n)^{-2} \left(\frac{1}{\sqrt{n}} \mathbb{M}_n \left(\partial_{2,j}(\hat{f}_n) - \partial_{2,j}(f_\star) \right) + \frac{1}{\sqrt{n}} \mathbb{M}_n \left(\partial_{2,j}(f_\star) \right) + \left(\mathcal{T}[\hat{f}_n] \right) \right) \tag{D.1}$$

under the null hypothesis (3), where $\mathbb{M}_n$ is the empirical process operator defined as $\mathbb{M}_n(f) := \sqrt{n}(\rho_n^2 - \mathbb{E})(f)$. By the way ϵ_n-order is taken in (9), we have $U(C', \epsilon_n)^{-2} = o(n^{1/2})$. Note also that by the CLT, $\mathbb{M}_n(\partial_{2,j}(f_\star)) \xrightarrow{d} N\left(0, \mathrm{Var}\left(\partial_{2,j}(f_\star)\right)\right) = 0$ almost surely under the null hypothesis. Hence, $\frac{U(C', \epsilon_n)^{-2}}{\sqrt{n}} \mathbb{M}_n \left(\partial_{2,j}(f_\star) \right) \xrightarrow{p} 0$.

To deal with the first term of (D.1), we will use Theorem 2.11.23. from Van der Vaart and Wellner (1996) to show that the process $\{\mathbb{M}_n e_{n,h} : h \in K \subseteq \mathcal{F}\}$, for an arbitrary $\|\cdot\|_{L^2(P)}$-compact K and index space $\mathcal{E}_n = \{e_{n,h} : h \in K\}$. Take an index function $e_{n,h} := \partial_{2,j}(f_\star + U(C', \epsilon_n)h) - \partial_{2,j}(f_\star) = U(C', \epsilon_n)\partial_{2,j}(h)$ converging to a tight Gaussian process, where the last equality for $e_{n,h}$ holds under the null hypothesis. By the way $\mathcal{F}$ is defined in (11), $\exists M_3 > 0$ such that $\forall h \in K$, $\left\| \frac{\partial h}{\partial x_j} \right\|_{L^2(P)} \leq M_3$. Because $\frac{\partial h}{\partial x_j} \in \mathcal{C}^1(\mathbb{X})$, Lemma 2. from Chen and Shen (1998) gives $\exists M_3 > 0$ such that $\forall h \in K$, $\frac{\partial h}{\partial x_j} \leq M_3$.

Then, the envelope function E_n can be defined as

$$E_n := 2M_3 U(C', \epsilon_n)^2,$$

which satisfies

$$\mathbb{E}(E_n^2) = O(1), \text{ and, } \mathbb{E}(E_n^2 \mathbb{1}_{E_n > \eta\sqrt{n}}) \to 0, \ \forall \eta > 0.$$

As $e_{n,s} - e_{n,t} = U(C', \epsilon_n)^2(\partial_{2,j}(s) - \partial_{2,j}(t)) \to 0$ pointwise for any $s, t \in K \times K$, we have

$$\sup_{\|s-t\|_{L^2(P)} \leq \delta_n} \mathbb{E}\left((e_{n,s} - e_{n,t})^2\right) \to 0, \ \forall \text{ sequence } \delta_n \to 0,$$

and also,

$$|e_{n,s} - e_{n,t}| \leq U(C', \epsilon_n)^2(\partial_{2,j}(s) - \partial_{2,j}(t))$$

$$\implies |e_{n,s} - e_{n,t}| \leq |E_n(\boldsymbol{x})| \left| \frac{\partial(s-t)}{\partial x_j} \right|.$$

Thus we can use the same reasoning as in Lemma 3.2.1 proof to get

$$\mathcal{N}\left(\varepsilon\|E_n\|_{L^2(P)}, \mathcal{E}_n, L^2(P)\right) \leq \mathcal{N}\left(\frac{\varepsilon}{2}, \mathcal{F}', \|\cdot\|_\infty\right) \lesssim \left(\frac{2}{\varepsilon}\right)^{\frac{d}{\lfloor 0.5d \rfloor + 1}},$$

where $\mathcal{F}' = \left\{\frac{\partial f}{\partial x_j} : f \in \mathcal{F}\right\} \subset \mathbb{H}^{\lfloor 0.5d \rfloor + 1,2}(\mathbb{R}^d, \langle x \rangle) \mid \mathbb{X}$, and the last asymptotic inequality comes from Corollary 4. of Nickl and Pötscher (2007). This implies

$$\int_0^{\delta_n} \sqrt{\log\left(\mathcal{N}\left(\varepsilon\|E_n\|_{L^2(P)}, \mathcal{E}_n, L^2(P)\right)\right)} d\varepsilon \to 0, \ \forall \text{ sequence } \delta_n \downarrow 0.$$

Hence, by the conclusion of Theorem 2.11.23 from Van der Vaart and Wellner (1996), $\mathbb{M}_n\left(\partial_{2,j}(\hat{f}_n) - \partial_{2,j}(f_\star)\right)$ is asymptotically tight in $\ell^\infty(K)$ and converges to a tight centered Gaussian process. This leads to $\frac{U(C',\epsilon_n)^{-2}}{\sqrt{n}}\mathbb{M}_n\left(\partial_{2,j}(\hat{f}_n) - \partial_{2,j}(f_\star)\right) \xrightarrow{p} 0$ (this holds as we have $o(1) = U(C', \epsilon_n) = \omega(n^{-0.25})$ from the statement (10), which implies $\omega(1) = U(C', \epsilon_n)^{-2} = o(n^{0.5})$). By the Slutsky's Theorem, the last term of (D.1), and Theorem 3.2.2, $U(C', \epsilon_n)^{-2}\rho_n^2\left(\partial_{2,j}(\hat{f}_n)\right) \xrightarrow{d} U(C', \epsilon_n)^{-2}\mathcal{T}[\hat{f}_n] \xrightarrow{d} \mathcal{T}[h^{(max)}]$. □

E Additional Theorems

In this section, we use some of the theorems or the formulas that (i) are central in the proving section, or; (ii) have different versions so the specific version being used in this paper has to be written down here (such as the Bernstein's Inequality).

Theorem 14.5 of Anthony and Bartlett (2009) *Suppose that a feed-forward real-output multi-layer network has the following properties*

- *The network has $l \geq 2$ layers, with connections only between adjacent layers.*
- *There are W weights in the network.*
- *For some b, each computation unit maps into the interval $[-b, b]$ and each computation unit in the first layer has a non-decreasing activation function.*
- *There are constants $V > 0$ and $L > 1/V$, so that for each unit in all but the first layer of the network, the vector w of weights associated with that unit has $\|w\|_1 \leq V$, and the unit's activation function $s : \mathbb{R} \to [-b, b]$ satisfies the Lipschitz condition $|s(\alpha_1) - s(\alpha_2)| \leq L|\alpha_1 - \alpha_2|$ for all $\alpha_1, \alpha_2 \in \mathbb{R}$. (Here, $\|w\|_1$ is the sum of absolute values of the entries of w.)*

For the class $\mathcal{F}$ of functions computed by the network described above, if $\epsilon \leq 2b$, then

$$\mathcal{N}_\infty(\epsilon, \mathcal{F}, m) \leq \left(\frac{4embW(LV)^l}{\epsilon(LV-1)}\right)^W.$$

.

Bernstein's Inequality *Let $d_1, \ldots, d_n$ be independent, zero-mean random variables. Suppose that*

$$\mathbb{E}\left(\left|d_i^k\right|\right) \leq \frac{k!}{4!}\left(\frac{L}{5}\right)^{k-4}, \text{ for all integer } k > 3.$$

Define $A_k := \sum_{i=1}^n \mathbb{E}\left(d_i^k\right)$. Then, for $0 < \sqrt{\gamma} \leq \frac{5\sqrt{2A_2}}{4L}$, the following holds with probability larger than $1 - 2e^{-\gamma}$

$$\left|\sum_{i=1}^n d_i - \frac{A_3\gamma}{3A_2}\right| < \sqrt{2A_2\gamma}\left[1 + \frac{A_4\gamma}{6A_2^2}\right]$$

Theorem 2.11.23 of Van der Vaart and Wellner (1996) *For each n, let $\mathcal{E}_n = \{e_{n,t} : t \in T\}$ be a class of measurable functions indexed by a totally bounded semimetric space (T, ρ). Suppose that given envelope functions E_n, assume that*

$$\mathbb{E}\left(E_n^2\right) = O(1),$$
$$\mathbb{E}\left(E_n^2 \mathbb{1}_{\{E_n > \eta\sqrt{n}\}}\right) \to 0, \forall \eta \to 0,$$
$$\sup_{\rho(s,t) < \delta_n} \mathbb{E}\left((e_{n,s} - e_{n,t})^2\right) \to 0, \forall \delta_n \downarrow 0.$$

Suppose also that

$$\int_0^{\delta_n} \sqrt{\log(\mathcal{N}(z\|E_n\|_{L^2(P)}, \mathcal{E}_n, L^2(P)))}dz \to 0, \forall \delta_n \downarrow 0.$$

Then, the sequence $\{\mathbb{G}_n e_{n,t} : t \in T\}$, *with* $\mathbb{G}_n e_{n,t} := n^{-1/2} \sum_{i=1}^{n} (e_{n,t}(\mathbf{X}_i, \varepsilon_i) - \mathbb{E}(e_{n,t}))$, *is asymptotically tight in* $\ell^\infty(T)$ *and converges in distribution to a tight Gaussian process provided that the sequence of covariance functions* $\mathbb{E}(e_{n,s}e_{n,t}) - \mathbb{E}(e_{n,s})\ E(e_{n,t})$ *converges pointwise on* $T \times T$.

Theorem 3.2.2 of Van der Vaart and Wellner (1996) *Let* $\mathbb{G}_n$, $\mathbb{G}$ *be stochastic processes indexed by a metric space* $\mathcal{F}$ *such that* $\mathbb{G}_n \xrightarrow{d} \mathbb{G}$ *in* $\ell^\infty(K)$ *for every compact* $K \subseteq \mathcal{F}$. *Suppose that almost all sample paths* $h \mapsto \mathbb{G}(h)$ *are upper semicontinuous and possess a unique maximum at a (random) point* $\hat{h}$, *which as a random map in* $\mathcal{F}$ *is tight. If the sequence* $\hat{h}_n$ *is uniformly tight and satisfies* $\mathbb{G}_n(\hat{h}_n) \geq \sup_h \mathbb{G}_n(h) - o_P(1)$, *then* $\hat{h}_n \xrightarrow{d} \hat{h}$ *in* $\mathcal{F}$.

F Conducting Significance Tests: Further Elaboration

The provided discussion presents a discretization methodology structured around three core phases, strategically developed to evaluate the statistical significance of neural network models when approximating a distinct unknown function. Additional comprehensive information can be located in Section 3.3. The ensuing elaboration expounds upon these consecutive stages:

Step 1: Approximating ζ-cover of $\mathcal{F}$

- This step involves creating an approximate set of neural networks that can effectively approximate a given function space $\mathcal{F}$.
- A number of neural networks are sampled, either with the same architecture as the trained model or other types that serve as "correct approximators."
- The weight parameters for these neural networks are sampled from a specific distribution called *Glorot normal distribution*.
- The collection of these neural networks forms an approximate ζ-cover of the function space $\mathcal{F}$.

Step 2: Simulate a sample from a multivariate normal and find $\hat{h}^{(max)}$

- A sample is drawn from a multivariate normal distribution with specific properties.
- Using this sample, empirical covariance or variance values are computed for pairs of neural networks from the set created in Step 1.
- The neural network with the highest value from this multivariate normal sample is identified as an approximate arg max of a certain function $\mathbb{G}$.
- This network is denoted as $\hat{h}^{(max)}$.

Step 3: Generate one approximate sample of $\mathcal{T}_j[h^{(max)}]$

- Using the neural network $\hat{h}^{(max)}$ identified in Step 2, an approximate sample of a certain value $\mathcal{T}_j[h^{(max)}]$ is generated.
- This sample is obtained by applying a specific calculation involving partial derivatives and a coefficient ρ_n^2.

Final Testing Process:

- The whole process of the three steps is repeated a certain number of times denoted as n_p.
- This repetition generates multiple samples of $\mathcal{T}_j[h^{(max)}]$.

- After obtaining these samples, a significance testing is conducted at a specified confidence level (α).
- The values of a test statistic involving the neural network's derivatives are compared to the quantile of the samples of $\mathcal{T}_j[h^{(max)}]$. It is important to note that the test statistic is computed utilizing the trained model and its derivative on the training data-set.
- If the test statistic value exceeds the quantile value, a null hypothesis $\mathcal{H}_0$ is rejected in favor of an alternative hypothesis $\mathcal{H}_1$ at the given confidence level.
- If the test statistic value doesn't exceed the quantile value, the null hypothesis is not rejected.

In essence, this methodology entails the construction of neural network approximations, the simulation of samples, and a comparative analysis of derivatives. This collective process serves to quantify the neural network's efficacy in approximating a distinct unknown function. Repetition of this procedure numerous times underpins the establishment of statistical significance.

G Details of the Training Procedure for the Empirical Study

The estimation/training of the neural networks means finding the weights and biases, $\gamma_{u,j,k}$ and $\gamma_{u,j,0}$, respectively, by minimizing a loss function, which in our case is the least square estimator (6). Due to high degree of non-linearity, non-convexity, and rich parametrization in neural networks, minimizing objective function (6) is a challenging task which requires a high degree of regularizations. A common approach to tackle this problem is to use stochastic gradient descent (SGD) to train a neural network. The SGD does not use the entire sample to calculate the gradient at each iteration, instead, it evaluates the gradient from a small subset of data.[18]

Specifying the neural network architecture is the first step of training. Guided by GKX, we consider five models where the shallowest network, denoted by NN1, has a single hidden layer of 32 nodes, respectively; NN2 has two hidden layers with 32 and 16 neurons for each successive layer, respectively; NN3 has three hidden layers with 32, 16, and 8 neurons, respectively; NN4 has four hidden layers with 32, 16, 8, and 4 neurons, respectively; and NN5 has five hidden layers with 32, 16, 8, 4, and 2 neurons, respectively.

Like GKX, we use the same optimizer and learning rate in our model. However, we use dropout regularization instead of $l1$ regularization to prevent overfitting. In each hidden layer, we apply dropout with a retention probability of 0.95 after the tanh transformation. Dropout is a regularization technique in neural network training that involves randomly setting a fraction of input units to zero during training. This has the effect of reducing overfitting and improving the generalization ability of the model (Srivastava et al., 2014).

We also follow the ensemble approach adopted by GKX to produce stable prediction results and reduce prediction variance due to the stochastic nature of the optimization. We first randomly initialize 9 models with the same NN architecture. We then train these models with the training sample separately in parallel. The hyperparameters of these models are tuned with the validation sample. We fit each model for 1024 epochs and save the fitted model only if the validation loss decreases in this epoch. We finally construct predictions by averaging the forecasts from these 9 model fits (Dietterich, 2000).

We do not consider using a rolling or extended window. Instead, similar to Chen et al. (2023), we use a single trained model to make predictions for the entire out-of-sample testing period. For example, to predict excess returns in January 2000, we take a trained model - which was obtained by using the training data from January 1975 to December 1986, and the validation data from January 1987 to December 1991 - and use the information up to December 1999.

[18] See GKX and Bianchi et al. (2021) for a detailed discussion about the training of a neural network and necessary regularization techniques of training.

