Can deep neural networks outperform Fama-MacBeth regression and other supervised learning approaches in stock returns prediction with asset-pricing factors?

Huei-Wen Teng*

Yu-Hsien Li†

February 11, 2023

Abstract

In asset pricing, most studies focus on finding new factors, such as macroeconomic factors or firm characteristics, to explain risk premiums. Investigating whether these factors help forecast stock returns remains active research in finance and computer science. This paper conducts an extensive comparative analysis using a large set of pricing factors. It compares out-of-sample stock-level and portfolio-level prediction performance among neural networks, the traditional Fama-MacBeth regression, and other supervised learning algorithms such as regression and tree-based algorithms. Our analysis shows the benefit of employing neural networks, and deeper neural networks enjoy marginal improvements in terms of prediction.

Keywords:asset pricing, Fama-MacBeth regression, elastic net, regression tree, boosting, neural network

JEL Codes: C14, C45, C53, C55, C58, G12,

*Department of Information Management and Finance, National Yang Ming Chiao Tung University, Taiwan. E-mail: venteng@gmail.com.

†Taishin International Bank, Taipei, Taiwan. E-mail: k8508011@gmail.com.

1 Introduction

Understanding the relationship between cross-sectional stock returns and systematic risk has been one of finance's most fruitful research areas. The celebrated capital asset pricing model (CAPM) initiated in Sharpe (1964) explains the variation of stock returns by market excess returns. Shortly after Roll and Ross (1980), Fama and French (1993), and Fama and French (1996), firm characteristics are founded as critical roles in empirical asset pricing. Fama-French 3-factor model, Carhart 4-factor model, and the q model with four factors in Hou et al. (2015) attempt to identify the most influential factors in asset pricing theory.

With the availability of voluminous data, hundreds of factors are investigated and employed for returns prediction. Goyal and Welch (2008) examine the usefulness of pricing factors and find them to perform poorly in predicting the risk premium. Cochrane (2011) identify pricing factors that can provide helpful information for stock returns. Jeremiah et al. (2013) summarize more than three hundred characteristics from previous articles. To avoid data snooping bias, Harvey et al. (2016) propose a higher threshold for evaluating a new factor model with a t-statistic greater than 3.0 in a study with more than 300 pricing factors. Mclean and Pontiff (2016) compare the prediction power for factors in finance, accounting, and economics in cross-sectional stock returns.

The collinearity problem among predictors usually deteriorates statistical inference in a standard regression. A classical econometric approach, the Fama-MacBeth (FM) regression allows solving the collinearity problem (Fama and MacBeth, 1973b). Lewellen (2015) apply the FM regression with up to 15 firm characteristics and find characteristic-based regressions outperform Fama-French three-factor model in out-of-sample data, andGreen et al. (2017) apply FM regression with 12 characteristics to predict stock returns.

In contrast to traditional econometric methods, machine learning includes a wide range of general methodologies, including variable selection, dimension reduction, and prediction with general non-linear models Hastie et al. (2009). Machine learning can be split into unsupervised learning and supervised learning. In unsupervised learning, data are not labeled, and usual tasks include dimension reduction and clustering. In supervised learning, data are labeled, and the main task is to build a flexible non-linear function to map explanatory variables to the response variable. Commonly supervised learning approaches include regression, tree-based methods, and neural networks. With great success in the field of computer science, applications of machine learning in finance are rapidly growing. For example, to reduce estimation errors through dimension reduction, Kelly et al. (2019) propose an instrument principle component analysis to find latent factors to improve predictability.

A neural network is a flexible non-linear function, and a selective view of neural networks in finance can be found in Fan et al. (2021). For example, Tsang and Wong

(2019) propose a deep neural network representing decision steps to solve the multi-period portfolio optimization problem in a discrete-time setting subject to investment constraints. For feature engineering, Feng et al. (2019) propose additional deep factors through deep learning to improve the out-of-sample forecast performance, in contrast to using only Fama-French factors. An auto-encoder model is a well-known deep learning method often used for dimension reduction. For example, Heaton et al. (2016) apply an auto-encoder model to select a small group of stocks to construct an equal-weight portfolio with value closely tracing the target index, and Gu et al. (2021) propose an auto-encoder model to estimate the compensated aggregate risk exposures through non-linear combinations of firm characteristics for individual stock returns.

Studies on predicting stock returns are fast-growing after the existence of predictability of stock returns using past price data has been indicated in the seminal paper by Lo et al. (2000). Instead of predicting the price itself or even its returns, Leitch and Tanner (1991), Christoffersen and Diebold (2006), and Nyberg (2011) suggest forecasting the sign of the stock return. Since White (1988) applies neural networks to predict IBM daily stock returns, there is an increasing interest in forecasting using neural networks. For example, Ticknor (2013) proposes a Bayesian regularized neural network. Variants of architectures in neural networks are proposed: Sang and Di Pierro (2019) improve technical analysis with Long Short-Term Memory (LSTM) neural network, Yu et al. (2020) employ LLE-BP neural network, Chudziak (2022) apply feedforward and recurrent neural networks on NYSE stocks. In the above studies to forecast stock returns, most studies use past price data as predictors because of the ease of data retrieval. Interested readers can find a survey on recent progress in stock returns prediction through neural networks in Jiang (2021).

This paper aims to compare traditional econometric and machine learning methods in return predictability using factors from asset pricing. As an extension to Chen and Guestrin (2016), Green et al. (2017), and Gu et al. (2020), we employ eight macroeconomic factors and 102 firm characteristics and compare three types of approaches: regression-based methods, tree-based methods, and neural networks. We consider simple linear regression, Huber loss function, elastic net, and Fama-MacBeth regression for regression-based methods. For tree-based methods, we use the random forest, boosted regression tree, and XGBoost. To investigate whether a deeper neural network improves prediction, we consider a neural network using one to five layers. We compare the prediction performance of each model by evaluating the out-of-sample R-squares in stock-level and portfolio-level analysis.

The rest of this paper is organized as follows. Section 2 gives and explores the data, and Section 3 reviews three types of models: regression, tree-based algorithms, and neural networks. Section 4 summarizes empirical analysis. The last section concludes.

2 Data

Section 2.1 describes how the data are collected, and Section 2.2 explains how we preprocess the data. Section 2.3 explores the data further.

2.1 Data description

We collect and compute eight macroeconomic factors as described in Goyal and Welch (2008) from https://sites.google.com/view/agoyal145. Furthermore, we follow Green et al. (2017) to calculate the 102 firm characteristics using accounting data from on Center for Research in Security Prices (CRSP), Compustat, and I/B/E/S, and apply SAS scripts from Jeremiah Green's website https://sites.google.com/site/jeremiahrgreenacctg/home. CRSP stores data in the primary stock markets, such as NYSE, AMEX, and Nasdaq. We also extend the study period to 2017. Therefore, our data cover 38 years from January 1980 to December 2017, including roughly 18,428 stocks. The average number of stocks exceeds about 4,500 each month. Because of the enormous size of stocks considered in this paper, it isn't easy to list them as an appendix.

Furthermore, we set a one-month Treasury bill rate as a proxy for the risk-free rate to calculate the excess returns. We provide detailed descriptions of these eight macroeconomic predictors and 102 firm characteristics in Table 8 in Appendix A and summary statistics for raw data in Table 9 in Appendix B, respectively. For the dependent variable, we retrieve data on the monthly stock returns of all firms listed in CRSP, and the monthly stock returns are treated as the ground truth value.

2.2 Data preprocessing

To start, we synchronize data monthly. Among these 102 firm characteristics, 63 are updated annually, 15 are updated quarterly, and 24 are updated monthly. In addition, because these accounting data are released publicly with a delay, we avoid the forward-looking bias as follows. For example, if the fiscal year of a firm ends in December, the quarterly accounting data will be observable at the end of April of the following year. Thus, to predict returns at month $t+1$, we use the most recent monthly-updated firm characteristics at month t, quarterly-updated data at month $(t-4)$, and the annual-updated data at month $(t-6)$.

For outliers, we winsorize data at the 1st and 99th percentiles. A further investigation of the data, Table 9 shows that scales of each predictor vary drastically, which may deteriorate the robustness of some machine learning algorithms. This is possible because if a specific variable has a larger magnitude of the variance than others or outliers exist in the data set, they might dominate the objective function(James et al., 2021; Hastie et al., 2009). On the other hand, Table 9 shows that 93% of firm characteristics have

missing values, and only about 7% of them have complete data.

To avoid estimation errors caused by missing values and unequal scales in predictors, we normalize data as follows. We normalize each non-binary predictor by subtracting its sample mean and dividing its sample standard deviation, calculated using the complete data. Because the mean value of the complete data has been zero after normalization, we plug the missing value with zero.

2.3 Exploratory data analysis

We rank variables according to their feature importance from the largest to the smallest and compare their feature importances in Table 1 and Figure 1. *baspread* produces the most considerable feature importance (0.02191), followed by *mve* (0.1786), *mom6m* (0.01585), *idiovol* (0.01568), and *retvol* (0.01501).

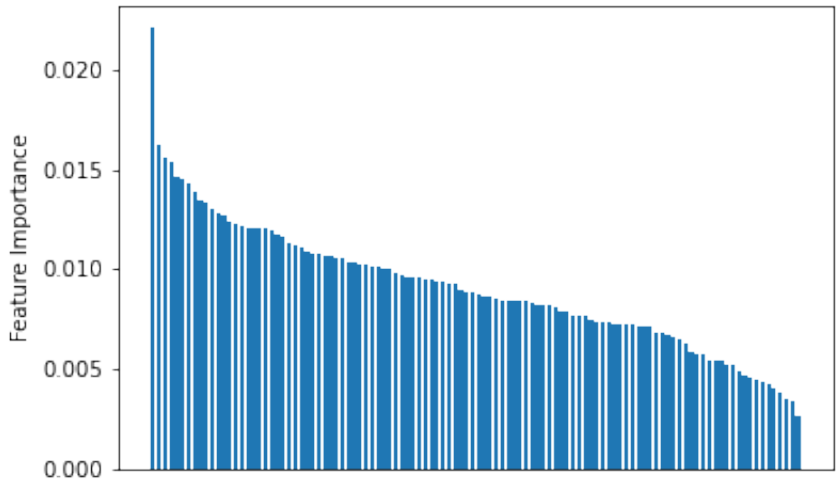


Figure 1: **The feature importance of each variable**. We rank variables according to their feature importances and compare the feature importances using a bar plot.

To check whether the multicollinearity problem exists, we calculate each predictor's variance inflation factor (VIF) and the absolute correlations between all predictors. Multicollinearity will increase standard errors in estimated parameters. As a rule of thumb, VIF larger than ten is considered an indicator of multicollinearity. In addition, the absolute correlation between two predictors larger than 0.2 is considered an indicator of multicollinearity.

Table 2 lists summary statistics of VIFs and absolute correlations: About 9% of firm characteristics whose VIFs are larger than ten, and about 7% of predictors have absolute correlations larger than 0.2. Thus, the multicollinearity problem exists in the

Table 1: **Feature importance** This table lists variables' acronym and their feature importance from largest to small.

