Machine Learning in Empirical Asset Pricing Models

1st Huei Wen Teng
Department of Information Management and Finance
National Chiao Tung University
Hsinchu, Taiwan
hwteng@nctu.edu.tw

2nd Yu-Hsien Li
Taishin International Bank
Taipei, Taiwan
k8508011@gmail.com

3rd Shang-Wen Chang
Department of Applied Mathematics
National Chiao Tung University
Hsinchu, Taiwan
ckjc80622@gmail.com

***Abstract*—Although machine learning has achieved great success in computer science, its performance in the canonical problem of asset pricing in finance is yet to be fully investigated. To compare machine learning techniques and traditional models, we use 8 macroeconomic predictors and 102 firm characteristics to predict stock returns in a monthly basis. It is shown that the neural network outperforms others: Specifically, when building bottom-up portfolios based on the predicted stock-level returns for both buy-and-hold and long-short strategies, XGBoost and neural networks produce portfolios with the highest Sharpe ratios. Limitations and challenges in using machine learning techniques in empirical asset pricing models are also discussed.**

***Index Terms*—Empirical asset pricing models, return prediction, Fama-MacBeth regression, elastic net, machine learning, random forest, regression tree, XGBoost, neural network**

I. Introduction

Since Sharpe (1964) proposed the capital asset pricing model (CAPM) to explain the variation of stock returns with market excess returns, there have been numerous studies to understand the relationship between cross-section stock returns and systematic risk. Shortly after Roll and Ross (1980), Fama and French (1993), and Fama and French (1996), firm characteristics are founded to play a critical role in empirical asset pricing field as well . Cochrane (2011) identified the pricing factors that are able to provide useful information about stock returns, and Jeremiah et al. (2013) and Harvey et al. (2016) summarize more than three hundreds characteristics from prior articles. Therefore, along with the proliferation of the available factors, it is of considerable importance to propose an adequate method to identify crucial factors for returns prediction and is able to accommodate a large number of factors.

Machine learning techniques are capable of variable selection with regularization methods and dimension reduction methods that can avoid overfitting. Also, collinearity poses less problem to machine learning techniques while traditional methods in general fail to overcome it. In literature, Gu et al. (2019), Tsang and Wong (2019), and Gu et al. (2020) have applied machine learning techniques in finance, including tree-based models, feed-forward neural network, and auto-encoder model. The powerful predictability of machine learning has been demonstrated for cross-section and time-series stock returns.

As an extension to Gu et al. (2020), we consider Fama-MacBeth regression and XGBoost (Chen and Guestrin, 2016; Green et al., 2017). We include 8 macroeconomic factors and 102 firm characteristics and we compare the prediction performance of each model by evaluating the R^2 of stock-level returns and predicted portfolio-level returns in out-of-sample. We aim to show that machine learning techniques outperform traditional methods both in stock-level returns prediction the resulting portfolios.

The rest of this paper is organized as follows. Section II provides a list of models for comparison and describes data, and Section III summarizes empirical results. The last section concludes.

II. Models and Data

To begin, we assume that there is a constant procedure of determining the return of next period $r_{i,t+1}$ of firm i when given macroeconomic and firm characteristics $z_{i,t}$ of firm i in this period. Each firm represents a data point $(z_{i,t}, r_{i,t+1})$ where $z_{i,t} \in \mathbb{R}^{110}$ and $r_{i,t+1} \in \mathbb{R}$. We assume that the return-generating procedure is constant along time. Thus, for all models except Fama-MacBeth regression, data points from different time t are not treated any differently.

In terms of equations, we consider a pooled model: We assume that the rates of next month is

$$r_{i,t+1} = g(z_{i,t}; \Theta) + \epsilon_{i,t},$$

where g, θ are constant along t Different models will determine different ways to estimate $g(\cdot; \Theta)$. In general, we use loss inspired by MSE for convenience purposes.

As for the data, we have collected monthly stock returns of all firms on Center for Research in Security Prices (CSRP). The number of stocks N_t fluctuates through time, but it always exceed 4,500; In total, there are roughly 18000 unique stocks. For the factors, we follow Green et al. (2017) to use 102 firm characteristics and 8 macroeconomic factors defined in Goyal and Welch (2008). Finally, we use one-month Treasury-bill rate as a proxy for risk-free rate. Detailed information about

Huei-Wen Teng gratefully acknowledges the The author gratefully acknowledges the support by the Ministry of Science and Technology of Taiwan, R.O.C., under grant number MOST 108-2118-M-009-001-MY2 and grant number MOST 108-2634-F-163-001 through Pervasive Artificial Intelligence Research Labs.

the firm characteristics and macroeconomic indicators can be found in Li (2020).