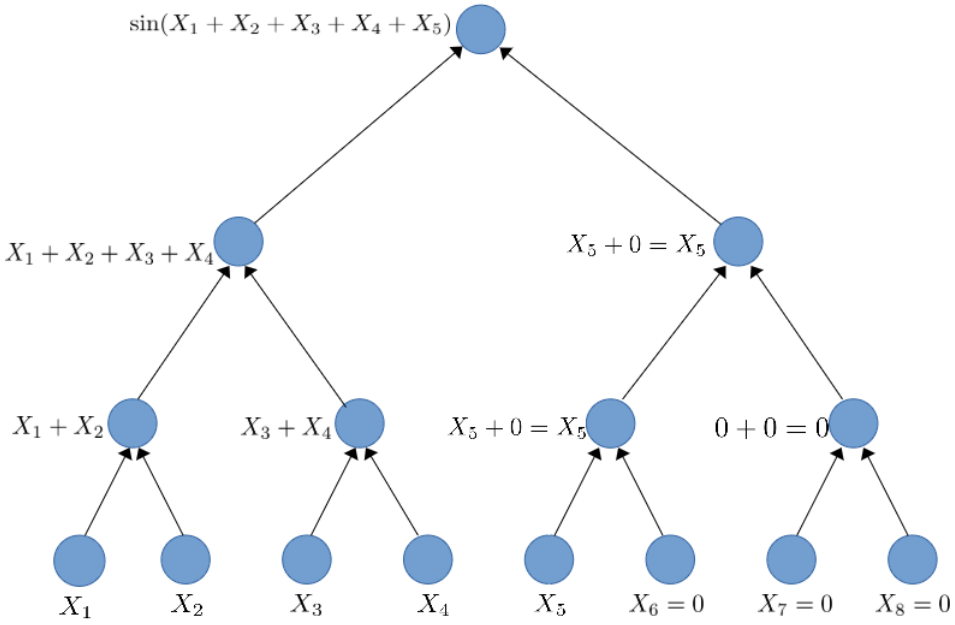
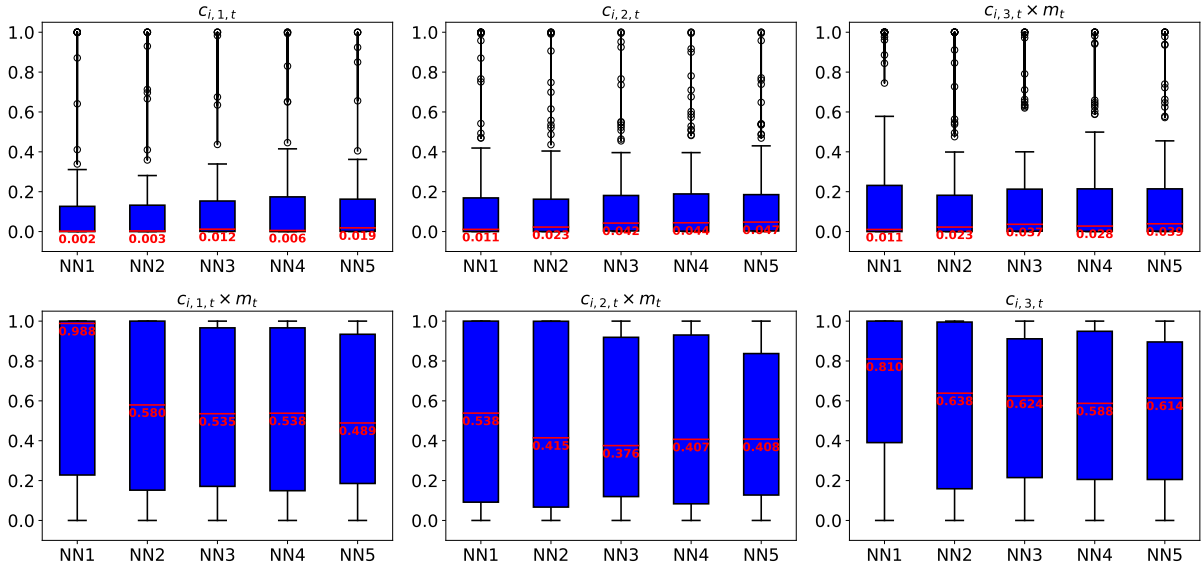
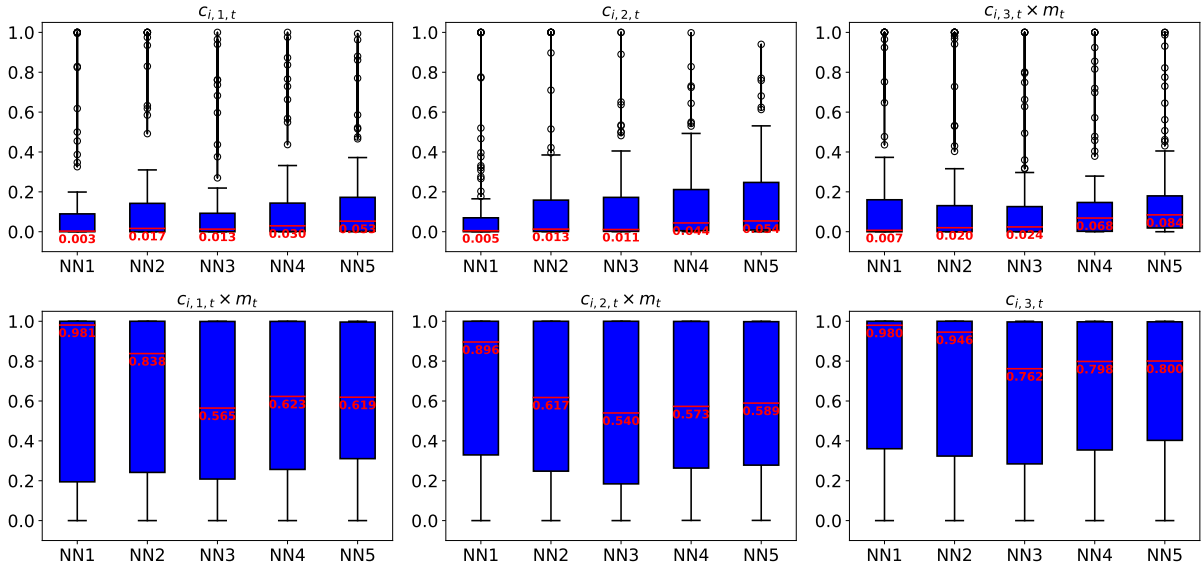


Figure A1: The binary tree visualization of $\sin(X_1 + X_2 + X_3 + X_4 + X_5)$

This binary tree visual representation illustrates the function $\sin(X_1 + X_2 + X_3 + X_4 + X_5)$. The lowest layer contains 8 nodes, representing the inputs X_1, X_2, X_3, X_4, X_5, and the other three nodes representing 0 ($X_6 = X_7 = X_8 = 0$). The second layer, located above the lowest layer, has 4 nodes that represent the sum functions $f_1(X_1, X_2) = X_1 + X_2$, $f_2(X_3, X_4) = X_3 + X_4$, $f_3(X_5, X_6) = X_5 + 0 = X_5$, and $f_4(X_7, X_8) = 0 + 0 = 0$. These second-layer nodes take input from the nodes in the bottom layer. The third layer, above the second layer, has 2 nodes that represent the sum functions taking input from the second layer nodes, $f_5(f_1(X_1, X_2), f_2(X_3, X_4)) = f_1(X_1, X_2) + f_2(X_3, X_4) = X_1 + X_2 + X_3 + X_4$ and $f_6(f_3(X_5, X_6), f_4(X_7, X_8)) = f_3(X_5, X_6) + f_4(X_7, X_8) = X_5$. The topmost layer, the output layer, has a single node that represents the composition function $f_7(f_5(f_1(X_1, X_2), f_2(X_3, X_4)), f_6(f_3(X_5, X_6), f_4(X_7, X_8))) = \sin(f_5(f_1(X_1, X_2), f_2(X_3, X_4)) + f_6(f_3(X_5, X_6), f_4(X_7, X_8))) = \sin(X_1 + X_2 + X_3 + X_4 + X_5)$.

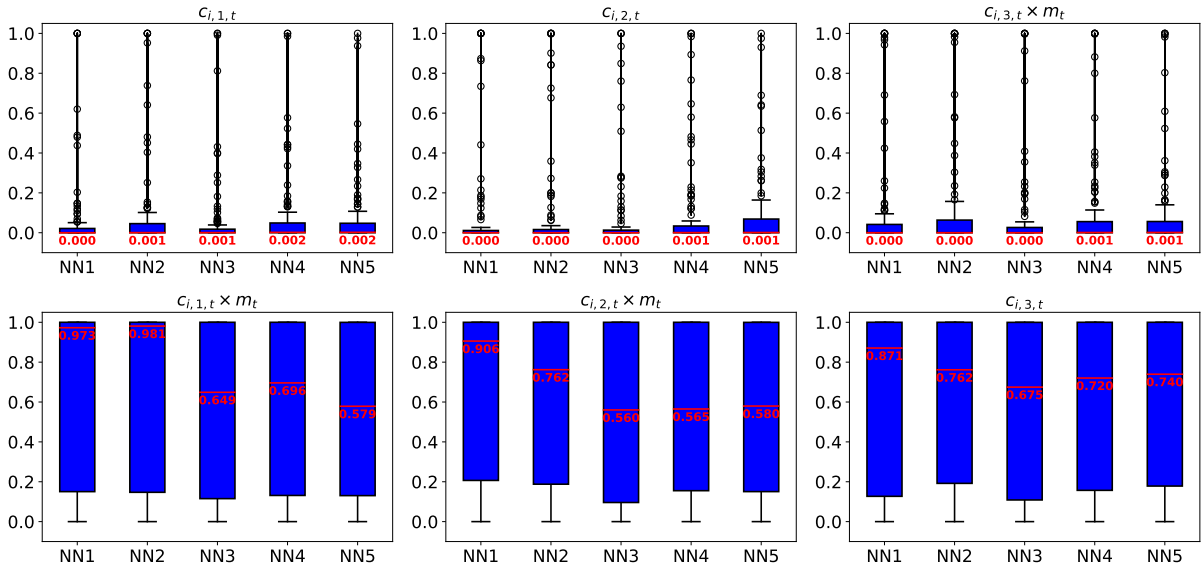


(a) $(N = 200, T = 120)$

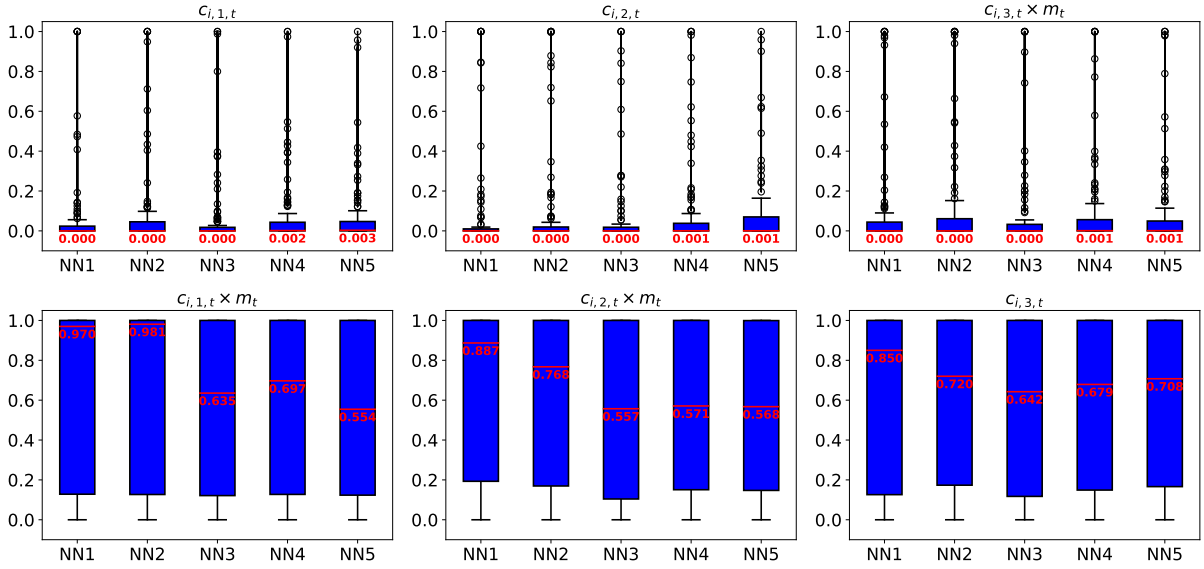


(b) $(N = 100, T = 180)$

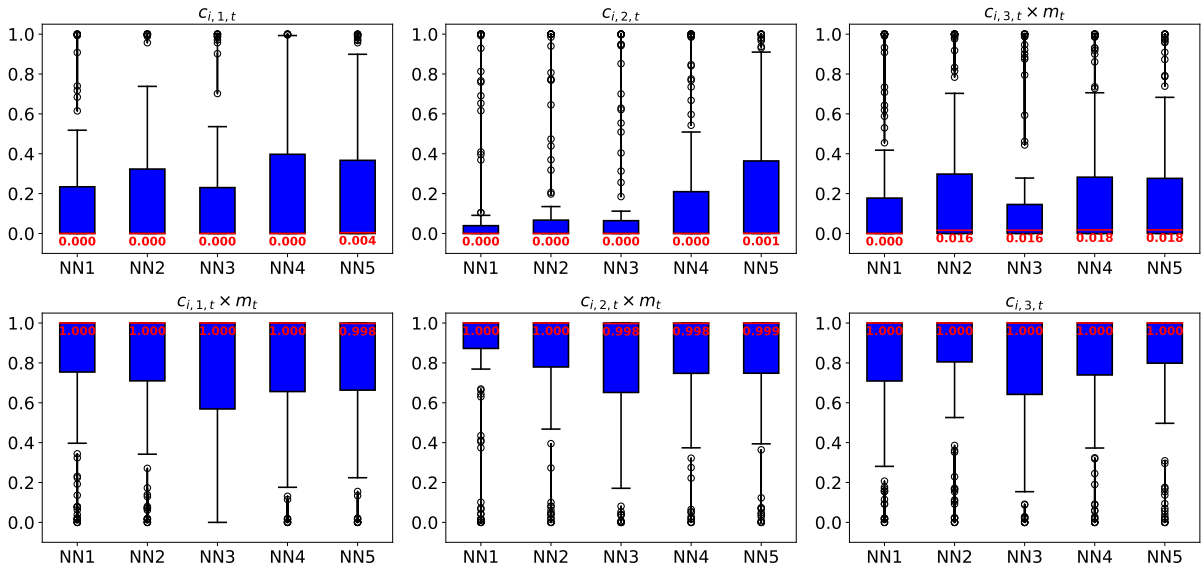
Figure A2: Distribution of p-values for covariates across simulation choices. This figure presents the distribution of 100 p-values for 6 covariates for five neural networks when $P_c = 50$ and the discretization choice is $(m = 500, one\ layer, n_p = 1{,}000)$. Panel (a) and (b) show the distribution for the simulation choice $(N = 200, T = 120)$ and $(N = 100, T = 180)$, respectively. $c_{i,1,t}$, $c_{i,2,t}$, and $c_{i,3,t} \times m_t$ are the true covariates. $c_{i,1,t} \times m_t$, $c_{i,2,t} \times m_t$, and $c_{i,3,t}$ are covariates that are correlated to one of the true covariates. For each Monte Carlo repetition, we compute the p-value for each covariate based on its test statistic scaled by $((N \times T/3)^{-0.245})^{-2}$, and the 1,000 $\mathcal{T}_j[h^{(max)}]$ samples. Thus, a total of 100 Monte Carlo repetitions lead to a total of 100 p-values for each covariate. The red line marks the median level of the 100 p-values. The number underneath the red line is the median p-value, rounded to three-digits. From NN1 to NN5, the number of hidden layers increases from 1 to 5 and the number of neurons in each hidden layer decreases from 32 to 2 according to the pyramid rule.



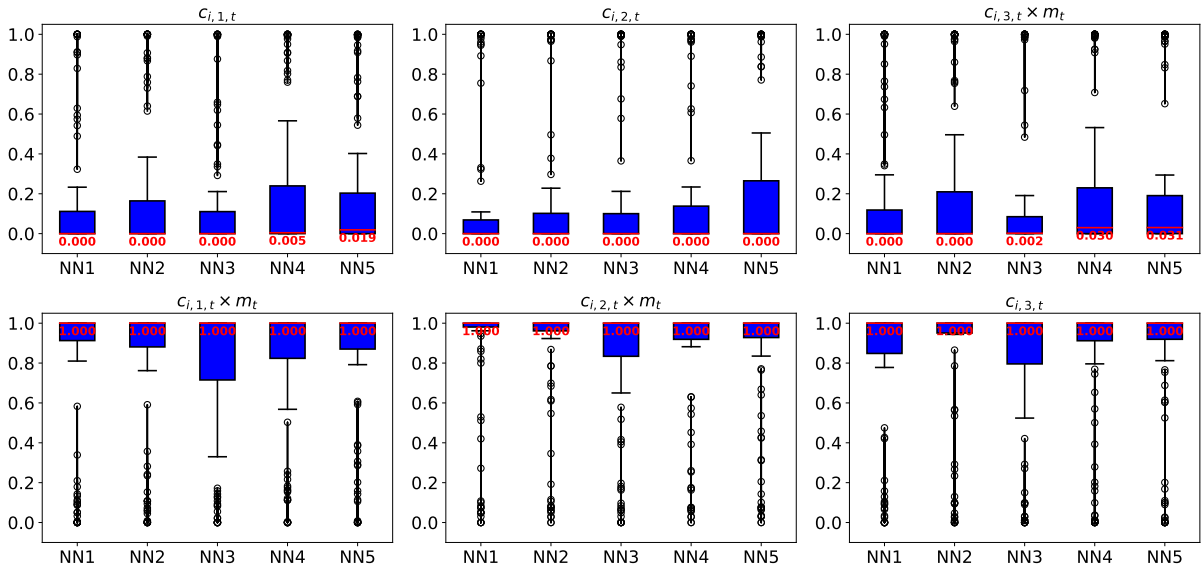
(a) $(m = 500, one\ layer, n_p = 2,000)$



(b) $(m = 300, one\ layer, n_p = 1,000)$



(c) $(m = 500, \textit{two layers}, n_p = 1,000)$



(d) $(m = 500, \textit{three layers}, n_p = 1,000)$

Figure A3: Distribution of p-values for covariates across discretization choices. This figure presents the distribution of 100 p-values for 6 covariates for five neural networks when $P_c = 100$ and the simulation choice is $(N = 200, T = 180)$. Panel (a), (b), (c), and (d) show the distribution for the discretization choice $(m = 500, \textit{one layer}, n_p = 2,000)$, $(m = 300, \textit{one layer}, n_p = 1,000)$, $(m = 500, \textit{two layers}, n_p = 1,000)$, and $(m = 500, \textit{three layers}, n_p = 1,000)$, respectively. $c_{i,1,t}$, $c_{i,2,t}$, and $c_{i,3,t} \times m_t$ are the true covariates. $c_{i,1,t} \times m_t$, $c_{i,2,t} \times m_t$, and $c_{i,3,t}$ are covariates that are correlated to one of the true covariates. For each Monte Carlo repetition, we compute the p-value for each covariate based on its test statistic and the 1,000 $\mathcal{T}_j[h^{(max)}]$ samples. Thus, a total of 100 Monte Carlo repetitions lead to a total of 100 p-values for each covariate. The red line marks the median level of the 100 p-values. The number underneath the red line is the median p-value, rounded to three-digits. From NN1 to NN5, the number of hidden layers increases from 1 to 5 and the number of neurons in each hidden layer decreases from 32 to 2 according to the pyramid rule.

