Evaluation of Machine Learning in Empirical Asset Pricing

Ze Yu Zhong

Tuesday 3rd September, 2019

Contents

1. Introduction

1.1. Topic

This thesis aims to evaluate the application of machine learning algorithms in empirical asset pricing. While there has been a significant recent interest in applying machine learning to the problem of predicting asset returns, there is little literature that focuses on how well these algorithms are at capturing true underlying variables in determining stock returns. 12 different simulated datasets ranging from linear to highly non-linear data generating processes incorporating observed phenomena of cross sectional correlation, persistence, and stochastic volatility, in addition to real world data will be used to assess the performance of linear models, elastic net models, random forests and neural networks. Model performance will be assessed according to their out of sample predictive R^2 and errors, in addition to whether or not they were able to identify the correct variables in the data generating process according to a variable importance metric.

1.2. Background Literature and Motivations

This paper is motivated by evaluating the performance of machine learning algorithms in empirical asset pricing, focusing on how well they deal with the many unique problems in financial returns data. Here, we define "performance" to refer to two forms of metrics conventional in the literature: R^2 (and more specifically, out of sample R^2), and forecast errors (see 2.10 for more details).

We first begin by defining "factors" with the more contemporary definition as suggested by Harvey et al. (2016): a collection of regressors to be used in pricing returns that can be used to proxy for unknown underlying risk factors due to their correlation with cross sectional returns. Most notably, their definition rejects the more strict view that risk factors should be variables that have unpredictable variation through time, and that they should be able explain cross sectional return patterns. Harvey

et al. (2016) further groups factors into the two broad categories of "common" and individual firm "characteristics." "Common" factors under this definition can be viewed as proxies for sources of risk constant across all firms, such as macroeconomic variables. They note that individual firm characteristics are unlikely to satisfy the more strict definition because they are often pre-known and display limited time series variation.

Because of this less strict definition, factors introduced and used in the literature often exhibit properties which makes them unsuitable for inclusion in models, such as high persistence, high levels of non-stationarity and cross sectional correlation.

Goetzmann and Jorion (1993) and Ang and Bekaert (2006) note the persistence present in dividend ratio factors. This means that movements in dividend ratios are dominated by movements in price and therefore dividend ratios are correlated with lagged dependent variables on the right hand side of the regression equation. This violates the assumptions of independent regressors required for traditional regression models (ordinary least squares) to be unbiased, resulting in t statistics which are biased upwards and increase with time horizon due to autocorrelated errors. Importantly, Goetzmann and Jorion (1993) show that corrections to t statistics using the Generalized Method of Moments and Newey-West standard errors also appear to be biased upwards, making them unreliable.

Goyal and Welch (2003) provide a more comprehensive study on the performance of lagged dividend price ratios, with specific focus on out of sample predictive performance both in terms of R^2 and forecast errors. They conclude that while models incorporating dividend related factors were able to achieve higher in sample performance prior to 1990 than the historical mean, they could not have outperformed the historical mean *out of sample*. Goyal and Welch (2003) attribute this to the increasing persistence and non-stationarity of dividend ratios, noting that they have become like random walks as of 2001. This mirrors the sentiment of (Lettau and Ludvigson (2001), Schwert (2003) and others) who conclude that models incorporating dividend ratios seemed to break down in the 2000s due to a changing economic environment despite having performed well in the 1990s.

Despite the controversy, the prevailing tone within the literature was that various factors such as dividend ratios, earnings price ratio, interest and inflation and other financial indicators were able to predict excess returns, with Lettau and Ludvigson (2001) remarking that this was now "widely accepted." However, Welch and Goyal (2008) extend upon the work of Goyal and Welch (2003) by including a more comprehensive set of variables and time horizons. They conclude that not a single variable had any statistical forecasting power. Crucially, they demonstrate the non-robustness of models incorporating these factors by showing that the significance values of some factors change with the choice of sample periods.

Despite this, the literature has continued to produce more factors: quantitative trading firms were using 81 factor models as the norm by 2014 (Hsu and Kalesnik, 2014), and Harvey and Liu (2019) currently document well over 600 different factors suggested in the literature.

The dramatic increase in the number of factors in of itself poses challenges to traditional statistical techniques. Harvey et al. (2016) detail the false discovery problem when the number of potential factors is extremely high. The significance of a factor in a traditional regression setting is determined by a single hypothesis test, which by construction carries a level of significance α controlling the type I error rate: the probability of finding a factor that is significant but is not. When the number of potential factors is very high, it is very likely that a factor will be concluded as significant by pure chance. Harvey et al. (2016) produce a multiple testing framework to mitigate this, and conclude that many of the historically discovered would have been deemed significant by chance.

Furthermore, Feng et al. (2019) note that the number of potential factors discovered in the literature

has increased to the same scale as, if not greater, than the number of stocks considered in a typical portfolio, or the time horizon, producing highly inefficient covariances in a standard cross sectional regression. Moreover, when the number of factors exceeds the sample size, traditional cross sectional regressions become infeasible and do not produce solutions altogether.

It does not help that many factors are cross sectionally correlated, meaning that factors which are discovered to be significant may simply be so because they are correlated with a true, underlying factor and do not provide independent information themselves, a concern which Cochrane (2011) calls the multidimensional challenge. Freyberger et al. (2017) notes that this is especially challenging for traditional regression models, which make strong functional form assumptions and are sensitive to outliers.

More recently, machine learning algorithms have emerged within the literature and appear to be well suited to the task of predicting asset returns. The definition of machine learning can be vague and is often context specific; Hastie et al. (2009) in *An Introduction to Statistical Learning* describes statistical (machine) learning as a vast set of tools for understanding data, and *supervised* learning specifically as the process of building a statistical model for the prediction or estimation of an output based on input(s). In the context asset pricing, we use the term to refer to a diverse collection of:

1. high-dimensional models for statistical prediction,
2. the "regularization" methods for model selection and mitigation of overfitting input data,
3. and the efficient systematic methods for searching potential model specifications.

The high dimensional and hence flexible nature of machine learning brings more hope to approximating unknown and likely complex data generating processes that underlie excess returns. The flexibility however, comes at a cost of potentially overfitting in sample data (referred to as training data in the machine learning literature), generalizing poorly and producing poor forecasts. The regularization aspect of machine learning explicitly guards against problem and emphasizes out of sample performance. Finally, machine learning offers tools which are designed to produce an an optimal model specification from all possible models with manageable computational cost, all in a systematically consistent way.

Machine learning has seen some applications in the empirical asset pricing literature. Kozak et al. (2017), Rapach and Zhou (2013) and Freyberger et al. (2017) all apply shrinkage and selection methods from machine learning in factor selection.

Most importantly, portfolios constructed using machine learning have been demonstrated to outperform traditional models in predicting stock returns (Gu et al. (2018), Hsu and Kalesnik (2014) and Feng et al. (2018)) in terms of out of sample predictive R^2 and Sharpe Ratios. Gu et al. (2018) attribute this to machine learning's ability to evaluate and consider non-linear complexities among factors that cannot be feasibly achieved using traditional techniques.

However, there is little work done on how machine learning actually recognises and deals with the challenges of returns prediction documented in the literature. Prior work has been done by Gu et al. (2018), however; only basic simulation designs which were not representative of real financial data were considered. In particular, their design did not consider cross sectionally correlated factors.

Furthermore, Feng et al. (2018) in particular use cross validation as part of their model building procedure, destroying the temporal aspect of returns data, in addition to only using a handful of factors. Gu et al. (2018) produce models using a training sample which ends in the 1970s to ultimately produce forecasts for the most recent 30 years. Given the non-robustness of financial data affecting even traditional regressions which are considered to be more inflexible, more research should be done into the robustness of more flexible machine learning methods with regards to sample selection.

For the aspect of factor selection specifically, Gu et al. (2018) concludes that all of the machine methods agree on the same subset of important factors. However, while their factor importance metrics for regression-tree based methods and neural networks (the most complex methods considered) are mostly consistent, they have differences in terms of the relative importance of each factor, in addition to completely different conclusions for the dividend yield factor.

This paper will be the first in focusing on how machine learning algorithms perform in environments with problems exhibited by financial returns data through extending the simulation designs of Gu et al. (2018). In addition, these algorithms will once again be evaluated on real world data, but with only more recent and representative data included in order to test their short term robustness in predicting stock returns. These two aspects of the study together are able to offer a better glimpse as to how "black box" machine learning algorithms deal with the challenges present in asset pricing, if at all.

2. Methodology

2.1. Overall Model Design

Each model will be presented and explained so that a reader without any machine learning background can understand the basic idea behind each model. Details such as the computational and specific algorithm used for each model are included in the Appendix (8.2). This is because there are many variations of algorithms available, and more importantly, specific understanding of how the algorithm works is not necessary.