A. Models

The models can be classified as traditional models and machine learning techniques. We present all models that we will use to estimate the regression function in the following section, which includes parameters, estimator and loss function. We first consider the following four traditional models.

(1) Ordinary Least Squares (OLS)

$$g(z_{i,t};\theta) = \theta^T z_{i,t}$$

$$\mathcal{L}(\theta) = \frac{1}{T}\sum_{t=1}^{T}\frac{1}{N_t}\sum_{i=1}^{N_t}\big((r_{i,t} - g(z_{i,t};\theta)\big)^2$$

(2) Least Squares with Huber loss

$$g(z_{i,t};\theta) = \theta^T z_{i,t}$$

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

where $H(s;\xi) = \begin{cases} s^2 \text{ if } |s| \le \xi \\ 2\xi|s| - \xi^2 \text{ if } |s| > \xi \end{cases}$

(3) ElasticNet (ENet)

$$g(z_{i,t};\theta) = \theta^T z_{i,t}$$

$$\mathcal{L}(\theta) = \frac{1}{T}\sum_{t=1}^{T}\frac{1}{N_t}\sum_{i=1}^{N_t}\big((r_{i,t} - g(z_{i,t};\theta)\big)^2$$

$$+\lambda(1-\rho)\sum_{j=1}^{P}|\theta_j| + \frac{1}{2}\lambda\rho\sum_{j=1}^{P}\theta_j^2$$

(4) Fama-MacBeth regression (FM)

$$\Theta_t = (\alpha_t, \theta_t)$$

$$\Theta = (\Theta_1, ..., \Theta_T)$$

$$g_t(z_{i,t};\Theta_t) = r_{f,t} + \alpha_t + \theta_t^T z_{i,t}$$

$$g(z_{i,t};\Theta_t) = \bar{\alpha} + \bar{\theta}^T z_{i,t}$$

$$\mathcal{L}_t(\Theta_t) = \frac{1}{N_t}\sum_{i=1}^{N_t}\big(r_{i,t+1} - g(z_{i,t};\Theta_t)\big)^2$$

where Θ_t's are independently found

In addition, we consider the following five machine learning techniques.

(5) Regression Tree (for illustration purposes only, not actually used as a model in this study)

$$\Theta = (..., \theta^{(k)}, ..., C^{(k)}, ...)$$

where $\theta^{(k)} \in \mathbb{R}, \{C_k\}_{k=1}^{K}$ is a partition of $\mathbb{R}^1$

$$g(z_{i,t};\Theta) = \sum_{k=1}^{K}\theta^{(k)}\mathrm{I}_{z_{i,t}\in C^{(k)}}$$

$$\mathcal{L}(\Theta) = \sum_{t=1}^{T}\sum_{i=1}^{N_t}\ell(r_{i,t+1}, g(z_{i,t};\theta)) = \sum_{k=1}^{K}\sum_{z_{i,t}\in C_k}\ell(r_{i,t+1}, \theta^{(k)})$$

(6) Random Forest (RF)

$$\Theta_b = (..., \theta_b^{(k)}, ..., C_b^{(k)}, ...)$$

$$\Theta = (..., \theta_1^{(k)}, ..., C_1^{(k)}, ..., \theta_B^{(k)}, ..., C_B^{(k)}, ...)$$

$$g_b(z_{i,t};\Theta_b) = \sum_{k=1}^{K}\theta_b^{(k)}\mathrm{I}_{z_{i,t}\in C_b^{(k)}}$$

$$g(z_{i,t};\Theta) = \frac{1}{B}\sum_{b=1}^{B}g_b(z_{i,t}, \Theta_b)$$

$$\mathcal{L}_b(\Theta_b) = \sum_{k=1}^{K}\sum_{z_{i,t}\in C_b^{(k)}}\ell(r_{i,t+1}, \theta_b^{(k)})$$

where Θ_b's are independently found.

