Stevens Institute of Technology

Financial Engineering

Machine Learning Applications in Empirical Asset Pricing

Author:
Eric Mozeika
Andrew Shields
Dheemanth Sriram

Supervisor:
Dr. Cristian Homescu

May 14, 2020

Abstract

The use of Machine Learning (ML) in financial markets has become a popular topic in recent years. We apply these concepts to empirical asset pricing in an attempt to further understand their potential improvements over classical asset pricing models. In addition, we run several types of ML models and perform comparative analysis of their benefits against each other and across different sets of market data such as returns from industries, stocks and commodities. We then deploy a backtesting framework and implement an equally-weighted long-short portfolio in order to study each model's ability to forecast returns. Finally, we investigate methods to improve the accuracy of these models with the goal of boosting portfolio performance. Our analysis consists of two parts: (1) determining the model's ability to make accurate forecasts and (2) ascertaining its ability to create superior portfolio returns in comparison to classical models.

1 Introduction

In the early 1960s, the Capital Asset Pricing Model (CAPM) was invented and laid the groundwork for asset pricing and risk premia analysis. The CAPM at its core is a single factor model which can describe an asset's expected return by its sensitivity to the overall market's excess return.

$$\mathbb{E}(R_i) = R_f + \beta_i(\mathbb{E}(R_m) - R_f) \tag{1}$$

In 1992 Fama and French expanded on the CAPM to create the Fama-French Three-Factor Model (FF3). Expanding on the baseline of the CAPM, this model includes two additional factors used to describe the expected returns of an asset. Since then, there have been several additions to these models which now include the Carhart Four-Factor Model and the Fama-French Five-Factor model. However with the recent emergence of big data and the advancements in computational power, it has become increasingly likely that these models are too sparse to capture an accurate amount of risk premium association using only a few linear factors.

We begin our analysis in similar fashion to the work of Rapach et al. [1] by testing several machine learning (ML) models on lagged returns data of 30 industries. Our goal is to study the interactions of these industries and add in supplementary factors which we believe will help capture an asset's risk premium. Leaning on the work of Borisenko [2] and Gu et al. [3] we find that some of the most influential factors in pricing asset return premia include return momentum and market state features.

We follow this by running comparative analysis of our models across different datasets, in an effort to examine how the degree of granularity in the market affects the predictability of each model. As a benchmark we compare the forecasting errors of these models to simple statistical methods found in the Madridakis [4] M4 competition paper. We find that as the level of granularity increases, the need for more complex and flexible models becomes ever more important.

Finally, we explore different methods to improve upon these models. One approach changes the problem setting from a regression to a classification approach, while another method we explore is an ensemble approach inspired by Monetro-Mason et al [5] in their feature-based forecast model averaging paper.

We will elaborate on our analysis in two separate settings; the first is a purely modelling approach where we examine the forecasting error of our models and the latter is in a portfolio scenario where we determine the investment implications of each model's predictive accuracy. The outline for this paper is as follows: we given a brief review of the literature in this area and our motivations upon reading them. We then give a formal review of all methodology used for

the machine learning applications. We discuss the empirical study performed in our work. We then elaborate on our results and offer descriptive analysis. We finish up with our conclusions and set the stage for further research.

2 Literature Review

We divide our review on existing literature into two components. First we explore the different ML models used in the area and second we dive into these models and examine the space of potential viable features to used for predictive regression.

2.1 Model Selection

Recently, due to the rush of big data in the financial markets, machine learning has becoming increasingly popular. We highlight the related literature that motivated our research. Weigand [6] reviews and discusses several approaches most commonly used in empirical asset pricing. Using this as a guideline we dive deeper and break our research down into categorizing different types of models.

One of the most popular approaches used in machine learning is to use regularization techniques to improve a model. Gu et al [3] deploys a penalized linear model in the form of Elastic Net Linear Regression. The Elastic Net function can be described as a combination of Lasso and Ridge penalties. In Rapach et al [1] the authors used an interesting procedure which they named "OLS post-LASSO" (Ordinary Least Squares post LASSO). This procedure uses Lasso regression in the variable selection process. Once the dimensionality of the data has been reduced, Rapach then utilizes regular OLS regression on only the selected variables.

In addition to linear models, more flexible models have been also used in recent years. One subset of these models found in literature is tree-based methods. Gu et al [3] use regression trees and random forests in their work. The Random Forest approach can be considered a bagging technique, which is also a form of regularization. Decision trees have a tendency to overfit their data, a phenomenon that arises when the model is too dependent on its original data. Boosting is another form of regularization, and can be applied to decision trees. Leiw and Mayster [7] also use Random Forests (RF) in their analysis when they attempt to forecast exchange-traded funds (ETFs). Their approach was used in a classification setting but can still be applied to a regression-based approach - which is the main focus of our research.

We also find in recent literature that the use of Neural Networks (NN) is very common throughout the field. This is not surprising as the latest advancements in computation power has driven the field of deep learning to massive popularity in the artificial intelligence community. Feng et al [8] use deep learning on a variety of financial fundamental data. This data is often found in an asset's quarterly and annual filings. They attempt to use these factors to predict an asset's excess returns. Gu et al [3] use a very similar approach but apply it to a broader set of factors. Additionally, Leiw and Mayster [7] use their NNs in a classification setting and Borisenko [2] relates a very interesting approach by using only factors based on an asset's return series.

Stepping away from the financial world, we scanned forecasting literature for insights on potential new ideas not fully explored in asset pricing. Makridakis [4] provides an analysis of the M4 forecasting competition. In this work we find one approach very intriguing and do further research into the creator's work. Montero-Manson [5] performs a variation of a common stacking ensemble that creates a weighted average of several statistical forecasting methods using a ML meta-learner. We incorporate learnings from this approach in our project.

2.2 Feature Selection

There is a wealth of literature that attempts to predict asset returns, particularly focused on equities. There are two basic approaches: the time-series approach and the cross-sectional approach. The time-series approach typically regresses portfolio-level returns on macro-economic variables to determine drivers of returns. The second approach runs cross-sectional regressions of future returns on lagged predictor variables. We follow the latter approach.

In the equities context, the predictor variables used in the regression equation are often a mix of fundamental accounting ratios, such as Earnings-to-Price, Book-to-Market, or Return on Assets. The cross-sectional approach measures a stock's financial ratio, and then compares it against the same metric across its peer universe. This comparison is performed for each time slice. The metric is often normalized to allow for proper comparison and to remove outliers. The general thesis behind this approach is that in the cross-section, firms that have more favorable metrics should have higher returns.

There is a whole subset of literature that focuses on another brand of predictors, those determined by prior returns of the asset in question. These include popular momentum indicators, which is simply the cumulative return of the asset over a certain period. There are other indicators such as beta, idiosyncratic risk, and alpha which are a decomposition of the CAPM model comparing the asset's return to a market benchmark. There is actually significant empirical evidence that some of these returns-based measures are sufficient as an investment style on their, without needing to include fundamental characteristics

long thought to be the primary drivers of stock returns.

The presence of the momentum phenomena has been studied across asset classes such as commodities, and seen in the work of Mifre et al [9]. This paper finds that momentum returns in commodities can be related to an economic rationale found in the work of Keynes (1930) and Hicks (1939) in their theory of normal backwardation. Implementation of commodity strategies is inherently limited to a few assets in question as opposed to thousands in the equities space. Commodities also do not face the restrictions of short selling found in equities, making implementation easier.

In recent years there has been a huge focus by quants and others seeking out the best predictors of asset return premia. This area has resulted in hundreds of indicators used in published research. Gu et al [3] studies this 'factor zoo', and concludes that the majority of these indicators have not been able to provide consistent risk-adjusted returns, either by themselves or in the context of a portfolio. One reason for this is that many of these variables have been developed from linear relationships, while in reality a linear approach is not sufficient to map the true behavior of a financial instrument and the features that drive it.

We chose to focus our work on returns-based indicators for two primary reasons. First, it is generally easier to compile information across asset classes when the only input needed is price. Fundamental data on stocks is notoriously tough to assemble, as it requires significant pre-processing and error checking. In addition to it being easy to gather and proces, Gu et al [3] work showed that returns-based predictors have actually been the most reliable predictors for an asset's return return over time. While there were several pieces of academic research that informed our approach, three papers primarily influenced the direction for our work.