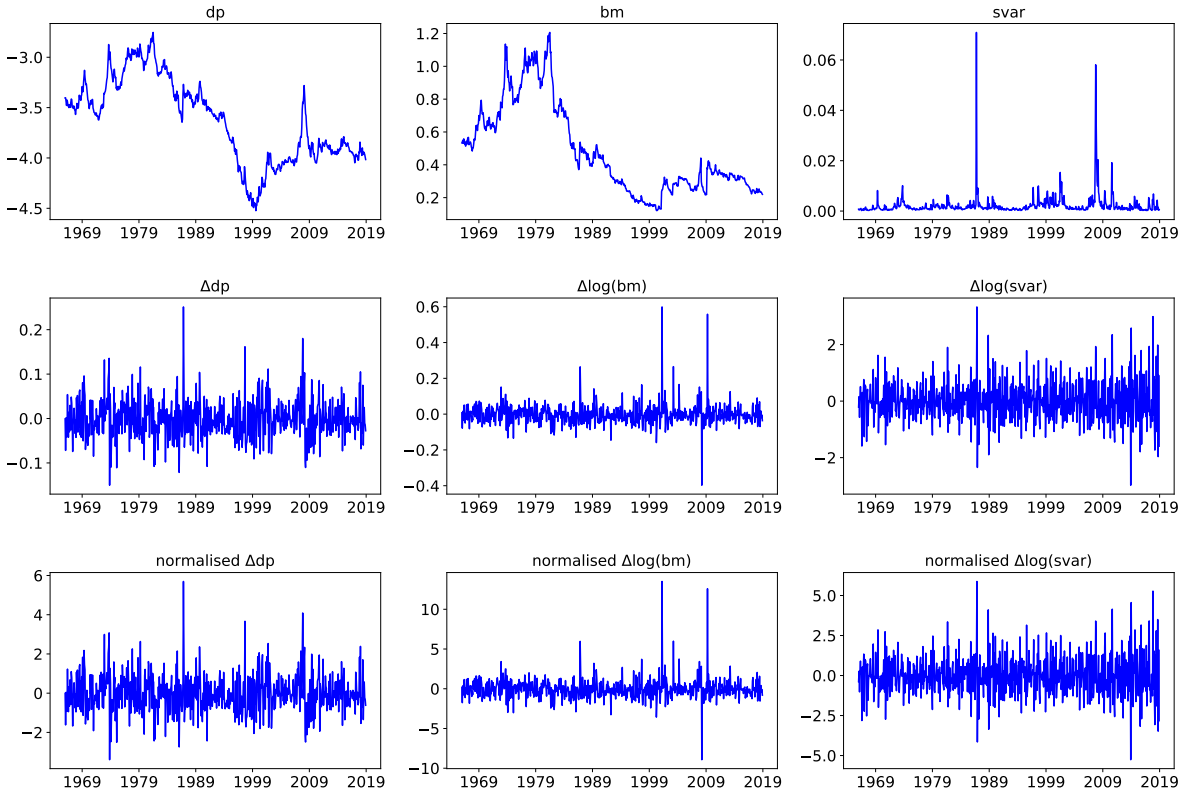


Figure A4: The times series of three predictors

The macro variables are first transformed and then normalised using the mean and standard deviation of training sample (1967-1986). See Table A6 for the definitions of these predictors.

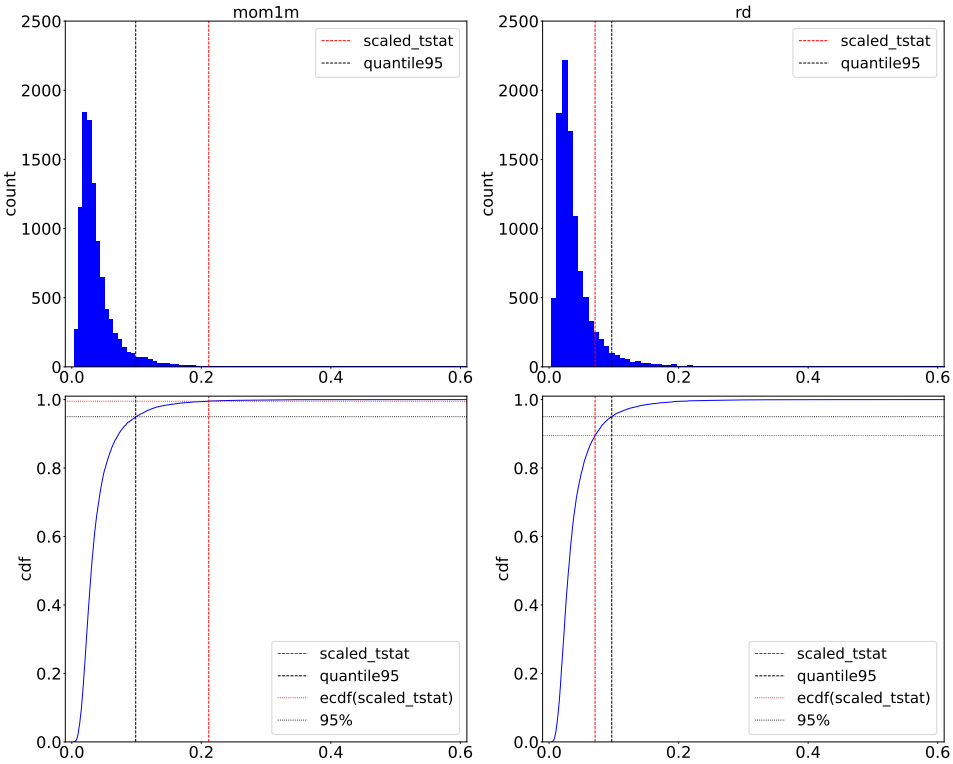


Figure A5: The approximation of asymptotic distributions for CpM predictors.

The neural network has three-hidden layers with 32, 16, and 8 neurons. Top panel shows approximated probability density functions while bottom exhibits cumulative distribution functions. mom1m: 1-month momentum. rd: R&D increase. cdf: cumulative distribution function.

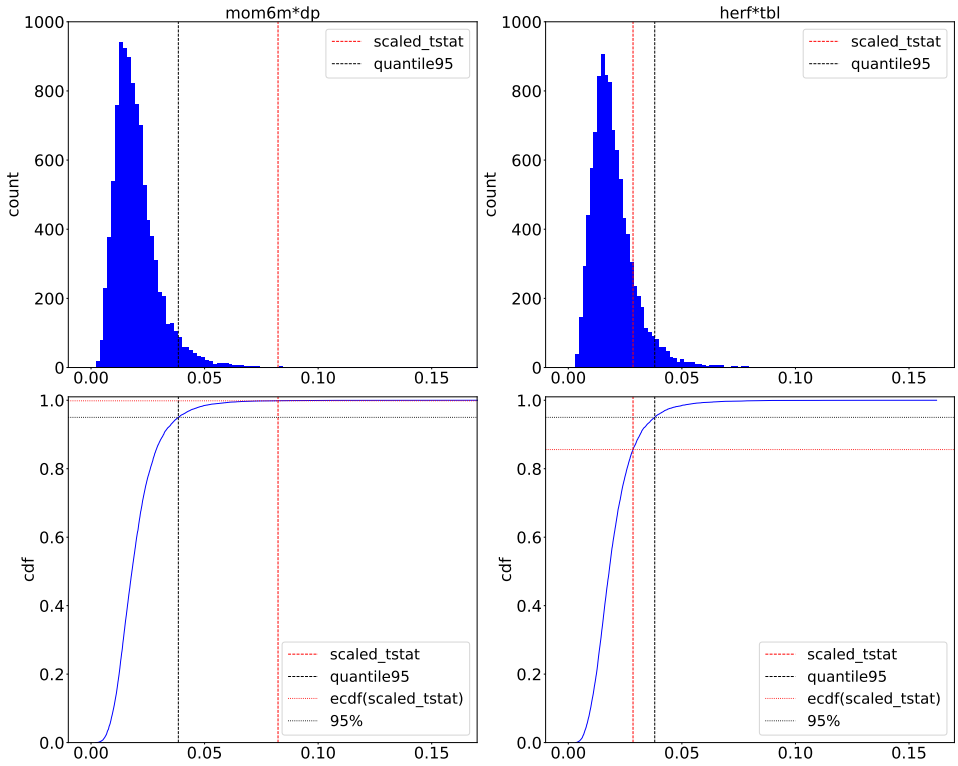


Figure A6: The approximation of asymptotic distributions for CtM predictors.

The neural network has three-hidden layers with 32, 16, and 8 neurons. Top panel shows approximated probability density functions while bottom exhibits cumulative distribution functions. mom6m: 6-month momentum. dp: dividend-price ratio. tbl: treasury-bill rate. herf: industry sales concentration. cdf: cumulative distribution function.

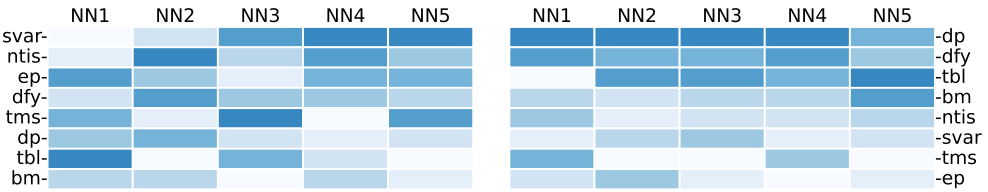


Figure A7: The importance of macroeconomic predictors where the CpM is the universe of predictors.

Rankings of 8 macroeconomic predictors in terms of overall model contribution. Predictors are ordered based on the sum of their ranks over all models, with the most influential predictors on the top and the least influential on the bottom. Columns correspond to the individual models, and the color gradients within each column indicate the most influential (dark blue) to the least influential (white) variables. See Table A5 for the definitions of stock characteristics. tstat: test statistic. LOOM: leave-one-out-metric.

Table A1: The notation and abbreviation table.

Notation	Description
capital letters	Random variables
small letters	Realization of random variables/deterministic variables or functions
bold capital letters	Random vectors
bold small letters	Realization of random vectors/any deterministic vectors
i.i.d.	Independent and identically distributed
$\vec{\alpha}$ (classic vector notation)	multi-index notation
N_f	The complexity of f
$\mathcal{C}^0(\mathbb{X})$	The set of continuous functions on $\mathbb{X}$
$\mathcal{C}^m(\mathbb{X})$	The set of m-times differentiable functions on $\mathbb{X}$
$a_n \lesssim b_n$	$\exists R > 0\ \exists N_1\ \in \mathbb{N}\ \forall n \geq N_1,\ a_n \leq R \cdot b_n \wedge a_n \sim b_n \Rightarrow \lim_{n\to\infty} \frac{a_n}{b_n} = 1$
$N(\epsilon, \mathcal{D}, \rho)$	The covering number for a pseudo-metric space $(\mathcal{D}, \rho)$
$\mathbb{H}^{p,v}(\mathbb{X})$	Sobolev space on $\mathbb{X}$ with weak differentiability order p and norm v
$L^p(P)$	The L^p space defined on probability space $(\mathbb{X}, P)$
$\mathcal{F}_{DNN}$	A smooth neural network space with fixed-depth L_d
$\mathcal{F}_{DNN,H_n}$	The subset of $\mathcal{F}_{DNN}$ whose members have fixed-width H_n
ρ_n	A semi-norm, defined as $\rho_n(g(\boldsymbol{x})) \coloneqq \sqrt{\frac{1}{n}\sum_{i=1}^{n}(g(\boldsymbol{x_i}))^2}$
$\equiv$	Equivalence of functions
CpM	Characteristics plus Macroeconomic
CtM	Characteristics times Macroeconomic
NN	Neural Network
MLP	Multi-layer perceptrons
HG	Horel and Giesecke (2020)
GKX	Gu et al. (2020)

Table A2: **Summary statistics of scaled test statistics for covariates when $P_c = 50$.** This table reports the statistics of scaled test statistics for 6 covariates for 100 Monte Carlo repetitions when $P_c = 50$ and the simulation choice is $(N = 200, T = 180)$. $c_{i,1,t}$, $c_{i,2,t}$, and $c_{i,3,t} \times m_t$ are the true covariates. $c_{i,1,t} \times m_t$, $c_{i,2,t} \times m_t$, and $c_{i,3,t}$ are covariates that are correlated to one of the true covariates. The remaining 94 covariates are noise. Panel A, B, C, D, and E report the statistics for NN1, NN2, NN3, NN4, and NN5, respectively. From NN1 to NN5, the number of hidden layers increases from 1 to 5 and the number of neurons in each hidden layer decreases from 32 to 2 according to the pyramid rule. The test statistics are scaled by $(n^{-0.245})^{-2}$, where $n = N \times T/3$.

Panel A: NN1								
Covariate	count	mean	std	min	25%	50%	75%	max
$c_{i,1,t}$	100	2.38E-02	2.14E-02	4.40E-10	6.47E-03	1.86E-02	3.37E-02	1.03E-01
$c_{i,2,t}$	100	2.09E-02	1.74E-02	1.28E-11	9.36E-03	1.76E-02	2.76E-02	8.96E-02
$c_{i,3,t} \times m_t$	100	1.93E-02	1.72E-02	2.56E-11	6.56E-03	1.65E-02	2.76E-02	9.26E-02
$c_{i,1,t} \times m_t$	100	3.48E-03	7.17E-03	1.28E-10	1.79E-05	1.98E-04	3.65E-03	4.58E-02
$c_{i,2,t} \times m_t$	100	2.51E-03	5.01E-03	1.26E-10	1.15E-05	2.56E-04	2.51E-03	2.88E-02
$c_{i,3,t}$	100	3.11E-03	6.55E-03	1.72E-11	1.51E-05	3.89E-04	3.50E-03	4.73E-02
Noise	9,400	9.50E-06	1.91E-05	5.01E-12	1.82E-06	3.79E-06	9.50E-06	4.04E-04
Panel B: NN2								
Covariate	count	mean	std	min	25%	50%	75%	max
$c_{i,1,t}$	100	2.14E-02	1.94E-02	6.49E-12	6.92E-03	1.64E-02	3.08E-02	1.03E-01
$c_{i,2,t}$	100	1.90E-02	1.56E-02	5.82E-12	8.34E-03	1.69E-02	2.61E-02	7.89E-02
$c_{i,3,t} \times m_t$	100	1.83E-02	1.62E-02	6.00E-12	6.38E-03	1.54E-02	2.47E-02	9.33E-02
$c_{i,1,t} \times m_t$	100	3.61E-03	7.58E-03	4.65E-12	4.09E-05	4.86E-04	3.46E-03	4.49E-02
$c_{i,2,t} \times m_t$	100	2.81E-03	5.22E-03	5.26E-12	1.85E-05	5.21E-04	2.90E-03	3.26E-02
$c_{i,3,t}$	100	3.39E-03	7.07E-03	5.21E-12	3.53E-05	5.93E-04	3.60E-03	3.94E-02
Noise	9,400	1.89E-05	3.65E-05	2.62E-12	2.41E-06	6.65E-06	1.94E-05	5.41E-04
Panel C: NN3								
Covariate	count	mean	std	min	25%	50%	75%	max
$c_{i,1,t}$	100	2.06E-02	1.82E-02	3.50E-05	7.05E-03	1.54E-02	2.95E-02	1.08E-01
$c_{i,2,t}$	100	1.84E-02	1.46E-02	4.16E-05	8.24E-03	1.69E-02	2.53E-02	7.27E-02
$c_{i,3,t} \times m_t$	100	1.77E-02	1.55E-02	3.53E-06	7.39E-03	1.45E-02	2.26E-02	9.07E-02
$c_{i,1,t} \times m_t$	100	3.94E-03	8.49E-03	1.35E-06	9.56E-05	9.74E-04	3.68E-03	4.88E-02
$c_{i,2,t} \times m_t$	100	3.11E-03	5.34E-03	5.08E-07	8.17E-05	1.02E-03	3.70E-03	3.34E-02
$c_{i,3,t}$	100	3.66E-03	7.85E-03	9.62E-07	8.51E-05	9.10E-04	3.81E-03	5.33E-02
Noise	9,400	3.33E-05	6.78E-05	1.58E-07	3.36E-06	1.08E-05	3.30E-05	1.73E-03

Table A2 Continue.