Acronym	FI	Acronym	FI	Acronym	FI
baspread	0.0219	*tms*	0.0103	*svar*	0.0075
mve	0.0179	*beta*	0.0103	*egr*	0.0073
mom6m	0.0159	*std_turn*	0.0101	*sue*	0.0073
idiovol	0.0157	*cashdebt*	0.0101	*sgr*	0.0073
retvol	0.0150	*salecash*	0.0100	*dfy*	0.0072
mom1m	0.0149	*roaq*	0.0099	*convind*	0.0071
betasq	0.0135	*ep*	0.0099	*pchsaleinv*	0.0071
cashpr	0.0133	*roeq*	0.0098	*chnanalyst*	0.0070
tb	0.0132	*aeavol*	0.0097	*cfp*	0.0070
std_dolvol	0.0132	*pchcurrat*	0.0096	*sp*	0.0069
indmom	0.0132	*chinv*	0.0096	*invest*	0.0069
herf	0.0130	*pchdepr*	0.0095	*pchsale_pchrect*	0.0067
mom12m	0.0128	*pchquick*	0.0094	*sin*	0.0067
bm_ia	0.0124	*zerotrade*	0.0093	*quick*	0.0067
rd_mve	0.0123	*ear*	0.0093	*dy*	0.0065
pchcapx_ia	0.0123	*cash*	0.0092	*acc*	0.0065
pricedelay	0.0123	*chcsho*	0.0092	*hire*	0.0064
mom36m	0.0122	*gma*	0.0092	*currat*	0.0061
cfp_ia	0.0119	*turn*	0.0092	*IPO*	0.0059
realestate	0.0119	*bm*	0.0089	*sfe*	0.0057
chmom	0.0116	*chpmia*	0.0087	*divi*	0.0055
depr	0.0116	*absacc*	0.0086	*agr*	0.0054
orgcap	0.0115	*nanalyst*	0.0086	*securedind*	0.0053
secured	0.0115	*ms*	0.0083	*roavol*	0.0053
maxret	0.0112	*pctacc*	0.0083	*rd*	0.0052
ill	0.0112	*pchgm_pchsale*	0.0082	*stdacc*	0.0052
saleinv	0.0110	*chtx*	0.0081	*stdcf*	0.0048
salerec	0.0110	*grltnoa*	0.0081	*EP_Welch*	0.0045
lev	0.0108	*ps*	0.0081	*chfeps*	0.0043
$pchsale_pchinvt$	0.0107	*chempia*	0.0080	*ntis*	0.0043
$pchsale_pchxsga$	0.0107	*grcapx*	0.0080	*dp*	0.0040
mve_ia	0.0105	*operprof*	0.0080	*divo*	0.0037
dolvol	0.0105	*lgr*	0.0079	*fgr5yr*	0.0033
roic	0.0105	*chatoia*	0.0079	*BM_Welch*	0.0032
tang	0.0105	*rd_sale*	0.0077	*disp*	0.0027
cinvest	0.0104	*age*	0.0077	*tbl*	0.0024
rsup	0.0104	*nincr*	0.0076		

data. Although a simple approach to solving the multicollinearity problem is to use these predictors, we still use all these predictors to see if machine learning could overcome the multicollinearity problem.

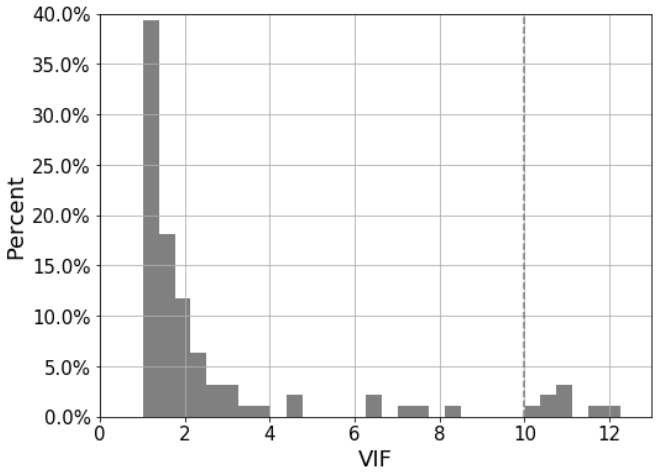
Figure 2 provides box plots of the autocorrelation function (ACF) for each firm characteristic at lags 1, 2, and 6. Most of the average ACF of characteristics excesses 0.25, and 80% of them are significantly different from 0 at the 5% level with the Ljung-Box test. This indicates that serial correlations exist in most of the characteristics. The serial correlation will also cause estimation errors. However, solving serial correlation is out of

Table 2: Descriptive statistics on VIFs and cross-correlations among firm characteristics

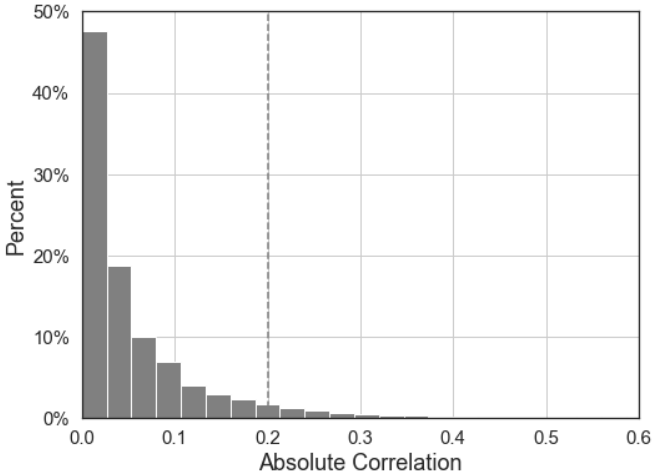
A. Statistics of VIFs and cross-correlation

	Avg.	std	kurt	min	25%	50%	75%	max
VIFs	2.81	2.9	3.13	1.02	1.21	1.63	2.42	12.26
Absolute cross-correlations	0.06	0.09	20.42	0.00	0.01	0.03	0.08	0.95

B. Distribution of VIFs



c. Distribution of the absolute cross-correlations among characteristics.



the scope of our studies.

Figure 2: Boxplots for ACF at lags 1, 2, and 6 for each firm characteristic



3 Study plan

To train the model and evaluate prediction performance for each approach, we split the 38-year data into a training period of 25 years (1980/1 - 2004/12), a validation period of 6 years (2005/1 - 2010/12), and a test period of the remaining seven years (2011/1 - 2017/12). We use the data from the training and validation periods to train a model and obtain estimated parameters. Then, we predict stock returns in the test period using estimated model parameters without re-estimating parameters. Our procedures are motivated by Green et al. (2017) and Gu et al. (2020).

In addition, these twelve approaches can be divided into three categories: regression, tree-based algorithms, and neural networks. Extensive studies on these approaches allow us to compare the classical Fama-Macbeth regression and state-of-the-art machine learning algorithms in empirical asset pricing theory predicting cross-sectional returns. Table 3 lists a total of twelve approaches, which are briefly reviewed in Sections 3.2, 3.3, and 3.4.

Table 3: **A summary of twelve approaches**. This table summarizes twelve approaches compared in this paper, which belong to one of the three categories: regression, tree-based algorithm and neural networks.

Index	Approach	Abbreviation	Category
1	Ordinary least square regression	OLS	Regression
2	OLS with Huber loss function	Huber	Regression
3	Elastic Net	ENet	Regression
4	Fama-MacBeth regression	MF	Regression
5	Random forest	RF	Tree-based algorithm
6	Gradient boosted regression tree	GBRT	Tree-based algorithm
7	XGBoost	XBG	Tree-based algorithm
8	Neural network with one layer	NN1	Neural Network
9	Neural network with two layers	NN2	Neural Network
10	Neural network with three layers	NN3	Neural Network
11	Neural network with four layer	NN4	Neural Network
12	Neural network with five layer	NN5	Neural Network

3.1 The general model

To start, we consider the general form of the asset pricing model to predict the future return:

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

where

$$E_t(r_{i,t+1}) = g(z_{i,t}; \theta), \tag{2}$$

with the p-dimensional predictor $z_{i,t}$. Here, the subscript t indexes time for $t = 1, ..., T$, and the subscript i indexes the i-th firm for $i = 1, ..., N_t$. The p-variate parameter vector is denoted as $\theta = (\theta_1, \ldots, \theta_p)$.

Note that the subscript in N_t indicates the number of firms at time t may be time-variant, and the conditional expectation $g(\cdot)$ is a general function. Equation (2) uses predictors (including macroeconomic factors and firm characteristics) at time t to predict the stock return at time $t + 1$.

Denote the predicted return at $t + 1$ using all information upto time t by

$$\hat{r}_{i,t+1} = g(z_{i,t}; \hat{\theta}).$$

Because $g(\cdot)$ does not depend on i nor t, this is indeed also known as a pooled model in panel data analysis (Hill et al., 2018). This model differs from traditional asset pricing models that use cross-sectional data at each time point or consider a time series model for each firm.

Parameters are decided to minimize the mean squared prediction error (MSE) between the predicted return $\hat{r}_{i,t+1} = g(z_{i,t}; \hat{\theta})$ and the realized return $r_{i,t+1}$: The parameter θ needs to minimize the loss function, which is defined as

$$\mathcal{L}(\theta) = \frac{1}{T} \sum_{t=1}^{T} \frac{1}{N_t} \sum_{i=1}^{N_t} (r_{i,t+1} - \hat{r}_{i,t+1})^2. \tag{3}$$

3.2 Regression-based methods

To start, the ordinary least squares (OLS) model, also known as multiple linear regression, assumes the conditional expectation $g(\cdot)$ to be a linear combination of predictors,

$$g(z_{i,t}; \theta) = z'_{i,t}\theta. \tag{4}$$

Albeit its simplicity, OLS remains a useful and robust model in explaining the significance of predictors Hill et al. (2018).

Moreover, we consider the Huber robust objective function:

$$\mathcal{L}_H(\theta) = \frac{1}{T} \sum_{t=1}^{T} \frac{1}{N_t} \sum_{i=1}^{N_t} H(r_{i,t+1} - g(z_{i,t}; \theta); \xi), \tag{5}$$

where the Huber loss function is

$$H(s;\xi) = \begin{cases} s^2, & \text{if} \quad \mid s \mid \leq \xi; \\ 2\xi \mid s \mid -\xi^2, & \text{if} \quad \mid s \mid > \xi. \end{cases}$$

The Huber loss function allows us to avoid extreme values Huber (1973). Note that $H(\cdot)$ combines the squared loss and absolute loss functions. When the error is smaller than the hyperparameter ξ, it uses the squared loss function. Otherwise, it uses the absolute loss function adjusted by ξ. The hyperparameter ξ can be a hurdle to deciding whether the difference between the realized and expected returns is large enough.

Next, to solve the large p problem, we consider the penalized linear model with a shrinkage method. The shrinkage method refers to shrinking parameters by imposing a penalty for the size of the parameters. The penalized linear model uses the simple linear model in equation (4) but adds a penalty term into the objective function to regularize parameters:

$$\mathcal{L}(\theta;\cdot) = \underbrace{\mathcal{L}(\theta)}_{\text{Loss Function}} + \underbrace{\phi(\theta;\cdot)}_{\text{Penalty}}.$$

Particularly, we consider the elastic-net penalty function,

$$\phi(\theta;\lambda,\rho) = \lambda(1-\rho)\sum_{j=1}^{P} \mid \theta_j \mid + \frac{1}{2}\lambda\rho\sum_{j=1}^{P}\theta_j^2.$$

where λ and ρ are hyperparameters (Zou and Hastie, 2005). Note that when $\rho = 0$, the penalized linear model reduces to the LASSO regression, which imposes the L_1 LASSO penalty $\sum_{j=1}^{P} \mid \theta_j \mid$ to the parameters (Tibshirani, 1996). When $\rho = 1$, the penalized linear model reduces to the ridge regression, which imposes the L_2 penalty for the parameters (Marquardt and Snee, 1975).