In Rapch et al [1], the author's work explored how improvements could be made on simple linear models when using lagged returns as predictors. It used returns data from 30 industry portfolios and attempted to predict future returns in each industry by using lagged returns of the other industries. The economic premise was simple but reasonable. For instance, the authors argued that the behavior of industries such as banks should proceed the behavior of other industries such as who rely on banks to finance their activities. Returns of commodity industries should inform the future returns of industries that rely on commodities as inputs, such as food and beverage industries.

Other papers we reviewed focused on how using non-linear models could improve on return prediction. Gu et al [3] conducts an in depth study on how neural networks accommodates a more expansive list of predictor variables and allows advancements in the frontier of risk premia measurement.

Rapch et al [1] provided us with the initial inspiration for the foundation of our project. Gu et al [3] pointed us in the direction of using neural networks. Borisenko [2] provided us with an idea about the data and features that we wanted to investigate. He took a similar approach to Gu et al, but delved deeper into returns-based indicators. He focused on using momentum predictors from varying time periods, and also used alpha, beta, and idiosyncratic volatility as features. Interestingly he departed from the norm in his approach to predicting returns by using a classification method as opposed to regression. We tested both approaches and compared their successes and shortcomings.

As we surveyed the literature, we addressed several questions which form the thesis of our work. The questions are as follows: 1)What are the current ML models being used in this area?, 2)Why are these methods performing better (worse) than classical models?, 3)Does the level of data granularity improve (hinder) a model's performance?, 4)Is data overfitting limiting ML's contribution or is there further improvement to be made by incorporating additional features?, and 5)How can we improve upon these models and their limitations?

3 Methodology

This section describes the theoretical framework of the several machine learning models being used throughout our analysis. In addition to these models, we will also introduce several simple statistical forecasting models which serve as a benchmark for our results. In the context of our approach, each model is attempting to capture the interactions between their respective features. In the most general form we can describe our returns for an asset to be its expected returns with an additional error.

$$r_{t+1} = \mathbb{E}(r_{t+1}) + \epsilon_{t+1} \tag{2}$$

Leaning on this we can further describe the expectation to be generalized to some function

$$\mathbb{E}(r_{t+1}) = f(x_t) \tag{3}$$

Using the above formula we assume our function to be a model containing an asset's respective predictors pertaining information only up to time t. This is a crucial step since we now have a function explaining an assets return at time $(t+1)$ independent of all information at needed at time $(t+1)$. Each of the subsequent subsections elaborate of the various models being used throughout our research.

3.1 Linear Model with Regularization

Simple linear regression using ordinary least squares (OLS) takes the form in our context

$$r_{t+1} = \mathbf{X}_t \beta_t + \epsilon_{t+1} \tag{4}$$

where $\mathbf{X}_t$ is our predictor variable matrix containing information up to time t, β_t is our vector of weights and ϵ_{t+1} is our remaining residual. Our objective function is to find weights for each of the predictor variables which minimize our loss function.

$$\mathcal{L}(\beta) = \frac{1}{N}\sum_{i=1}^{N}(r_{t+1} - \mathbf{X}_t\beta_t)^2 \tag{5}$$

Given the nature of financial time series, we run into several issues when performing basic linear regression. One major issue deals with the tendency for OLS to overfit the data due to a high noise-to-signal ratio. To combat this we improve on our simple OLS model to add regularization terms to it. The model with which we had the most success is Elastic Net, which is a combination of the two different types of penalty terms of LASSO and Ridge regression.

The LASSO regression is a penalty term added to the loss function of the OLS equation. This term is a regularization in the L^1 norm which adds the absolute value of the variable weights. The Ridge regression is a term also added to the loss function of the OLS. This penalty is in the L^2 norm which adds the squared value of the variable weights. Both of these regularizations have different purposes. The LASSO and Ridge penalties promote different scenarios as the importance of the penalty increases. The LASSO leads to a sparse model since it forces the weights to zero. The Ridge penalty has a different effect, resulting in shrinkage as the weights become very small but not actually zero. The formulae for this model's loss function is as follows:

$$\mathcal{L}(\beta) = \frac{1}{N}\sum_{i=1}^{N}(r_{t+1} - \mathbf{X}_t\beta_t)^2 + \lambda\left(\frac{1-\alpha}{2}\sum_{j=1}^{P}\beta_j^2 + \alpha\sum_{j=1}^{P}|\beta_j|\right) \tag{6}$$

where λ is the loss function penalty parameter and α is the amount proportioned to the LASSO penalty. We can clearly see from the special cases of $\lambda = 1$ the function becomes a pure LASSO regression and $\lambda = 0$ the function becomes a pure Ridge regression.

3.2 Decision Trees and Random Forests

Among the various tree-based methods at our disposal, we chose to work with decision trees and random forests. These methods are different from linear models since they are nonparametric, meaning they allow for significantly more flexibility in the model as compared to a linear parametric approach. In our attempt to model an asset's expected returns, the first model we deploy is a regression tree. The regression tree works by partitioning our data into subgroups where the observations have similar values. Using these partition points the model can make future predictions by evaluating in which partition the new data belongs.

Random Forests (RF) are classified as an ensemble method, since they combines many decision trees together in a model. Singular decision trees are often very sensitive to hyperparameters and as a result often are prone to overfitting. A way we can address this is through random forests. RFs use a technique known as bootstrap aggregation or "bagging". This approach creates an averaging of the values found from the different single trees, creating a more robust and stable prediction.

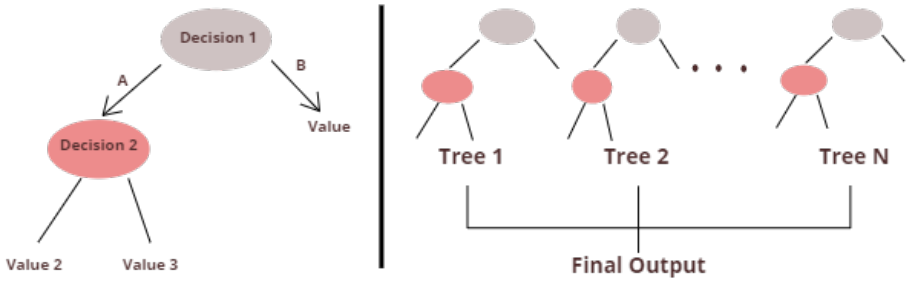

Figure 1: Decision Trees and Random Forest
Note: The diagram on the left is the workflow of a decision tree and the diagram on the right is the workflow of a random forest

3.3 Neural Networks

In recent years there have been many advancements in the field of computer science, specifically in terms of computational power. With the introduction of powerful graphics processing units (GPUs), deep learning concepts can now be tested more easily. As a result, the popularity of these models has grown tremendously. For our research we implement a fully connected feed-forward network.

A feed-forward network consists of three pieces, the input layer, a group of the hidden layers, and the output layer. The input layer receives the data and through some transformation, which is commonly referred to as an "activation function". This function is pushed sequential through the hidden layers, hence its name "feed-forward". The outputs of each hidden layer become the inputs of the next layer before they reach the output layer. The reason our neural network is also titled "fully connected" deals with the fact that every node in a layer is connected to every node in the subsequent layer.

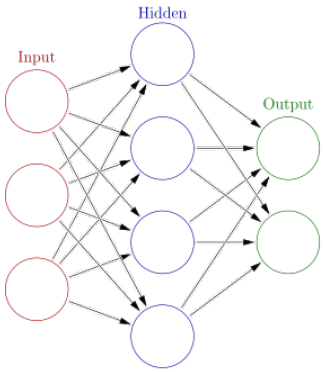

Figure 2: Feed Forward Network
Source: https://upload.wikimedia.org/wikipedia/commons/thumb/4/46/Colored_neural_network.svg/280px-Colored_neural_network.svg.png

We test several different types of network architectures but primarily use the rectified linear unit activation function (ReLU) for all hidden layers. The activation function be described as the following.

$$f(x) = max(x, 0) \tag{7}$$

3.4 Stacking Ensemble

In addition to the several commonly used ML models we also test a method which seems to be attracting popularity in recent years. This method can be described as a stacking ensemble, which in simple terms is a stacking of several machine learning models together to form a weighted average of these methods. We added a unique flavor to this traditional method by allowing the model to make predictions using either machine learning or simple statistical forecasting methods.