Panel D: NN4								
Covariate	count	mean	std	min	25%	50%	75%	max
$c_{i,1,t}$	100	2.65E-02	2.57E-02	1.59E-05	9.03E-03	1.81E-02	4.19E-02	1.58E-01
$c_{i,2,t}$	100	2.32E-02	1.97E-02	1.62E-05	9.24E-03	1.99E-02	2.87E-02	1.01E-01
$c_{i,3,t} \times m_t$	100	2.23E-02	2.00E-02	1.04E-05	8.29E-03	1.69E-02	3.10E-02	1.05E-01
$c_{i,1,t} \times m_t$	100	4.66E-03	9.40E-03	6.67E-07	1.03E-04	1.09E-03	5.18E-03	6.76E-02
$c_{i,2,t} \times m_t$	100	3.73E-03	6.47E-03	6.16E-07	7.72E-05	1.21E-03	4.83E-03	4.45E-02
$c_{i,3,t}$	100	4.59E-03	1.00E-02	7.85E-07	1.26E-04	9.03E-04	4.54E-03	6.69E-02
Noise	9,400	4.06E-05	8.50E-05	5.39E-09	4.11E-06	1.29E-05	3.92E-05	2.87E-03
Panel E: NN5								
Covariate	count	mean	std	min	25%	50%	75%	max
$c_{i,1,t}$	100	2.76E-02	2.59E-02	2.02E-04	9.41E-03	1.99E-02	4.01E-02	1.68E-01
$c_{i,2,t}$	100	2.45E-02	1.99E-02	1.69E-04	1.07E-02	2.21E-02	3.43E-02	1.00E-01
$c_{i,3,t} \times m_t$	100	2.39E-02	2.21E-02	3.31E-05	9.55E-03	1.82E-02	3.18E-02	1.24E-01
$c_{i,1,t} \times m_t$	100	5.08E-03	1.02E-02	8.74E-07	1.87E-04	1.55E-03	5.05E-03	6.90E-02
$c_{i,2,t} \times m_t$	100	4.14E-03	6.96E-03	1.49E-06	1.90E-04	1.79E-03	4.88E-03	4.53E-02
$c_{i,3,t}$	100	5.10E-03	1.13E-02	1.63E-06	2.00E-04	1.25E-03	5.10E-03	7.85E-02
Noise	9,400	5.95E-05	1.27E-04	9.53E-08	5.15E-06	1.76E-05	5.63E-05	3.88E-03

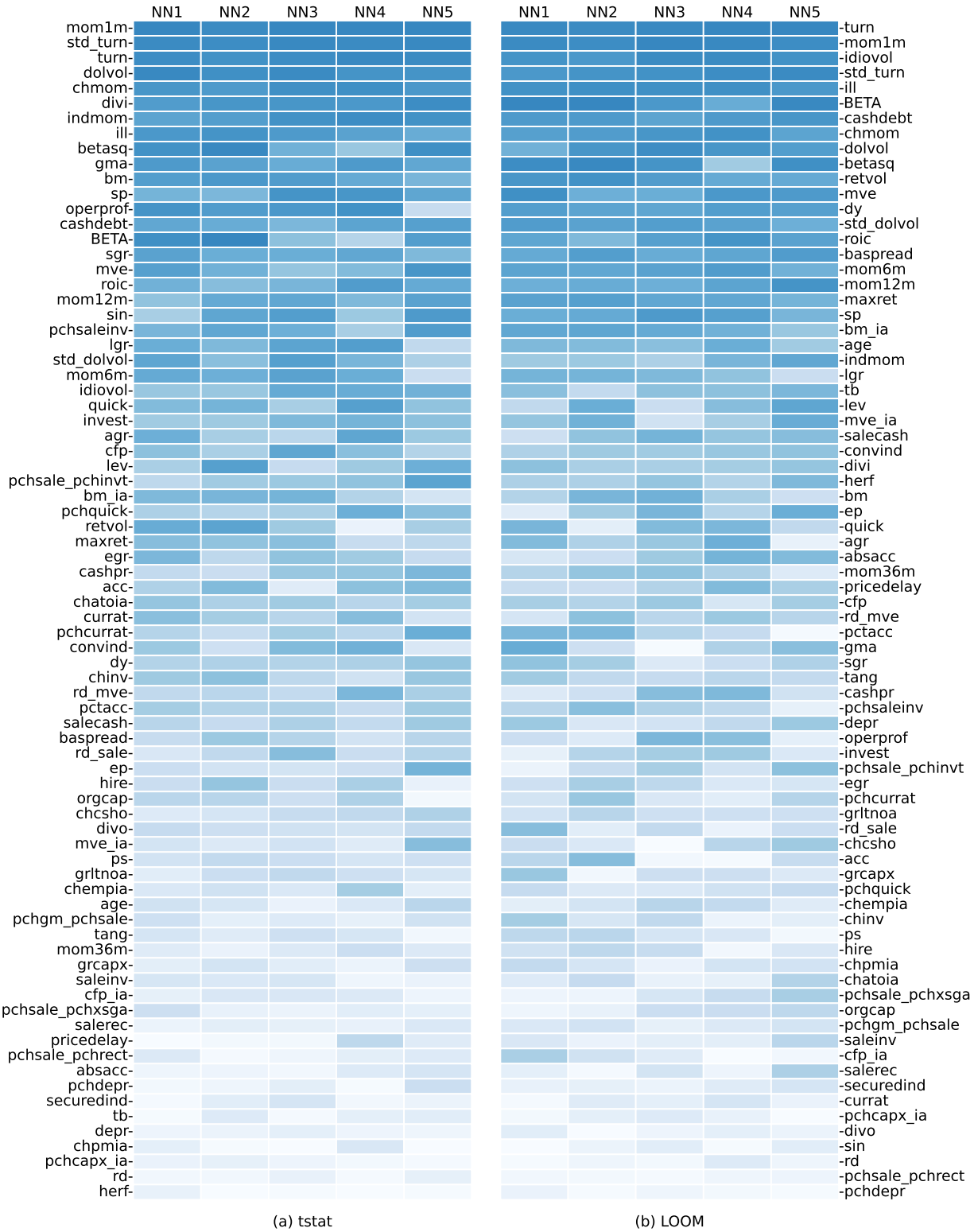


Figure A8: The importance of characteristic predictors where the CpM is the universe of predictors.

Rankings of 78 characteristic predictors in terms of overall model contribution. Predictors are ordered based on the sum of their ranks over all models, with the most influential predictors on the top and the least influential on the bottom. Columns correspond to the individual models, and the color gradients within each column indicate the most influential (dark blue) to the least influential (white) variables. See Table A5 for the definitions of stock characteristics. tstat: test statistics. LOOM: leave-one-out-metric.

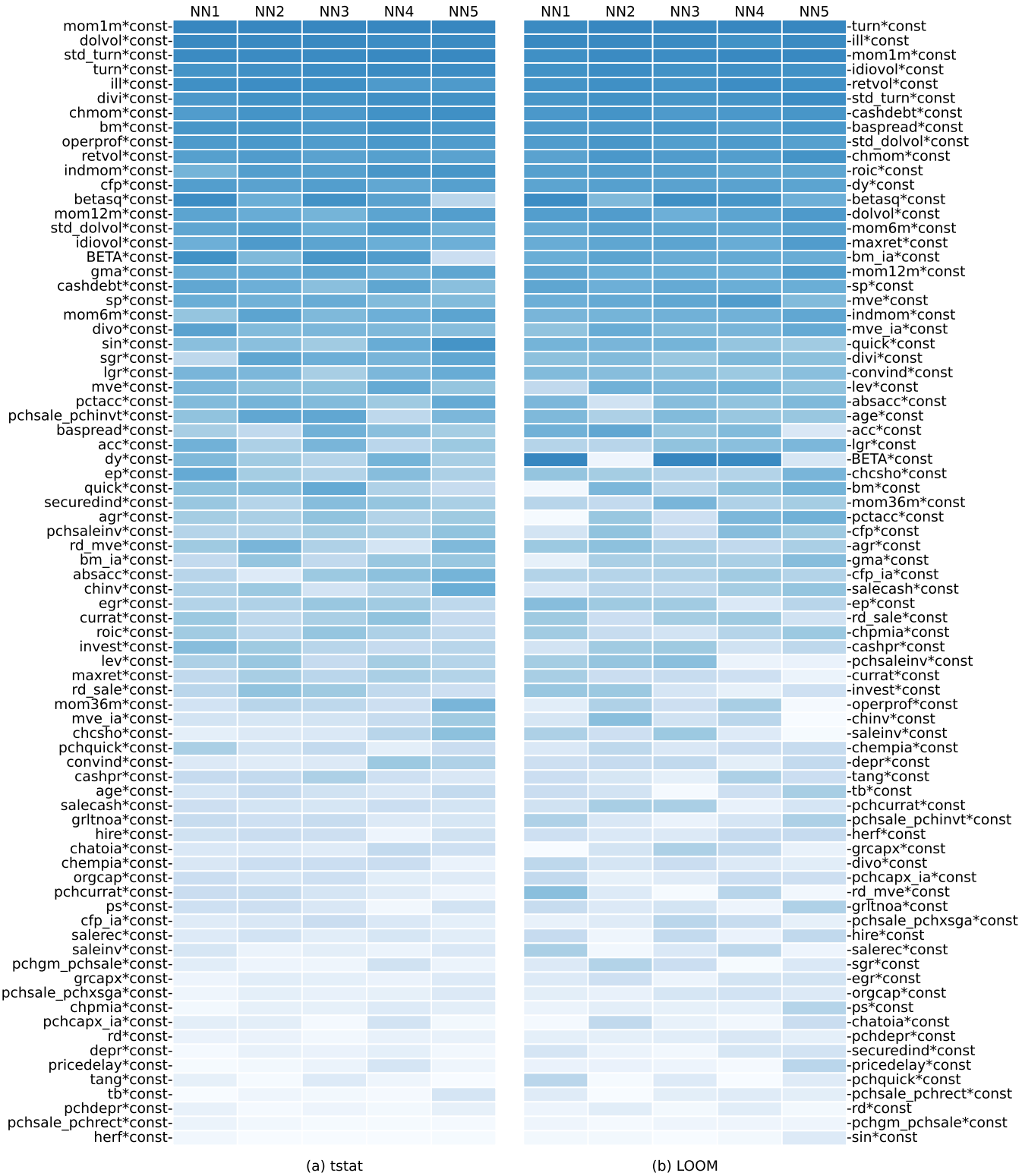


Figure A9: The importance of characteristic predictors where the CtM is the universe of predictors.

Rankings of 78 characteristic predictors in terms of overall model contribution. Predictors are ordered based on the sum of their ranks over all models, with the most influential predictors on the top and the least influential on the bottom. Columns correspond to the individual models, and the color gradients within each column indicate the most influential (dark blue) to the least influential (white) variables. See Table A5 for the definitions of stock characteristics. tstat: test statistics. LOOM: leave-one-out-metric.

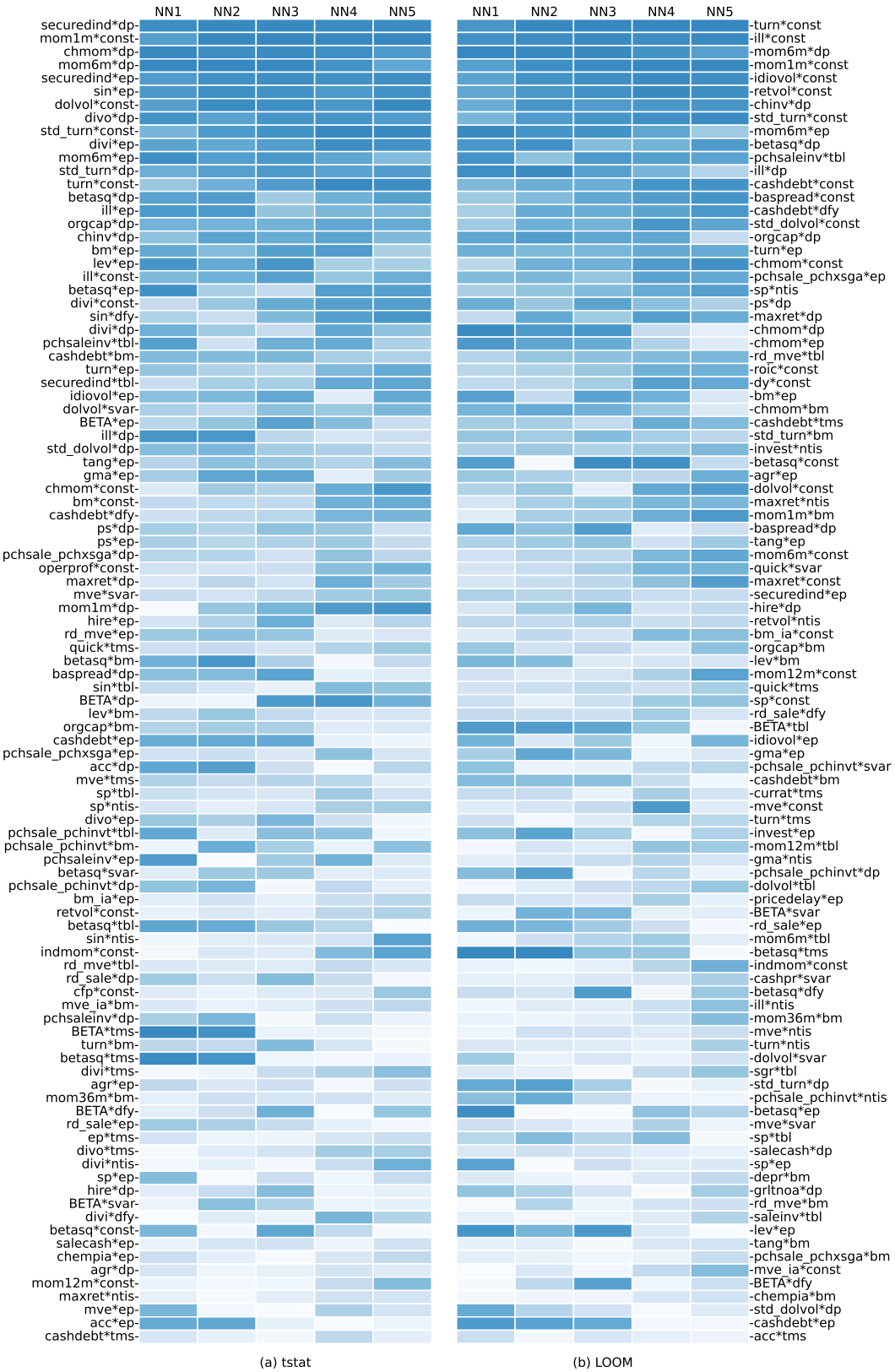
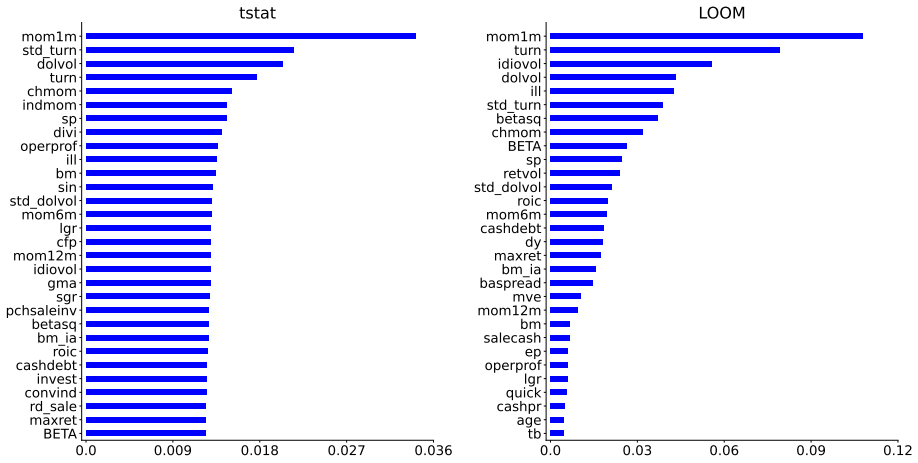
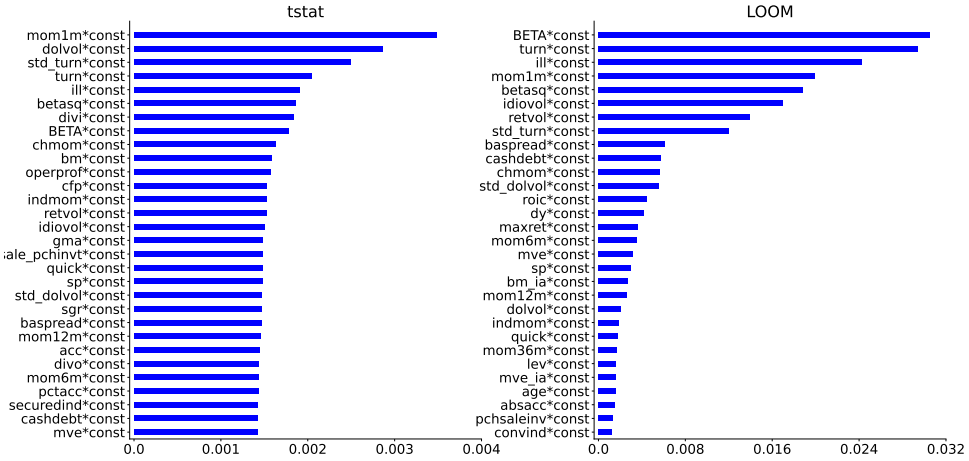


Figure A10: The importance of top-100 predictors where the CtM is the universe of predictors.