Lastly, we follow Lewellen (2015) that uses both the intercept and coefficients for each predictor to conduct the well-known Fama-MacBeth (FM) two-step regression to predict future stock returns to avoid estimation errors in the cross-sectional regression (Fama and MacBeth, 1973a). In the first step, we run a cross-sectional regression at time t:

$$r_{i,t+1} = r_{f,t} + \alpha_t + z'_{i,t}\theta_t + \epsilon_{i,t} \tag{6}$$

where $r_{i,t+1}$ is the return for the i-th firm at time $t+1$ period for $i = 1, 2, \ldots, N_t$ and $t = 1, 2, \ldots, T$. In addition, α_t is the intercept, $r_{f,t}$ is the risk-free rate at time t, $z_{i,t}$ is the risk factors of the ith stock at time t. The estimated coefficients are denoted by $\hat{\alpha}_t$ and $\hat{\theta}_t$ (also called as "factor exposure").

In the second step, we compute the averages of the time series of intercepts and

coefficients by

$$\begin{aligned}\bar{\alpha} &= \frac{1}{T}\sum_{t=1}^{T}\hat{\alpha}_t,\\ \bar{\theta} &= \frac{1}{T}\sum_{t=1}^{T}\hat{\theta}_t.\end{aligned}$$

Then we predict the stock return for the i-th firm at time $t+1$ using the average factor exposure by

$$\hat{r}_{i,t} = \bar{\alpha} + z'_{i,t}\bar{\theta}$$

3.3 Tree-based algorithms

A regression tree is a special type of nonlinear function in Equation (1) (Breiman et al., 2017). Regression trees use a tree to represent the recursive partition among predictors. Each of the terminal nodes or leaves of the tree represents a cell of the partition. The ideal model must distinguish among the potential outcomes in terms of minimum forecast errors. A regression tree with K terminal nodes (leaves) and depth L can be formally expressed as

$$g(z_{i,t};\theta,L) = \sum_{k=1}^{2^L}\theta^{(k)}\mathbf{I}\{z_{i,t}\in C_k(L)\}, \tag{7}$$

where $C_k(L)$ is a one of the K partitions of the data, the constant $\theta^{(k)}$ is defined to be the average of outcomes $r_{i,t+1}$ in partition k, and L is depth of the tree.

We choose the most popular l_2 loss function associated with the forecast error for each branch $C_k(L)$ called "impurity" for the model. The parameters $\Theta = \{\theta^{(k)}, C_k(L)\}_k^{2^L}$ are found by minimizing the objective function

$$\mathcal{L}(\Theta) = \arg\min_{\Theta}\sum_{k=1}^{2^L}\sum_{z_{i,t}\in C_k(L)}\mathcal{L}(r_{i,t+1},\theta^{(k)}). \tag{8}$$

Although the regression tree allows us to analyze data with complicated features interaction easily, one drawback is that we often need more robust results in testing data since the regression tree tends to be deep. Therefore, we use "ensemble" methods as a remedy. Ensemble methods include bagging and boosting, and they use a combination of simple models and aggregate them to improve prediction.

We consider the random forest model an example of bagging (Breiman, 2001). To be more specific, we assume that the average of B bagging trees identically distribute a random variable, each with variance σ and positive pairwise correlation ρ, so that we can

get the variance of the average of trees.

$$\rho\sigma^2 + \frac{1-\rho}{B}\sigma^2$$

We can observe the second term disappears as B increases, but the first term remains, which indicates high correlation will limit the benefits from taking an average of bagging trees. Given a tree as:

$$\hat{g}_b(z_{i,t}; \hat{\theta}_b, L) = \sum_{k=1}^{2^L} \hat{\theta}_b^{(k)} \mathbf{I}_{\{z_{i,t} \in C_k(L)\}}$$

In random forest, we need to tune two parameters, the depth D of trees and the number of trees B in the validation samples. The random forest model will aggregate all of the trees by the average of their outcomes of them.

$$\hat{g}(z_{i,t}; L, B) = \frac{1}{B} \sum_{b=1}^{B} \hat{g}_b(z_{i,t}; \hat{\theta}_b, L).$$

The basic idea of boosting is to use a combination of weak learners to construct a powerful "committee". The weak learner is a model with little predictive power that performs slightly better than a random guess. The boosting method is to sequentially apply the weak learner to repeatedly revise them from the estimation errors and then aggregate the results into one prediction value.

The gradient boosting regression tree (GBRT) uses additive training (Friedman, 2002). That is, we maintain the original model and then add a new one that takes forward stage-wise procedure to fit the prediction residuals from the original model. By doing so, we can obtain fewer estimation biases by combining these two models into an ensemble prediction of the outcome. An example of an additive training process is:

$$\begin{aligned}
&\hat{r}_{i,t+1}^{(0)} = 0 \\
&\hat{r}_{i,t+1}^{(1)} = \hat{r}_{i,t+1}^{(0)} + g_1(z_{i,t}) = 0 + g_1(z_{i,t}) \\
&\hat{r}_{i,t+1}^{(2)} = \hat{r}_{i,t+1}^{(1)} + g_2(z_{i,t}) = g_1(z_{i,t}) + g_2(z_{i,t}) \\
&\cdots \\
&\hat{r}_{i,t+1}^{(b)} = \hat{r}_{i,t+1}^{(b-1)} + g_b(z_{i,t}) = \sum_{b=1}^{B} g_b
\end{aligned}$$

At each step in the forward stage-wise procedure, one must solve

$$\hat{\Theta}_b = \arg\min_{\Theta_b} \sum_{t=1}^{T} \sum_{i=1}^{N} \mathcal{L}(r_{i,t+1}, g_{b-1}(\cdot) + g_b(\cdot; \Theta_b)) \tag{9}$$

for the region set and constants $\Theta_b = \{\theta^{(k)}, C_k(L)\}_1^{2^L}$ of the next tree, given the current

model $g_{b-1}(\cdot)$. Given the regions $C_k(L)$ in each tree, we can find the optimal constants $\theta_b^{(k)}$ in each region is equal to the mean of current residuals $r_{i,t+1} - g_{b-1}(\cdot)$ in each corresponding region.

The weak learner becomes overfitting when fitting the training data, so it usually needs to choose a pre-specific parameter L to control the depth of each weak learner in the boosting procedure. Besides, not only controlling the depth of each weak learner (L), we also add another shrinkage factor into the second step, ν, as a regularization strategy (also known as learning rate). This implementation of shrinkage factor, $\nu \in (0, 1]$, is applied to scale the contribution of each tree when it is added to the approximation which fits the residuals from $b-1$ step. Empirically, it has been found that smaller values of ν and correspondingly larger values of depth L make the model perform better in testing data. The GBRT is a sum of regression trees,

$$g(z_{i,t}; B, v, L) = \sum_{b=1}^{B} v\hat{g}_b(\cdot). \tag{10}$$

The boosting trees have three tuning parameters (L, ν, B).

Chen and Guestrin (2016) propose a novel-tree-based algorithm, called XGBoost. Reviewing equation (10) in GBRT, both GBRT and XGBoost are special cases of gradient boosting tree-based ensemble method. Two major differences between GBRT and XGBoost are addressed below. First, we add weights to each leaf in XGBoost, representing the specific influence on an individual leaf, unlike using the same score on an individual leaf in the normal regression tree. Secondly, a regularization term in the objective function of XGBoost is added to control the complexity of the model and avoid the overfitting problem. Given B trees, the XGBoost model considers

$$\hat{r}_{i,t+1} = g(z_{i,t}; B, v, L, \gamma, \lambda) = \sum_{b=1}^{B} vg_b(\cdot), g_b \in \mathcal{F},$$

where $\mathcal{F} = \{g(z_{i,t}) = w_{q(z_{i,t})}\}(q : \mathbb{R}^P \to M, w \in \mathbb{R}^M)$ is the space of regression trees. M is the number of leaves in tree, and each $w_q(z_{i,t})$ represents the weight of corresponding q-th leaf in each independent tree structure.

$$\mathcal{L}(\Theta) = \sum_{i=1}^{N}\sum_{t=1}^{T} l(r_{i,t+1}, \hat{r_{i,t+1}}) + \sum_{b=1}^{B} \Omega$$
$$\text{where } \Omega(g_b) = \gamma M + \frac{1}{2}\lambda\|w\|^2$$

where l means a differentiable convex loss function. We use mean squared error (MSE) here. The second term, Ω, is used to penalize the complexity of the model structure and makes the model simpler and smoother. When we set all of the regularization parameters

to zero, the objective function will become the same as traditional GBRT.

Besides, since our objective function is differentiable, we can directly optimize the objective function at each step by using second-order approximation during the additive training process. This unique property can also help us to derive how good the model is given the tree structure.

$$\begin{aligned}\mathcal{L}^{(b)} &= \sum_{i=1}^{N}\sum_{t=1}^{T} l(r_{i,t+1}, \hat{r}_{i,t+1}^{(b)}) + \Omega(g_i) \\ &= \sum_{i=1}^{N}\sum_{t=1}^{T} l(r_{i,t+1}, \hat{r}_{i,t+1}^{(b-1)} + g_b(z_{i,t})) + \Omega(g_b) \\ &\simeq \sum_{i=1}^{N}\sum_{t=1}^{T} [l(r_{i,t+1}, \hat{r}_{i,t+1}^{(b-1)}) + f_i g_b(z_{i,t}) + \frac{1}{2} s_i g_b^2(z_{i,t})] + \Omega(g_b)\end{aligned}$$

where f_i and s_i are defined as $f_i = \partial_{\hat{r}_{i,t+1}^{(b-1)}} l(r_{i,t+1}, \hat{r}_{i,t+1}^{(b-1)})$ and $s_i = \partial^2_{\hat{r}_{i,t+1}^{(b-1)}} l(r_{i,t+1}, \hat{r}_{i,t+1}^{(b-1)})$, respectively. Then, we expand $\Omega(g_b)$ and simplify the objective by removing the constants.

$$\mathcal{L}^{(b)} = \sum_{i=1}^{N}\sum_{t=1}^{T} [f_i g_b(z_{i,t}) + \frac{1}{2} s_i g_b^2(z_{i,t})] + rM + \frac{1}{2}\lambda \sum_{j=1}^{M} w_j^2$$

Define $\mathbf{I}_j = \{i \mid q(z_{i,t}) = j\}$ as the instance set of j-th leaf, and we can rewrite the objective function.

$$\mathcal{L}^{(b)} = \sum_{j=1}^{M} [(\sum_{i \in \mathbf{I}_j} f_i) w_j + \frac{1}{2}(\sum_{i \in \mathbf{I}_j} s_i + \lambda) w_j^2] + \gamma M$$

For a fixed structure tree, we could obtain the optimal w_j^*

$$w_j^* = -\frac{\sum_{i \in \mathbf{I}_j} f_i}{\sum_{i \in \mathbf{I}_j} s_i + \lambda}$$

Then, we calculate the corresponding optimal value by

$$\mathcal{L}^{(b)}(q) = -\frac{1}{2} \sum_{j=1}^{M} \frac{(\sum_{i \in \mathbf{I}_j} f_i)^2}{\sum_{i \in \mathbf{I}_j} s_i + \lambda} + \gamma M$$

The above equation can be used as a scoring function to measure the quality of tree structure q.

3.4 Neural networks

The neural network has been one of the most well-known and influential machine learning techniques for prediction. Briefly speaking, the architecture of a deep neural network consists of two types of hyperparameters: the number of hidden layers and the number of neurons in each hidden layer. The number of hidden layers controls the depth of the network. A "deep" neural network has many hidden layers, whereas a "shallow" neural network has a small number of hidden layers. Neurons in the hidden layer connect input variables and contribute to output variables through activation functions.