As depicted in the diagram in the appendix, our original data is fed into several forecasting methods, or "base learners". These methods are then used to create predictions. These predictions are then used as features to train another machine learning, "meta-learner", which will then give us a weighted average of the base learners' predictions.

3.5 Performance Evaluation

We chose to define our performance evaluation by two methods. The first is a more formal approach of computing an error metric for our forecasts. In financial literature, it is very common to see a metric R^2_{OOS} or R-squared (out of sample) to measure the performance of the model's forecasts. However, we

did not feel like this metric was sufficient in evaluating the power of a model. This is in part due to the intrinsic nature of financial time series, which have high noise-to-signal ratio. We therefore scoured time series forecasting literature to find more suitable metrics. The M4 forecasting competition uses symmetric mean absolute percentage error (sMAPE), mean absolute scaled error (MASE), and a combination of those two metrics to evaluate a forecast's accuracy. We use these two metrics and add in a third commonly used metric. The MSE, sMAPE, and MASE and their associated formulae can be expressed as follows:

$$MSE = \frac{1}{T}\sum_{t=1}^{T}(y_t - \hat{y}_t)^2 \tag{8}$$

$$sMAPE = 100\%\frac{2}{T}\sum_{t=1}^{T}\frac{|y_t - \hat{y}_t|}{|y_t| + |\hat{y}_t|} \tag{9}$$

$$MASE = \frac{\sum_{t=1}^{T}|Y_t - \hat{Y}_t|}{\frac{1}{T-1}\sum_{t=2}^{T}|Y_t - Y_{t-1}|} \tag{10}$$

The second method we choose is a comparative approach to simple statistical methods commonly used in forecasting. We record the error metrics gained by these methods and compare them to our ML models to gain a more intuitive sense of our error values. These adjusted errors can be thought of as the increase (decrease) in predictive accuracy gained (lost) by deploying a machine learning model for prediction. The simple models we use are a Naïve, Drift, and Prevailing Mean (PM) methods. Their forecasting functions are as follows noting that h is the forecast horizon:

$$Naïve = \hat{y}_{T+h|T} = y_T \tag{11}$$

$$Drift = \hat{y}_{T+h|T} = y_T * h\frac{y_T - y_1}{T-1} \tag{12}$$

$$PM = \hat{y}_{T+h|T} = \frac{1}{T}\sum_{t=1}^{T} y_t \tag{13}$$

4 Application: An Empirical Study

4.1 Data

We used three subsets of data for this project, adding each sequentially as we increased the complexity of the project. We used 30 Industry value-weighted portfolio monthly returns from December 1959 to December 2019. We sourced the data from the Kenneth French's website. We then added monthly returns for 17 commodity and 7 currency futures contracts, using a continuous futures

price to account for changes in contract month. The data was sourced from Bloomberg.

Finally, we expanded our analysis to a much broader universe and collected monthly returns data for common stocks from 1970 – 2019. The data was sourced from CRSP. We also obtain the Treasury-bill rate to proxy for the risk-free rate from which we calculate individual excess returns. We used the value-weighted index found on Kenneth French's library as a proxy for the market index. We used this index to compute various statistics such as beta and alpha on individual securities. Finally, we used the SP 500 index and several Fama-French indexes as benchmarks.

One time-consuming portion of this project was collecting, cleaning, and warehousing all of the data, which we did in a SQL database. We filtered the vast universe of stock prices through to proxies, market cap and price, selecting the largest 100/200/500 in market cap and stocks trading above 5 dollars per share. This reduces the possibility of mispricing and deals with the issue of missing data.

Lagged returns were the simplest returns predictors we tested. The basic premise is that information diffuses slowly across the investor base, and that events occurring in one industry may not be immediately felt in other industries that are linked economically. As an example firms in many industries rely on the financial industry for financing. When the financial industry experiences a positive environment, the industries with which they interact often do as well. Financial firms become more willing to provide credit on favorable terms, and their borrowers benefit as do the customers of the borrower. In the presence of information friction, one could expect lagged financial industry returns to positively affect future returns in many other industries.

The results from using simple lagged returns as predictors were decent, but we wanted to see what else we could add to improve our models. We then turned to Borisenko [2] and others who used other slightly more sophisticated returns-based predictors. Specifically, we computed measures of momentum across each asset in question. This was done on a rolling basis and across several different timeframes. In this manner we span several of the momentum indicators tested across literature. Jegadeesh et al [10] researches the 1m reversal factor. Asness et al [11] covers the 12-1m momentum factor, one that is commonly used across many standard industry risk models. In simple terms, momentum is a measure of cumulative returns over a period of months, linked geometrically.

We then added in as predictor variables alpha, beta, and idiosyncratic volatility. These were used in tandem in Borisenko [2], but studied frequently across literature. Daniel et al [12] documents the ability of beta to predict returns in the context of momentum. Hühn et al [13] documents the ability of alpha. Idiosyncratic volatility is covered extensively by Ang et al [14]. In simple terms,

we ran an OLS regression of the returns of each asset in question against Fama French's Value-weighted index. We then returned the coefficients from that equation as betas, the intercept as alphas, and the error as idiosyncratic return. We did so on a rolling basis and as in the case of momentum over several different timeframes. While we tested various timeframes for each variable, we mainly focused on timeframes between 1 and 24 months.

In addition, we computed a type of momentum that removed the most recent month's return, in order to remove the aforementioned reversal effect. However, the portfolio-level results did not markedly improve with the inclusion of this variable.

At the start of the project, we recognized the need for the addition of some type of variable to define the state of the market as we supposed that this would directly affect the returns of the strategy. Borisenko [2] provided us with the idea to add a 'market state' variable, which computed returns and volatility of the market and added them into the model directly as predictor variables. He thoroughly evaluated the relationship between the performance of various predictor variables and the state of the market. In addition to using this variable, we also added the VIX index as another measure of market stress. We noticed that the addition of these variables had a positive effect on the performance of the model.

It is critical to clarify a point on feature normalization. Our predictions are based on an asset's score for a particular predictor versus the rest of the cross-section for that month. This means that for each asset, each month we compute a standardized score of that predictor variable vs. all the rest of the assets that month. The basic thesis is that if we suppose that stocks with high momentum outperform stocks with low momentum, then stocks with a high 12-month momentum -score during a particular month should be added to the long leg of the portfolio, and those with low 12-month momentum -score should be added to the short leg of the portfolio. Of course the model does not presuppose these relationships, but rather learns them in the course of predicting returns.

4.2 Backtesting / Portfolio Construction

We first outline the iterative process of our forecasting method and then move to our portfolio construction. For a given dataset we allocate all given information for that asset up to time t exclusive. This is used as our training data which we feed into our ML model. For our labels we feed it the one month ahead returns. This allows use to train our model to regression last months predictor variables to the future month's return. Once our model is fitted to our training data we then feed it our information at time t which is used to make a forecasting for time $t + 1$. We do this for every asset for a given month. We then step for our time and repeat the process. However this time the original month at time t is now indexed as $t - 1$ and included inside the training data.

The new month t is the month we originally forecasted, but instead we give it the asset's actual returns to forecast the next month. This process is run through the length of our backtesting period.

Once we have built all the forecasted returns we can sort them month by month. We catalog the forecasted top performers and the bottom performers separately. We then implement our portfolio strategy. Our strategy is an equally-weighted long and short portfolio. Using this strategy we buy or go long the top performers of the month and sell or go short the bottom performers. We reset our portfolio every month. For the majority of our research we give our model an initial training period of 10 years or 120 months of data to start our backtest. For our ensemble method we give it 20 years so it can use 10 years to train the base learners and then 10 years to train the meta learner. We also create quintile sorted portfolios for the majority or our research. This means we go long the top 20% of assets in our universe and short the bottom 20% of assets.