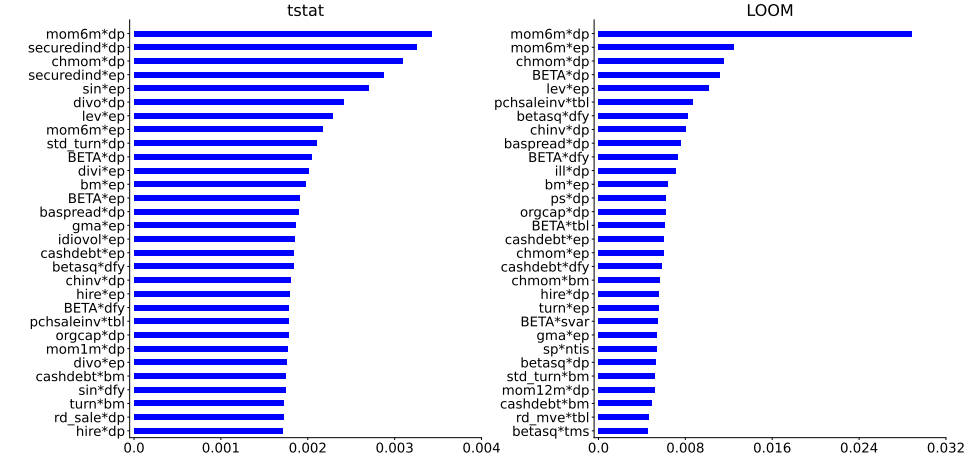
Rankings of top-100 predictors in terms of overall model contribution. Predictors are ordered based on the sum of their ranks over all models, with the most influential predictors on the top and the least influential on the bottom. Columns correspond to the individual models, and the color gradients within each column indicate the most influential (dark blue) to the least influential (white) variables. See Tables A5 and A6 for the definitions of stock characteristics and macroeconomic predictors. tstat: test statistics. LOOM: leave-one-out-metric.



(a) Top-30 significant characteristics in NN3 where the universe of predictors is CpM



(b) Top-30 significant characteristics in NN3 where the universe of predictors is CtM



(c) Top-30 significant interaction predictors in NN3 where the universe of predictors is CtM

Figure A11: Top-30 normalized tstat and LOOM predictors in NN3.

See Tables A5 and A6 for the definitions of stock characteristics and macroeconomic predictors. tstat: test statistics. LOOM: leave-one-out-metric.

Table A3: **Summary statistics of scaled test statistics for covariates when $P_c = 100$.** This table reports the statistics of scaled test statistics for 6 covariates for 100 Monte Carlo repetitions when $P_c = 100$ and the simulation choice is $(N = 200, T = 180)$. $c_{i,1,t}$, $c_{i,2,t}$, and $c_{i,3,t} \times m_t$ are the true covariates. $c_{i,1,t} \times m_t$, $c_{i,2,t} \times m_t$, and $c_{i,3,t}$ are covariates that are correlated to one of the true covariates. The remaining 194 covariates are noise. Panel A, B, C, D, and E report the statistics for NN1, NN2, NN3, NN4, and NN5, respectively. From NN1 to NN5, the number of hidden layers increases from 1 to 5 and the number of neurons in each hidden layer decreases from 32 to 2 according to the pyramid rule. The test statistics are scaled by $(n^{-0.215})^{-2}$, where $n = N \times T/3$.

Panel A: NN1								
Covariate	count	mean	std	min	25%	50%	75%	max
$c_{i,1,t}$	100	1.62E-02	1.44E-02	2.36E-11	4.86E-03	1.27E-02	2.29E-02	6.83E-02
$c_{i,2,t}$	100	1.42E-02	1.16E-02	1.01E-10	6.09E-03	1.20E-02	1.97E-02	5.88E-02
$c_{i,3,t} \times m_t$	100	1.33E-02	1.17E-02	9.55E-11	4.64E-03	1.13E-02	1.84E-02	6.51E-02
$c_{i,1,t} \times m_t$	100	2.36E-03	4.78E-03	5.06E-11	7.95E-06	1.63E-04	2.30E-03	2.72E-02
$c_{i,2,t} \times m_t$	100	1.75E-03	3.57E-03	2.74E-11	6.22E-06	2.44E-04	1.83E-03	2.14E-02
$c_{i,3,t}$	100	2.13E-03	4.40E-03	1.48E-10	1.54E-05	2.81E-04	2.51E-03	3.19E-02
Noise	19,400	6.74E-06	1.38E-05	1.80E-11	1.29E-06	2.63E-06	6.60E-06	4.00E-04
Panel B: NN2								
Covariate	count	mean	std	min	25%	50%	75%	max
$c_{i,1,t}$	100	1.47E-02	1.35E-02	1.29E-12	4.19E-03	1.15E-02	2.13E-02	6.23E-02
$c_{i,2,t}$	100	1.29E-02	1.08E-02	1.25E-12	5.49E-03	1.12E-02	1.76E-02	5.54E-02
$c_{i,3,t} \times m_t$	100	1.25E-02	1.14E-02	1.13E-12	3.93E-03	1.08E-02	1.75E-02	6.66E-02
$c_{i,1,t} \times m_t$	100	2.30E-03	4.71E-03	1.15E-12	1.41E-05	1.57E-04	2.37E-03	2.78E-02
$c_{i,2,t} \times m_t$	100	1.70E-03	3.38E-03	1.16E-12	7.65E-06	3.18E-04	2.13E-03	2.10E-02
$c_{i,3,t}$	100	2.13E-03	4.38E-03	1.29E-12	1.65E-05	3.88E-04	2.09E-03	2.87E-02
Noise	19,400	9.52E-06	1.92E-05	6.37E-13	1.49E-06	3.52E-06	9.60E-06	4.87E-04
Panel C: NN3								
Covariate	count	mean	std	min	25%	50%	75%	max
$c_{i,1,t}$	100	1.56E-02	1.49E-02	7.47E-06	5.09E-03	1.08E-02	2.25E-02	8.28E-02
$c_{i,2,t}$	100	1.37E-02	1.15E-02	7.67E-07	5.71E-03	1.21E-02	1.75E-02	5.94E-02
$c_{i,3,t} \times m_t$	100	1.32E-02	1.16E-02	3.13E-06	5.01E-03	9.97E-03	1.81E-02	6.24E-02
$c_{i,1,t} \times m_t$	100	2.62E-03	5.55E-03	1.22E-09	3.37E-05	4.53E-04	2.72E-03	3.79E-02
$c_{i,2,t} \times m_t$	100	2.11E-03	3.86E-03	3.79E-07	3.13E-05	6.55E-04	2.59E-03	2.48E-02
$c_{i,3,t}$	100	2.54E-03	5.33E-03	4.97E-08	5.48E-05	5.06E-04	2.64E-03	3.21E-02
Noise	19,400	1.65E-05	3.46E-05	7.46E-10	1.97E-06	5.49E-06	1.60E-05	9.23E-04

Table A3 Continue.

Panel D: NN4								
Covariate	count	mean	std	min	25%	50%	75%	max
$c_{i,1,t}$	100	1.37E-02	1.40E-02	1.07E-05	3.85E-03	9.14E-03	2.00E-02	8.06E-02
$c_{i,2,t}$	100	1.20E-02	1.09E-02	2.87E-06	4.63E-03	1.02E-02	1.57E-02	5.63E-02
$c_{i,3,t} \times m_t$	100	1.13E-02	9.94E-03	4.38E-06	3.80E-03	9.08E-03	1.66E-02	4.54E-02
$c_{i,1,t} \times m_t$	100	2.17E-03	4.37E-03	2.09E-08	4.29E-05	4.04E-04	2.48E-03	3.01E-02
$c_{i,2,t} \times m_t$	100	1.76E-03	3.28E-03	2.08E-07	2.48E-05	6.27E-04	2.24E-03	2.35E-02
$c_{i,3,t}$	100	2.24E-03	4.64E-03	3.17E-07	4.88E-05	4.58E-04	2.26E-03	2.52E-02
Noise	19,400	1.61E-05	3.45E-05	1.50E-09	1.76E-06	5.10E-06	1.52E-05	1.02E-03
Panel E: NN5								
Covariate	count	mean	std	min	25%	50%	75%	max
$c_{i,1,t}$	100	1.34E-02	1.43E-02	4.48E-05	3.99E-03	8.53E-03	1.95E-02	9.10E-02
$c_{i,2,t}$	100	1.20E-02	1.14E-02	3.57E-05	3.68E-03	9.61E-03	1.56E-02	5.68E-02
$c_{i,3,t} \times m_t$	100	1.11E-02	9.79E-03	9.62E-06	3.99E-03	8.77E-03	1.57E-02	4.83E-02
$c_{i,1,t} \times m_t$	100	2.29E-03	4.62E-03	2.01E-07	6.33E-05	5.88E-04	2.38E-03	3.04E-02
$c_{i,2,t} \times m_t$	100	1.82E-03	3.32E-03	1.74E-07	6.57E-05	6.00E-04	2.24E-03	2.48E-02
$c_{i,3,t}$	100	2.31E-03	4.88E-03	2.45E-07	6.18E-05	4.24E-04	2.14E-03	2.87E-02
Noise	19,400	2.26E-05	4.87E-05	4.47E-08	2.03E-06	6.57E-06	2.09E-05	1.49E-03

Table A4: **Variable significance frequencies after multiple comparison adjustment.** This table reports the variable selection frequencies after multiple comparison adjustment for 6 covariates for five neural networks (NN1...NN5) when $P_c = 100$ and the simulation choice is $(N = 200, T = 180)$. The variable significance frequency after multiple comparison adjustment for a covariate is calculated as the proportion of significance test results showing that the covariate is significant at the 0.025% level, out of 100 Monte Carlo repetitions. $c_{i,1,t}$, $c_{i,2,t}$, and $c_{i,3,t} \times m_t$ are true covariates. $c_{i,1,t} \times m_t$, $c_{i,2,t} \times m_t$, and $c_{i,3,t}$ are covariates that are correlated to one of the three true covariates. From NN1 to NN5, the number of hidden layers increases from 1 to 5 and the number of neurons in each hidden layer decreases from 32 to 2 according to the pyramid rule. We consider five discretization choices: $(m = 500, 1, n_p = 1,000)$, $(m = 300, 1, n_p = 1,000)$, $(m = 500, 1, n_p = 2,000)$, $(m = 500, 2, n_p = 1,000)$, and $(m = 500, 3, n_p = 1,000)$. Their results are reported in Panel A, B, C, D, and E, respectively.

Panel A: $m = 500$, *one layer*, $n_p = 1,000$						
Method	$c_{i,1,t}$	$c_{i,2,t}$	$c_{i,3,t} \times m_t$	$c_{i,1,t} \times m_t$	$c_{i,2,t} \times m_t$	$c_{i,3,t}$
NN1	64	61	59	6	2	4
NN2	56	56	56	4	2	4
NN3	55	57	57	6	2	5
NN4	48	51	51	5	2	5
NN5	46	49	49	6	2	5
Panel B: $m = 300$, *one layer*, $n_p = 1,000$						
Method	$c_{i,1,t}$	$c_{i,2,t}$	$c_{i,3,t} \times m_t$	$c_{i,1,t} \times m_t$	$c_{i,2,t} \times m_t$	$c_{i,3,t}$
NN1	57	61	60	4	2	3
NN2	55	55	55	4	2	5
NN3	53	57	56	5	3	5
NN4	47	51	48	3	2	5
NN5	41	50	49	5	1	3
Panel C: $m = 500$, *one layer*, $n_p = 2,000$						
Method	$c_{i,1,t}$	$c_{i,2,t}$	$c_{i,3,t} \times m_t$	$c_{i,1,t} \times m_t$	$c_{i,2,t} \times m_t$	$c_{i,3,t}$
NN1	56	60	55	4	2	4
NN2	52	55	56	4	2	5
NN3	53	56	55	5	2	6
NN4	45	49	48	4	2	5
NN5	39	46	46	6	2	5
Panel D: $m = 500$, *two layers*, $n_p = 1,000$						
Method	$c_{i,1,t}$	$c_{i,2,t}$	$c_{i,3,t} \times m_t$	$c_{i,1,t} \times m_t$	$c_{i,2,t} \times m_t$	$c_{i,3,t}$
NN1	60	61	51	4	5	3
NN2	56	57	48	4	3	5
NN3	56	59	46	5	4	5
NN4	51	53	42	4	3	5
NN5	46	49	39	4	3	4
Panel E: $m = 500$, *three layers*, $n_p = 1,000$						
Method	$c_{i,1,t}$	$c_{i,2,t}$	$c_{i,3,t} \times m_t$	$c_{i,1,t} \times m_t$	$c_{i,2,t} \times m_t$	$c_{i,3,t}$
NN1	56	62	55	8	4	8
NN2	56	60	53	7	2	6
NN3	54	63	50	8	5	7
NN4	49	56	44	7	2	6
NN5	42	54	41	7	2	7

Table A5: Description of stock characteristics

No.	Acronym	Stock Characteristic	Data Source	Frequency	Definition
1	absacc	Absolute accruals	Compustat	Annual	Absolute value of working capital accruals *(acc)*
2	acc	Working capital accruals	Compustat	Annual	(Income before extraordinary items *(ib)* − operating activities/net cash flows *(oancf)*) / average of total assets *(at)* in year t and year t-1. If *oancf* is missing, then set numerator to (change in current assets *(act)* − change in cash/cash equivalents *(che)*) − change in current liabilities *(lct)* + change in debt included in current liabilities *(dlc)* + change in income taxes payable *(txp)* − depreciation and amortization expense *(dp)*)
3	age	# years since first Compustat coverage	Compustat	Annual	The number of years since the first coverage in Compustat
4	agr	Asset growth	Compustat	Annual	Percentage change in total assets *at*
5	baspread	Bid-ask spread	CRSP	Monthly	Monthly average of daily (bid-ask spread/average of daily spread)
6	beta	Beta	CRSP	Monthly	Estimated market beta from weekly returns and equal weighted market returns for 3 years ending month t-1 with at least 52 weeks of returns
7	betasq	Beta squared	CRSP	Monthly	Market beta *(beta)* Squared
8	bm	Book-to-market	Compustat+CRSP	Annual	Book value of equity *(ceq)* in Compustat / fiscal-year-end market capitalization *(mve_f)* in CRSP
9	bm_ia	Industry-adjusted book to market	Compustat+CRSP	Annual	Book-to-market *(bm)* − industry mean of book-to-market *(bm)*
10	cashdebt	Cash flow to debt	Compustat	Annual	Income before extraordinary items and depreciation *(ib+dp)* / average of total liabilities *(lt)* for last two years.
11	cashpr	Cash productivity	Compustat	Annual	Fiscal-year-end market capitalization *(mve_f)* + long-term debt *(dltt)* − total assets *(at)*) / Cash and cash equivalents *(che)*
12	cfp	Cash flow to price ratio	Compustat	Annual	Operating activities/net cash flows *(oancf)* / fiscal-year-end market capitalization *(mve_f)*. If *oancf* is missing, then set numerator to (Income before extraordinary items *(ib)* − ((change in current assets *(act)* − change in cash and cash equivalents *(che)*) − (change in current liabilities *(lct)* − change in debt included in current liabilities *(dlc)* − change in income taxes payable *(txp)*) − depreciation and amortization expense *(dp)*))
13	cfp_ia	Industry-adjusted cash flow to price ratio	Compustat	Annual	Cash flow to price ratio *(cpr)* − industry mean of cash flow to price ratio
14	chatoia	Industry-adjusted change in asset turnover	Compustat	Annual	Change in asset turnover *(chato)* − industry mean of change in asset turnover