We apply the traditional "feed-forward" neural network. Our network consists of an input layer, several hidden layers, and an output layer of predicted stock returns. A deep neural network nonlinearly transforms the factors through the hidden layers and aggregates the information of the previous hidden layers to predict stock returns in the output layer. The neural network's architecture directly affects the prediction results: A "shallow" neural network may result in poor model fits, but a "deep" neural network may lead to an overfitting problem. As a result, deciding a proper architecture of a neural network is of critical importance. Unfortunately, there is no general theory to decide the optimal architecture. The optimal architecture is decided using model validation.

We use ReLU as our activation function for hyper-parameters because it could overcome the vanishing gradient problem and allow the neural network to be learned faster. The ReLU activation function is:

$$\text{ReLU}(h) = \begin{cases} 0, & \text{if } h < 0. \\ h, & \text{otherwise.} \end{cases}$$

Moreover, in training a neural network, we set 0.2 for the dropout, set 'adam' for the optimizer, and set epochs as 100 and batch size as 10000.

The recursive output formulation for each neuron in our neural network:

$$h_k^{(l)} = \text{ReLU}(h^{(l-1)'}\theta_k^{(l-1)}), \tag{11}$$

with final output

$$g(z;\theta) = h^{(L-1)'}\theta^{(L-1)}. \tag{12}$$

Define $K^{(l)}$ as the number of neurons in each layer $l = 1, \ldots, L$ and $h_k^{(l)}$ as the output of neuron k ($k = 1, \ldots, K$) in layer l ($l = 1, \ldots, L$). Then, we define the output vector of l-th layer as $h^{(l)} = (h_1^{(l)}, \ldots, h_k^{(l)})$, which includes an intercept and the predictors. Thus, we can recursively initialize the network by setting the input layer as $h(0) = (1, z_1, \ldots, z_N)'$. Again, we choose MSE as the objective function to estimate the weight parameters. To decide the best architecture, we use neural networks with one to five hidden layers: The simplest network contains a single hidden layer of 32 neurons denoted NN1, and NN2 has

two hidden layers with 32 and 16 neurons, respectively. Following this manner, NN5 has five hidden layers with 32, 16, 8, 4, and 2 neurons, respectively. For neural networks, we use 'relu' as the activation function, set 0.2 for the dropout, set 'adam' for the optimizer, and set epochs as 100 and batch size as 10000.

Similar to Gu et al. (2020), we use an ensemble approach in training our neural network to reduce the prediction variation. It is commonly known that random initial weights lead to different weights inferred from the same data of the same neural network. To avoid estimation error due to random start, the ensemble method uses M random seeds to generate initial weights to yield M different weights, then predicts the value by taking an average of predicted values using these M weights (Dietterich, 2002).

4 Empirical analysis

For each model, we compare stock-level out-of-sample Ruares in Section 4.1. In addition, we form bottom-up equal-weight portfolios to confirm robustness for each model in Section 4.2.

4.1 Out-of-sample stock-level analysis

Let $\hat{r}_{i,t+1}$ denote the predicted return of the i-th stock at time $t+1$. We define the out-of-sample root mean squared prediction error (RMSE) between real returns and predicted returns as:

$$\text{RMSE} = \sqrt{\frac{1}{N}\sum_{i\in I, t\in\tau}\left(r_{i,t+1}-\hat{r}_{i,t+1}\right)^2},$$

where N is the total observations in the testing period. Then, we define the out-of-sample mean absolute prediction error (MAE) between real returns and predicted returns using each method.

$$\text{MAE} = \frac{1}{N}\sum_{i\in I, t\in\tau}\left|r_{i,t+1}-\hat{r}_{i,t+1}\right|.$$

Table 4 calculates the out-of-sample RMSE and MAE between real returns and predicted returns using each method. Regarding RMSE and MSE, neural networks produce the most negligible value (except for NN with four layers), followed by ENet, FM, tree-based, Huber, and OLS methods. However, MAE is generally much smaller than RMSE, indicating that errors may be skewed to the right. In other words, large values of out-of-sample errors may exist in our data analysis.

Furthermore, we define the out-of-sample R-squared, denoted as R^2_{oos}, for cross-sectional stock returns as a measure to compare the performance of forecasting each

Table 4: **RMSE and MAE**. This table reports the out-of-sample RMSE and MAE at the stock level for each method

Model	OLS	Huber	ENet	FM	RF	GBRT	XGB	NN1	NN2	NN3	NN4	NN5
RMSE	0.160	0.155	0.152	0.153	0.153	0.154	0.153	0.152	0.152	0.152	0.435	0.152
MAE	0.102	0.094	0.089	0.089	0.090	0.091	0.090	0.089	0.089	0.089	0.411	0.089

excess stock return as

$$R^2_{oos} = 1 - \frac{\sum_{i\in I, t\in\tau}(r_{i,t+1} - \hat{r}_{i,t+1})^2}{\sum_{i\in I, t\in\tau} r^2_{i,t+1}},$$

where I is the set for firms τ is the time set in the testing period. The R^2_{oos} is a convenience measure that allows us to compare the prediction performance of each method. Note its denominator is a constant across all methods, but its numerator measures the squared prediction errors given a method. Hence, a model of a higher R^2_{oos} is preferred because it produces fewer prediction errors.

Table 5 reports R^2_{oos} for a portfolio including all stocks, large-cap portfolio (stocks ranked top 400 by market values), and small-cap portfolio (including stocks ranked bottom 400 by market values) to compare these twelve models summarized in Table 3. For portfolios including all stocks, it is shown that neural networks outperform tree-based algorithms and regressions.

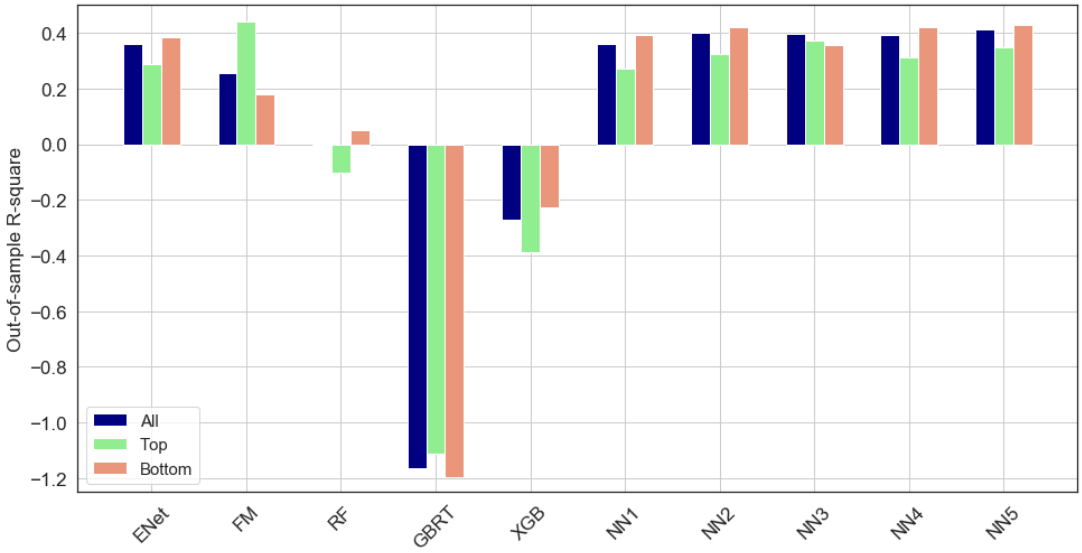
Neural networks outperform all methods in predicting stock returns: R^2_{oos} is 0.36% for NN1 and reaches its peaks at 0.41% for NN5. However, a deeper neural network provides marginal benefits in R^2_{oos}. Unfortunately, tree-Based methods produce negative R^2_{oos}: FM (0.00%), GBRT (-1.16%), and XGB (-0.27%). It is likely that we only consider shallow trees in this study. Finally, the OLS fails to predict returns well, producing the least R^2_{oos} of -9.41%. Then, the Huber produces the second least R^2_{oos} of -3.11%. Two other regression models produce positive R^2_{oos}: ENet produces R^2_{oos} of 0.36%, and FM produces R^2_{oos} of 0.26%. Because ENet tends to choose a smaller set of explanatory variables but produces higher R^2_{oos}, this indicates that including all the predictors does not quarantine better prediction.

One can draw a similar conclusion for the large-cap and small-cap portfolios. Neural network dominates regressions and tree-based algorithms, and Tree-based algorithms generally produce negative R^2_{oos}. Interestingly, both ENet and FM are competitive with neural networks. FM dominates other models for large-cap portfolios and produces the highest R^2_{oos} of 0.44%.

To see whether predicted returns are close to real returns, we provide scatter plots of the predicted returns using each method and real returns and overlay a 45-degree straight line in Figures 3 and 4. First, we see many real returns are pretty significant.

Table 5: **Stock-level R^2_{oos} (%) for the portfolio including all stocks (All), and large-cap portfolio (L), and small-cap portfolio (S)** The large-cap portfolio includes top 400 stocks by market values and small-cap portfolio include bottom 400 stocks by market values. The lower panel provides a visual comparison of the R^2_{oos} (by omitting OLS and Huber due to their large negative values

	OLS	Huber	ENet	FM	RF	GBRT	XGB	NN1	NN2	NN3	NN4	NN5
Portfolio												
All	-9.41	-3.11	0.36	0.26	0.00	-1.16	-0.27	0.36	0.40	0.40	0.39	0.41
Large	-9.39	-2.75	0.29	0.44	-0.10	-1.11	-0.39	0.27	0.32	0.37	0.31	0.35
Small	-9.40	-3.16	0.39	0.18	0.05	-1.20	-0.23	0.39	0.42	0.36	0.42	0.43



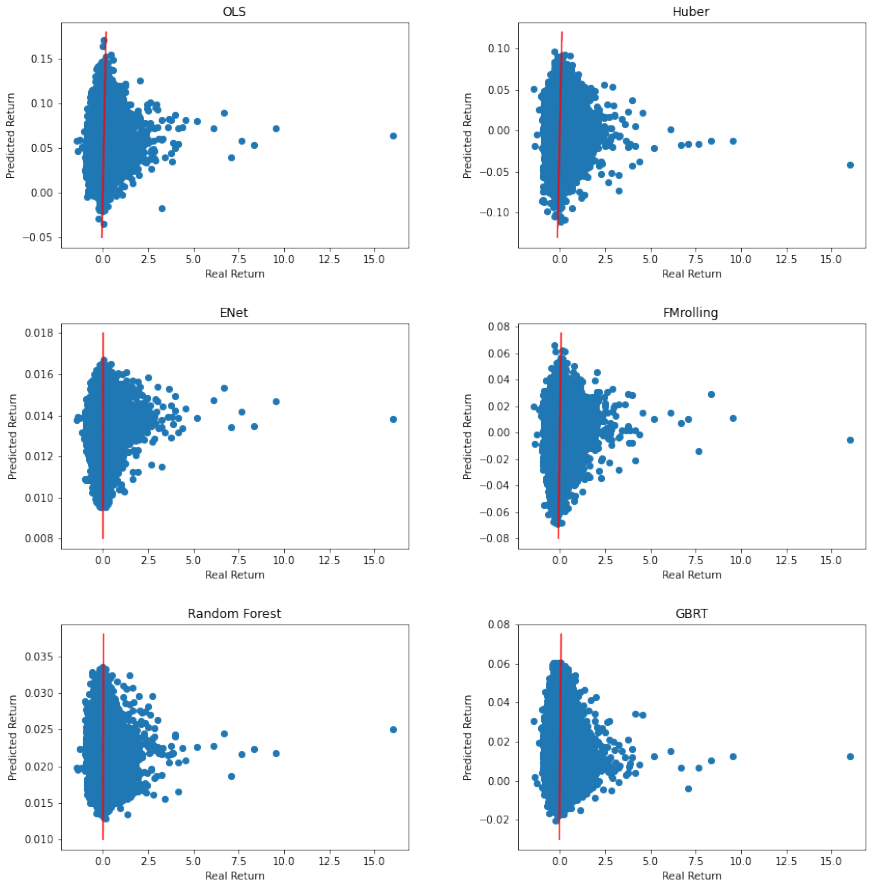
A closer look at the data indicates that about 0.1% data have returns higher than 1.545 and about 0.1% have returns lower than −0.703. However, this is not the case in the predicted returns. Each model yields predicted returns with narrower ranges. This is because the general model (2) is a pooled model, which takes the same parameter across firm and time. As a result, the predicted return does not vary as much as the real return. It is also surprising that the regression-based methods, OLS and ENet, only yield positive predicted returns.