Admittedly, our approach is naïve as well. At times gains might come primarily from the short leg, and one could improve portfolio performance by eliminating the long leg altogether. We highlight this behavior later in the paper. In addition, market conditions likely have a significant impact on the behavior of the portfolio. The point is that a static approach to portfolio construction is not robust.

4.3 Model Tuning

With linear regression, you get a straight line that fits (often poorly) the data. As you go non-linear you get curvier lines that try to fit the data. This leads to the well-documented bias-variance tradeoff. As you add parameters and non-linearity model predictability improves and variance is reduced. There is a gradient at which overfitting occurs and the model's bias is too high.

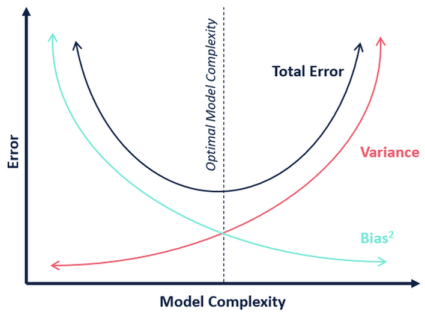

Figure 3: BiasVariance Tradeoff
Source: https://files.ai-pool.com/a/c457c503205d0740d3efc553bdb74b0b.png

With machine learning algorithms there are often ways to tune the degree of non-linearity and in the case of penalized linear models in selecting the best predictor variables. This is where cross-validation comes into play and is where a researcher selects the model with the best fit. In non-financial settings, typical convention is to split the data 60/20/20, where 60% is used for training, 20% for testing, and 20% is randomly picked for cross validation. However, with time series data, one must consider serial correlation. When serial correlation is present, information is lost when cross validation is done at random. To avoid this issue, we trained an expanding window of data, then predicted the next window sequentially. We conducted cross-validation by testing the performance of hyperparameters over each model run.

Several of the papers we reviewed gave a nod to the importance of selecting hyperparameters when running various models, in order to optimize performance. We tested various combinations in a limited sense, but did not make this a major focus of our work. As with comparing the performance of machine learning models however, we did find that certain hyperparameters produced consistently better results. Our approach was to run the models iteratively across a combination of hyperparameters to determine which ones did the best job predicting. When we used LASSO, we noticed that a smaller regularization penalty worked better, presumably as it allowed more explanatory variables. When combining the approaches of LASSO and Ridge in an Elastic Net model, we found that models which used slightly higher regularization penalties but had a smaller total penalty from LASSO performed better. While improvements could be gained from further hyperparameter optimization, we believe that the nature of financial data imposes a ceiling on how good the results can be from optimizing parameters.

5 Results and Analysis

We tested the application of ML models on a broad range of datasets and theories in question. We have attempted to separate our results into subsections to give the reader the best chance in absorbing the information pertaining to each individual question. In addition to the statistical results we also provided analysis and intuition on these numbers in an effort to bring the theory behind the numbers to light.

5.1 Foundational Analysis

We begin our results section with the outcome of using lagged returns as predictor variables for other industries. We tested several different types of ML models over the same predictor variables. We found that results were interesting

but unimpressive for ML models when using lagged returns. MSE's were in the neighborhood of 40, and the highest Sharpe ratio was around 0.5.

Table 1: Industry Lagged Returns Models

Model	**MSE**	**Sharpe**	**Mean Return**
OLS	43.00	0.50	0.03
LASSO	40.01	0.52	0.03
Elastic Net($\alpha = 1$, L^1 ratio $= 0.1$)	40.26	0.58	0.04

As the table shows, Elastic net was one of the best performing models. Using this model as our test case, we turned to finding the best hyperparameter combinations for this model. We generally found that a smaller regularization penalty worked better, presumably as it did not overshrink the model's coefficients.

Table 2: Enet Hyperparameter Optimization

Alpha	**L^1 Ratio**	**MSE**	**Sharpe**	**Returns**
1	0.01	39.7	0.08	0.00
1	0.9	39.1	0.23	0.01
3	0.01	39.1	0.15	0.01
3	0.9	38.7	0.01	0.00
10	0.01	38.8	0.17	0.01
10	0.9	38.8	-0.22	-0.01
100	0.01	38.8	-0.08	-0.01
100	0.9	38.8	-0.22	-0.01

5.2 Comparative Analysis of Forecasting Errors

In an effort to build a greater intuition of our forecasting errors we decided to use an approach similar to the M4 competition. We built adjusted errors based on a Naïve error scale. Simply put we divide our model's MSEs by the MSE obtained from the Naïve approach. This allows us to analyze how much better (worse) our prediction is then a simple statistical forecast. We also include an Auto Arima method to see if ML models can beat a popular time series modelling approach.

We can clearly see that implementing a more sophisticated approach than Naïve yields better results across the board, except in the case of one model. This is a positive result eluding to the finding that ML models offer additional

information. However, when comparing them to Auto Arima models it seems that they might be limited in how much information they can extract. Using this we look to further improve our models by adding additional information.

Table 3: Naïve Adjusted Errors

Stocks			**Industries**		
Model	**MSE**	**Adjusted Error**	**Model**	**MSE**	**Adjusted Error**
Naïve	76.96	1	Naïve	62.01	1
AutoArima	37.22	0.48	AutoArima	30.07	0.48
Enet(3, 0.9)	37.58	0.49	Enet(1, 0.5)	31.49	0.51
Decision Tree	70.47	1.14	Decision Tree	39.75	0.52
Neural Network	29.16	0.38			

5.3 Returns-Based Features

We then sought to improve our results by adding factors to our model. Our idea was that lagged returns were not sufficient as predictor variables alone and that including additional features would help our models capture more information on the asset's return. We add return based characteristics as predictor variables to our industry dataset.

Table 4: Adding Return Based Characteristics to Industries

Model	**MSE**	**Sharpe**	**Mean Return**
Highest Enet	39.92	0.54	0.03
Elastic Net($\alpha = 1$, L^1 ratio $= 0.1$)	39.99	0.36	0.04
Bagging Regressor	46.63	0.31	0.01

Unfortunately, our results were not a statistically significant improvement over our original less-complex model. Not only were these results disappointing, they were also surprising since it is well document in the literature that these characteristics usually offer better results. We then decided to test different types of data and see if giving the model more granular data might offer better results.

5.3.1 Different Asset Classes

We retrieved returns data for stocks from different periods, beginning in 1980 and testing each following decade with the largest stocks that were present at the start of the decade and were actively trading throughout the entire sampling period. At the beginning of the 1980s, there were slightly over 300 tickers that met our criteria. Rather than re-scanning the available universe each month, we ran the model on these stocks over the period of testing. We computed momentum and the other aforementioned returns-based measures for each of

these stocks. We then ran the model for the period from 1980 through 2019. We repeated this approach for 1990 through 2019, and 2000 through 2019.

Table 5: Model Runs for Various Stock Periods

Dataset	**MSE**	**Sharpe**	**Returns**
Stocks:1980s	65.42	0.43	0.02
Stocks:1990s	59.74	0.21	0.01
Stocks:2000s	44.00	0.41	0.01

Again, we were surprised to find that there was not much improvement over the results from industries. MSE was actually worse, as were portfolio results. While we feel that our sampling approach gathered a sufficient universe, it is possible that the high volatility on an individual asset makes it difficult to predict. Perhaps more rigorous attempt at resampling stocks each month could also provide better results.

We achieved more encouraging portfolio results when we extended the analysis to other asset classes. It appears that momentum variables are more viable when used to predict future returns in a portfolio of commodity and currency futures. Perhaps these markets are less efficient or perhaps it is due to the significant presence of commodity trading advisors (CTA) in the investment base. These investors notoriously incorporate trend following strategies into their approach, suggesting that momentum is an important factor in these markets.

We noticed attractive MSEs in certain commodities, notably currencies and less volatile commodities such as gold. In others, the variance was much greater. The overall MSE of a portfolio of 17 commodities and 7 currencies was actually worse than the MSE of a portfolio composed of industries or stocks. Yet portfolio returns were better, highlighting the conundrum of the two competing objective functions of MSE and Sharpe. When combining commodities and industries together in a cross-asset portfolio, we saw portfolio performance improve into the realm of a viable strategy. This behavior was consistent across models we tested. So the question is why. While more work needs to be done on this front, we suspect that the better results are due at least in part to diversification. Commodities have been well-documented as an excellent portfolio diversification tool, providing an inflation hedge and low correlation at the portfolio level. Indeed, constructing a portfolio with assets of varying risk and return characteristics improves risk-adjusted return over market cycles.