Table A5 (Continue)

No.	Acronym	Stock Characteristic	Data Source	Frequency	Definition
15	chcsho	Change in shares outstanding	Compustat	Annual	Percent change in common shares outstanding *(csho)*
16	chempia	Industry-adjusted change in employees	Compustat	Annual	Change in employees *(chemp)* − industry mean of change in employees
17	chinv	Change in inventory	Compustat	Annual	Change in total inventory *(invt)* / last-two-years average total assets *(at)*
18	chmom	Change in 6-month momentum	CRSP	Monthly	Cumulative returns from months t-6 to t-1 − cumulative returns from months t-12 to t-7
19	chpmia	Industry-adjusted change in profit margin	Compustat	Annual	Change in profit margin *(chpm)* − industry mean of change in profit margin
20	convind	Convertible debt indicator	Compustat	Annual	An indicator equal to 1 if the firm has convertible debt outstanding, and 0 otherwise.
21	currat	Current ratio	Compustat	Annual	Current Assets *(act)* / Current Liabilities *(lct)*
22	depr	Depreciation / PP&E	Compustat	Annual	Depreciation *(dp)* / PP&E *(ppent)*
23	divi	Dividend initiation	Compustat	Annual	An indicator equal to 1 if company pays dividends in year t but did not in year t-1
24	divo	Dividend omission	Compustat	Annual	An indicator equal to 1 if company does not pay dividend in year t but did in year t-1
25	dolvol	Dollar trading volume	CRSP	Monthly	Natural log of (trading volume *(vol)* ∗ price per share *(prc)*) in month t-2
26	dy	Dividend to price	Compustat	Annual	Total dividends *(dvtt)* / fiscal-year-end market capitalization *(mve_f)*
27	egr	Growth in common shareholder equity	Compustat	Annual	Percent change in book value of equity *(ceq)*
28	ep	Earnings to price	Compustat	Annual	Income before extraordinary items *(ib)* / fiscal-year-end market capitalization *(mve_f)*
29	gma	Gross profitability	Compustat	Annual	(Revenues *(revt)* - cost of goods sold *(cogs)*) / total assets *(at)* in year t-1
30	grcapx	Growth in capital expenditures	Compustat	Annual	Percent change in capital expenditures *(capx)*) from year t-2 to year t
31	grltnoa	Growth in long term net operating assets	Compustat	Annual	Growth in long-term net operating assets
32	herf	Industry sales concentration	Compustat	Annual	Sum of squared percent of sales *(sale)* in industry in a fiscal year for each company
33	hire	Employee growth rate	Compustat	Annual	Percent change in number of employees *(emp)*

Table A5 (Continue)

No.	Acronym	Stock Characteristic	Data Source	Frequency	Definition
34	idiovol	Idiosyncratic return volatility	CRSP	Monthly	Standard deviation of residuals of weekly returns *(idioret)* on weekly equal weighted market returns for 3 years prior to month end
35	ill	Illiquidity	CRSP	Monthly	Average of daily (absolute return/dollar volume) *(|ret|/(prc*vol))*
36	indmom	Industry momentum	CRSP	Monthly	Average of industry 12-month momentum *(mom12m)*
37	invest	Capital expenditures and inventory	Compustat	Annual	(Change in gross property, plant, and equipment *(ppegt)* + change in inventories *(invt)*) / total assets *(at)* in year t-1
38	lev	Leverage	Compustat	Annual	Total liabilities *(lt)*) / fiscal-year-end market capitalization *(mve_f)*
39	lgr	Growth in long-term debt	Compustat	Annual	Percent change in total liabilities *(lt)*
40	maxret	Maximum daily return	CRSP	Monthly	Maximum daily return from returns during calendar month t-1
41	mom12m	12-month momentum	CRSP	Monthly	11-month cumulative returns ending one month before month end
42	mom1m	1-month momentum	CRSP	Monthly	1-month cumulative return
43	mom36m	36-month momentum	CRSP	Monthly	Cumulative returns from months t-36 to t-13
44	mom6m	6-month momentum	CRSP	Monthly	5-month cumulative returns ending one month before month end
45	mve	Size	CRSP	Monthly	Natural log of market capitalization *(prc*shrout)* at end of month t-1
46	mve_ia	Industry-adjusted size	Compustat	Annual	Fiscal-year-end market capitalization *(mve_f)* − industry mean of fiscal-year-end market capitalization
47	operprof	Operating profitability	Compustat	Annual	Revenue *(revt)*- cost of goods sold *(cogs)* - SG&A expense *(xsga)* - interest expense) *(xint)* / common shareholders' equity *(ceq)* in prior year
48	orgcap	Organizational capital	Compustat	Annual	Capitalized SG&A expenses
49	pchcapx_ia	Industry adjusted % change in capital expenditures	Compustat	Annual	Percent change in capital expenditures *(capx)* − industry mean of percent change in capital expenditures
50	pchcurrat	% change in current ratio	Compustat	Annual	Percent change in current ratio *(currat)*
51	pchdepr	% change in depreciation	Compustat	Annual	Percent change in (depreciation/PP&E) *(depr)*
52	pchgm_pchsale	% change in gross margin - % change in sales	Compustat	Annual	Percent change in gross margin *(sale-cogs)* − percent change in sales *(sale)*
53	pchquick	% change in quick ratio	Compustat	Annual	Percent change in quick ratio
54	pchsale_pchinvt	% change in sales - % change in inventory	Compustat	Annual	Percent change in sales *(sale)* − percent change in inventory *(invt)*
55	pchsale_pchrect	% change in sales - % change in receivables	Compustat	Annual	Percent change in sales *(sale)* − percent change in receivables *(rect)*

Table A5 (Continue)

No.	Acronym	Stock Characteristic	Data Source	Frequency	Definition
56	pchsale_pchxsga	% change in sales - % change in SG&A	Compustat	Annual	Percent change in sales *(sale)* − percent change in SG&A *(xsga)*
57	pchsaleinv	% change sales-to-inventory	Compustat	Annual	Percent change in sales-to-inventory *(sale/inv)*
58	pctacc	Percent accruals	Compustat	Annual	(Income before extraordinary items *(ib)* − operating activities/net cash flows *(oancf)*) / absolute income before extraordinary items
59	pricedelay	Price delay	CRSP	Monthly	
60	ps	Financial statements score	Compustat	Annual	Sum of 9 indicator variables
61	quick	Quick ratio	Compustat	Annual	(current assets *(act)* − inventory *(invt)*) / current liabilities *(lct)*
62	rd	R&D increase	Compustat	Annual	An indicator equal to 1 if the percent change in (R&D expense / total assets) *(xrd/at)* is greater than 5%
63	rd_mve	R&D to market capitalization	Compustat	Annual	R&D expense *(xrd)* / fiscal-year-end market capitalization *(mve_f)*
64	rd_sale	R&D to sales	Compustat	Annual	R&D expense *(xrd)* / sales *(sale)*
65	retvol	Return volatility	CRSP	Monthly	Standard deviation of daily returns from month t-1
66	roic	Return on invested capital	Compustat	Annual	(Earnings before interest and taxes *(ebit)* - non-operating income *(nopi)*) / book value of invested capital *(lt+ceq-che)*
67	salecash	Sales to cash	Compustat	Annual	Sales *(sale)* / cash and cash equivalents *(che)*
68	saleinv	Sales to inventory	Compustat	Annual	Sales *(sale)* / inventories *(invt)*
69	salerec	Sales to receivables	Compustat	Annual	Sales *(sale)* / receivables *(rect)*
70	securedind	Secured debt indicator	Compustat	Annual	An indicator equal to 1 if company has secured debt obligations
71	sgr	Sales growth	Compustat	Annual	Percent change in sales *(sale)*
72	sin	Sin stocks	Compustat	Annual	An indicator equal to 1 if a company's primary industry classification is in smoke or tobacco, beer or alcohol, or gaming.
73	sp	Sales to price	Compustat	Annual	Sales *(sale)* / fiscal-year-end market capitalization *(mve_f)*
74	std_dolvol	Volatility of liquidity (dollar trading volume)	CRSP	Monthly	Monthly standard deviation of log daily dollar trading volume *(prc*vol)*
75	std_turn	Volatility of liquidity (share turnover)	CRSP	Monthly	Monthly standard deviation of daily share turnover
76	tang	Debt capacity/firm tangibility	Compustat	Annual	(Cash holdings *(che)* + 0.715 receivables *(rect)* + 0.547 inventory *(invt)* + 0.535 PP&E *(ppent)*) / total assets *(at)*
77	tb	Tax income to book income	Compustat	Annual	Current tax expense *(txfo+txfed)* / maximum federal tax rate *(tr)* / income before extraordinary items *(ib)* If *(txfo)* or *(txfed* is missing, set numerator to *(txt-txdi)*
78	turn	Share turnover	CRSP	Monthly	Average of trading volume *(vol)* in months t-3, t-2 and t-1 / number of shares outstanding *(shrout)* in month t

Table A6: Description of macroeconomic predictors.

No.	Acronym	Macroeconomic Predictor	Data Source	Frequency	Definition	Transformation
1	dp	dividend-price ratio	Robert Shiller's website+S&P Corporation	Monthly	The difference between the log of dividends and the log of prices. Dividends are 12-month moving sums of dividends paid on the S&P 500 index.	Δx_t
2	ep	earnings-price ratio	Robert Shiller's website+S&P Corporation	Monthly	The difference between the log of earnings and the log of prices. Earnings are 12-month moving sums of earnings on the S&P 500 index.	Δx_t
3	bm	book-to-market ratio	Value Line's website	Monthly	The ratio of book value to market value for the Dow Jones Industrial Average.	$\Delta \log(x_t)$
4	nits	net equity expansion	CRSP	Monthly	12-month moving sums of net issues by NYSE listed stocks divided by the total end-of-year market capitalization of NYSE stocks.	Δx_t
5	tbl	Treasury-bill rate	NBER+FRED	Monthly	Treasury-bill rates from 1920 to 1933 are the Yields On Short-Term United States Securities, Three-Six Month Treasury Notes and Certificates, Three Month Treasury Bills for United States. Treasury-bill rates from 1934 to 2005 are the 3- Month Treasury Bill: Secondary Market Rate.	Δx_t
6	tms	term spread	FRED+NBER+Stocks, Bonds, Bills and Inflation Yearbook	Monthly	The difference between the long term yield on government bonds and the Treasury-bill rate.	x_t
7	dfy	default spread	FRED	Monthly	The difference between BAA and AAA-rated corporate bond yields.	Δx_t
8	svar	stock variance	G. William Schwert+CRSP	Monthly	Sum of squared daily returns on the S&P 500.	$\Delta \log(x_t)$

Table A7: Variable significant test results of characteristics where the universe of predictors is CtM.

Predictors are sorted based on their t-stat values within each model, and averaging the predictors rankings among the five NN models to obtain an overall influence in terms of t-stat. In front of each predictor, the scaled t-stats for each model are reported, while their corresponding p-values are underneath of them. We use the red (green) color for predictors that their p-value is less (larger) than 10%. The darkness of color represents its significant. See Table A5 for the definitions of stock characteristics. NN*i*: a multi-layer perceptron with *i*-hidden layer.

char	NN1	NN2	NN3	NN4	NN5	char	NN1	NN2	NN3	NN4	NN5	char	NN1	NN2	NN3	NN4	NN5
mom1m	0.082	0.076	0.084	0.107	0.090	pctacc	0.025	0.029	0.034	0.036	0.033	cashpr	0.023	0.028	0.034	0.035	0.030
	0.000	0.001	0.001	0.000	0.002		0.056	0.068	0.074	0.088	0.086		0.075	0.083	0.077	0.100	0.100
dolvol	0.078	0.074	0.069	0.054	0.053	pchsale_pchinvt	0.025	0.029	0.036	0.035	0.032	age	0.022	0.027	0.033	0.035	0.031
	0.000	0.001	0.003	0.019	0.019		0.058	0.068	0.066	0.092	0.085		0.089	0.088	0.090	0.097	0.095
std_turn	0.052	0.054	0.060	0.077	0.055	baspread	0.024	0.028	0.035	0.037	0.032	salecash	0.022	0.027	0.032	0.035	0.031
	0.001	0.004	0.006	0.003	0.016		0.061	0.075	0.064	0.081	0.090		0.079	0.090	0.094	0.095	0.095
turn	0.043	0.040	0.049	0.068	0.046	acc	0.026	0.028	0.035	0.036	0.032	grltnoa	0.023	0.027	0.033	0.035	0.030
	0.005	0.020	0.015	0.007	0.026		0.053	0.080	0.069	0.091	0.086		0.076	0.089	0.088	0.093	0.099
ill	0.049	0.040	0.046	0.042	0.036	dy	0.025	0.028	0.033	0.037	0.032	hire	0.022	0.027	0.033	0.034	0.031
	0.002	0.018	0.024	0.051	0.064		0.055	0.074	0.082	0.082	0.092		0.080	0.091	0.092	0.106	0.096
divi	0.033	0.036	0.044	0.050	0.039	ep	0.026	0.028	0.033	0.037	0.031	chatoia	0.021	0.026	0.032	0.035	0.031
	0.021	0.028	0.027	0.025	0.052		0.049	0.076	0.083	0.081	0.090		0.096	0.098	0.105	0.094	0.099
chmom	0.029	0.035	0.039	0.045	0.041	quick	0.025	0.028	0.035	0.036	0.031	chempia	0.021	0.027	0.033	0.035	0.030
	0.034	0.032	0.045	0.040	0.046		0.055	0.071	0.067	0.087	0.096		0.091	0.083	0.092	0.093	0.103
bm	0.033	0.033	0.038	0.044	0.037	securedind	0.025	0.028	0.034	0.036	0.031	orgcap	0.022	0.027	0.033	0.035	0.030
	0.019	0.040	0.049	0.043	0.059		0.059	0.083	0.075	0.082	0.094		0.078	0.092	0.090	0.097	0.101
operprof	0.031	0.032	0.038	0.042	0.036	agr	0.024	0.028	0.034	0.036	0.032	pchcurrat	0.023	0.027	0.032	0.035	0.030
	0.025	0.048	0.050	0.050	0.064		0.061	0.077	0.076	0.086	0.088		0.078	0.086	0.097	0.103	0.106
retvol	0.028	0.031	0.037	0.040	0.034	pchsaleinv	0.024	0.028	0.034	0.036	0.032	ps	0.022	0.027	0.032	0.034	0.031
	0.044	0.048	0.059	0.062	0.074		0.066	0.081	0.079	0.086	0.086		0.083	0.091	0.094	0.106	0.100
indmom	0.026	0.031	0.037	0.042	0.038	rd_mve	0.024	0.029	0.034	0.035	0.032	cfp_ia	0.020	0.026	0.033	0.035	0.030
	0.052	0.053	0.059	0.049	0.058		0.064	0.069	0.082	0.098	0.087		0.105	0.097	0.090	0.099	0.105
cfp	0.029	0.030	0.037	0.039	0.035	bm_ia	0.023	0.028	0.033	0.036	0.032	salerec	0.021	0.027	0.032	0.035	0.030
	0.034	0.061	0.058	0.067	0.072		0.075	0.074	0.090	0.084	0.086		0.101	0.091	0.102	0.099	0.103
betasq	0.050	0.029	0.045	0.040	0.031	absacc	0.024	0.026	0.034	0.036	0.033	saleinv	0.022	0.025	0.032	0.034	0.030
	0.002	0.068	0.025	0.064	0.093		0.066	0.105	0.077	0.085	0.083		0.082	0.108	0.101	0.104	0.100
mom12m	0.027	0.029	0.035	0.040	0.035	chinv	0.024	0.028	0.033	0.036	0.033	pchgm_pchsale	0.020	0.025	0.031	0.035	0.030
	0.046	0.067	0.068	0.060	0.067		0.066	0.079	0.092	0.090	0.080		0.108	0.112	0.105	0.097	0.105
std_dolvol	0.027	0.030	0.035	0.041	0.033	egr	0.024	0.028	0.034	0.036	0.031	grcapx	0.018	0.026	0.031	0.034	0.030
	0.047	0.062	0.065	0.057	0.080		0.063	0.079	0.074	0.087	0.096		0.141	0.102	0.107	0.103	0.100

Table A7 (cont.)