All asset excess monthly returns denoted as $r_{i,t+1}$ are modelled as an additive prediction error model conditional on the true and unobservable information set available to market participants up to and including time t, $\mathcal{F}_t$):

$$r_{i,t+1} = E(r_{i,t+1}|\mathcal{F}_t) + \epsilon_{i,t+1} \tag{1}$$

where

$$E(r_{i,t+1}|\mathcal{F}_t) = g^*(z_{i,t}) \tag{2}$$

with stocks indexed as $i = 1, \ldots, N$ and time periods by $t = 1, \ldots, T$. $g^*(z_{i,t})$ represents the model approximation using the P dimensional predictor set $z_{i,t}$. We allow $g^*(z_{i,t})$ to be a flexible function of the predictor set $z_{i,t}$, and most notably, not depend on i or t directly. This means that we do not re-estimate a model for each time period, or independently estimate a model for each stock. Note that $g^*(z_{i,t})$ only contains information in time t for individual stock i, meaning that while the model and its parameters will be estimated using $\mathcal{F}_t$ for stock i, predictions for $r_{i,t+1}$ will only use information at time t as an input, analogous to using variables lagged by one period.

2.2. Sample Splitting

Imperative to any machine learning technique is the establishment of how the dataset is to be split into training, validation and test sets. The training set is used to initially build the model and provide initial estimates of parameters, whereas the validation set is used to tune model parameters to optimise out of sample performance, thus preventing overfitting. The validation set acts as a simulation of out of sample testing, whereas the test set is used only for evaluation, and is thus truly out of sample.

There are three main approaches to splitting temporal data (such as financial data).

The first is to decide arbitrarily on a single training, validation and test set. This method is straightforward and the least computationally intensive, but is limited and inflexible in evaluating how models perform when more recent data is provided for training.

The second method is a "rolling window" method, where a fixed size or "window" for the training and validation set is first chosen. This window then incrementally move forwards in time to include more recent data, with a set of forecasts for the test sets made for all possible windows.

The third is a "recursive" method, which is the same as the rolling window method, but different in that the training set always contains previous data, with only the validation set staying fixed in size and "rolling" forwards. Hence, it is also referred to as a "growing window."

Both the rolling window and recursive schemes are very computationally intensive. Therefore, a hybrid of the rolling and recursive schemes was considered: the training set is increased by one year with each refit, the validation set remains one year in length but moves forward by one year, and forecasts are made using that model for the subsequent year. Cross validation was not done to maintain the temporal ordering of the data.

2.3. Loss Function

The choice of the loss function used in models is imperative to machine learning. The loss functions considered are Mean Absolute Error (MAE), Mean Squared Error and Root Mean Squared Error (MSE, and RMSE).

2.3.1. Mean Absolute Error

The mean absolute error (MAE) is simply the average magnitude of errors. Because of this, it places equal weighting to all magnitudes of errors and is more robust to outliers.

$$\text{MAE} = \frac{1}{n}\sum_{j=i}^{n}|y_j - \hat{y_j}| \tag{3}$$

It should be noted that minimizing the MAE criterion is equivalent to minimizing 0.5 quantile loss.

2.3.2. Mean Squared Error and Root Mean Squared Error

The mean squared error (MSE) and root mean squared error (RMSE) are quadratic scoring methods. This means that they place higher weight on large errors. Models that minimize this metric are therefore more sensitive to outliers.

$$\text{MSE} = \frac{1}{n}\sum_{j=i}^{n}(y_j - \hat{y_j})^2 \tag{4}$$

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{j=i}^{n}(y_j - \hat{y_j})^2} \tag{5}$$

2.4. Linear Model

The least complex model considered is the simple linear regression model, also known that ordinary least squares in the case of the using mean squared error as the loss function. Linear models struggle with

high-dimensionality. Nevertheless, despite being expected to perform poorly it was implemented as a "control."

The simple linear model assumes that the underlying conditional expectation $g^*(z_{i,t})$ can be modelled as a linear function of the predictors and the parameter vector θ:

$$g(z_{i,t};\theta) = z'_{i,t}\theta \tag{6}$$

This model can capture non-linearities only if the predictor set $z^*_{i,t}$ contains specified non-linear transformations or interaction terms.

Note that computing this model with respect to minimizing the mean squared error yields the pooled ordinary least squares estimator (POLS), while minimizing the mean absolute error corresponds to quantile regression predicting the 0.5th quantile.

2.5. Penalized Linear Model

Penalized linear models have the same underlying statistical model as simple linear models, but differ in their addition of a new penalty term in the loss function:

$$\mathcal{L}(\theta;.) = \underbrace{\mathcal{L}(\theta)}_{\text{Loss Function}} + \underbrace{\phi(\theta;.)}_{\text{Penalty Term}} \tag{7}$$

Several choices exist for the choice of penalty function $\phi(\theta;.)$. We restrict our scope to the popular "elastic net" penalty (Zou and Hastie, 2005):

$$\phi(\theta;\lambda,\rho) = \lambda(1-\rho)\sum_{j=1}^{P}|\theta_j| + \frac{1}{2}\lambda\rho\sum_{j=1}^{P}\theta_j^2 \tag{8}$$

The elastic net has two hyperparameters: λ, which controls the overall magnitude of the loss, and ρ, which controls the shape of the penalization.

The $\rho = 1$ case corresponds to ridge regression proposed by Hoerl and Kennard (1970), which uses l_2 that shrinks all coefficients closer to 0, but not to 0. Ridge regression is therefore a shrinkage method which prevents coefficients from becoming too large and overpowering. For $0 < \rho < 1$, the elastic net aims to produce parsimonious models through both shrinkage and selection.

The $\rho = 0$ case corresponds to the popular LASSO and uses absolute (l_1) parameter penalization proposed by Tibshirani (1996), which geometrically allows the coefficients to be shrunk to 0. This allows it to impose sparsity, and can be thought of as a variable selection tool.

By combining the properties of LASSO and ridge regression, the elastic net aims to produce parsimonious models through both coefficient shrinkage and selection.

The hyperparameters λ and ρ are both tuned using the validation sample (see 8.2).

2.6. *Classification and Regression Trees*

Classification and regression trees are fully non-parametric models that can capture complex multi-way interactions. A tree "grows" in a series of iterations. With each iteration, a split ("branch") is made along one predictor such that it is the best split available at that stage with respect to minimizing the loss function. These steps are continued until each observation is its own node, or more commonly until the stopping criterion is met. The eventual model slices the predictor space into rectangular partitions, and predicts the unknown function $g^*(z_{i,t})$ with the average value of the outcome variable in each partition.

The prediction of a tree, $\mathcal{T}$, with K "leaves" (terminal nodes), and depth L is

$$g(z_{i,t};\theta,K,L) = \sum_{k=1}^{K} \theta_k \mathbf{1}_{z_{i,t}\in C_k(L)} \tag{9}$$

where $C_k(L)$ is one of the K partitions in the model.

For this study, only recursive binary trees (the most common and easy to implement) are considered. Though trees were originally proposed and fit with respect to minimizing mean squared error, they can be grown with respect to a variety of loss functions, including mean absolute error, mean squared error:

$$H(\theta,C) = \frac{1}{|C|}\sum_{z_{i,t}\in C} L(r_{i,t+1}-\theta) \tag{10}$$

where $|C|$ denotes the number of observations in set C (partition). Given C, it is clear that the optimal choice for minimising the loss function when it is mean squared error is simply $\theta = \frac{1}{|C|}\sum_{z_{io,t}\in C} r_{i,t+1}$ i.e. the average of the partition, and the median of the partition when the loss function is mean absolute error.

Trees, grown to a deep enough level, are highly unbiased and flexible. The trade-off is their high variance and instability. Thus, an ensemble method called "Random Forest" was proposed by Breiman (2001) to regularize trees by combining many different trees into a single prediction.

2.7. *Random Forests*

Random Forests are an extension of trees that attempt to address some of their problems. A random forest algorithm creates B different bootstrap samples from the training dataset, fits an overfit (and hence low bias) regression tree to each using only a random subset m size from all available predictors (also known as dropout), and then averages their forecasts as the final output. The overfit trees means that the underlying trees has low bias, and the dropout procedure means that they have low correlation. Thus, averaging these low bias, uncorrelated trees results in a low bias, yet stable model. Specific details of the random forest algorithm are detailed in the appendix.

Random forests were the most computationally intensive, and for feasibility purposes an ensemble of only $B = 30$ trees with $m = 20$ at most was considered for the simulation study, and $B = 50$ trees with $m = 40$ at most was considered for the empirical study.

2.8. Neural Networks

2.8.1. Introduction

Neural networks have theoretical underpinnings as "universal approximators" for any smooth predictive association, (Hornik et al. (1989)). They are arguably the most complex type of model available, able to capture several non-linear interactions through their many layers, hence its other name "deep learning." On the flipside, their high flexibility often means that they are among the most parameterized and least interpretable models, earning them the reputation as a black box model.

The scope of this paper is limited to traditional "feed-forward" networks. The feed forward network consists of an "input layer" of scaled data inputs, one or more "hidden layers" which interact and non-linearly transform the inputs, and finally an output layer that aggregates the hidden layers and transform them a final time for the final output.