4.2 Out-of-sample portfolio-level analysis

Indeed, whether the stock-level out-of-sample RMSE, MAE, and R^2 are ideal measures for model fitting or prediction is questionable. Specifically, R^2_{oos}'s are very low (<.5%). To investigate whether predicted returns using traditional methods or machine learning benefits portfolio management, we construct a bottom-up equal-weight portfolio as follows:

1. We calculate one-month-ahead predicted returns for each method at the end of each month.

Figure 3: **Scatter plots of the predicted return versus the real return using regression-based methods, RF, and GBRT**. This figure provides scatter plots of the predicted versus real returns. To see if predicted returns are close to real returns, we also overlay a 45-degree straight line in red color.



2. We rank the stocks by their predicted returns and sort them into ten deciles.

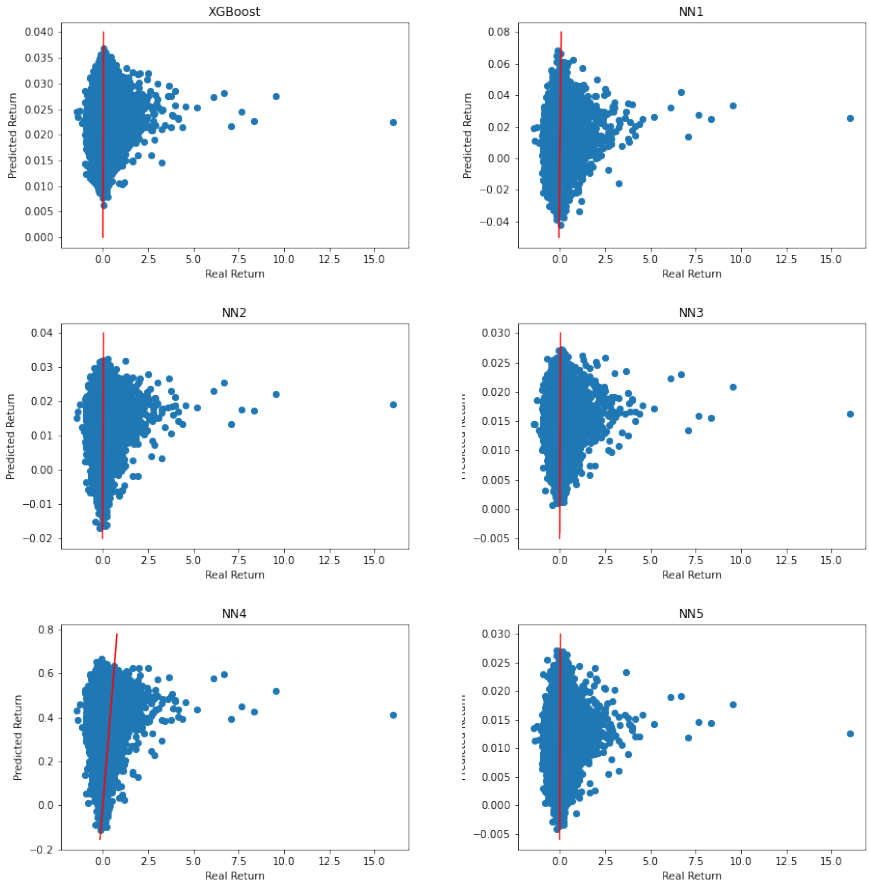
3. We construct an equal-weight portfolio based on those deciles in the last step.

4. We obtain ten portfolios from the stocks with the highest expected returns (decile 10) to the stocks with the lowest expected returns.

Similar to Chen et al. (1993), we split stocks into ten subgroups according to their predicted returns. More specifically, the decile one portfolio (also called the low portfolio) comprises stocks with the lowest 10% expected returns. In contrast, the decile ten portfolio (also called the so-called high portfolio) comprises stocks with the highest 10% expected returns. Stocks in each group are assigned equal weight to form a portfolio.

In Section 2.1, there are, on average, about 4,500 stocks each month. Thus, the high and low portfolios are composed of about 450 stocks. Consequently, listing stocks at each decile is formidable, and we only observe their portfolio values. We remark that these bottom-up equal-weight portfolios allow us to evaluate the prediction and robustness of

Figure 4: **Scatter plots of the predicted return versus the real return using XGB and neural networks**. This figure provides scatter plots of the predicted versus real returns. To see if predicted returns are close to real returns, we also overlay a 45-degree straight line in red color.



each method. And they also provide investors with a systematic procedure to manage their wealth, including multiple assets, and diversify financial market risk.

Let $N_{p,t}$ denote the number of stocks for the portfolio p at time t. Then, the weight of the i-th stock in this portfolio is $\frac{1}{N_{p,t}}$. Given the weight of stock i in portfolio p. The predicted return for portfolio p is hence

$$\hat{r}_{p,t+1} = \sum_{i=1}^{N_{p,t}} \frac{1}{N_{p,t}} \times \hat{r}_{i,t+1}$$

With $\hat{r}_{p,t+1}$, we calculate the portfolio-level R^2_{oos} as

$$R^2_{oos} = 1 - \frac{\sum_{t\in\tau}(r_{p,t+1} - \hat{r}_{p,t+1})^2}{\sum_{t\in\tau} r^2_{p,t+1}}.$$

Similar to the stock-level R^2_{oos}, a model yields higher portfolio-level R^2_{oos} is preferred because produces less prediction error for the portfolio.

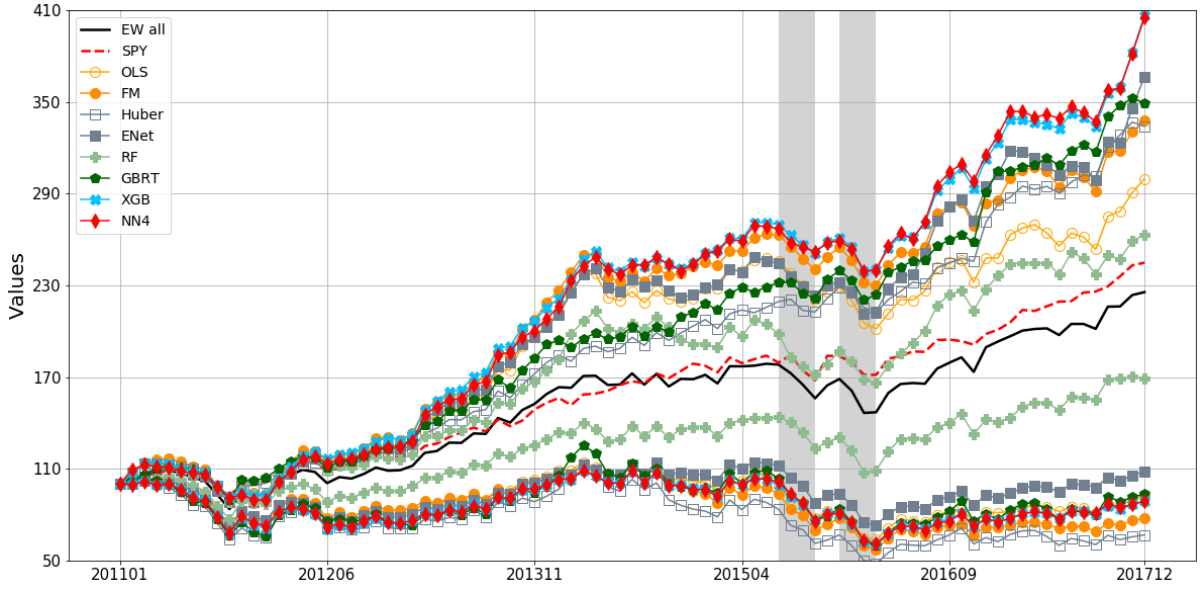
Table 6 compares the portfolio-level R^2_{oos} for the top-bottom equal-weight portfolios. Neural networks dominate other models because they produce positive portfolio-level R^2_{oos} for most top-bottom equal-weight portfolios. In addition, deeper neural networks produce higher portfolio-level R^2_{oos}. In fact, $NN4$ and $NN5$ produces the highest R^2_{oos} of 0.16 and 0.15, respectively. On the contrary, OLS, Huber, and GBRT produce negative portfolio-level R^2_{oos}, and their prediction deteriorates for higher portfolios because their portfolio-level R^2_{oos} decreases from low to high portfolios. However, it is also true that this R^2_{oos} remains low.

Table 6: **Portfolio-level R^2_{oos} (%) for the bottom-up equal-weight portfolios**. In this table, we report the R^2_{oos} for 10-decile portfolios using OLS, OLS with Huber loss function (Huber), elastic net (ENet), Fama-MacBeth regression (FM), random forest (RF), gradient boosting regression tree (GBRT), XGBoost (XGB), and neural networks with one to five layers (NN1-NN5). The ten equal-weight portfolios of each model are constructed based on ten deciles, which sort the stocks into ten deciles from high to low stock-level return forecasts. Low (L) is the equal-weight portfolio constructed by stocks with the lowest returns forecasts. High (H) is the equal-weight portfolio constructed by stocks with the highest returns forecasts.

	OLS	Huber	ENet	FM	RF	GBRT	XGB	NN1	NN2	NN3	NN4	NN5
Portfolio												
L	-0.39	-0.03	-0.04	0.00	-0.03	-0.05	-0.11	-0.02	-0.02	0.00	-0.03	-0.03
2	-0.75	-0.07	0.02	0.01	0.03	-0.09	-0.07	0.01	0.01	0.02	0.01	0.00
3	-0.96	-0.18	0.02	0.05	0.07	-0.14	-0.03	0.03	0.03	0.03	0.03	0.01
4	-0.98	-0.25	0.05	0.06	0.00	-0.19	-0.01	0.06	0.06	0.06	0.04	0.03
5	-1.13	-0.38	0.07	0.07	0.01	-0.18	0.00	0.06	0.07	0.09	0.06	0.07
6	-1.17	-0.48	0.04	0.06	-0.03	-0.18	0.01	0.06	0.09	0.09	0.06	0.09
7	-1.22	-0.62	0.07	0.06	0.00	-0.17	-0.01	0.07	0.07	0.08	0.05	0.09
8	-1.47	-0.70	0.06	0.09	-0.02	-0.20	-0.05	0.05	0.05	0.05	0.07	0.13
9	-1.39	-0.90	0.05	0.02	-0.02	-0.15	-0.07	0.04	0.04	0.07	0.10	0.12
H	-1.99	-1.28	0.12	-0.07	0.00	-0.15	0.11	0.09	0.10	0.01	0.16	0.15

Figure ?? provides a time series plot for the value of the high and low portfolios constructed from each model. The equal-weight portfolio and S&P 500 index are included as a benchmark. We only depict NN4 for simplicity for neural networks because it produces the highest portfolio-level R^2_{oos} for the high portfolio. The lines above and below the equal-weight portfolio refer to the high and low portfolios. A model outperforms others if the difference between the values of the high and low portfolios is more significant. One observes that XGBoost and NN4 dominate other models because they produce the most significant differences between high and low portfolios. Most models (besides RF) generate similar portfolio values for the low portfolio. In particular, NN4 and XGB produce the highest portfolio values for the high portfolio. B boosting (GBRT and XGBoost) among tree-based algorithms vastly outperforms bagging (random forest). The random forest produces the most negligible difference between the high and low portfolios.