char	NN1	NN2	NN3	NN4	NN5	char	NN1	NN2	NN3	NN4	NN5	char	NN1	NN2	NN3	NN4	NN5
idiovol	0.026	0.032	0.036	0.038	0.033	currat	0.025	0.028	0.034	0.036	0.031	pchsale_pchxsga	0.018	0.026	0.031	0.034	0.030
	0.046	0.048	0.064	0.070	0.080		0.059	0.078	0.080	0.087	0.095		0.133	0.103	0.105	0.103	0.100
BETA	0.048	0.029	0.043	0.041	0.031	roic	0.024	0.028	0.034	0.036	0.031	chpmia	0.018	0.025	0.031	0.035	0.030
	0.002	0.070	0.030	0.056	0.095		0.060	0.078	0.076	0.089	0.096		0.139	0.108	0.103	0.098	0.102
gma	0.026	0.029	0.036	0.037	0.033	invest	0.025	0.028	0.033	0.035	0.031	pchcapx_ia	0.020	0.026	0.031	0.035	0.030
	0.047	0.068	0.065	0.076	0.080		0.058	0.077	0.084	0.090	0.092		0.103	0.108	0.114	0.100	0.107
cashdebt	0.026	0.029	0.034	0.039	0.032	lev	0.024	0.028	0.033	0.036	0.031	rd	0.019	0.025	0.031	0.034	0.030
	0.045	0.069	0.075	0.068	0.088		0.062	0.072	0.086	0.083	0.092		0.122	0.111	0.108	0.101	0.104
sp	0.026	0.029	0.035	0.037	0.032	maxret	0.023	0.028	0.033	0.036	0.031	depr	0.018	0.025	0.031	0.034	0.030
	0.049	0.071	0.063	0.075	0.085		0.071	0.076	0.083	0.088	0.091		0.139	0.105	0.108	0.101	0.106
mom6m	0.025	0.030	0.034	0.038	0.034	rd_sale	0.024	0.028	0.034	0.035	0.031	pricedelay	0.018	0.025	0.031	0.035	0.030
	0.057	0.062	0.074	0.077	0.076		0.070	0.076	0.079	0.093	0.094		0.132	0.117	0.104	0.097	0.105
divo	0.028	0.029	0.035	0.037	0.032	mom36m	0.022	0.028	0.033	0.035	0.032	tang	0.020	0.025	0.032	0.034	0.030
	0.038	0.074	0.072	0.080	0.086		0.087	0.081	0.084	0.098	0.084		0.108	0.118	0.098	0.101	0.102
sin	0.025	0.028	0.034	0.038	0.038	mve_ia	0.022	0.026	0.033	0.035	0.032	tb	0.018	0.025	0.031	0.034	0.031
	0.057	0.072	0.077	0.070	0.055		0.086	0.096	0.094	0.094	0.090		0.158	0.112	0.110	0.104	0.096
sgr	0.023	0.030	0.035	0.037	0.033	chcsho	0.020	0.026	0.032	0.036	0.032	pchdepr	0.020	0.025	0.031	0.034	0.030
	0.067	0.064	0.065	0.079	0.080		0.109	0.097	0.093	0.091	0.086		0.112	0.125	0.105	0.107	0.102
lgr	0.025	0.029	0.034	0.037	0.033	pchquick	0.024	0.027	0.033	0.034	0.031	pchsale_pchrect	0.019	0.025	0.031	0.034	0.030
	0.053	0.062	0.080	0.078	0.081		0.062	0.091	0.084	0.100	0.095		0.139	0.117	0.112	0.110	0.105
mve	0.025	0.028	0.034	0.039	0.032	convind	0.021	0.026	0.032	0.036	0.031	herf	0.018	0.024	0.031	0.033	0.030
	0.057	0.078	0.076	0.070	0.084		0.095	0.099	0.100	0.087	0.091		0.141	0.132	0.114	0.111	0.107
scaling factor	55.054	50.966	51.511	77.893	97.026												

Table A8: Variable significant test results of top-34 interaction predictors where the universe of them is CtM.

Predictors are sorted based on their t-stat values within each model, and averaging the predictors rankings among the five NN models to obtain an overall influence in terms of t-stat. In front of each predictor, the scaled t-stats for each model are reported, while their corresponding p-values are underneath of them. We use the red (green) color for predictors that their p-value is less (larger) than 10%. The darkness of red (green) color shows its significant (insignificant) level. See Tables A5 and A6 for the definitions of stock characteristics and macroeconomic predictors. NNi: a multi-layer perceptron with i-hidden layer.

interaction	NN1	NN2	NN3	NN4	NN5	interaction	NN1	NN2	NN3	NN4	NN5	interaction	NN1	NN2	NN3	NN4	NN5
securedind*dp	0.114	0.074	0.078	0.068	0.051	chinv*dp	0.044	0.045	0.043	0.047	0.035	ill*dp	0.093	0.055	0.039	0.039	0.033
	0.000	0.000	0.002	0.008	0.023		0.004	0.011	0.030	0.033	0.067		0.000	0.004	0.042	0.065	0.078
chmom*dp	0.146	0.071	0.074	0.062	0.041	bm*ep	0.066	0.037	0.047	0.050	0.034	std_dolvol*dp	0.047	0.038	0.040	0.040	0.034
	0.000	0.002	0.001	0.011	0.045		0.000	0.026	0.019	0.025	0.074		0.003	0.023	0.037	0.059	0.076
mom6m*dp	0.181	0.081	0.082	0.058	0.037	lev*ep	0.099	0.042	0.055	0.040	0.034	tang*ep	0.035	0.036	0.041	0.040	0.035
	0.000	0.000	0.002	0.013	0.059		0.000	0.015	0.011	0.057	0.073		0.013	0.028	0.034	0.060	0.068
securedind*ep	0.088	0.062	0.069	0.061	0.046	betasq*ep	0.112	0.034	0.038	0.050	0.039	cashdebt*dfy	0.032	0.033	0.039	0.043	0.036
	0.000	0.003	0.003	0.010	0.031		0.000	0.034	0.051	0.028	0.054		0.025	0.047	0.049	0.049	0.062
sin*ep	0.093	0.062	0.065	0.054	0.045	sin*dfy	0.036	0.033	0.042	0.050	0.041	ps*dp	0.039	0.033	0.041	0.041	0.033
	0.000	0.002	0.004	0.019	0.033		0.012	0.043	0.034	0.027	0.045		0.008	0.038	0.034	0.055	0.081
divo*dp	0.101	0.046	0.058	0.056	0.040	divi*dp	0.059	0.035	0.038	0.046	0.035	ps*ep	0.038	0.033	0.040	0.041	0.033
	0.000	0.010	0.008	0.017	0.047		0.001	0.033	0.049	0.037	0.070		0.010	0.038	0.041	0.058	0.080
divi*ep	0.070	0.043	0.048	0.063	0.043	pchsaleinv*tbl	0.078	0.032	0.043	0.045	0.034	pchsale_pchxsga*dp	0.035	0.033	0.037	0.042	0.034
	0.000	0.014	0.018	0.012	0.038		0.000	0.046	0.032	0.040	0.075		0.014	0.039	0.053	0.053	0.077
std_turn*dp	0.060	0.050	0.050	0.048	0.039	cashdebt*bm	0.048	0.037	0.042	0.040	0.034	mve*svar	0.033	0.032	0.038	0.041	0.035
	0.001	0.008	0.013	0.029	0.047		0.004	0.027	0.035	0.061	0.077		0.021	0.049	0.047	0.060	0.072
mom6m*ep	0.114	0.053	0.052	0.046	0.035	turn*ep	0.043	0.033	0.039	0.043	0.037	quick*tms	0.032	0.033	0.037	0.040	0.035
	0.000	0.005	0.012	0.033	0.068		0.005	0.039	0.040	0.049	0.056		0.023	0.043	0.057	0.062	0.069
betasq*dp	0.077	0.054	0.041	0.045	0.039	securedind*tbl	0.033	0.034	0.040	0.046	0.038	sp*tbl	0.032	0.032	0.037	0.041	0.033
	0.000	0.004	0.038	0.040	0.054		0.019	0.035	0.039	0.035	0.057		0.022	0.050	0.055	0.057	0.082
ill*ep	0.092	0.055	0.041	0.043	0.036	dolvol*svar	0.037	0.033	0.041	0.041	0.036						
	0.000	0.005	0.037	0.046	0.063		0.010	0.043	0.035	0.056	0.065						
orgcap*dp	0.052	0.040	0.043	0.046	0.037	BETA*ep	0.035	0.036	0.046	0.042	0.033						
	0.003	0.020	0.032	0.034	0.061		0.016	0.031	0.023	0.051	0.082						
scaling factor	55.054	50.966	51.511	77.893	97.026												

Table A9: Variable significant test results of bottom-48 interaction predictors where the universe of them is CtM.

Predictors are sorted based on their t-stat values within each model, and averaging the predictors rankings among the five NN models to obtain an overall influence in terms of t-stat. In front of each predictor, the scaled t-stats for each model are reported, while their corresponding p-values are underneath of them. We use the green color for predictors that their p-value is larger than 10%. The darkness of green color shows its insignificant level. See Tables A5 and A6 for the definitions of stock characteristics and macroeconomic predictors. NN*i*: a multi-layer perceptron with *i*-hidden layer.

interaction	NN1	NN2	NN3	NN4	NN5	interaction	NN1	NN2	NN3	NN4	NN5	interaction	NN1	NN2	NN3	NN4	NN5
chcsho*svar	0.017	0.025	0.031	0.034	0.030	pricedelay*tms	0.017	0.024	0.030	0.034	0.030	depr*tbl	0.019	0.023	0.029	0.033	0.029
	0.170	0.123	0.115	0.106	0.109		0.169	0.133	0.126	0.104	0.104		0.128	0.152	0.131	0.115	0.110
pchsale_pchrect*tbl	0.020	0.024	0.030	0.034	0.030	tang*dfy	0.019	0.024	0.030	0.033	0.029	chpmia*dfy	0.018	0.023	0.029	0.034	0.030
	0.126	0.128	0.126	0.101	0.105		0.125	0.123	0.117	0.110	0.111		0.149	0.164	0.135	0.105	0.108
tb*tms	0.018	0.024	0.031	0.034	0.030	convind*ntis	0.019	0.024	0.030	0.033	0.030	pchcapx_ia*tbl	0.018	0.023	0.030	0.034	0.029
	0.136	0.129	0.109	0.102	0.106		0.118	0.138	0.122	0.113	0.109		0.156	0.144	0.126	0.109	0.112
pchcapx_ia*ntis	0.020	0.024	0.030	0.034	0.030	depr*ntis	0.019	0.024	0.030	0.033	0.029	rd*ntis	0.018	0.024	0.030	0.033	0.029
	0.115	0.126	0.119	0.104	0.105		0.124	0.125	0.120	0.111	0.115		0.149	0.134	0.126	0.114	0.113
pchsale_pchrect*svar	0.018	0.025	0.031	0.034	0.030	chpmia*svar	0.018	0.023	0.030	0.033	0.030	pchsale_pchxsga*tbl	0.017	0.023	0.030	0.034	0.029
	0.141	0.124	0.110	0.105	0.107		0.144	0.141	0.121	0.115	0.105		0.174	0.157	0.124	0.109	0.109
chatoia*tbl	0.020	0.024	0.030	0.034	0.030	convind*dfy	0.018	0.024	0.030	0.034	0.030	tb*svar	0.017	0.023	0.030	0.033	0.029
	0.114	0.134	0.115	0.105	0.107		0.150	0.142	0.121	0.108	0.106		0.154	0.136	0.120	0.111	0.107
tang*svar	0.018	0.024	0.030	0.034	0.030	pchcapx_ia*dfy	0.018	0.023	0.029	0.034	0.030	pchdepr*svar	0.017	0.023	0.030	0.033	0.029
	0.136	0.122	0.115	0.104	0.104		0.152	0.150	0.141	0.108	0.106		0.155	0.140	0.114	0.117	0.110
pchsale_pchrect*ntis	0.020	0.025	0.030	0.033	0.030	convind*tbl	0.018	0.024	0.030	0.033	0.030	herf*svar	0.017	0.024	0.030	0.033	0.030
	0.120	0.127	0.123	0.115	0.106		0.141	0.135	0.115	0.117	0.108		0.172	0.139	0.121	0.115	0.109
absacc*ntis	0.020	0.024	0.031	0.033	0.030	herf*tms	0.018	0.024	0.030	0.034	0.029	pchdepr*tbl	0.017	0.024	0.029	0.033	0.029
	0.112	0.131	0.114	0.113	0.104		0.147	0.140	0.125	0.103	0.113		0.157	0.139	0.147	0.113	0.110
pchgm_pchsale*ntis	0.020	0.025	0.031	0.033	0.029	pricedelay*ntis	0.018	0.023	0.030	0.033	0.030	pchgm_pchsale*dfy	0.018	0.024	0.029	0.033	0.029
	0.112	0.122	0.114	0.115	0.110		0.140	0.157	0.122	0.113	0.108		0.145	0.132	0.129	0.117	0.110
pchsale_pchrect*dfy	0.020	0.023	0.030	0.034	0.030	herf*ntis	0.019	0.024	0.029	0.034	0.029	tb*ntis	0.019	0.023	0.029	0.033	0.029
	0.107	0.150	0.121	0.103	0.111		0.130	0.127	0.128	0.107	0.110		0.129	0.146	0.132	0.113	0.110
convind*svar	0.018	0.025	0.031	0.034	0.030	tb*tbl	0.017	0.023	0.030	0.034	0.030	pricedelay*dfy	0.017	0.023	0.029	0.033	0.030
	0.138	0.125	0.116	0.110	0.106		0.165	0.144	0.116	0.108	0.106		0.155	0.165	0.134	0.112	0.105
pchcapx_ia*svar	0.019	0.024	0.031	0.033	0.030	depr*dfy	0.019	0.023	0.030	0.033	0.029	pricedelay*svar	0.017	0.023	0.030	0.033	0.029
	0.136	0.137	0.116	0.114	0.108		0.116	0.143	0.123	0.111	0.109		0.153	0.149	0.131	0.116	0.106
chpmia*ntis	0.019	0.023	0.030	0.034	0.030	indmom*dfy	0.018	0.023	0.030	0.034	0.030	pricedelay*tbl	0.018	0.024	0.029	0.033	0.029
	0.118	0.140	0.121	0.105	0.106		0.144	0.162	0.123	0.106	0.106		0.151	0.139	0.135	0.115	0.116
rd*dfy	0.019	0.024	0.030	0.034	0.030	pchdepr*ntis	0.018	0.024	0.030	0.033	0.029	herf*dfy	0.017	0.022	0.029	0.033	0.029
	0.132	0.143	0.115	0.107	0.110		0.147	0.135	0.125	0.114	0.109		0.170	0.178	0.136	0.106	0.114
absacc*svar	0.019	0.024	0.031	0.033	0.029	chpmia*tbl	0.017	0.023	0.029	0.034	0.030	herf*tbl	0.016	0.023	0.028	0.033	0.029
	0.123	0.138	0.115	0.111	0.112		0.155	0.165	0.134	0.106	0.105		0.184	0.152	0.144	0.117	0.108
scaling factor	55.054	50.966	51.511	77.893	97.026												

Table A10: Quantile of characteristics where the universe of predictors is the CpM.