More specifcally, a neural network consists of layers denoted by $l = 0, 1, \ldots, L$, with $l = 0$ denoting the input layer and $l = L$ denoting the output layer, and $K^{(l)}$ denoting the number of neurons in each hidden layer. The input layer is defined using predictors, $x^{(0)} = (1, z_1, \ldots, z_N)'$. The output of neuron k in layer l is then $x_k^{(l)}$. Next, define the vector of outputs for this layer as $x^{(l)} = (1, x_1^{(l)}, \ldots, x_{K^(l)}^{(l)})'$. The recursive output formula for the neural network at each neuron in layer $l > 0$ is then:

$$x_k^{(l)} = \alpha(x^{(l-1)'}\theta_k^{l-1}), \tag{11}$$

where $\alpha()$ represents the activation function for that layer (see next section) with the final output

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

Note that the specification of a constant "1" at the beginning of each layer is the same as specifying a bias term as is popular in other parametrizations.

Neural networks with up to 5 hidden layers were considered, each named NNX where X represents the number of hidden layers. The number of neurons is each layer was chosen according to the geometric pyramid rule (Masters, 1993): NN1 has 32 neurons, NN2 has 32 and 16 neurons in the first and second hidden layers respectively, NN3 has 32, 16, and 8 neurons, NN4 has 32, 16, 8, and 4 neurons, and NN5 has 32, 16, 8, 4, 2 neurons respectively. All units are fully connected; that is, each neurons receives input from all neurons the layer before it (see Figure 1). This mimics the methodology in Gu et al. (2018).

2.8.2. Activation Function

Several choices of activation functions exist in the literature, one of the most popular being the ReLU activation function (see Lecun et al. (2015) and Ramachandran et al. (2017), among others):

$$\text{ReLU}(x) = max(0, x) \tag{13}$$

owing to its high computational speed, ability to introduce sparseness into neural networks, and fewer vanishing gradient problems, as repeatedly applying ReLU tends to saturate less compared to other activation functions such as sigmoid and tanh. However, during the study it was observed that ReLU suffered tremendously from the "dying-ReLU problem," where ReLU neurons receive weight updates that fail to activate (output a 0), hence making it unable to receive further weight updates and learn. This resulted in networks which outputted the same value for the majority of, if not all inputs. To mitigate this, a variant of ReLU known as Leaky ReLU was used instead:

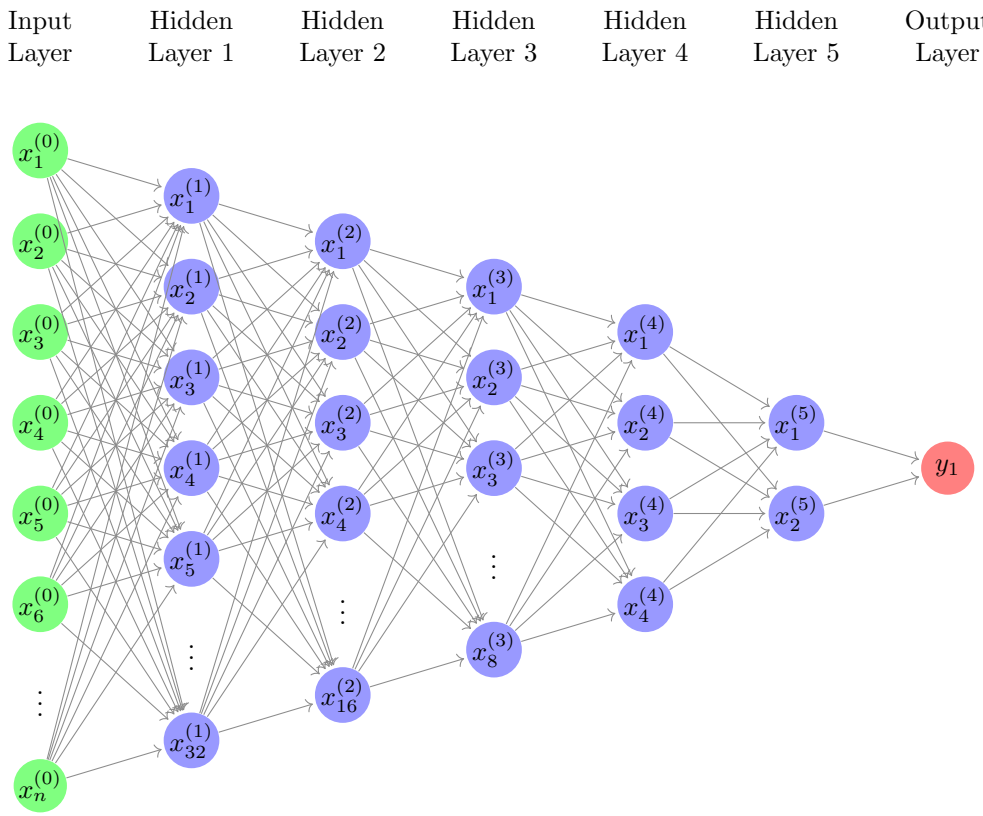

Figure 1. Neural Network 5 (most complex considered), without biases terms drawn

$$\text{lReLU}(x) = \begin{cases} x & \text{if} \quad x > 0; \\ 0.01x & \text{otherwise} \end{cases} \tag{14}$$

This activation function assigns a small non-zero gradient to each weight, hence enabling them to receive further weight updates.

2.8.3. Computation

RS: this, like all neural networks, is a mess. Re write

The neural network's weight and bias parameters for each layer are estimated by minimizing the loss function with respect to the parameters, i.e. by calculating the partial derivative with respect to a specific weight or bias element.

The solution to this is typically found via backpropagation, an iterative alogrithm similar to Gauss-Newton steps and produces a local minimum. These steps can be visualized as a descent towards the local minimum of the loss function; it is thus also known as "gradient descent." Note that the lack of a global minimum is actually desirable, as global minimums tend to be overfit solutions to the problem, (Choromanska et al., 2014). A common solution is to use "stochastic gradient descent" (SGD) where instead of optimising the loss function with respect to the entire training sample, only a small, random subset of the data (batch) is used at each optimisation step.

RS: Define learning rate more properly

Due the noisiness (randomness) introduced by SGD, the path towards the local minimum is more of

a quick "zig zag" and has the potential to "overshoot" the local minimum and not converge to a solution. This is typically controlled via a hyperparameter known as the learning rate, which controls the step size of each gradient descent iteration. Note that it is also common to apply a learning rate hyperparameter in non SGD environments as a way to control the magnitude of each step towards the solution to assist with convergence. The learning rate is to be tuned so that the solution path descends quickly enough to be computationally feasible, but slow enough so that it does not overshoot the local minimum and not converge. In this paper, the learning rate shrinkage algorithm which adaptively shrinks the learning rate of each individual weight separately as convergence occurs known as ADAM was employed (see Kingma and Ba (2014)). It should be noted that ADAM still requires a baseline learning rate which acts as a upper bound on the learning rate to be supplied, and this was tuned manually.

2.8.4. Batch Normalization

"Batch normalization" is a technique for addressing a phenomenon known as internal covariate shift, a particularly prevalent problem in training deep, complex neural networks, (Ioffe and Szegedy, 2015). Internal covariate shift occurs when the distributions of each layers' inputs change as the parameters of the previous layer change, resulting in the need for much slower learning rates and more careful initialization of parameters. More crucially, batch normalization has been observed to help normalizing the outputs of activation layers, alleviating issue related to vanishing or exploding gradients. Additionally, significant gains in computational speed may also be achieved.

2.8.5. Initialization

Finally, multiple random starting values for the weights and biases (seeds) were used in training neural networks, with the resulting predictions averaged in an ensemble model, Hansen and Salamon (1990). This regularizes the variance associated with the initial starting values for the weights and biases.

2.9. Simulation Design

2.9.1. Overall Design

RS: double check conformability of everything, make sure that the errors are corrected

Though Gu et al. (2018) explore the performance of machine learning on simulated returns series, their design used factors are uncorrelated across i, and, in particular, that the factors which enter the return equation are uncorrelated with the factors that do not enter the return equation. As note by Harvey et al. (2016) and many others, this is not what is observed in practice.

Therefore, we simulate an extension: a latent factor model with stochastic volatility for excess return, r_{t+1}, for $t = 1, \ldots, T$:

$$r_{i,t+1} = g\left(z_{i,t}\right) + \beta_{i,t+1} v_{t+1} + e_{i,t+1}; \quad z_{i,t} = (1, x_t)' \otimes c_{i,t}, \quad \beta_{i,t} = (c_{i1,t}, c_{i2,t}, c_{i3,t}) \tag{15}$$

$$e_{i,t+1} = \exp\left(\frac{\sigma^2_{i,t+1}}{2}\right) \varepsilon_{i,t+1}; \tag{16}$$

$$\sigma^2_{i,t+1} = \omega + \gamma \sigma^2_t + w_{t+1} \tag{17}$$

Let v_{t+1} be a 3×1 vector of errors, and $w_{t+1} \sim N(0,1)$ and $\varepsilon_{i,t+1} \sim N(0,1)$ scalar error terms. The parameters of these are tuned such that the R squared for each individual return series was around 5%,

and the annualized volatility of each return series approximately 22%, as is often observed empirically.