We also see that the inclusion of a market state variable aided in model's predictability and boosted returns, albeit the effect was marginal and dependent on the period tested. The results are intuitive and can be interpreted as follows. Different investment styles work in different market regimes. For example, when

Table 6: Cross-Asset Approach

Dataset	**Model**	**MSE**	**Sharpe**	**Returns**
Commodities	Enet(1,0.01)	59.93	0.74	0.08
Commod + Ind	Enet(1,0.01)	48.07	0.86	0.05

market volatility is low, stocks with higher long term momentum scores generally exhibit higher returns. When market volatility is high, stocks with favorable shorter-term momentum scores tend to better detect changing market dynamics. This behavior is well documented across financial literature, and explored in depth in the work of Borisenko [2]. We believe that the model is able to use the knowledge of market state to help it inform its prediction of expected returns. Our results show that holding all else equal - this is indeed the case.

Table 7: Addition of Market State Features

Predictors	**MSE**	**Sharpe**	**Returns**
Momentum	56.82	-0.082	-0.006
Momentum + Mkt State	56.78	0.001	0.002
Momentum + Vix	57.94	0.224	0.008

5.4 Different ML Methods

In the spirit of Borisenko [2], we changed our problem from one of regression to one of classification. Where with a regression approach one tries to directly estimate excess returns, a classification approach determines whether a stock is above or below the median return for that month. Rather than output an expected return, classification returns the probability of the return being above or below the median. In this approach, the estimated probability is directly proportional to the expected return. A classification approach has a few advantages, one being that the labels have the same distribution over time and same magnitude. This can serve to alleviate the problem of time-varying cross-sectional dispersion in returns. This approach saw slight but consistent gains in the data we tested.

In addition to changing the modelling approach we also implemented a stacking ensemble method. While stacking ensembles appear to be the only "free lunch" in modelling, many time series stacking ensembles do not statistically outperform because the residuals from each of its base learners are extremely correlated with each other. Therefore to combat this behavior we also fed our model several statistical forecasting methods in hopes to reduce the multicollinearity effect. Our results were very pleasing.

Table 8: Classification vs. Regression

Dataset	Prediction Method	MSE	Sharpe	Returns
Industry	Regression	42.8	0.5	0.02
Industry	Classification	40.0	0.3	0.02
Stocks	Regression	44.0	0.4	0.01
Stocks	Classification	43.9	0.5	0.02
Commod + Ind	Regression	39.5	1.2	0.07
Commod + Ind	Classification	39.4	1.5	0.08

Table 9: Ensemble Methods vs. Statistical Methods

Stocks				**Industries**			
Model	**MSE**	**Adjusted Error**	**Sharpe**	**Model**	**MSE**	**Adjusted Error**	**Sharpe**
Naive	76.96	1	-0.56	Naive	62.01	1	-0.05
AutoArima	37.22	0.48	0.46	AutoArima	30.07	0.48	0.19
Ensemble 1	16.53	0.21	1.13	Ensemble 1	13.59	0.22	1.20
Ensemble 2	16.81	0.22	0.82	Ensemble 2	13.24	0.21	0.99
Ensemble 3	16.93	0.22	0.76	Ensemble 3	13.46	0.97	0.22

Table 10: Portfolios 2010-2017 with all Predictor Variables

	Mean Return	Ann. Volatility	Sharpe
S&P 500	12.0%	11.8%	1.08
Stocks	0.8%	3.9%	0.22
Industry	-0.2%	4.5%	-0.01
Ensemble 1 (Stocks)	6.9%	6.4%	1.11
Ensemble 2 (Stocks)	5.9%	7.7%	0.80
Ensemble 3 (Stocks)	4.7%	6.8%	0.74
Ensemble 1 (Ind)	9.1%	7.9%	1.19
Ensemble 2 (Ind)	7.2%	7.4%	1.01
Ensemble 3 (Ind)	7.4%	8.1%	0.96

Out of the numerous models tested the ensemble proved to be one of the few able to reduce MSE and increase Sharpe ratios. This was a huge breakthrough since many models were not able to improve on the forecasting and portfolio measures simultaneously.

5.5 Benchmark Analysis

Investors seek out a strategy that can provide them with reliable capital appreciation potential throughout all market cycles. While one could passively invest in the market as a whole or some subset of the market via ETFs, a whole industry is dedicated to providing products that are designed to outperform a passive strategy. These products are generally benchmarked to an index with

similar characteristics to a passive vehicle in which the investor could invest as an alternative. Active managers put significant emphasis on their ability to provide better returns than a benchmark. Investors are not only concerned about relative performance, but also about the volatility they experience. Volatility is often measured in standard deviation terms. The former represents the numerator in the Sharpe ratio, and the latter the denominator. In a similar spirit, we compare the performance of our strategy to the SPY, an exchange-traded fund that mimics the broader market. We also compare it against the Fama French momentum index.

Table 11: 2002-2019; Momentum & Market State Predictors

	Mean Return	**Ann. Volatility**	**Sharpe**
S&P 500	4.7%	14.0%	0.41
FF-Mom	-1.3%	15.9%	-0.02
Industry	-0.4%	5.3%	-0.05
Commodity + Industry	5.0%	6.4%	0.81

Our best strategies do indeed provide better risk-adjusted returns than both benchmarks, due in part to relative performance but also in terms of relative risk. While we ran our models over several periods and with many variations, there were many instances in which they outperformed the broader market. This is a compelling result.

5.6 Market Regime Analysis

We have highlighted how the model's predictability falters some during periods of high stress. Despite this shortcoming, our investment strategy still performs well over time. We captured the returns from the models above during the 2008 financial crisis. Our results show that during a period when the S&P declined over 30%, our strategy of combining Commodities and Industries lost just shy of 7%, and realized a volatility of less than a third of the broader market.

Table 12: July 2007 - Feb 2009; Momentum & Market State Predictors

	Mean Return	**Ann. Volatility**	**Sharpe**
S&P 500	-36.3%	18.7%	-1.88
FF-Mom	23.1%	17.2%	1.42
Industry	3.3%	6.3%	0.56
Commodity + Industry	-8.0%	5.5%	-1.44

It is also interesting and important to dissect the returns during the period after the 2008 crisis when the market rebounded sharply. Notice that while the

broader equity market rebounded sharply, momentum returns as represented by Fama French's index saw sharp drawdowns.

Table 13: March 2009 - Feb 2010; Momentum & Market State Predictors

	Mean Return	**Ann. Volatility**	**Sharpe**
S&P 500	50.1%	13.4%	3.82
FF-Mom	-54.4%	36.0%	-1.40
Industry	-6.9%	9.0%	-0.73
Commodity + Industry	17.3%	8.5%	2.08

Momentum crashes are documented throughout financial literature. It is important to understand that the industry-standard momentum index is comprised of stocks that have performed well over a trailing period. In the instance of 12-1 month momentum, the highest bucket of the index represents stocks that have experienced large cumulative returns over the prior 12 months, leaving out the latest month. Borisenko [2] provided interesting insight into why this occurred. He relates how during depressed markets, this standard momentum strategy accumulates substantial negative and asymmetric market beta which leads to a crash when the market rebounds. What this means is that these momentum indexes are comprised of low beta stocks, which have recently outperformed high beta stocks. When that trend reverses, the momentum strategy that goes long top momentum stocks (low beta) and short low momentum stocks (high beta) experiences crashes. The Fama-French indexes and others with similar methodology are not designed to account for this changing dynamic. This is where the market state variable comes in. Recall that we use several timeframes to measure momentum. In a period of volatility, a shorter-term measure of momentum would be able to better capture changing market dynamics. By using information about the time-varying characteristics of momentum, the model is able to adjust to predict returns based on which type of momentum should perform better. This is readily evident in the results table above.