Figure 5: Time series plots for the value of the high and low portfolios constructed from each model in the testing period



Note: We depict the portfolio value of the equal-weight portfolio with all of the stocks (EW all) in black solid line and S&P 500 index (SPX) in red dash line as the benchmark. The lines above and below benchmarks refer to the high and low portfolios, respectively. The gray area denotes periods with larger drawdown (2015/7-2015/10 and 2015/12-2016/03).

Lastly, we define maximum drawdown as

$$\text{MaxDD} = \max_{0 \leq t_1 \leq t_2 \leq T} (P_{t_1} - P_{t_2})$$

where P_t is the value from period 0 through t. Table ?? reports maximum drawdowns of the bottom-up equal-weight portfolios constructed from a different model. Huber produces the least maximum drawdowns of 11.1 and 10.1 for the low and high portfolios, respectively, but it produces the highest maximum drawdown of 40.3 for the long-short portfolio. As for NN4, it produces moderate maximum drawdowns for the bottom-up equal portfolios and the long-short portfolio.

Table 7: Maximum drawdowns for the bottom-up equal-weight portfolio constructed using different models

	OLS	FM	Huber	ENet	RF	GBRT	XGB	NN4
L	11.8	12.1	11.1	11.7	14.1	13.1	11.1	12.2
2	14.9	15.2	18.1	16.5	14.4	18.1	16.9	18.6
3	15.5	14.5	17.4	18.8	18.2	15.9	17.6	17.4
4	15.5	14.3	16.5	16.8	14.8	14.6	16.3	14.4
5	13.8	13.9	12.6	14.8	12.4	12.5	15.4	14
6	12.6	13.8	12.5	10.8	14.1	11.5	13.3	13.7
7	15.1	12.6	13.2	14.4	16	11.8	12.8	10.9
8	13.4	14.4	10.7	12.6	15.3	10.8	9.7	10.5
9	15	12.3	12.7	10.7	13.6	12.6	10.5	10.9
H	15.3	15.4	10.1	14.7	15.5	12.8	16.2	15
H-L	21.1	24.9	40.3	13	-12	31.2	22	19.1

5 Conclusion

We investigate asset pricing models and compare the performance of three models: regression, tree-based algorithms, and neural networks. To evaluate these models, we construct the bottom-up equal-weight portfolios based on the predicted returns using different models. In addition to existing literature, we include the Fama-MacBeth regression and tree-based algorithms for comparisons.

Our empirical analysis demonstrates the benefits of neural networks. They produce higher stock-level R^2_{oos}, higher portfolio-level R^2_{oos}, they can produce more significant differences between the high and low portfolio, and they produce moderate maximum drawdown. We also find marginal improvement for deeper neural networks, and a neural network with four layers is optimal. As deep neural network remains an active research area, one extension is to exploit serial correlations for the predictors shown in Figure 2 using recurrent neural networks, for example. Applying advanced neural networks is worthy of further research in the field of empirical asset pricing Fan et al. (2021).

References

Breiman, L. (2001). Random forests. *Machine learning 45*(1), 5–32.

Breiman, L., J. H. Friedman, R. A. Olshen, and C. J. Stone (2017). *Classification and Regression Trees*. New York: Routledge.

Chen, N.-F., R. Kan, and M. H. Miller (1993). Are the discounts on closed-end funds a sentiment index? *The Journal of Finance 48*(2), 795–800.

Chen, T. and C. Guestrin (2016). Xgboost: A scalable tree boosting system. *in Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, New York, USA*.

Christoffersen, P. F. and F. X. Diebold (2006). Financial asset returns, direction-of-change forecasting, and volatility dynamics. *Management Science 52*(8), 1273–1287.

Chudziak, A. (2022). Predictability of stock returns using neural networks: Elusive in the long term. *Expert Systems with Applications*, 119203.

Cochrane, J. H. (2011). Presidential address: Discount rate. *The Journal of Finance 66*(4), 1047–1108.

Dietterich, T. G. (2002). Ensemble learning. *The handbook of brain theory and neural networks 2*(1), 110–125.

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

Fama, E. F. and K. R. French (1996). Multifactor explanations of asset pricing anomalies. *The Journal of Finance 51*(1), 55–84.

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

Fama, E. F. and J. D. MacBeth (1973b). Risk, return, and equilibrium: Empricial tests. *The Journal of Political Economy 81*(3), 607–636.

Fan, J., C. Ma, and Y. Zhong (2021). A selective overview of deep learning. *Statistical Science 36*(2), 264–290.

Feng, G., N. G. Polson, and J. Xu (2019). Deep learning in asset pricing. *Technical report, University of Chicago*.

Friedman, J. H. (2002). Stochastic gradient boosting. *Computational Statistics and Data Analysis 38*(4), 367–378.

Goyal, A. and I. Welch (2008). A comprehensive look at the empirical performance of equity premium prediction. *The Review of Economic Studies 21*(4), 1455–1508.

Green, J., J. R. M. Hand, and X. F. Zhang (2017). The characteristics that provide independent information about average U.S. monthly stock returns. *The Review of Finance Studies 30*(12), 4389–4436.

Gu, S., B. Kelly, and D. Xiu (2020). Empirical asset pricing via machine learning. *The Review of Financial Studies 33*(5), 2223–2273.

Gu, S., B. Kelly, and D. Xiu (2021). Autoencoder asset pricing models. *Journal of Econometrics 222*(1), 429–450.

Harvey, C. R., Y. Liu, and H. Zhu (2016). ... and the cross-section of expected returns. *The Review of Financial Studies 29*(1), 5–68.

Hastie, T., R. Tibshirani, J. H. Friedman, and J. H. Friedman (2009). *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. New York: Springer.

Heaton, J., N. Polson, and J. Witte (2016). Deep learning for finance: Deep portfolios. *Applied Stochastic Models in Business and Industry 33*(1), 3–12.

Hill, R. C., G. C. Lim, and W. E. Griffiths (2018). *Introduction to Econometrics*. Wiley.

Hou, K., C. Xue, and L. Zhang (2015). Digesting anomalies: An investment approach. *The Review of Financial Studies 28*(3), 650–705.

Huber, P. J. (1973). Robust regression: asymptotics, conjectures and monte carlo. *The Annals of Statistics*, 799–821.

James, G., D. Witten, T. Hastie, and R. Tibshirani (2021). *An Introduction to Statistical Learning: With Applications in R* (2 ed.). New York, NY: Springer.

Jeremiah, G., H. J. RM, and Z. X. Frank (2013). The supraview of return predictive signals. *Review of Accounting Studies 18*(3), 692–730.

Jiang, W. (2021). Applications of deep learning in stock market prediction: recent progress. *Expert Systems with Applications 184*, 115537.

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

Leitch, G. and J. E. Tanner (1991). Economic forecast evaluation: profits versus the conventional error measures. *The American Economic Review*, 580–590.

Lewellen, J. (2015). Cross-section of expected stock return. *Critical Finance Review* (4), 1–44.

Lo, A. W., H. Mamaysky, and J. Wang (2000). Foundations of technical analysis: Computational algorithms, statistical inference, and empirical implementation. *The Journal of Finance 55*(4), 1705–1765.

Marquardt, D. W. and R. D. Snee (1975). Ridge regression in practice. *The American Statistician 29*(1), 3–20.

Mclean, R. D. and J. Pontiff (2016). Does academic research destroy stock return predictability? *The Journal of Finance 71*(1), 5–32.

Nyberg, H. (2011). Forecasting the direction of the us stock market with dynamic binary probit models. *International Journal of Forecasting 27*(2), 561–578.

Roll, R. and S. A. Ross (1980). An empirical investigation of the arbitrage pricing theory. *The Journal of Finance 35*(5), 1073–1103.

Sang, C. and M. Di Pierro (2019). Improving trading technical analysis with tensorflow long short-term memory (lstm) neural network. *The Journal of Finance and Data Science 5*(1), 1–11.

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

Tibshirani, R. (1996). Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society: Series B (Methodological) 58*(1), 267–288.

Ticknor, J. L. (2013). A bayesian regularized artificial neural network for stock market forecasting. *Expert Systems with Applications 40*(14), 5501–5506.

Tsang, K. H. and H. Y. Wong (2019). Deep-learning solution to portfolio selection with serially-dependent returns. *SIAM Journal on Financial Mathematics 11*(2), 593–619.

White, H. (1988). Economic prediction using neural networks: The case of ibm daily stock returns. In *ICNN*, Volume 2, pp. 451–458.

Yu, Z., L. Qin, Y. Chen, and M. D. Parmar (2020). Stock price forecasting based on lle-bp neural network model. *Physica A: Statistical Mechanics and Its Applications 553*, 124197.

Zou, H. and T. Hastie (2005). Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society: Series B (Statistical Methodology) 67*(2), 301–320.

Appendix A: Variable descriptions

Table 8: **Descriptions of 8 macroeconomic predictors and 102 firm characteristics**. The last column lists data source (D), frequency (F), and type (T). For data source: Goyal's website (G), Compustat (C), CRSP (CR), and I/B/E/S (I); For frequency: monthly (M), quarterly (Q), and annually (Q); For type: indicator (I), ratio (R), value (V)

Acronym	Definition	Author(s)	Year, Journal	D/F/T
Macroeconomic predictors				
dp	Dividend Price Ratio (d/p) is the difference between the log of dividends and the log of prices.	Goyal and Welch	2008, RFS	G/M/R
ep_GoyalWelch	Earnings Price Ratio (e/p) is the difference between the log of earnings and the log of prices.	Goyal and Welch	2008, RFS	G/M/R
bm_GoyalWelch	Book-to-Market Ratio (b/m) is the ratio of book value to market value for the Dow Jones Industrial Average.	Goyal and Welch	2008, RFS	G/M/R
ntis	Net Equity Expansion (ntis) is the ratio of 12-month moving sums of net issues by NYSE listed stocks divided by the total end-of-year market capitalization of NYSE stocks.	Goyal and Welch	2008, RFS	G/M/R
tbl	Treasury Bills (tbl) rate is from the 3-Month Treasury Bill	Goyal and Welch	2008, RFS	G/M/V
tms	The Term Spread (tms) is the dfference between the long term yield on government bonds and the T-bill.	Goyal and Welch	2008, RFS	G/M/V
dfy	The default Yield Spread (dfy) is the difference between the BAA and AAA-rate corporate bond yield.	Goyal and Welch	2008, RFS	G/M/V

(*continued*)