The matrix C_t is an $N \times P_c$ vector of latent factors, where the first three columns correspond to $\beta_{i,t}$, across the $1 \leq i \leq N$ dimensions, while the remaining $P_c - 3$ factors do not enter the return equation. The $P_x \times 1$ vector x_t is a 3×1 multivariate time series, and ε_{t+1} is a $N \times 1$ vector of idiosyncratic errors.

2.9.2. Simulating Characteristics

A simulation mechanism for C_t that gives some correlation across the factors and across time was used. First consider drawing normal random numbers for each $1 \leq i \leq N$ and $1 \leq j \leq P_c$, according to

$$\overline{c}_{ij,t} = \rho_j \overline{c}_{ij,t-1} + \epsilon_{ij,t}; \quad \rho_j \sim \mathcal{U}\left(\frac{1}{2}, 1\right) \tag{18}$$

Then, define the matrix

$$B := \Lambda\Lambda' + \frac{1}{10}\mathbb{I}_n, \quad \Lambda_i = (\lambda_{i1}, \ldots, \lambda_{i4}), \quad \lambda_{ik} \sim N(0, \lambda_{sd}),\ k = 1, \ldots, 4 \tag{19}$$

where λ_{sd} values of 0.01 and 0.1 were considered to explore different degrees of cross sectional correlation. This is then transformed into a correlation matrix W via the Cholesky Decomposition:

$$W = (\text{diag}(B))^{\frac{-1}{2}} (B) (\text{diag}(B))^{\frac{-1}{2}} \tag{20}$$

To build in cross-sectional correlation, from the $N \times P_c$ matrix $\bar{C}_t$, we simulate characteristics according to

$$\widehat{C}_t = W\overline{C}_t \tag{21}$$

Finally, the "observed" characteristics for each $1 \leq i \leq N$ and for $j = 1, \ldots, P_c$ are constructed according to:

$$c_{ij,t} = \frac{2}{n+1} \text{rank}\left(\hat{c}_{ij,t}\right) - 1. \tag{22}$$

with the rank transformation normalizing all predictors to be within $[-1, 1]$.

2.9.3. Simulating Macroeconomic Series

For simulation of x_t, a 3×1 multivariate time series, we consider a Vector Autoregression (VAR) model, a generalization of the univariate autoregressive model to multiple time series:

$$x_t = Ax_{t-1} + u_t, \quad u_t \sim N\left(\mu = (0,0,0)', \Sigma = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}\right)$$

where we specify A:

RS: clean up

Other more complex and interactive matrix specifications were briefly explored, but these did not appear to have a significant impact on results. More complex designs were observed to only affect the variable importance metrics, but only to a small degree.

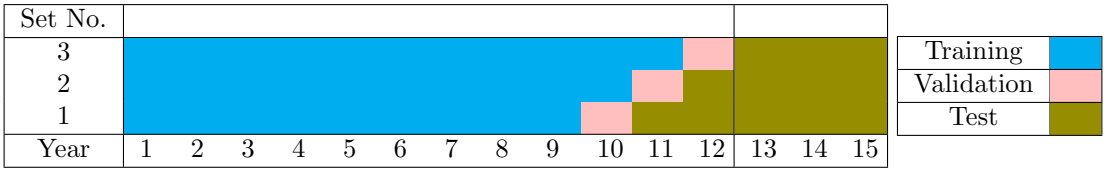

Figure 2. Sample Splitting Procedure

$$(1)\ A = \begin{pmatrix} .95 & 0 & 0 \\ 0 & .95 & 0 \\ 0 & 0 & .95 \end{pmatrix} \quad (2)\ A = \begin{pmatrix} 1 & 0 & .25 \\ 0 & .95 & 0 \\ .25 & 0 & .95 \end{pmatrix} \quad (3)\ A = \begin{pmatrix} .99 & .2 & .1 \\ .2 & .90 & -.3 \\ .1 & -.3 & -.99 \end{pmatrix} \tag{23}$$

2.9.4. Simulating Return Series

Three different function for $g(z_{i,t})$ were considered:

$$(1)\ g_1(z_{i,t}) = (c_{i1,t}, c_{i2,t}, c_{i3,t} \times x'_t)\,\theta_0; \quad \theta_0 = (0.02, 0.02, 0.02)'$$
$$(2)\ g_2(z_{i,t}) = \left(c^2_{i1,t}, c_{i1,t} \times c_{i2,t}, \operatorname{sgn}(c_{i3,t} \times x'_t)\right)\theta_0; \quad \theta_0 = (0.04, 0.035, 0.01)'$$
$$(3)\ g_3(z_{i,t}) = \left(1[c_{i3,t} > 0], c^3_{i2,t}, c_{i1,t} \times c_{i2,t} \times 1[c_{i3,t} > 0], \operatorname{logit}(c_{i3,t})\right)\theta_0; \quad \theta_0 = (0.04, 0.035, 0.01, 0.01)'$$

$g_1(z_{i,t})$ allows the characteristics to enter the return equation linearly, and $g_2(z_{i,t})$ allows the characteristics to enter the return equation interactively and non-linearly. These two specifications correspond to the simulation design proposed by Gu et al. (2018).

$g_3(z_{i,t})$ allows the characteristics to enter in a highly complex and non-linear fashion.

θ^0 was tuned such that the cross sectional R^2 and predictive R^2 were approximately 5%.

RS: change this

The simulation design results in $3 \times 3 = 12$ different simulated datasets, each with $N = 200$ stocks, $T = 180$ periods and $P_c = 100$ characteristics. Each design was simulated 30 times to assess the robustness of machine learning algorithms. The number of simulations was kept low for computational feasibility reasons.

2.9.5. Sample Splitting

If viewed as monthly periods, $T = 180$ corresponds to 15 years. The training sample was set to start from $T = 108$ or 9 years, a validation set 3 years in length, and a test set 3 in length. We employ the hybrid growing window approach as described earlier in section 2.2 (see Figure 2 for a graphical representation). The sample splitting scheme was chosen in a way which assisted with the stability of the advanced machine learning models, particularly neural networks.

Other popular schemes in the forecasting literature such as using an "inner" rolling window validation loop to find the best hyperparameters on average, finally aggregating them in an "outer" loop for a robust error were considered but not implemented for a variety of reasons. Firstly, many of the models are computationally too intensive for this to be feasible for this to be feasible. More importantly, during the model fitting process it was observed that the optimal hyperparameters for the different rolling windows

were highly unstable. Thus, this would have made the selection of the best hyperparameters on average across all windows significantly less meaningful.

2.10. Model Evaluation

2.10.1. R Squared

Overall predictive performance for individual excess stock returns were assessed using the out of sample R^2:

$$R^2_{OOS} = 1 - \frac{\sum_{(i,t)\in\mathcal{T}_3}(r_{i,t+1} - \widehat{r}_{i,t+1})^2}{\sum_{(i,t)\in\mathcal{T}_3}(r_{i,t+1} - \bar{r}_{i,t+1})^2} \tag{24}$$

where $\mathcal{T}_3$ indicates that the fits are only assessed on the test subsample, which is never used for training or tuning.

2.10.2. Diebold-Mariano Test

The Diebold-Mariano test (Diebold and Mariano (2002), and Harvey et al. (1997)) is a procedure which compares the forecast accuracy of two forecast methods. It is different to the overall R squared metric because it tests whether or not the models' forecast accuracy is significantly different.

Under the null hypothesis of the Diebold-Mariano test:

$$S_1^* = \left[\frac{n+1-2h+n^{-1}h(h-1)}{n}\right]^{1/2} S_1; \quad S_1^* \sim N(0,1) \tag{25}$$

$$S_1 = \left[\hat{V}(\bar{d})\right]^{-1/2}\bar{d} \tag{26}$$

$$\hat{\gamma}_k = n^{-1}\sum_{t=k+1}^{n}(d_t - \bar{d})(d_{t-k} - \bar{d}) \tag{27}$$

$$V(\bar{d}) \approx n^{-1}\left[\gamma_0 + 2\sum_{k=1}^{h-1}\gamma_k\right] \tag{28}$$

where d_t represents the difference series between the forecast errors of the two models $e_1t - e_2t$, $\hat{\gamma}_k$ represents the sample kth autocovariance for d_t, and S_1 represents the original unmodified Diebold Mariano test statistic.

As all models in this paper will be producing forecasts for an entire cross section of stocks, e_1t and e_2t will instead represent the cross sectional average forecast errors for each model.

2.11. Variable Importance

The importance of each predictor j is denoted as VI_j, and is defined as the reduction in predictive R-Squared from setting all values of predictor j to 0, while holding the remaining model estimates fixed. These were then normalized to sum to 1 within each model as a way to assess the relative importance of each predictor for each model. As VI_j can sometimes be negative, this was achieved by shifting all VI_j positively by the magnitude of the smallest VI_j, then dividing all VI_j by the total:

$$VI_{j,norm} = \frac{VI_j + \min(VI_j)}{\Sigma VI_j + \min(VI_j)} \tag{29}$$

Table 1: Simulation Results

Figure 3. Simulation Variable Importance Plots

This mechanism was chosen because the other popular normalization mechanism "softmax" was observed to be unable to preserve the distances between each original VI_j well enough, thus squishing together all importance metrics.