5.7 Feature Importance Analysis

Neural networks are at times criticized for their black box nature and lack of interpretability. Several of the papers we reviewed offer ways to interpret variables. Borisenko [2] used gradients and Hessians of estimated probabilities with respect to the input variables to measure the drivers of prediction. By doing this he was able to provide very detailed descriptions of which horizons of momentum (or other features) were high and low predictors of expected return. He also highlighted the extent to which these variables are modulated by market volatility. He was able to show that the predictive power of longer-term measures falls when market volatility rises.

A crude approach to measure variable importance is to run the model with

a certain set of features, tally the results and then run the model again with an additional feature and note the difference in result. We did this at times, in particular noting the positive effect that adding a market state variable had on our results. This approach is oversimplified however, as it provides little color on which predictions had a positive or negative relationship with return. It also provides no insight into the time-varying nature of predictor variables.

For our penalized linear models, we had the ability to extract the coefficients – for each asset, for each predictor variable, for each date. This provided an interesting look into what variables drove the model. Although we conducted this analysis on a limited basis and only on industries, we found that the predictor variables displayed a significant amount of time variation. That is to say that a predictor variable's coefficient was not static but modulated throughout time. Each asset also had a different coefficient value for each predictor. For example with the Beer industry, the 1-month momentum score had a positive coefficient value. This could be interpreted as follows: when Beer had high short-term momentum vs. the rest of the industries, it is likely that the model predicted it would have a high return. The opposite was true for the Smoking products industry.

We took a simple average of predictor coefficients across industries over time and what we found generally corroborated the findings of the aforementioned papers. On average across industries, short-term momentum had a negative coefficient. This could be loosely interpreted as the reversal factor, where assets with high short-term return tend to mean revert. Conversely, 12-month momentum had a positive coefficient on average. This supports the general thesis that firms with positive long term momentum often continue to generate high returns.

In addition we noticed behavior that coincided neatly with the aforementioned momentum crash that happened in 2009. Long-term momentum peaked as a positive coefficient in March 2009, which we know was a long-term market bottom after the Great Recession. The coefficient then 'crashed' to around zero over the next 7 months. Presumably this was due to a higher composition of lower-beta stocks leading into the market bottom. The behavior of short-term momentum was even more extreme. The coefficient peaked in late 2008 in positive territory, which is counter to the traditional mean-reverting behavior and negative coefficient generally seen. It then crashed hard over the next three years to deeply negative territory, reverting back to its normal mean-reverting behavior. The reason for the crash is presumably the same as for the crash in long-term momentum but on a magnified scale. Low beta stocks underperformed dramatically as the market rotated into cyclical stocks. As the market continued to go higher, investors rotated across stocks within industries and across industries themselves.

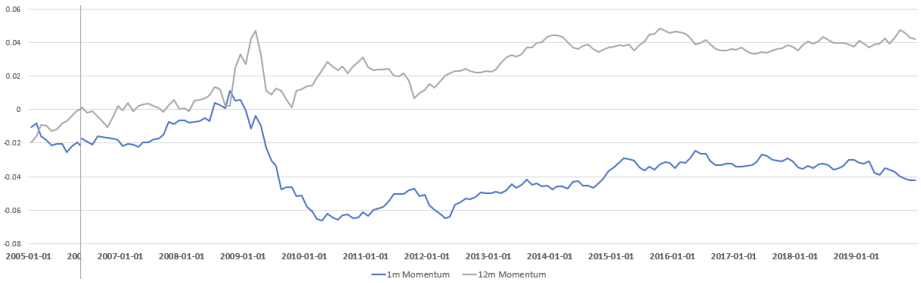

Figure 4: Model Coefficient Behavior

Note: This chart shows the average coefficients of industries from an Elastic Net model run. Coefficients are pictured for the 1-month momentum and 12-month momentum predictors

Although this analysis leaves some rigor to be desired, we take comfort in the fact that our findings confirm existing research on the topic. This is an part of our work that we would like to explore further in the future.

5.8 Performance Driver Analysis

An interesting lens under which to review performance is to decompose the sources of return from the long leg and the short leg. A positive spread between returns of the long leg versus the short leg translates into positive performance for the strategy. The model is doing a good enough job of forecasting returns and constructing a portfolio. Of the 216 monthly periods surveyed for the Commodity + Industry portfolio, 134 months showed longs outperforming shorts - a positive return for the portfolio. Of the 134 months with a positive portfolio return, the long leg of the portfolio drove the majority of the performance. In the remaining 37 months, the gains from the short leg dominated those of the long leg.

Table 14: Breakout Performance by Long/Short Leg

	Yes	**No**	**Total**
Long >Short (+ Performance)	134	82	216
+ Performance Driven by Short Leg	37	97	134

It is interesting to note that the strategy performed well during the biggest returns for the stock market, suggesting that adding the market state variable was fruitful. The strategy generally seemed to avoid the aforementioned momentum crashes. As mentioned previously, typical momentum indexes suffer from a meaningful tilt to low beta and suffer greatly when high beta comes into favor.

Table 15: Market Highs vs Lows Strategy Performance

Top and Bottom 10 Portfolio Returns					
Date	**SPX**	**FF-Mom**	**Long Return**	**Short Return**	**Spread**
10/1/2002	8.6	-5.3	1.8	3.2	-1.5
4/1/2003	8.1	-9.5	3.4	1.2	2.2
3/1/2009	8.5	-11.4	2.9	4.6	-1.6
4/1/2009	9.4	-34.4	8.6	2.9	5.7
7/1/2009	7.4	-5.4	3.5	2.0	1.4
9/1/2010	8.8	1.4	5.3	1.7	3.6
10/1/2011	10.8	-1.4	4.6	3.1	1.5
10/1/2015	8.3	-4.0	4.3	0.8	3.5
1/1/2019	7.9	-8.7	3.0	0.9	2.1
6/1/2019	6.9	-2.2	3.0	0.6	2.4
12/1/2002	-6.0	9.6	-2.5	1.4	-3.8
1/1/2003	-2.7	1.6	-1.5	1.5	-3.0
7/1/2004	-3.4	-2.3	-3.9	-0.3	-3.5
9/1/2004	0.9	5.2	-1.9	0.9	-2.8
9/1/2008	-9.1	0.4	-5.8	-1.9	-3.9
1/1/2009	-8.6	-1.8	-5.0	-1.4	-3.6
9/1/2011	-7.2	-2.6	-6.8	-3.0	-3.9
1/1/2014	-3.6	1.7	-1.9	1.2	-3.2
10/1/2018	-6.9	-1.8	-3.7	0.4	-4.1
5/1/2019	-6.6	7.6	-3.9	-0.4	-3.5

The strategy did not fare quite as well at market highs. In fact, 9/10 of its worst returns over that periods were in a negative market environment, notably during market crashes such as in 2008, 2009, and in 2018. In half of the cases, the model incorrectly allocated to short positions that outperformed the long positions it suggested. This behavior suggests the speed at which market crashes occur, and points to an area of future work. Perhaps higher-frequency market state predictor variables would improve performance.

5.9 Fama-French Comparison

We should point out that returns in our models cannot be explained by exposures to the market or to the traditional momentum factor. This can be seen from the correlation of returns between our strategies and that of the S&P 500. This suggests that the models are able to account for return drivers outside of what is explained by established risk factors. The low correlation suggests that these portfolios could be added to an existing strategy and provide diversification benefits, although more rigorous analysis would be required.

Table 16: Correlation Matrix

	S&P 500	**FF-Mom**	**Industry**	**Com + Ind**
S&P 500	1.00			
FF-Mom	-0.42	1.00		
Industry	-0.07	0.3	1.00	
Com + Ind	0.05	0.03	0.05	1.00

Note: Transaction Costs

We've performed a lot of analysis and tested a lot of variations on the data and in the models. We believe that the results are compelling and can be a viable strategy for an investor with similar testing capability to use in practice. Many academic papers are full of proofs and robustness checks, but some lack applicability. Furthermore much has been written about how transaction costs negate an investment strategy's viability. However we believe that an investor can employ our strategy and achieve good results even after accounting for transaction costs. In reality, our best performing results could be implemented with liquid futures in the case of commodities, and with ETFs in the case of industries. The strategy would buy and short at most 12 instruments each month. The investor avoids having to purchase illiquid small caps, both by construction of the data we tested and as a byproduct of our model results. While there is transaction cost involved and is not negligible, we believe that the investment environment today provides low cost structures that make it possible to achieve these returns without a significant drag from execution cost.