Table 8 – continued from previous page

Acronym	Definition	Author(s)	Year, Journal	D/F/T
svar	Stock Variance (svar): Stock Variance is computed as sum of squared daily returns on S&P 500.	Goyal and Welch	2008, RFS	G/M/V
Firm characteristics				
absacc	Absolute value of *acc*	Bandyopad, Huang, and Wirjanto	2010, WP	C/A/V
acc	Annual income before extraordinary items minus operating cash flows divided by average total assets	Sloan	1996, TAR	C/A/R
aeavol	Abnormal earnings announcement	Lerman, Livnat, and Mendenhall	2008, WP	C,CR/Q/V
age	Number of years since first Compustat coverage	Jiang, Lee, and Zhang	2005, RAS	C/A/V
agr	Annual percent change in total assets	Cooper, Gulen, and Schill	2008, JF	C/A/R
baspread	Monthly average of daily bid-ask spread divided by average of daily spread	Amihud and Mendelson	1989, JF	CR/M/R
beta	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	Fama and MacBeth	1973, JPE	CR/M/V
betasq	Market beta squared	Fama and MacBeth	1973, JPE	CR/M/V
bm	Book value of equity divided by end of fiscal year-end market capitalization	Rosenberg, Reid, and Lanstein	1985, JPM	C,CR/A/R
bm_ia	Industry adjusted book-tomarket ratio	Asness, Porter, and Stevens	2000, WP	C,CR/A/R
cash	Cash and Cash equivalents divided by average total assets	Palazzo	2012, JFE	C/A/R

(*continued*)

Table 8 – continued from previous page

Acronym	Definition	Author(s)	Year, Journal	D/F/T
cashdebt	Earnings before depreciation and extraordinary items divided by average total liabilities	Ou and Penman	1989, JAE	C/A/R
cashpr	Fiscal year-end market capitalization plus long-term debt minus total assets divided by cash and equivalents	Chandrashekar and Rao	2009, WP	C/A/R
cfp	Operating cash flows divided by fiscal-year-end market capitalization	Desai, Rajgopal, and Venkatachalam	2004, TAR	C/A/R
cfp_ia	Industry adjusted cfp	Asness, Porter, and Stevens	2000, WP	C/A/R
chatoia	2-digit SIC - fiscal-year mean-adjusted change in sales divided by average total assets	Soliman	2008, TAR	C/A/R
chcsho	Annual percent change in shares outstanding	Pontiff and Woodgate	2008, JF	C/A/R
chempia	Industry-adjusted change in unmber of empolyees	Asness, Porter, and Stevens	1994, WP	C/A/V
$chfeps$	Mean analyst forecast in month prior to fiscal period end date from I/B/E/S summary file minus same mean forecast for prior fiscal period using annual earnings forecasts	Hawkins, Daniel, and Chamberlin	1984, FAJ	I/A/V
chinv	Change in inventroy scaled by average total assets	Thomas and Zhang	2002, RAS	C/A/R
chmom	Cumulative returns from months $t-6$ to $t-1$ minus months $t-12$ to $t-7$	Titman, Wei, and Xie	2006, WP	CR/M/V
chnanalyst	Change in *nanalyst* from month $t-3$ to month t	Scherbina	2008, RFS	I/Q/R

(*continued*)

Table 8 – continued from previous page

Acronym	Definition	Author(s)	Year, Journal	D/F/T
chpmia	2-digit SIC - fiscal-year mean-adjusted change in income before extraordinary items divided by sales	Soliman	2008, TAR	C/A/R
chtx	Percent change in total taxes from quarter $t-4$ to t	Thomas and Zhang	2011, JAR	C/Q/R
cinvest	Change over one quarter in net PP&E divided by sales minus average of this variable for prior 3 quarters	Titman, Wei, and Xie	2004, JFQA	C/Q/R
convind	An indicator equal to 1 if company has convertible debt obligations	Valta	2016, JFQA	C/A/I
currat	Current assets divided by current liabilities	Ou and Penman	1989, JAE	C/A/R
depr	Depreciation divided by PP&E	Holthausen and Larcker	1992, JAE	C/A/R
disp	Standard deviation of analyst forecasts in month prior to fiscal period end date divided by the absolute value of the mean forecast	Diether, Malloy, and Scherbina	2002, JF	I/Q/R
divi	An indicator equal to 1 if company pays dividends but did not in prior year	Michaely, Thaler, and Womack	1995, JF	C/A/I
divo	An indicator equal to 1 if company does not pay dividend but did in prior year	Michaely, Thaler, and Womack	1995, JF	C/A/I
dolvol	Natural log of trading volume times price per share from month $t-2$	Chordia, Anshuman, and Subrahmanyam	2001, JFE	CR/M/V
dy	Total dividends divided by market capitalization at fiscal year-end	Litzenberger and Ramaswamy	1982, JF	C/A/R
ear	Sum of daily returns in three days around earnings announcement	Kishore et al.	2008, WP	C,CR/Q/V
egr	Annual percent change in book value of equity	Richardson et al.	2005, JAE	C/A/R

(*continued*)

Table 8 – continued from previous page

Acronym	Definition	Author(s)	Year, Journal	D/F/T
ep	Annual income before extraordinary items divided by end of fiscal year market cap	Basu	1977, JF	C/A/R
$fgr5yr$	Most recently available analyst forecasted 5-year growth	Bauman and Dowen	1988, FAJ	I/Q/V
gma	Revenues minus cost of goods sold divided by lagged total assets	Novy-Marx	2013, JFE	C/A/R
$grCAPX$	Percent change in capital expenditures from year $t-2$ to year t	Anderson and Garcia-Feijoo	2006, JF	C/A/R
grltnoa	Growth in long-term	Fairfield, Whisenant, and Yohn	2003, TAR	C/A/V
$herf$	2-digit SIC - fiscal-year sales concentration	Hou and Robinson	2006, JF	C/A/V
hire	Percent change in number of employees	Bazdresch, Belo, and Lin	2014, JPE	C/A/R
idiovol	Standard deviation of residuals of weekly returns on weekly equal weighted market returns for 3 years prior to month end	Ali, Hwang, and Trombley	2003, JFE	CR/M/V
ill	Average of daily absolute return divided by dollar volume	Amihud and Mendelson	2002, JFM	CR/M/R
indmom	Equal weighted average industry 12-month returns	Moskowitz and Grinblatt	1999, JF	CR/M/V
invest	Annual change in gross property, plant, and equipment + annual change in inventories all scaled by lagged total assets	Chen and Zhang	2010, JF	C/A/R
IPO	An indicator equal to 1 if first year available on CRSP monthly stock file	Loughran and Ritter	1995, JF	CR/M/I
lev	Total liabilities divided by fiscal year-end market capitalization	Bhandari	1988, JF	C/A/R

(*continued*)

Table 8 – continued from previous page

Acronym	Definition	Author(s)	Year, Journal	D/F/T
lgr	Annual percent change in total liabilities	Richardson et al.	2005, JAE	C/A/R
maxret	Maximum daily return from returns during calendar month $t-1$	Bali, Cakici, and Whitelaw	2011, JFE	CR/M/V
mom12m	11-month cumulative returns ending one month before month end	Jegadeesh	1990, JFE	CR/M/V
mom1m	1-month cumulative return	Jegadeesh and Titman	1993, JF	CR/M/V
mom36m	Cumulative returns from months $t-36$ to $t-13$	Jegadeesh and Titman	1993, JF	CR/M/V
mom6m	5-month cumulative returns ending one month before month end	Jegadeesh and Titman	1993, JF	CR/M/V
ms	Sum of 8 indicator variables for fundamental performance	Mohanram	2005, RAS	C/Q/I
mve	Natural log of market capitalization at end of month $t-1$	Banz	1981, JFE	CR/M/V
mve_ia	2-digit SIC industry-adjusted fiscal year-end market capitalization	Asness, Porter, and Stevens	2000, WP	C/A/V
nanalyst	Number of analysts covering stock	Elgers, Lo, and Pfeiffer	2001, TAR	I/Q/V
nincr	Number of consecutive quarters (up to 8 quarters) with an increase in earning over same quarter in the prior year	Barth, Elliott, and Finn	1999, JAR	C/Q/V
operprof	Revenue minus cost of goods sold - SG&A expense - interest expense divided by lagged common shareholders' equity	Fama and French	2015, JFE	C/A/R
orgcap	Capitalized SG&A expenses	Eisfeldt and Papanikolaou	2013, JF	C/A/V
pchcapx_ia	2-digit SIC - fiscal-year mean-adjusted percent change in capital expenditures	Abarbanell and Bushee	1998, TAR	C/A/R

(*continued*)

Table 8 – continued from previous page

Acronym	Definition	Author(s)	Year, Journal	D/F/T
pchcurrat	Percent change in *currat*	Ou and penman	1989, JAE	C/A/R
pchdepr	Percent change in *depr*	Holtahausen and Larcker	1992, JAE	C/A/R
pchgm_pchsale	Percent change in gross margin munus percent change in sales	Abarbanell and Bushee	1998, TAR	C/A/R
pchquick	Percent change in *quick*	Ou and Penman	1989, JAE	C/A/R
pchsale_pchinvt	Annual percent change in sales minus annual percent change in inventory	Abarbanell and Bushee	1998, TAR	C/A/R
pchsale_pchrect	Annual percent change in sales minus annual percent change in receivables	Abarbanell and Bushee	1998, TAR	C/A/R
pchsale_pchxsga	Annual percent change in sales minus annual percent change in SG&A	Abarbanell and Bushee	1998, TAR	C/A/R
pchsaleinv	Percent change in *saleinv*	Ou and Penman	1989, JAE	C/A/R
pctacc	Same as *acc* except that numerator is divided by the absolute value of annual income before extraordinary items	Hafzalla, Lundholm, and Van Winkle	2011, TAR	C/A/R
pricedelay	The proportion of variation in weekly returns for 36 months ending in month t explained by 4 lags of weekly market returns incremental to contemporaneous market return	Hou and Moskowitz	2005, RFS	CR/M/V
ps	Sum of 9 indicator variables to form fundamental health score	Piotroski	2000, JAR	C/A/I
quick	(current assets - inventroy) / current liabilities	Ou and Penman	1989, JAE	C/A/R

(*continued*)

Table 8 – continued from previous page

Acronym	Definition	Author(s)	Year, Journal	D/F/T
rd	An indicator variable equal to 1 if R&D expense as a percentage of total assets has an increase greater than 5%	Eberhart, Maxwell, and Sideeique	2004, JF	C/A/I
rd_mve	R&D expense divided by end-of-fiscal-year market capitalization	Guo, Lev, and Shi	2006, JBFA	C/A/R
rd_sale	R&D expense divided by sales	Guo, Lev, and Shi	2006, JBFA	C/A/R
realestate	Buildings and capitalized leases divided by gross PP&E	Tuzel	2010, RFS	C/A/R
retvol	Standard deviation of daily returns from month $t-1$	Ang et al.	2006, JF	CR/M/V
roaq	Income before extraordinary items divided by one quarter lagged total assets	Balakrishnan, Bartov, and Faurel	2010, JAE	C/Q/R
roavol	Standard deviation for 16 quarters of income before extraordinary items divided by average total assets	Francis et al.	2004, TAR	C/Q/R
roeq	Earnings before extraordinary items divided by lagged common shareholders' equity	Hou, Xue, and Zhang	2015, RFS	C/Q/R
roic	Annual earnings before interest and taxes minus nonoperating income divided by non-cash enterprise value	Brown and Rowe	2007, WP	C/A/R
rsup	Sales from quarter t minus sales from quarter $t-4$ divided by fiscal-quarter-end market capitalization	Kama	2009, JBFA	C/Q/R
salecash	Annual sales divided by cash and cash equivalents	Ou and Penman	1989, JAE	C/A/R
saleinv	Annual sales divided by total inventory	Ou and Penman	1989, JAE	C/A/R
salerec	Annual sales divided by accounts receivable	Ou and Penman	1989, JAE	C/A/R

(*continued*)