3. Study

3.1. Model

All machine learning methods are designed to approximate the empirical model $E_t(r_{i,t+1}) = g * (z_{i,t})$ defined in equation (2). The baseline set of stock-level covariates $z_{i,t}$ as:

$$z_{i,t} = x_t \otimes c_{i,t} \tag{30}$$

where $c_{i,t}$ is a $P_c \times 1$ matrix of characteristics for each stock i, and x_t is a $P_x \times 1$ vector of macroeconomic predictors (and are this common to all stocks, including a constant). $z_{i,t}$ is a $P \times 1$ vector of features for predicting individual stock returns ($P = P_c P_x$) and includes interactions between individual characteristics and macroeconomic characteristics.

4. Simulation Results

When the underlying data generating process is linear with no cross sectional correlation we see that the linear and penalized linear models offer the best performance. Random forests and neural networks offer performance that is not too far behind, and it is likely due to these models overfitting the training and validation sets, thus generalizing more poorly to the test set. In terms of variable importance, the linear and penalized linear models are able to consistently pick out the correct factors in the DGP, while the random forests and neural networks cannot.

Linear Models

The linear models performed very well in simple linear settings, with notably deteriorating performance as the specifications became more complex. They struggled greatly with cross sectional correlation even in specifications with no cross sectional correlation, as noted by the fact that they identify correlated interaction terms. When cross sectional correlation was increased

The introduction of non-linear interactions in the data generating process affected its performance, though it was still able to pick out transformations such as sgn and logit functions. This is likely due to these transformations not changing the direction of their terms.

Penalized Linear Models

The penalized linear models had markedly better performance than the linear models, though this was only apparent for more complex model specifications.

Of particular note is the instability of the optimal hyperparameter as the validation set moved forward in time. It was quite common to see optimal α values that changed from 1 (equivalent to the sparse case

Table 2: Diebold Mariano Tests for Simulation Study

of LASSO regression), to 0.01 (almost equivalent to dense case of ridge regression). Increasing the size of the validation set seemed to help a little, but not by a lot.

Similarly to the linear models, penalized linear models which minimized mean absolute error (corresponding to quantile regression) offered consistent improvements over their mean squared error versions.

Random Forests

The random forests exhibited higher performance and robustness compared the linear and penalized linear models.

The random forests struggle greatly with determining variable importance in the presence of cross sectional correlation. This is to be expected, likely due to how the random forest algorithms work. Recall that the random forest algorithm is an ensemble of tree models, with each tree model only having access to a subset of all available predictors. If this subset does not include the true data generating predictor, that particular tree will likely select the predictors which have the highest correlation with the true data generating predictor instead. Thus, the resulting ensemble model is likely to believe that cross sectionally correlated predictors are important.

Very computationally intensive, so only a conservative grid of hyperparameters was able to be implemented. In particular, a low ntree parameter was used for feasibility, and it was not feasible to explore which value of ntree stopped improving performance. It is generally recommended to set the ntree parameter as high as computationally feasible, as doing so typically only increases performance. This means that there is possibly more room for higher performance from random forest models.

Neural Networks

These were a disaster in general.

The neural networks had surprisingly very poor performance compared to the other models, in addition to being the most involved to fit well. Due to computational environment having access to GPUs, higher batch sizes were faster. However, higher batch sizes were observed to almost universally producing poorer solutions. Smaller batch sizes produced better solutions, up until the batch size was too small and the resulting noise introduced by the randomness of each batch overpowered the gains in performance. This well documented in other papers, and due to the computation resource limitations it is typically recommended to set these by trial and error, rather than through a systematic validation process. This is what was done, and it was found that batch sizes of 32 (default) or 64, max epochs of at least 100, and an early stopping patience parameter of at least 20 typically produced good results. Smaller batch sizes were slightly better, but computationally too intensive to implement.

The neural networks were observed to produce forecasts which did not make intuitive sense, and almost always found solutions which resulted in constant prediction values no matter the input. There aren't really solutions for explicitly addressing problems like these.

When the underlying data generating process becomes somewhat more non-linear (g2 specification), we start to see the linear and penalized linear models suffering in performance.

Moving to even more non linear DGPs (g3), the limitations of linear and penalized linear models becomes more apparent.

Once cross sectional correlation across factors is introduced into the DGP however, all models perform significantly worse.

5. Empirical Data

5.1. Data & Cleaning Procedure

We mimic the data procedure of Gu et al. (2018). This means that we obtain the dataset provided by Gu on his website. This dataset contains 94 stock level characteristics: 61 updated annually, 13 updated quarterly and 20 updated monthly. In addition, 74 industry dummies corresponding the the first two digits of the Standard Industrial Classification (SIC) codes are included. It is noted that this dataset contains all securities traded, including those with a CRSP share code other than 10 or 11 and thus includes instruments such as REITs and mutual funds, and those with a share price of less than \$5.

We detail our cleaning procedure of this dataset. To reduce the size and maintain computational feasibility, only shares traded on the NASDAQ exchange was included (filtered out using the PRIMEXCH variable). As the majority of the characteristics included in the dataset are only updated annually, this significantly decreases the overall variability present in the dataset. To increase the variability yet also maintain a large enough number of time periods, quarterly returns were constructed. This also had the benefit of decreasing the dataset size for computational feasibility.

We then follow Gu et al. (2018) and construct eight macroeconomic factors following the variable definitions in Welch and Goyal (2008): 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).

The full details of these factors are available in table X.

There exists a significant amount of missing data in the dataset. The dataset's columns were first examined, and any characteristics that had over 20% of their data were removed. However, as the amount of missing data increases dramatically going further back in time, a balance between using more periods at the cost of removing more characteristics versus using less periods but keeping more characteristics was needed. In the end, it was decided that any data before 1993 Q2 was to be filtered out, as there was a noticeable increase in data availability after this period.

Throughout all methods and analysis we define the baseline set of covariates as:

$$z_{i,t} = (1, x_t)' \otimes c_{i,t} \tag{31}$$

where $c_{i,t}$ is a P_c matrix of characteristics for each stock i, and $(1, x_t)'$ is a $P_x \times 1$ vector of macroeconomic predictors. $z_{i,t}$ is therefore a $P_x P_c$ vector of features for predicting individual stock returns and includes interactions between stock level characteristics and macroeconomic variables. The total number of covariates in this baseline set is $94 \times (8 + 1) + 74 = 920$.

The dataset was not normalized for all methods, as only penalized regression and neural networks are affected by normalization. For these two methods, the dataset was normalized such that each predictor column had 0 mean and 1 variance.

The final dataset spanned from 1993 Q3 to 2016 Q6 with 20,000

5.2. Empirical Data Model Fitting

We detail our model fitting procedure used specifically for the empirical dataset.

We try to mimic the procedure used in the simulation study. This means that the dataset was split such that the training and validation sets were split such that the training set was 1.5 times the length of the validation set, in order to predict a test set that is one year in length.

Table 3: Empirical Data Results

Figure 4. Simulation Variable Importance Plots

5.3. Empirical Data Results

In general, results from the simulation study were reproduced in the empirical study.

Performance

The linear models did the most poorly. The penalized linear models offered a substantial increase in performance. The random forest was the best performing model. The neural networks offered varying results.

Variable Importance

As the true underlying covariates for empirical returns is unknown, variable importance is only discussed with regards to how consistent each method is at picking out true covariates.

The linear and penalized linear models appear to be mostly consistent.

The neural networks in general identified similar groups of variables, but also omitted other variables.

6. Limitations

7. Conclusion

8. Appendix

8.1. Data

Table 4: Diebold Mariano Tests for Empirical Data

No.	Acronym	Firm Characteristic	Author(s)	Data Source	Frequency
1	absacc	Absolute Accruals	Bandyopadhyay et al. (2010)	Compustat	Annual
2	acc	Working capital accruals	Sloan (1996)	Compustat	Annual
3	aeavol	Abnormal earnings announcement volume	Lerman et al. (2008)	Compustat	Quarterly
4	age	# years since first Compustat coverage	Jiang et al. (2005)	Compustat	Annual
5	agr	Asset growth	Cooper et al. (2008)	Compustat	Annual
6	baspread	Bid-ask spread	Amihud and Mendelson (1989)	Compustat	Monthly
7	beta	Beta	Fama and MacBeth (1973)	Compustat	Monthly
8	betasq	Beta squared	Fama and MacBeth (1973)	Compustat	Monthly
9	bm	Book-to-market	Rosenberg et al. (1985)	Compustat	Annual
10	bm_ia	Industry-adjusted book to market	Asness et al. (2000)	Compustat	Quarterly
11	cash	Cash holdings	Palazzo (2012)	Compustat	Annual
12	cashdebt	Cashflow to debt	Ou and Penman (1989)	Compustat	Annual
13	cashpr	Cash productivity	Chandrashekar and Rao (2009)	Compustat	Annual
14	cfp	Cashflow to price ratio	Desai et al. (2004)	Compustat	Annual
15	cfp_ia	Industry-adjusted cashfow to price ratio	Asness et al. (2000)	Compustat	Annual
16	chatoia	Industry-adjusted change in asset turnover	Soliman (2008)	Compustat	Annual
17	chcsho	Change in shares outstanding	Pontiff and Woodgate (2008)	Compustat	Annual
18	chempia	Industry-adjusted change in employee	Asness et al. (2000)	Compustat	Annual
19	chinv	Change in inventory	Thomas and Zhang (2002)	Compustat	Annual
20	chmom	Change in 6-month momentum	Gettleman and Marks (2006)	Compustat	Monthly
21	chpmia	Industry-adjusted change in profit margin	Soliman (2008)	Compustat	Annual
22	chtx	Change in tax expense	Thomas and Zhang (2011)	Compustat	Quarterly
23	cinvest	Corporate investment	Titman et al. (2004)	Compustat	Quarterly
24	convind	Convertible debt indicator	Valta (2016)	Compustat	Annual
25	currat	Current ratio	Ou and Penman (1989)	Compustat	Annual
26	depr	Depreciation / PP&E	Holthausen and Larcker (1992)	Compustat	Annual
27	divi	Dividend initiation	Michaely et al. (1995)	Compustat	Annual