6 Conclusion

In summary, we were able to combine the good work of past research into an outcome that produced competitive results. At the start of our work, we believed that one single approach was sub-optimal and that by incrementally adding improvements to the data and the modelling we could gain ground. We found this to be true. Our unique approach in combining these techniques produces results that can be employed as an investment strategy with impressive results, even after accounting for transaction costs.

We conclude our findings with answers to many of the initial questions we posed. It is very apparent that machine learning provides significant improvement over simple linear models that are too sparse to capture an accurate amount of the risk premium association using only a few linear factors.

While building a complex model is great in theory, one is limited by the data they are using. In the case of financial markets, the low signal-to-noise nature of the data limits the ability of any model to produce sound forecasts.

Our work shows that adding more granularity and other asset classes delivers a model that can outperform simple statistical approaches.

Adding predictor variables is a tricky exercise where one must balance the tradeoff of improving the model's forecasting ability and overfitting. Penalized linear models help determine which variables have the biggest impact on the model's performance. Neural nets have the ability to weight different variables based on their importance, with the drawback that interpretability is limited and without computing power very slow to process.

As we conducted our research, we addressed several more questions. It does seem that returns-based indicators are able to work well in an investment context, despite them not adding all that much over simple lagged returns in terms of model accuracy. We noted a similar phenomenon with adding other asset classes to the portfolio. Though model predictability did not increase, portfolio performance did. We attribute this difference to the power of diversification. Indeed, constructing a portfolio with assets of varying risk and return characteristics improves risk-adjusted return over market cycles.

Finally, we made the biggest gains in performance when improving our forecasting approach. We did this in a variety of ways – via hyperparameter choices, by changing the prediction problem from regression to classification, and by employing an ensemble approach.

While we are pleased with our results, we admit that the project was broad in scope but limited in depth of certain areas. This leaves us with many outlets to explore further. We believe that providing more insight into which predictors added the most value is important, especially in a world where practitioners need to explain what drives returns to their investors. For further work it would be interesting to seem how some of the more complex models can work across assets and not be trained on a single series. This would allow the ability to extract information from asset classes with small amounts of data such as the recently popular cryptocurrencies.

We see promise in employing neural networks to model non-linear relationships that are readily apparent in financial market data. However our computational power limited our ability to employ this method as much as we would have liked, and we leave this as a future direction for our work. In this same vein we see room for improvement in our hyperparameter and algorithm optimization, where many methods have been explored both inside and out of a financial setting.

References

[1] David E Rapach, Jack K Strauss, Jun Tu, and Guofu Zhou. Industry return predictability: A machine learning approach. *The Journal of Financial Data Science*, 1(3):9–28, 2019.

[2] Dmitry Borisenko. Dissecting momentum: We need to go deeper. *Available at SSRN 3424793*, 2019.

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

[4] Spyros Makridakis, Evangelos Spiliotis, and Vassilios Assimakopoulos. The m4 competition: 100,000 time series and 61 forecasting methods. *International Journal of Forecasting*, 36(1):54–74, 2020.

[5] Pablo Montero-Manso, George Athanasopoulos, Rob J Hyndman, and Thiyanga S Talagala. Fforma: Feature-based forecast model averaging. *International Journal of Forecasting*, 36(1):86–92, 2020.

[6] Alois Weigand. Machine learning in empirical asset pricing. *Financial Markets and Portfolio Management*, 33(1):93–104, 2019.

[7] Jim Kyung-Soo Liew and Boris Mayster. Forecasting etfs with machine learning algorithms. *The Journal of Alternative Investments*, 20(3):58–78, 2017.

[8] Guanhao Feng, Jingyu He, and Nicholas G Polson. Deep learning for predicting asset returns. *arXiv preprint arXiv:1804.09314*, 2018.

[9] Joëlle Miffre and Georgios Rallis. Momentum strategies in commodity futures markets. *Journal of Banking & Finance*, 31(6):1863–1886, 2007.

[10] Narasimhan Jegadeesh and Sheridan Titman. Short-horizon return reversals and the bid-ask spread. *Journal of Financial Intermediation*, 4(2):116–132, 1995.

[11] Clifford S Asness, Tobias J Moskowitz, and Lasse Heje Pedersen. Value and momentum everywhere. *The Journal of Finance*, 68(3):929–985, 2013.

[12] Kent Daniel and Tobias J Moskowitz. Momentum crashes. *Journal of Financial Economics*, 122(2):221–247, 2016.

[13] Hannah Lea Hühn and Hendrik Scholz. Alpha momentum and price momentum. *International Journal of Financial Studies*, 6(2):49, 2018.

[14] Andrew Ang, Robert J Hodrick, Yuhang Xing, and Xiaoyan Zhang. High idiosyncratic volatility and low returns: International and further us evidence. *Journal of Financial Economics*, 91(1):1–23, 2009.

Appendix

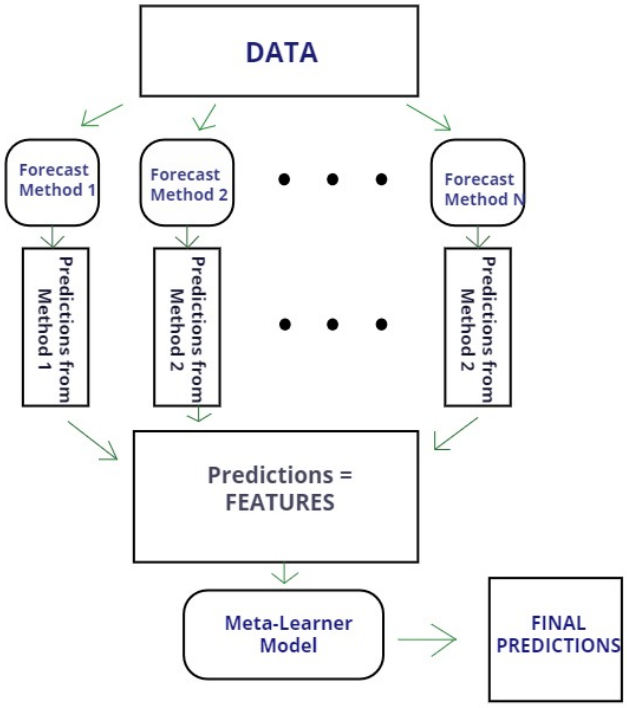

Figure 5: Stacking Ensemble Model

Table 17: Industry Data 2010-2017 Lagged vs All Predictors

Model	**MSE**	**Sharpe**	**Returns**
Enet(0.1-0.1)	34.19868	-0.10729	-0.00919
Enet(0.1-0.3)	34.09194	-0.10378	-0.00884
Enet(0.1-0.5)	33.9863	-0.1136	-0.00957
Enet(0.1-0.7)	33.88245	-0.09451	-0.00829
Enet(0.1-0.9)	33.78029	-0.10165	-0.0088
Enet(0.3-0.1)	33.91147	-0.14483	-0.01162
Enet(0.3-0.3)	33.61662	-0.05878	-0.00591
Enet(0.3-0.5)	33.33207	-0.07509	-0.0068
Enet(0.3-0.7)	33.06616	-0.01375	-0.00292
Enet(0.3-0.9)	32.81963	-0.00359	-0.00232
Enet(1-0.1)	33.10548	-0.05144	-0.0053
Enet(1-0.3)	32.37521	-0.05106	-0.00531