Predictors are sorted based on their t-stat values within each model, and averaging the predictors rankings among the five NN models to obtain an overall influence in terms of t-stat. In front of each predictor, 95% quantile is reported. See Table A5 for the definitions of stock characteristics. NN*i*: a multi-layer perceptron with *i*-hidden layer.

characteristics	NN1	NN2	NN3	NN4	NN5	characteristics	NN1	NN2	NN3	NN4	NN5
mom1m	0.099	0.098	0.099	0.086	0.069	currat	0.102	0.099	0.095	0.084	0.068
std_turn	0.101	0.101	0.097	0.085	0.068	pchcurrat	0.102	0.099	0.095	0.086	0.068
turn	0.099	0.098	0.096	0.086	0.067	dy	0.101	0.100	0.097	0.088	0.067
dolvol	0.100	0.099	0.096	0.088	0.067	convind	0.104	0.100	0.099	0.088	0.068
chmom	0.098	0.101	0.096	0.087	0.068	chinv	0.100	0.100	0.097	0.085	0.066
divi	0.098	0.102	0.097	0.091	0.069	rd_mve	0.101	0.098	0.096	0.085	0.069
indmom	0.100	0.099	0.096	0.085	0.067	pctacc	0.103	0.099	0.095	0.086	0.068
ill	0.101	0.099	0.095	0.089	0.069	salecash	0.100	0.100	0.098	0.084	0.066
betasq	0.099	0.101	0.095	0.084	0.070	baspread	0.098	0.101	0.100	0.087	0.067
gma	0.100	0.102	0.097	0.087	0.069	rd_sale	0.100	0.099	0.094	0.087	0.068
bm	0.101	0.101	0.096	0.084	0.069	ep	0.104	0.100	0.097	0.087	0.067
sp	0.100	0.102	0.093	0.084	0.069	hire	0.101	0.098	0.100	0.085	0.067
operprof	0.102	0.097	0.094	0.087	0.068	orgcap	0.101	0.099	0.097	0.088	0.068
cashdebt	0.101	0.102	0.098	0.087	0.067	chcsho	0.100	0.101	0.098	0.086	0.068
BETA	0.104	0.098	0.096	0.090	0.067	divo	0.103	0.101	0.097	0.087	0.068
sgr	0.100	0.101	0.098	0.085	0.070	mve_ia	0.101	0.098	0.096	0.085	0.066
mve	0.100	0.101	0.098	0.086	0.070	ps	0.101	0.100	0.097	0.086	0.068
roic	0.100	0.099	0.098	0.088	0.069	grltnoa	0.101	0.102	0.095	0.086	0.068
mom12m	0.099	0.098	0.096	0.088	0.068	chempia	0.101	0.102	0.095	0.088	0.069
sin	0.100	0.097	0.098	0.084	0.068	age	0.099	0.099	0.092	0.084	0.067
pchsaleinv	0.104	0.098	0.099	0.087	0.068	pchgm_pchsale	0.101	0.098	0.099	0.086	0.069
lgr	0.097	0.100	0.098	0.086	0.067	tang	0.099	0.098	0.097	0.084	0.069
std_dolvol	0.100	0.100	0.097	0.083	0.066	mom36m	0.098	0.100	0.094	0.086	0.067
mom6m	0.101	0.100	0.097	0.088	0.066	grcapx	0.098	0.099	0.096	0.087	0.068
idiovol	0.101	0.101	0.097	0.086	0.066	cfp_ia	0.102	0.100	0.097	0.086	0.068
quick	0.103	0.101	0.096	0.084	0.069	saleinv	0.103	0.099	0.095	0.086	0.066
invest	0.100	0.097	0.099	0.087	0.068	pchsale_pchxsga	0.099	0.101	0.094	0.087	0.067
agr	0.103	0.098	0.096	0.087	0.069	salerec	0.102	0.099	0.095	0.087	0.069
cfp	0.106	0.100	0.098	0.087	0.070	pricedelay	0.101	0.103	0.097	0.087	0.069
lev	0.100	0.102	0.096	0.085	0.068	pchsale_pchrect	0.100	0.098	0.097	0.087	0.067
pchsale_pchinvt	0.102	0.097	0.096	0.086	0.068	absacc	0.100	0.097	0.098	0.087	0.067
pchquick	0.104	0.099	0.095	0.086	0.068	pchdepr	0.097	0.100	0.093	0.086	0.068
bm_ia	0.099	0.103	0.096	0.084	0.069	securedind	0.100	0.096	0.096	0.086	0.067
retvol	0.100	0.099	0.097	0.088	0.068	tb	0.101	0.098	0.097	0.085	0.067
maxret	0.103	0.101	0.098	0.085	0.066	depr	0.101	0.103	0.094	0.089	0.068
egr	0.102	0.096	0.095	0.088	0.067	chpmia	0.101	0.103	0.097	0.086	0.067
chatoia	0.103	0.100	0.095	0.087	0.068	pchcapx_ia	0.101	0.097	0.096	0.087	0.069
acc	0.101	0.104	0.095	0.085	0.068	rd	0.102	0.098	0.097	0.087	0.070
cashpr	0.099	0.098	0.095	0.086	0.068	herf	0.100	0.101	0.099	0.087	0.068

Table A11: Quantile of characteristics where the universe of predictors is the CtM.

Predictors are sorted based on their t-stat values within each model, and averaging the predictors rankings among the five NN models to obtain an overall influence in terms of t-stat. In front of each predictor, 95% quantile is reported. See Table A5 for the definitions of stock characteristics. NNi: a multi-layer perceptron with i-hidden layer.

char	NN1	NN2	NN3	NN4	NN5	char	NN1	NN2	NN3	NN4	NN5
mom1m	0.0260	0.0309	0.0380	0.0417	0.0392	chinv	0.0261	0.0320	0.0384	0.0415	0.0393
dolvol	0.0261	0.0319	0.0382	0.0419	0.0391	egr	0.0256	0.0311	0.0375	0.0416	0.0392
std_turn	0.0256	0.0313	0.0382	0.0416	0.0388	currat	0.0255	0.0308	0.0376	0.0422	0.0391
turn	0.0255	0.0314	0.0377	0.0411	0.0392	roic	0.0257	0.0318	0.0379	0.0422	0.0391
ill	0.0260	0.0318	0.0378	0.0419	0.0398	invest	0.0260	0.0316	0.0381	0.0420	0.0391
divi	0.0263	0.0312	0.0380	0.0415	0.0394	lev	0.0256	0.0308	0.0381	0.0423	0.0390
chmom	0.0259	0.0307	0.0379	0.0420	0.0395	maxret	0.0260	0.0313	0.0379	0.0422	0.0396
bm	0.0256	0.0311	0.0379	0.0418	0.0390	rd_sale	0.0261	0.0313	0.0379	0.0420	0.0387
operprof	0.0253	0.0313	0.0377	0.0420	0.0393	mom36m	0.0256	0.0316	0.0375	0.0422	0.0395
retvol	0.0266	0.0309	0.0382	0.0426	0.0388	mve_ia	0.0256	0.0313	0.0382	0.0420	0.0393
indmom	0.0260	0.0316	0.0383	0.0419	0.0391	chcsho	0.0257	0.0311	0.0375	0.0421	0.0394
cfp	0.0258	0.0317	0.0382	0.0421	0.0400	pchquick	0.0257	0.0321	0.0375	0.0418	0.0392
betasq	0.0264	0.0312	0.0380	0.0425	0.0394	convind	0.0258	0.0316	0.0385	0.0418	0.0389
mom12m	0.0264	0.0315	0.0381	0.0419	0.0392	cashpr	0.0261	0.0313	0.0378	0.0424	0.0393
std_dolvol	0.0261	0.0319	0.0378	0.0419	0.0391	age	0.0258	0.0318	0.0380	0.0421	0.0393
idiovol	0.0255	0.0314	0.0388	0.0416	0.0391	salecash	0.0256	0.0318	0.0380	0.0420	0.0386
BETA	0.0258	0.0317	0.0376	0.0423	0.0397	grltnoa	0.0260	0.0318	0.0384	0.0425	0.0395
gma	0.0259	0.0315	0.0379	0.0423	0.0397	hire	0.0256	0.0316	0.0387	0.0424	0.0397
cashdebt	0.0256	0.0317	0.0380	0.0427	0.0395	chatoia	0.0261	0.0316	0.0382	0.0419	0.0399
sp	0.0261	0.0321	0.0376	0.0411	0.0391	chempia	0.0260	0.0315	0.0384	0.0412	0.0391
mom6m	0.0256	0.0316	0.0378	0.0424	0.0396	orgcap	0.0259	0.0317	0.0372	0.0429	0.0392
divo	0.0258	0.0317	0.0376	0.0422	0.0397	pchcurrat	0.0256	0.0316	0.0390	0.0421	0.0393
sin	0.0258	0.0312	0.0379	0.0416	0.0391	ps	0.0263	0.0315	0.0383	0.0420	0.0398
sgr	0.0255	0.0315	0.0379	0.0424	0.0394	cfp_ia	0.0259	0.0309	0.0383	0.0423	0.0390
lgr	0.0258	0.0306	0.0382	0.0423	0.0389	salerec	0.0260	0.0314	0.0384	0.0424	0.0395
mve	0.0260	0.0316	0.0381	0.0420	0.0393	saleinv	0.0251	0.0316	0.0378	0.0423	0.0394
pctacc	0.0259	0.0313	0.0377	0.0420	0.0398	pchgm_pchsale	0.0265	0.0317	0.0380	0.0419	0.0391
pchsale_pchinvt	0.0259	0.0316	0.0383	0.0423	0.0395	grcapx	0.0266	0.0313	0.0379	0.0421	0.0394
baspread	0.0260	0.0311	0.0377	0.0422	0.0395	pchsale_pchxsga	0.0256	0.0315	0.0380	0.0419	0.0392
acc	0.0260	0.0314	0.0375	0.0424	0.0395	chpmia	0.0258	0.0310	0.0377	0.0416	0.0389
dy	0.0258	0.0315	0.0381	0.0423	0.0395	pchcapx_ia	0.0259	0.0316	0.0381	0.0422	0.0393
ep	0.0261	0.0315	0.0380	0.0423	0.0396	rd	0.0255	0.0319	0.0382	0.0420	0.0390
quick	0.0256	0.0312	0.0381	0.0418	0.0394	depr	0.0253	0.0312	0.0383	0.0423	0.0388
securedind	0.0258	0.0320	0.0378	0.0420	0.0395	pricedelay	0.0251	0.0312	0.0377	0.0420	0.0394
agr	0.0256	0.0313	0.0384	0.0416	0.0393	tang	0.0250	0.0312	0.0385	0.0416	0.0390
pchsaleinv	0.0260	0.0314	0.0378	0.0421	0.0394	tb	0.0261	0.0316	0.0379	0.0419	0.0387
rd_mve	0.0264	0.0315	0.0377	0.0418	0.0395	pchdepr	0.0260	0.0321	0.0377	0.0417	0.0388
bm_ia	0.0265	0.0314	0.0383	0.0422	0.0392	pchsale_pchrect	0.0263	0.0316	0.0379	0.0426	0.0393
absacc	0.0258	0.0319	0.0377	0.0420	0.0387	herf	0.0258	0.0319	0.0378	0.0417	0.0394

Table A12: Quantile of 34 common interaction predictors where the universe of predictors is the CtM.

Predictors are sorted based on their t-stat values within each model, and averaging the predictors rankings among the five NN models to obtain an overall influence in terms of t-stat. In front of each predictor, 95% quantile is reported. See Tables A5 and A6 for the definitions of stock characteristics and macroeconomic predictors, respectively. NNi: a multi-layer perceptron with i-hidden layer.

interaction	NN1	NN2	NN3	NN4	NN5	interaction	NN1	NN2	NN3	NN4	NN5
securedind*dp	0.0258	0.0322	0.0378	0.0420	0.0389	divi*dp	0.0259	0.0312	0.0377	0.0424	0.0391
chmom*dp	0.0257	0.0320	0.0381	0.0428	0.0393	pchsaleinv*tbl	0.0258	0.0313	0.0381	0.0428	0.0387
mom6m*dp	0.0264	0.0314	0.0384	0.0417	0.0393	cashdebt*bm	0.0261	0.0314	0.0384	0.0427	0.0389
securedind*ep	0.0258	0.0320	0.0378	0.0425	0.0394	turn*ep	0.0256	0.0315	0.0373	0.0421	0.0388
sin*ep	0.0264	0.0313	0.0374	0.0415	0.0390	securedind*tbl	0.0257	0.0312	0.0379	0.0420	0.0390
divo*dp	0.0257	0.0312	0.0376	0.0421	0.0391	dolvol*svar	0.0256	0.0318	0.0381	0.0424	0.0396
divi*ep	0.0261	0.0312	0.0375	0.0419	0.0385	BETA*ep	0.0259	0.0318	0.0379	0.0422	0.0394
std_turn*dp	0.0258	0.0316	0.0377	0.0417	0.0389	ill*dp	0.0262	0.0316	0.0370	0.0421	0.0388
mom6m*ep	0.0266	0.0313	0.0378	0.0420	0.0393	std_dolvol*dp	0.0254	0.0310	0.0375	0.0419	0.0392
betasq*dp	0.0260	0.0310	0.0380	0.0419	0.0399	tang*ep	0.0255	0.0311	0.0375	0.0421	0.0394
ill*ep	0.0257	0.0313	0.0376	0.0419	0.0389	cashdebt*dfy	0.0261	0.0322	0.0385	0.0426	0.0390
orgcap*dp	0.0258	0.0310	0.0378	0.0415	0.0392	ps*dp	0.0261	0.0313	0.0372	0.0423	0.0393
chinv*dp	0.0258	0.0316	0.0381	0.0424	0.0395	ps*ep	0.0258	0.0310	0.0376	0.0427	0.0397
bm*ep	0.0258	0.0315	0.0386	0.0428	0.0396	pchsale_pchxsga*dp	0.0260	0.0316	0.0381	0.0423	0.0395
lev*ep	0.0255	0.0312	0.0380	0.0417	0.0389	mve*svar	0.0258	0.0315	0.0376	0.0427	0.0388
betasq*ep	0.0256	0.0307	0.0382	0.0420	0.0396	quick*tms	0.0264	0.0313	0.0385	0.0426	0.0388
sin*dfy	0.0259	0.0312	0.0385	0.0422	0.0393	sp*tbl	0.0258	0.0315	0.0376	0.0422	0.0387

Table A13: Quantile of 48 common interaction predictors where the universe of predictors is the CtM.

Predictors are sorted based on their t-stat values within each model, and averaging the predictors rankings among the five NN models to obtain an overall influence in terms of t-stat. In front of each predictor, 95% quantile is reported. See Tables A5 and A6 for the definitions of stock characteristics and macroeconomic predictors. NN*i*: a multi-layer perceptron with *i*-hidden layer.

interaction	NN1	NN2	NN3	NN4	NN5	interaction	NN1	NN2	NN3	NN4	NN5
chcsho*svar	0.0259	0.0322	0.0385	0.0415	0.0391	herf*tms	0.0256	0.0320	0.0378	0.0416	0.0388
pchsale_pchrect*tbl	0.0260	0.0314	0.0376	0.0414	0.0391	pricedelay*ntis	0.0262	0.0317	0.0377	0.0413	0.0386
tb*tms	0.0260	0.0313	0.0375	0.0419	0.0385	herf*ntis	0.0256	0.0313	0.0373	0.0420	0.0385
pchcapx_ia*ntis	0.0261	0.0315	0.0382	0.0415	0.0392	tb*tbl	0.0258	0.0312	0.0370	0.0414	0.0388
pchsale_pchrect*svar	0.0262	0.0315	0.0379	0.0415	0.0386	depr*dfy	0.0255	0.0311	0.0380	0.0419	0.0387
chatoia*tbl	0.0256	0.0323	0.0380	0.0412	0.0392	indmom*dfy	0.0259	0.0318	0.0378	0.0412	0.0386
tang*svar	0.0254	0.0316	0.0379	0.0414	0.0386	pchdepr*ntis	0.0256	0.0316	0.0379	0.0421	0.0391
pchsale_pchrect*ntis	0.0267	0.0316	0.0381	0.0418	0.0391	chpmia*tbl	0.0256	0.0319	0.0379	0.0420	0.0392
absacc*ntis	0.0259	0.0313	0.0381	0.0415	0.0390	depr*tbl	0.0265	0.0315	0.0377	0.0419	0.0391
pchgm_pchsale*ntis	0.0251	0.0324	0.0377	0.0417	0.0388	chpmia*dfy	0.0257	0.0317	0.0383	0.0417	0.0395
pchsale_pchrect*dfy	0.0255	0.0324	0.0374	0.0416	0.0386	pchcapx_ia*tbl	0.0259	0.0322	0.0389	0.0419	0.0388
convind*svar	0.0259	0.0318	0.0381	0.0410	0.0384	rd*ntis	0.0264	0.0320	0.0387	0.0414	0.0389
pchcapx_ia*svar	0.0266	0.0316	0.0379	0.0422	0.0386	pchsale_pchxsga*tbl	0.0260	0.0315	0.0373	0.0420	0.0392
chpmia*ntis	0.0256	0.0316	0.0379	0.0419	0.0390	tb*svar	0.0261	0.0314	0.0379	0.0416	0.0390
rd*dfy	0.0263	0.0316	0.0374	0.0414	0.0393	pchdepr*svar	0.0258	0.0315	0.0373	0.0417	0.0388
absacc*svar	0.0259	0.0318	0.0388	0.0419	0.0398	herf*svar	0.0261	0.0315	0.0376	0.0416	0.0389
pricedelay*tms	0.0259	0.0313	0.0379	0.0423	0.0389	pchdepr*tbl	0.0260	0.0320	0.0383	0.0419	0.0390
tang*dfy	0.0257	0.0310	0.0380	0.0421	0.0389	pchgm_pchsale*dfy	0.0257	0.0315	0.0380	0.0418	0.0391
convind*ntis	0.0260	0.0322	0.0381	0.0411	0.0391	tb*ntis	0.0254	0.0314	0.0373	0.0413	0.0387
depr*ntis	0.0255	0.0317	0.0375	0.0418	0.0384	pricedelay*dfy	0.0260	0.0322	0.0379	0.0422	0.0382
chpmia*svar	0.0254	0.0320	0.0377	0.0421	0.0388	pricedelay*svar	0.0252	0.0315	0.0382	0.0415	0.0388
convind*dfy	0.0258	0.0320	0.0379	0.0419	0.0385	pricedelay*tbl	0.0259	0.0317	0.0381	0.0422	0.0392
pchcapx_ia*dfy	0.0260	0.0315	0.0386	0.0419	0.0390	herf*dfy	0.0258	0.0314	0.0376	0.0418	0.0392
convind*tbl	0.0254	0.0314	0.0379	0.0415	0.0383	herf*tbl	0.0256	0.0312	0.0380	0.0421	0.0391