Table 8 – continued from previous page

Acronym	Definition	Author(s)	Year, Journal	D/F/T
secured	Total liability scaled secured debt	Valta	2016, JFQA	C/A/V
securedind	An indicator equal to 1 if company has secured debt obligations	Valta	2016, JFQA	C/A/I
sfe	Scaled earnings forecast	Elgers, Lo, and Pfeiffer	2001, TAR	I/Q/V
sgr	Annual percent change in sales	Lakonishok, Shleifer, and Vishny	1994, JF	C/A/R
sin	An indicator equal to 1 if company's primary industry classification is in smoke or tobacco, beer, or alcohol, or gaming	Hong and Kacperczyk	2009, JFE	C/A/I
sp	Annual revenue divided by fiscal year-end market capitalization	Barbee, Mukherji, and Raines	1996, FAJ	C/A/R
std_dolvol	Monthly standard deviation of daily dollar trading volume	Chordia, Subrahmanyam, and Anshuman	2001, JFE	CR/M/V
std_turn	Monthly standard deviation of daily share turnover	Chordia, Subrahmanyam, and Anshuman	2001, JFE	CR/M/V
stdacc	Standard deviation for 16 quarters of accruals scaled by sales	Bandyopad hyay, Huang, and Wirjanto	2010, WP	C/Q/V
stdcf	Standard deviation for 16quarters of cash flows divided by sales	Hung	2009, JEF	C/Q/V
sue	Unexpeceted quarterly earning divided by fiscal-quarter-end market cap	Rendelman, Jones, and Latane	1982, JFE	I/Q/R
tang	Cash holdings + 0.715 × receivables + 0.547 × inventroy + 0.535 × (PPE/total assets)	Almeida and Campello	2007, RFS	C/A/V
tb	Tax income to book income	Lev and Nissim	2004, TAR	C/A/R

(*continued*)

Table 8 – continued from previous page

Acronym	Definition	Author(s)	Year, Journal	D/F/T
turn	Average monthly trading volume for most recent 3 months scaled by number of shares outstanding in current month	Datar, Naik, and Radcliffe	1998, JFM	CR/M/V
zerotrade	Turnover weighted number of zero trading days for most recent 1 month	Liu	2006, JFE	CR/M/V

Appendix B: Descriptive statistics

Table 9: **Descriptive statistics of 8 macroeconomic predictors and 102 firm characteristics** This table reports the number of observations (N), missing ratio (%), average (Avg), stadard deviation (Std), minimum (Min), median (Med), and maximum (Max) for each variable.

Acronym	N	Miss.(%)	Avg	Std	Min	Med	Max
Macroeconomic predictors							
dp	456	-	-3.72	0.41	-4.52	-3.85	-2.75
ep_GoyalWelch	456	-	-2.94	0.45	-4.84	-2.94	-1.9
bm_GoyalWelch	456	-	0.4	0.23	0.12	0.32	1.21
ntis	456	-	0	0.02	-0.06	0.01	0.05
tbl	456	-	0.04	0.04	0	0.05	0.16
tms	456	-	0.02	0.01	-0.04	0.02	0.05
dfy	456	-	0.01	0	0.01	0.01	0.03
svar	456	-	0	0.01	0	0	0.07
Firm characteristics							
absacc	1,775,701	13.34	0.09	0.1	0	0.06	1.14
acc	1,775,701	13.34	-0.03	0.13	-1.12	-0.03	0.52
aeavol	1,794,393	12.42	0.87	2.1	-1	0.29	22.52
age	2,048,944	0	11.18	9	1	9	43
agr	1,914,858	6.54	0.16	0.45	-0.71	0.07	6.05
baspread	2,012,384	1.78	0.06	0.07	0	0.04	0.93
beta	2,028,819	0.98	1.09	0.67	-0.82	1.01	3.97
betasq	2,028,819	0.98	1.64	1.9	0	1.03	15.77
bm	2,048,944	0	0.7	0.64	-2.57	0.56	7.64
bm_ia	2,048,944	0	27.32	779.89	-1307.01	0.04	17158.65
cash	1,818,998	11.22	0.16	0.2	-0.14	0.07	0.98
cashdebt	1,977,499	3.49	-0.04	1.34	-99.68	0.11	2.18
cashpr	2,028,268	1.01	-1.04	57.15	-537.25	-0.29	594.9
cfp	1,891,095	7.7	0.05	0.6	-4.43	0.05	24.35
cfp_ia	1,891,095	7.7	14.54	335.7	-296.6	0	7031.6
chatoia	1,764,963	13.86	0	0.24	-1.4	0	1.19
chcsho	1,914,092	6.58	0.11	0.32	-0.89	0.01	2.65
chempia	1,909,832	6.79	-0.12	0.68	-24.52	-0.07	3.8
chfeps	1,099,111	46.36	0	0.36	-13.64	0	10.21
chinv	1,865,800	8.94	0.01	0.06	-0.29	0	0.36
chmom	1,901,565	7.19	0	0.58	-9.09	-0.01	8.65
chnanalyst	1,575,363	23.11	-0.02	1.58	-45	0	43
chpmia	1,884,184	8.04	0.16	12.58	-548.23	-0.01	116.44
chtx	1,794,192	12.43	0	0.01	-0.12	0	0.14
cinvest	1,788,933	12.69	-0.02	0.97	-26.2	0	27.64
convind	2,048,944	0	0.12	0.33	0	0	1
currat	1,981,187	3.31	3.31	4.97	0.14	1.94	58.24

(*continued*)

Table 9 – continued from previous page

Acronym	N	Miss.(%)	Avg	Std	Min	Med	Max
depr	1,960,419	4.32	0.28	0.41	-0.98	0.17	5.83
disp	912,932	55.44	0.15	0.41	0	0.04	9.97
divi	1,914,891	6.54	0.03	0.17	0	0	1
orgcap	1,507,419	26.43	0.01	0.01	0	0.01	0.09
pchcapx_ia	1,860,751	9.18	7.52	75.74	-237.44	-0.37	1629.63
pchcurrat	1,844,645	9.97	0.07	0.57	-0.89	-0.01	6.7
pchdepr	1,823,344	11.01	0.12	0.55	-0.86	0.03	7.82
pchgm_pchsale	1,888,403	7.84	-0.08	0.95	-12.29	0	4.77
pchquick	1,832,752	10.55	0.09	0.68	-0.92	-0.01	8.92
pchsale_pchinvt	1,508,643	26.37	-0.07	0.89	-11.57	0.02	3.96
pchsale_pchrect	1,835,471	10.42	-0.05	0.63	-7.69	0	3.38
pchsale_pchxsga	1,592,423	22.28	0.02	0.35	-1.45	0	4.68
pchsaleinv	1,489,019	27.33	0.17	1.13	-121.04	0.01	33.24
pctacc	1,775,689	13.34	-0.94	5.95	-64.75	-0.41	68.85
pricedelay	2,028,790	0.98	0.16	1.05	-15.61	0.07	13.47
ps	1,914,891	6.54	4.27	1.73	0	4	9
quick	1,970,271	3.84	2.61	4.41	0.09	1.31	53.93
rd	1,914,891	6.54	0.14	0.35	0	0	1
rd_mve	993,889	51.49	0.06	0.11	-0.03	0.03	2.25
rd_sale	977,364	52.3	0.71	5.69	-218.74	0.03	283.48
realestate	866,165	57.73	0.26	0.19	0	0.23	0.94
retvol	2,012,373	1.78	0.03	0.03	0	0.03	0.27
roaq	1,827,075	10.83	0	0.06	-0.51	0.01	0.14
roavol	1,551,046	24.3	0.03	0.05	0	0.01	0.85
divo	1,914,891	6.54	0.03	0.18	0	0	1
dolvol	1,979,178	3.4	11.33	3.04	-3.06	11.27	19.01
dy	2,043,740	0.25	0.02	0.03	-6.12	0	0.36
ear	1,806,061	11.85	0	0.08	-0.47	0	0.52
egr	1,914,703	6.55	0.14	0.72	-3.74	0.07	9.04
ep	2,048,944	0	-0.04	0.37	-8.22	0.04	0.44
fgr5yr	817,211	60.12	16.4	9.93	-43.5	14.6	85
gma	1,910,367	6.76	0.36	0.35	-0.96	0.31	1.77
grcapx	1,719,178	16.09	0.95	3.62	-12.85	0.12	60.64
grltnoa	1,441,881	29.63	0.09	0.16	-0.61	0.06	1.14
herf	2,048,934	0	0.08	0.09	0.01	0.05	1
hire	1,909,832	6.79	0.09	0.36	-0.77	0.02	4.22
idiovol	2,028,819	0.98	0.07	0.04	0	0.06	0.27
ill	1,953,829	4.64	0	0	0	0	0
indmom	2,048,787	0.01	0.14	0.3	-0.68	0.11	3.48
invest	1,850,781	9.67	0.08	0.18	-0.55	0.04	1.5
IPO	2,048,944	0	0.07	0.26	0	0	1
lev	2,043,342	0.27	2.24	4.81	0	0.65	75.63
lgr	1,908,518	6.85	0.26	0.82	-0.76	0.07	10.62
maxret	2,012,428	1.78	0.08	0.08	0	0.05	0.94

(*continued*)

Table 9 – continued from previous page

Acronym	N	Miss.(%)	Avg	Std	Min	Med	Max
mom12m	1,901,565	7.19	0.13	0.61	-0.96	0.05	12.19
mom1m	2,048,944	0	0.01	0.16	-0.7	0	2.2
mom36m	1,600,305	21.9	0.33	0.96	-0.98	0.16	16.07
mom6m	1,989,131	2.92	0.05	0.38	-0.92	0.02	8.13
ms	1,830,211	10.68	3.78	1.68	0	4	8
mve	2,048,944	0	11.88	2.24	2.36	11.76	18.86
mve_ia	2,048,944	0	-202.19	6084.34	-15100.04	-472.12	109419.34
nanalyst	1,600,797	21.87	5.06	6.81	0	2	56
nincr	1,830,211	10.68	1	1.37	0	1	8
operprof	1,910,212	6.77	0.81	1.16	-8.58	0.62	10.51
roeq	1,826,834	10.84	0	0.15	-2.15	0.02	1.43
roic	1,963,488	4.17	-0.12	1.01	-20.32	0.06	0.91
rsup	1,816,448	11.35	0.01	0.2	-3.84	0.01	1.51
salecash	2,032,645	0.8	55.86	172.67	-1230.91	8.85	2505.12
saleinv	1,608,081	21.52	28.38	70.53	-35.44	8.11	979
salerec	1,976,064	3.56	11.52	59.23	-21796	5.82	241.87
secured	1,205,427	41.17	0.58	0.53	0	0.58	4.93
securedind	2,048,944	0	0.47	0.5	0	0	1
sfe	1,083,621	47.11	-0.22	2.66	-100.79	0.04	2.47
sgr	1,888,596	7.83	0.2	0.62	-0.87	0.09	8.73
sin	2,048,944	0	0.01	0.09	0	0	1
sp	2,043,482	0.27	2.04	3.26	-4.13	0.98	40.52
std_dolvol	1,950,573	4.8	0.88	0.42	0	0.82	2.86
std_turn	1,955,627	4.55	4.19	6.6	0	2.14	137.96
stdacc	1,290,891	37	4.44	31.54	0	0.13	714.21
stdcf	1,290,891	37	10.18	71.88	0	0.15	1665.7
sue	1,818,201	11.26	0	0.12	-5.3	0	1.76
tang	1,965,850	4.06	0.54	0.16	0	0.54	0.98
tb	1,804,442	11.93	-0.11	1.72	-27.7	-0.07	11.35
turn	1,979,639	3.38	1.1	1.44	0	0.6	66.48
zerotrade	1,953,855	4.64	1.4	3.4	0	0	19.95