28	divo	Dividend omission	Michaely et al. (1995)	Compustat	Annual
29	dolvol	Dollar trading volume	Chordia et al. (2001)	Compustat	Monthly
30	dy	Dividend to price	Litzenberger and Ramaswamy (1982)	Compustat	Annual
31	ear	Earnings announcement return	Brandt et al. (2008)	Compustat	Quarterly
32	egr	Growth in common shareholder eq	Richardson et al. (2005)	Compustat	Annual
33	ep	Earnings to price	Basu (1977)	Compustat	Annual
34	gma	Gross profitability	Novy-Marx (2013)	Compustat	Annual
35	grCAPX	Growth in capital expenditures	Anderson and Garcia-Feijo (2006)	Compustat	Annual
36	grltnoa	Growth in long term net operating assets	Fairfield et al. (2003)	Compustat	Annual
37	herf	Industry sales concentration	Hou and Robinson (2006)	Compustat	Annual
38	hire	Employee growth rate	Belo et al. (2014)	Compustat	Annual
39	idiovol	Idiosyncratic return volatility	Ali et al. (2003)	Compustat	Monthly
40	ill	Illiquidity	Amihud (2002)	Compustat	Monthly
41	indmom	Industry momentum	Moskowitz and Grinblatt (1999)	Compustat	Monthly
42	invest	Capital expenditures and inventory	Chen and Zhang (2010)	Compustat	Annual
43	lev	Leverage	Bhandari (1988)	Compustat	Annual
44	lgr	Growth in long-term debt	Richardson et al. (2005)	Compustat	Annual
45	maxret	Maximum daily return	Bali et al. (2011)	Compustat	Monthly
46	mom12m	12-month momentum	Jegadeesh (1990)	Compustat	Monthly
47	mom1	1-month momentum	Jegadeesh and Titman (1993)	Compustat	Monthly
48	mom36m	36-month momentum	Jegadeesh and Titman (1993)	Compustat	Monthly
49	mom6m	6-month momentum	Jegadeesh and Titman (1993)	Compustat	Monthly
50	ms	Financial statement score	Mohanram (2005)	Compustat	Quarterly
51	mvel1	Size	Banz (1981)	Compustat	Monthly
52	mve_ia	Industry-adjusted size	Asness et al. (2000)	Compustat	Annual
53	nincr	Number of earnings increases	Barth et al. (1999)	Compustat	Quarterly
54	operprof	Operating profitability	Fama and French (2015)	Compustat	Annual
55	orgcap	Organizational capital	Eisfeldt and Papanikolaou (2013)	Compustat	Annual
56	pchcapx_ia	Industry adjusted % change in capital expenditures	Abarbanell and Bushee (1998)	Compustat	Annual

57	pchcurrat	% change in current ratio	Ou and Penman (1989)	Compustat	Annual	
58	pchdepr	% change in depreciation	Holthausen and Larcker (1992)	Compustat	Annual	
59	pchgm_pchsale	% change in gross margin - % change in sales	Abarbanell and Bushee (1998)	Compustat	Annual	
60	pchquick	% change in quick ratio	Ou and Penman (1989)	Compustat	Annual	
61	pchsale_pchinvt	% change in sales - % change in inventory	Abarbanell and Bushee (1998)	Compustat	Annual	
62	pchsale_pchrect	% change in sales - % change in A/R	Abarbanell and Bushee (1998)	Compustat	Annual	
63	pchsale_pchxsga	% change in sales - % change in SG	Abarbanell and Bushee (1998)	Compustat	Annual	
64	pchsaleinv	% change sales-to-inventory	Ou and Penman (1989)	Compustat	Annual	
65	pctacc	Percent accruals	Hafzalla et al. (2011)	Compustat	Annual	
66	pricedelay	Price delay	Hou and Moskowitz (2005)	Compustat	Monthly	
67	ps	Financial statements score	Piotroski (2000)	Categorical	Compustat	Annual
68	quick	Quick ratio	Ou and Penman (1989)	Compustat	Annual	
69	rd	R&D increase	Eberhart et al. (2004)	Compustat	Annual	
70	rd_mve	R&D to market capitalization	Guo et al. (2006)	Compustat	Annual	
71	rd_sale	R&D to sales	Guo et al. (2006)	Compustat	Annual	
72	realestate	Real estate holdings	Tuzel (2010)	Compustat	Annual	
73	retvol	Return volatility	Ang et al. (2006)	Compustat	Monthly	
74	roaq	Return on assets	Balakrishnan et al. (2010)	Compustat	Quarterly	
75	roavol	Earnings volatility	cite	Compustat	Quarterly	
76	roeq	Return on equity	Hou et al. (2015)	Compustat	Quarterly	
77	roic	Return on invested capital	Brown and Rowe (2007)	Compustat	Annual	
78	rsup	Revenue surprise	cite	Compustat	Quarterly	
79	salecash	Sales to cash	Ou and Penman (1989)	Compustat	Annual	
80	saleinv	Sales to inventory	Ou and Penman (1989)	Compustat	Annual	
81	salerec	Sales to receivables	Ou and Penman (1989)	Compustat	Annual	
82	secured	Secured debt	Valta (2016)	Compustat	Annual	
83	securedind	Secured debt indicator	Valta (2016)	Compustat	Annual	
84	sgr	Sales growth	Barbee Jr et al. (1996)	Compustat	Annual	
85	sin	Sin stocks	Hong and Kacperczyk (2009)	Compustat	Annual	

86	sp	Sales to price	cite	Compustat	Annual
87	std_dolvol	Volatility of liquidity (dollar trading volume)	Chordia et al. (2001)	Compustat	Annual
88	std_turn	Volatility of liquidity (share turnover)	Chordia et al. (2001)	Compustat	Monthly
89	stdacc	Accrual volatility	Bandyopadhyay et al. (2010)	Compustat	Monthly
90	stdcf	Cashflow volatility	Huang (2009)	Compustat	Quarterly
91	tang	Debt capacity/rm tangibility	Almeida and Campello (2007)	Compustat	Quarterly
92	tb	Tax income to book income	Lev and Ohlson (1982)	Compustat	Annual
93	turn	Share turnover	Datar et al. (1998)	Compustat	Monthly
94	zerotrade	Zero trading days	Liu (2006)	Compustat	Monthly

Table 6: Macroeconomic Factors

No.	Acronym	Macroeconomic Factor
1	macro_dp	Dividend Price Ratio
2	macro_ep	Earnings Price Ratio
3	macro_bm	Book to Market Ratio
4	macro_ntis	Net Equity Expansion
5	macro_tbl	Treasury Bill Rate
6	macro_tms	Term Spread
7	macro_dfy	Default Spread
8	macro_svar	Stock Variance

8.2. Computational Details

8.2.1. Linear Models

OLS used for fitting wrt MSE.

quantreg used for fitting wrt MAE. (minimizing 0.5 quantile loss is equivalent to minimizing mae).

8.2.2. Penalized Linear

The package *hqreg* was used to fit penalized regression models with respect to MSE and MAE.

This package efficiently calculates a regularization path of penalization values given a value for α. This means that it is much more efficient to instead only supply a grid for α, let the algorithm decide its own path of penalization values. The combination of these two parameters which produces the best results on the validation set were then chosen.

The more traditional method of supplying a grid of α and λ was also implemented at first, and this was observed to give almost identical results to the above implementation, but at a much higher computational cost.

8.2.3. Classification and Regression Trees

For full details of the Classification and Regression Tree algorithm see Breiman (1984).