Enet(1-0.5)	31.79691	0.071841	0.002555
Enet(1-0.7)	31.38366	-0.02299	-0.00371
Enet(1-0.9)	31.05192	-0.04817	-0.00526
Enet(3-0.1)	31.77556	0.001295	-0.0021
Enet(3-0.3)	30.80877	-0.02498	-0.00386
Enet(3-0.5)	30.29869	0.134381	0.0068
Enet(3-0.7)	30.01076	0.109017	0.005029
Enet(3-0.9)	29.82578	0.207914	0.011562
DTree(5)	41.79676	0.032835	0.000497
DTree(10)	54.72363	0.139888	0.005126
DTree(20)	67.61467	0.494263	0.017035
DTree(30)	67.0475	0.250613	0.009482
DTree(50)	68.69655	0.440228	0.015927
DTree(100)	70.47305	0.54678	0.018838
DTree(None)	70.32855	0.183514	0.005722
RF(10)	35.2744	0.102721	0.003954
RF(15)	34.28603	-0.20777	-0.01197
RF(20)	34.71993	-0.03996	-0.00241
RF(30)	33.3214	0.34416	0.01377
RF(50)	33.68386	0.164775	0.006912
NN - 1	29.15826	0.421462	0.016864
NN - 2	29.22014	0.160511	0.005933
NN - 3	29.21198	0.315435	0.010588
NN - 4	29.2048	-0.0832	-0.00368
NN - 5	29.21051	0.566807	0.021341
Ensemble 1	13.58752	1.197717	0.090629
Ensemble 2	13.47515	0.830859	0.065315
Ensemble 3	13.69495	0.869805	0.065709
Enet(0.1-0.1)	45.95449	18.15132	0.210664
Enet(0.1-0.3)	45.74563	18.11176	0.26462
Enet(0.1-0.5)	45.57178	18.12572	0.191809
Enet(0.1-0.7)	45.40342	18.14197	0.20851
Enet(0.1-0.9)	45.2514	18.15002	0.172848
Enet(0.3-0.1)	44.90682	18.16838	0.075274
Enet(0.3-0.3)	44.44932	18.20924	0.090495
Enet(0.3-0.5)	44.01281	18.24823	0.145695
Enet(0.3-0.7)	43.62897	18.2722	0.184906
Enet(0.3-0.9)	43.25548	18.31493	0.301548
Enet(1-0.1)	43.43036	18.29338	0.152717
Enet(1-0.3)	42.32968	18.40491	0.147843
Enet(1-0.5)	41.44001	18.47948	0.077542
Enet(1-0.7)	40.71492	18.51272	0.189204
Enet(1-0.9)	40.11382	18.48998	0.16978
Enet(3-0.1)	41.34804	18.4575	0.117829
Enet(3-0.3)	39.651	18.47734	0.25291

Enet(3-0.5)	38.65022	18.41491	0.355075
Enet(3-0.7)	38.00987	18.32848	0.420992
Enet(3-0.9)	37.58924	18.24881	0.517354
DTree(5)	38.75667	-0.02006	-0.00168
DTree(10)	54.34769	-0.10781	-0.00537
DTree(20)	67.83131	-0.01806	-0.00171
DTree(30)	71.28097	0.053589	0.001367
DTree(50)	69.52461	-0.12555	-0.00724
DTree(100)	69.85447	-0.17601	-0.00901
DTree(None)	71.58275	0.023368	6.13E-05
RF(10)	34.61173	-0.10368	-0.00535
RF(15)	33.881	-0.19717	-0.01066
RF(20)	32.96578	0.061315	0.00181
RF(30)	32.92662	-0.00712	-0.0012
RF(50)	32.12196	-0.16916	-0.00976
NN - 1	29.14529	0.318702	0.010624
NN - 2	29.16027	0.604075	0.024645
NN - 3	29.21061	0.234911	0.00719
NN - 4	29.21265	0.1974	0.006551
NN - 5	29.22733	0.20107	0.006815
Ensemble 1	13.58752	1.197717	0.090629
Ensemble 2	13.23931	0.990642	0.069618
Ensemble 3	13.45545	0.966169	0.074435

Table 18: Stock Data 2010-2017 Lagged vs All Predictors

Model	**MSE**	**Sharpe**	**Returns**
Enet(0.1-0.3)	44.31406	0.268311	0.012867
Enet(0.1-0.5)	44.18591	0.224243	0.010492
Enet(0.1-0.7)	44.06058	0.262212	0.012499
Enet(0.1-0.9)	43.93688	0.308738	0.014793
Enet(0.3-0.1)	44.13877	0.261732	0.012529
Enet(0.3-0.3)	43.77393	0.291029	0.013865
Enet(0.3-0.5)	43.42592	0.220922	0.009875
Enet(0.3-0.7)	43.09702	0.194828	0.008641
Enet(0.3-0.9)	42.79151	0.171125	0.007415
Enet(1-0.1)	43.20926	0.310595	0.014764
Enet(1-0.3)	42.24743	0.199432	0.008811
Enet(1-0.5)	41.43626	0.044033	0.000946
Enet(1-0.7)	40.75684	0.167923	0.007278
Enet(1-0.9)	40.17788	0.230253	0.010362
Enet(3-0.1)	41.37602	0.243735	0.011417
Enet(3-0.3)	39.7166	0.243876	0.011155

Enet(3-0.5)	38.7077	0.498058	0.025218
Enet(3-0.7)	38.06116	0.467233	0.022634
Enet(3-0.9)	37.63568	0.471939	0.023516
Dtree(5)	45.222	0.162122	0.005338
Dtree(10)	64.87218	-0.42977	-0.01717
Dtree(20)	86.49434	-0.21554	-0.00765
Dtree(30)	87.43809	-0.57902	-0.02056
Dtree(50)	86.76381	-0.53023	-0.01533
Dtree(100)	85.93127	-0.19952	-0.00707
Dtree(None)	89.17368	-0.40847	-0.01406
RF(10)	44.0405	0.023424	6.24E-05
RF(15)	42.14818	-0.02422	-0.00212
RF(20)	41.78031	-0.11243	-0.00571
RF(30)	41.46587	-0.42774	-0.02051
RF(50)	40.19865	0.087206	0.003001
Ensemble 1	16.53222	1.131609	0.069671
Ensemble 2	16.84487	0.918542	0.064502
Ensemble 3	16.93739	0.826853	0.053647

Enet(0.1-0.1)	45.95449	0.210664	0.009566
Enet(0.1-0.3)	45.74563	0.26462	0.011848
Enet(0.1-0.5)	45.57178	0.191809	0.008283
Enet(0.1-0.7)	45.40342	0.20851	0.009161
Enet(0.1-0.9)	45.2514	0.172848	0.007513
Enet(0.3-0.1)	44.90682	0.075274	0.00252
Enet(0.3-0.3)	44.44932	0.090495	0.003286
Enet(0.3-0.5)	44.01281	0.145695	0.005997
Enet(0.3-0.7)	43.62897	0.184906	0.007924
Enet(0.3-0.9)	43.25548	0.301548	0.014072
Enet(1-0.1)	43.43036	0.152717	0.006718
Enet(1-0.3)	42.32968	0.147843	0.006479
Enet(1-0.5)	41.44001	0.077542	0.002734
Enet(1-0.7)	40.71492	0.189204	0.008918
Enet(1-0.9)	40.11382	0.16978	0.007795
Enet(3-0.1)	41.34804	0.117829	0.004978
Enet(3-0.3)	39.651	0.25291	0.012742
Enet(3-0.5)	38.65022	0.355075	0.018294
Enet(3-0.7)	38.00987	0.420992	0.020252
Enet(3-0.9)	37.58924	0.517354	0.026401
Dtree(5)	48.08273	0.154131	0.005295
Dtree(10)	63.82789	0.414628	0.017303
Dtree(20)	85.91167	0.507556	0.020483
Dtree(30)	86.6067	-0.01421	-0.00159
Dtree(50)	85.60581	0.428376	0.016016

Dtree(100)	86.23681	0.456406	0.016961
Dtree(None)	84.782	0.198169	0.007097
RF(10)	44.51922	-0.33493	-0.01885
RF(15)	42.20038	-0.26414	-0.01432
RF(20)	41.24452	-0.21069	-0.01034
RF(30)	40.52058	-0.09751	-0.0051
RF(50)	39.7489	0.028684	0.000214
Ensemble 1	16.53222	1.131609	0.069671
Ensemble 2	16.81335	0.821506	0.059622
Ensemble 3	16.9327	0.757096	0.048535