Algorithm 1: Classification and Regression Tree

Initialize ;

for *d from 1 to L* **do**

for *i in* $C_l(d-1), l = 1, \ldots, 2^{d-1}$ **do**

For each feature $j = 1, 2, \ldots, P$, and each threshold level α, define a split as $s = (j, \alpha)$ which divides $C_l(d-1)$ into C_{left} and C_{right}:

$$C_{left}s = \{z_j \leq \alpha\} \cap C_l(d-1); C_{right}s = \{z_j > \alpha\} \cap C_l(d-1)$$

Define the impurity function:

$$\mathcal{L}(C, C_{left}, C_{right}) = \frac{|C_{left}|}{|C|} H(C_{left}) + \frac{|C_{right}|}{|C|} H(C_{right})$$

where

$$H(C) = \frac{1}{|C|} \sum^{z_{i,t} \in C} (r_{i,t+1} - \theta)^2, \theta = \frac{1}{|C|} \sum_{z_{i,t} \in C} r_{i,t+1}$$

and $|C|$ denotes the number of observations in set C

Find the optimal split

$$s^* \leftarrow \underset{s}{argmin} \mathcal{L}(C(s), C_{left}(s), C_{right}(s))$$

Update nodes (partition the data):

$$C_{2l-1}(d) \leftarrow C_{left}(s^*), C_{2l}(d) \leftarrow C_{right}(s^*)$$

end

end

Result: The prediction of a regression tree is:

$$g(z_{i,t}; \theta, L) = \sum_{k=1}^{2^L} \theta_k \mathbf{1}_{z_{i,t} \in C_k(L)}; \theta_k = \frac{1}{|C_k(L)|} \sum_{z_{i,t} \in C_k(L)} r_{i,t+1}$$

8.2.4. *Random Forest*

For full details of the Random Forest algorithm see Breiman (2001).

Algorithm 2: Random Forest

for b *from* 1 *to* B **do**

Draw bootstrap samples $(z_{i,t}, r_{i,t+1}), (i,t) \in Bootstrap(b)$ from the dataset Grow a tree T_b using Algorithm, using only a random subsample, say $\sqrt{P}$ of all features Denote the resulting bth tree as

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

end

Result: The final random forest prediction is given by the output of all trees:

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

8.2.5. *Neural Networks*

Fit using keras package (cite)

ADAM algorithm for stochastic gradient descent and learning rate shrinkage as detailed by Kingma and Ba (2014).

Algorithm 3: Early stopping via validation

Initialize $j = 0$, $\epsilon = \infty$ and select the patience parameter p (max iterations)
while j ¡ p **do**
  Update θ using the training algorithm Calculate the prediction error from the validation sample, denoted as ϵ'
  if $\epsilon' < \epsilon$ **then**
    $j \leftarrow 0$
    $\epsilon \leftarrow \epsilon'$
    $\theta' \leftarrow \theta$
  else
    $j \leftarrow j + 1$
  end
end
Result: θ' is the final parameter estimate

Batch Normalization Algorithm as detailed by Ioffe and Szegedy (2015).

Algorithm 4: Batch Normalization for one activation over one batch

Input: Values of x for each activation over a batch $\mathcal{B} = x_1, x_2, \ldots, x_N$
$\mu_{\mathcal{B}} \leftarrow \frac{1}{N} \sum_{i=1}^{N} x_i$
$\sigma_{\mathcal{B}}^2 \leftarrow \frac{1}{N} \sum_{i=1}^{N} (x_i - \mu_{\mathcal{B}})^2$
$\hat{x_i} \leftarrow \frac{x_i - \mu_{\mathcal{B}}}{\sqrt{\sigma_{\mathcal{B}}^2 + \epsilon}}$
$y_i \leftarrow \gamma \hat{x_i} + \beta := BN_{\gamma,\beta}(x_i)$
Result: $y_i = BN_{\gamma,\beta}(x_i) : i = 1, 2, ..., N$

8.2.6. Tuning of Simulated Datasets

The simulated datasets were tuned according to the following statistics: average individual time series R squared, average annualized volatility, and cross sectional R squared, and true/predictive R squared.

The methodology for evaluating average time series R squared and cross sectional R squared is consistent with that detailed by Cochrane (2005). The steps are reproduced here for reference; for complete details refer to Cochrane (2005).

First evaluate the following OLS model:

$$R_{it} = a_i + \beta_i' f_{it} + \epsilon_{it} \tag{32}$$

where f_{it} represents the *true* factors in the returns process. The corresponding R-squared value for this time series regression is calculated across all stocks and averaged to yield the average time series R-Squared.

A cross sectional regression for the risk premia is then run across assets of average returns on the factor coefficients:

$$\bar{R_{it}} = \alpha_i = \beta_i' \lambda \tag{33}$$

where the β_i' are the estimated coefficients from each time series regressions run previously. The corresponding R squared for this regression is the cross sectional R squared.

The true R squared is a measure of signal to noise ratio i.e. how much of the returns data is due to This is simply calculated by running a pooled ordinary least squares regression on the entire panel, using the underlying g() as the "predictions.' The resulting R squared value is therefore a measure of how much of the panel can be explained by the g() term exclusive of any noise or error terms.

References

Abarbanell, J. S., Bushee, B. J., 1998. Abnormal returns to a fundamental analysis strategy. Accounting Review pp. 19–45.

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

Almeida, H., Campello, M., 2007. Financial constraints, asset tangibility, and corporate investment. The Review of Financial Studies 20, 1429–1460.

Amihud, Y., 2002. Illiquidity and stock returns: cross-section and time-series effects. Journal of financial markets 5, 31–56.

Amihud, Y., Mendelson, H., 1989. The effects of beta, bid-ask spread, residual risk, and size on stock returns. The Journal of Finance 44, 479–486.

Anderson, C. W., Garcia-Feijo, L., 2006. Empirical evidence on capital investment, growth options, and security returns. The Journal of Finance 61, 171–194.

Ang, A., Bekaert, G., 2006. Stock return predictability: Is it there? The Review of Financial Studies 20, 651–707.

Ang, A., Hodrick, R. J., Xing, Y., Zhang, X., 2006. The Cross-Section of Volatility and Expected Returns. The Journal of Finance 61, 259–299.

Asness, C. S., Porter, R. B., Stevens, R. L., 2000. Predicting stock returns using industry-relative firm characteristics. Available at SSRN 213872 .

Balakrishnan, K., Bartov, E., Faurel, L., 2010. Post loss/profit announcement drift. Journal of Accounting and Economics 50, 20–41.

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

Bandyopadhyay, S. P., Huang, A. G., Wirjanto, T. S., 2010. The accrual volatility anomaly. Unpublished Manuscript, University of Waterloo .

Banz, R. W., 1981. The relationship between return and market value of common stocks. Journal of financial economics 9, 3–18.

Barbee Jr, W. C., Mukherji, S., Raines, G. A., 1996. Do salesprice and debtequity explain stock returns better than bookmarket and firm size? Financial Analysts Journal 52, 56–60.

Barth, M. E., Elliott, J. A., Finn, M. W., 1999. Market rewards associated with patterns of increasing earnings. Journal of Accounting Research 37, 387–413.

Basu, S., 1977. Investment performance of common stocks in relation to their price-earnings ratios: A test of the efficient market hypothesis. The journal of Finance 32, 663–682.

Belo, F., Lin, X., Bazdresch, S., 2014. Labor hiring, investment, and stock return predictability in the cross section. Journal of Political Economy 122, 129–177.

Bhandari, L. C., 1988. Debt/equity ratio and expected common stock returns: Empirical evidence. The journal of finance 43, 507–528.

Brandt, M. W., Kishore, R., Santa-Clara, P., Venkatachalam, M., 2008. Earnings announcements are full of surprises. SSRN eLibrary .

Breiman, L., 1984. Classification and Regression Trees. Routledge.

Breiman, L., 2001. Random forests. Machine learning 45, 5–32.

Brown, D. P., Rowe, B., 2007. The productivity premium in equity returns. Available at SSRN 993467 .

Chandrashekar, S., Rao, R. K., 2009. The productivity of corporate cash holdings and the cross-section of expected stock returns. McCombs Research Paper Series No. FIN-03-09 .

Chen, L., Zhang, L., 2010. A better three-factor model that explains more anomalies. Journal of Finance 65, 563–595.

Chordia, T., Subrahmanyam, A., Anshuman, V. R., 2001. Trading activity and expected stock returns. Journal of Financial Economics 59, 3–32.

Choromanska, A., Henaff, M., Mathieu, M., Arous, G. B., LeCun, Y., 2014. The Loss Surfaces of Multilayer Networks. arXiv:1412.0233 [cs] ArXiv: 1412.0233.

Cochrane, J. H., 2005. Asset pricing. Princeton Univ. Press, Princeton, NJ, rev. ed ed., oCLC: 265653065.

Cochrane, J. H., 2011. Presidential Address: Discount Rates. The Journal of Finance 66, 1047–1108.

Cooper, M. J., Gulen, H., Schill, M. J., 2008. Asset Growth and the Cross-Section of Stock Returns. The Journal of Finance 63, 1609–1651.

Datar, V. T., Naik, N. Y., Radcliffe, R., 1998. Liquidity and stock returns: An alternative test. Journal of Financial Markets 1, 203–219.

Desai, H., Rajgopal, S., Venkatachalam, M., 2004. Value-glamour and accruals mispricing: One anomaly or two? The Accounting Review 79, 355–385.

Diebold, F. X., Mariano, R. S., 2002. Comparing predictive accuracy. Journal of Business & economic statistics 20, 134–144.

Eberhart, A. C., Maxwell, W. F., Siddique, A. R., 2004. An examination of long-term abnormal stock returns and operating performance following R&D increases. The Journal of Finance 59, 623–650.

Eisfeldt, A. L., Papanikolaou, D., 2013. Organization capital and the cross-section of expected returns. The Journal of Finance 68, 1365–1406.

Fairfield, P. M., Whisenant, J. S., Yohn, T. L., 2003. Accrued earnings and growth: Implications for future profitability and market mispricing. The accounting review 78, 353–371.

Fama, E. F., French, K. R., 2015. A five-factor asset pricing model. Journal of financial economics 116, 1–22.

Fama, E. F., MacBeth, J. D., 1973. Risk, return, and equilibrium: Empirical tests. Journal of political economy 81, 607–636.

Feng, G., Giglio, S., Xiu, D., 2019. Taming the Factor Zoo: A Test of New Factors. Tech. Rep. w25481, National Bureau of Economic Research, Cambridge, MA.

Feng, G., He, J., Polson, N. G., 2018. Deep Learning for Predicting Asset Returns. arXiv:1804.09314 [cs, econ, stat] ArXiv: 1804.09314.

Freyberger, J., Neuhierl, A., Weber, M., 2017. Dissecting characteristics nonparametrically. Tech. rep., National Bureau of Economic Research.

Gettleman, E., Marks, J. M., 2006. Acceleration strategies. Tech. rep., Working Paper, Seton Hall Univeristy.

Goetzmann, W. N., Jorion, P., 1993. Testing the predictive power of dividend yields. The Journal of Finance 48, 663–679.

Goyal, A., Welch, I., 2003. Predicting the equity premium with dividend ratios. Management Science 49, 639–654.

Gu, S., Kelly, B., Xiu, D., 2018. Empirical asset pricing via machine learning. Tech. rep., National Bureau of Economic Research.

Guo, R.-J., Lev, B., Shi, C., 2006. Explaining the Short-and Long-Term IPO Anomalies in the US by R&D. Journal of Business Finance & Accounting 33, 550–579.

Hafzalla, N., Lundholm, R., Matthew Van Winkle, E., 2011. Percent accruals. The Accounting Review 86, 209–236.

Hansen, L. K., Salamon, P., 1990. Neural network ensembles. IEEE Transactions on Pattern Analysis & Machine Intelligence pp. 993–1001.

Harvey, C. R., Liu, Y., 2019. A Census of the Factor Zoo. Social Science Research Network p. 7.

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

Harvey, D., Leybourne, S., Newbold, P., 1997. Testing the equality of prediction mean squared errors. International Journal of Forecasting 13, 281–291.

Hastie, T., Tibshirani, R., Friedman, J. H., 2009. The elements of statistical learning: data mining, inference, and prediction. Springer series in statistics, Springer, New York, NY, second ed.

Hoerl, A. E., Kennard, R. W., 1970. Ridge regression: Biased estimation for nonorthogonal problems. Technometrics 12, 55–67.

Holthausen, R. W., Larcker, D. F., 1992. The prediction of stock returns using financial statement information. Journal of accounting and economics 15, 373–411.

Hong, H., Kacperczyk, M., 2009. The price of sin: The effects of social norms on markets. Journal of Financial Economics 93, 15–36.

Hornik, K., Stinchcombe, M., White, H., 1989. Multilayer feedforward networks are universal approximators. Neural networks 2, 359–366.

Hou, K., Moskowitz, T. J., 2005. Market frictions, price delay, and the cross-section of expected returns. The Review of Financial Studies 18, 981–1020.

Hou, K., Robinson, D. T., 2006. Industry concentration and average stock returns. The Journal of Finance 61, 1927–1956.

Hou, K., Xue, C., Zhang, L., 2015. Digesting Anomalies: An Investment Approach. Review of Financial Studies 28, 650–705.

Hsu, J., Kalesnik, V., 2014. Finding smart beta in the factor zoo. Research Affiliates (July) .

Huang, A. G., 2009. The cross section of cashflow volatility and expected stock returns. Journal of Empirical Finance 16, 409–429.

Ioffe, S., Szegedy, C., 2015. Batch normalization: Accelerating deep network training by reducing internal covariate shift. arXiv preprint arXiv:1502.03167 .

Jegadeesh, N., 1990. Evidence of predictable behavior of security returns. The Journal of finance 45, 881–898.

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

Jiang, G., Lee, C. M., Zhang, Y., 2005. Information uncertainty and expected returns. Review of Accounting Studies 10, 185–221.

Kingma, D. P., Ba, J., 2014. Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 .

Kozak, S., Nagel, S., Santosh, S., 2017. Shrinking the cross section. Tech. rep., National Bureau of Economic Research.

Lecun, Y., Bengio, Y., Hinton, G., 2015. Deep learning. Nature 521, 436–444.

Lerman, A., Livnat, J., Mendenhall, R. R., 2008. The High-Volume Return Premium and Post-Earnings Announcement Drift. SSRN Electronic Journal .

Lettau, M., Ludvigson, S., 2001. Consumption, Aggregate Wealth, and Expected Stock Returns. The Journal of Finance 56, 815–849.

Lev, B., Ohlson, J. A., 1982. Market-based empirical research in accounting: A review, interpretation, and extension. Journal of Accounting research pp. 249–322.

Litzenberger, R. H., Ramaswamy, K., 1982. The effects of dividends on common stock prices tax effects or information effects? The Journal of Finance 37, 429–443.

Liu, W., 2006. A liquidity-augmented capital asset pricing model. Journal of financial Economics 82, 631–671.

Masters, T., 1993. Practical Neural Network Recipes in C++. Morgan Kaufmann, google-Books-ID: 7Ez_Pq0sp2EC.

Michaely, R., Thaler, R. H., Womack, K. L., 1995. Price reactions to dividend initiations and omissions: Overreaction or drift? The Journal of Finance 50, 573–608.

Mohanram, P. S., 2005. Separating winners from losers among lowbook-to-market stocks using financial statement analysis. Review of accounting studies 10, 133–170.

Moskowitz, T. J., Grinblatt, M., 1999. Do industries explain momentum? The Journal of finance 54, 1249–1290.

Novy-Marx, R., 2013. The other side of value: The gross profitability premium. Journal of Financial Economics 108, 1–28.

Ou, J. A., Penman, S. H., 1989. Financial statement analysis and the prediction of stock returns. Journal of accounting and economics 11, 295–329.

Palazzo, B., 2012. Cash holdings, risk, and expected returns. Journal of Financial Economics 104, 162–185.

Piotroski, J. D., 2000. Value investing: The use of historical financial statement information to separate winners from losers. Journal of Accounting Research 38, 1–52.

Pontiff, J., Woodgate, A., 2008. Share issuance and cross-sectional returns. The Journal of Finance 63, 921–945.

Ramachandran, P., Zoph, B., Le, Q. V., 2017. Searching for Activation Functions. arXiv:1710.05941 [cs] ArXiv: 1710.05941.

Rapach, D., Zhou, G., 2013. Forecasting Stock Returns. In: *Handbook of Economic Forecasting*, Elsevier, vol. 2, pp. 328–383.

Richardson, S. A., Sloan, R. G., Soliman, M. T., Tuna, I., 2005. Accrual reliability, earnings persistence and stock prices. Journal of accounting and economics 39, 437–485.

Rosenberg, B., Reid, K., Lanstein, R., 1985. Persuasive evidence of market inefficiency. The Journal of Portfolio Management 11, 9–16.

Schwert, G. W., 2003. Anomalies and market efficiency. Handbook of the Economics of Finance 1, 939–974.

Sloan, R. G., 1996. Do Stock Prices Fully Reflect Information in Accruals and Cash Flows about Future Earnings? The Accounting Review 71, 289–315.

Soliman, M. T., 2008. The use of DuPont analysis by market participants. The Accounting Review 83, 823–853.

Thomas, J., Zhang, F. X., 2011. Tax expense momentum. Journal of Accounting Research 49, 791–821.

Thomas, J. K., Zhang, H., 2002. Inventory changes and future returns. Review of Accounting Studies 7, 163–187.

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

Titman, S., Wei, K. J., Xie, F., 2004. Capital investments and stock returns. Journal of financial and Quantitative Analysis 39, 677–700.

Tuzel, S., 2010. Corporate real estate holdings and the cross-section of stock returns. The Review of Financial Studies 23, 2268–2302.

Valta, P., 2016. Strategic default, debt structure, and stock returns. Journal of Financial and Quantitative Analysis 51, 197–229.

Welch, I., Goyal, A., 2008. A Comprehensive Look at The Empirical Performance of Equity Premium Prediction. Review of Financial Studies 21, 1455–1508.

Zou, H., Hastie, T., 2005. Regularization and variable selection via the Elastic Net. Journal of the Royal Statistical Society, Series B 67, 301–320.