(7) Gradient Boosting Regression Tree (GBRT)

$$\Theta_b = (..., \theta_b^{(k)}, ..., C_b^{(k)}, ...)$$

$$\Theta = (..., \theta_1^{(k)}, ..., C_1^{(k)}, ..., \theta_B^{(k)}, ..., C_B^{(k)}, ...)$$

$$g_b(z_{i,t};\Theta_b) = \sum_{k=1}^{K}\theta_b^{(k)}\mathrm{I}_{z_{i,t}\in C_b^{(k)}}$$

$$g(z_{i,t};\Theta) = \sum_{b=1}^{B}g_b(z_{i,t}, \Theta_b)$$

$$\mathcal{L}_b(\Theta_b) = \sum_{t=1}^{T}\sum_{i=1}^{N_t}\ell\big(r_{i,t+1}, g_{b-1}(z_{i,t};\Theta_{b-1}) + g_b(z_{i,t};\Theta_b)\big)$$

where Θ_b's are sequentially found.

(8) XGBoost

$$\Theta_b = (..., \theta_b^{(k)}, ..., C_b^{(k)}, ...)$$

$$\Theta = (..., \theta_1^{(k)}, ..., C_1^{(k)}, ..., \theta_B^{(k)}, ..., C_B^{(k)}, ...)$$

$$g_b(z_{i,t};\Theta_b) = \sum_{k=1}^{K}\theta_b^{(k)}\mathrm{I}_{z_{i,t}\in C_b^{(k)}}$$

$$g(z_{i,t};\Theta) = \sum_{b=1}^{B}g_b(z_{i,t}, \Theta_b)$$

$$\Omega(g_b(\cdot;\Theta_b)) = \gamma K + \frac{1}{2}\lambda\sum_{k=1}^{K}(\theta_b^{(k)})^2$$

$$\mathcal{L}_b(\Theta_b) = \sum_{t=1}^{T}\sum_{i=1}^{N_t}\ell\Bigg(r_{i,t+1}\ ,\ g_{b-1}(z_{i,t};\Theta_{b-1}) + g_b(z_{i,t};\Theta_b)\Bigg)$$

$$+\Omega(g_b(\cdot;\Theta_b)))$$

where Θ_b's are sequentially found.

(9) Neural Network (NN) We will try five different neural networks with number of hidden layers L ranging from 1 to 5.

For $L = 1$,

$$h^{(0)} = \begin{pmatrix} 1 \\ z \end{pmatrix} \in \mathbb{R}^{p+1}$$

$$\Theta = (\theta^{(0)}, \theta^{(1)})$$

$$h_k^{(1)} = \text{ReLU}\big((h^{(0)})^T \theta_k^{(0)}\big)$$

for $k = 1 \sim K_1$

$$g(z; \Theta) = (h^{(1)})^T \theta^{(1)}$$

$$\mathcal{L}(\Theta) = \sum_{t=1}^{T} \sum_{i=1}^{N_t} \big(r_{i,t+1} - g(z_{i,t}; \Theta)\big)$$

For $L > 1$,

$$h^{(0)} = \begin{pmatrix} 1 \\ z \end{pmatrix} \in \mathbb{R}^{p+1}$$

$$\Theta = (\theta^{(0)}, ..., \theta^{(L)})$$

$$h_k^{(l)} = \text{ReLU}\big((h^{(l-1)})^T \theta_k^{(l-1)}\big)$$

for $k = 1 \sim K_l$ and $l = 0 \sim L$

$$g(z; \Theta) = (h^{(L)})^T \theta^{(L)}$$

$$\mathcal{L}(\Theta) = \sum_{t=1}^{T} \sum_{i=1}^{N_t} \big(r_{i,t+1} - g(z_{i,t}; \Theta)\big)$$

B. Data preprocessing

First, In practice, data are released to the public with a delay. If we want to predict returns at month $t + 1$, we use the most recent monthly-updated firm characteristics at the end of month t, quarterly-updated data at the end of month $t - 4$, and the annual-updated data at the end of month $t - 6$. Take quarterly-updated data as an example, these accounting characteristics are assumed to be known four months after the end of the fiscal year. Thus, if fiscal year of a firm ends in the prior December, its quarterly accounting data can be observable by the end of April of the next year.

Second, We winsorize all characteristics at the 1st and 99th percentiles. Third, the average and standard deviation of each factor change over time. If we use the original data without normalization, the estimation error is likely to increase. For example, tree-based model will automatically determine the splitting features and values during the training process. However, the values of firm characteristics and macroeconomic predictors in validation or testing periods change a lot and thus differ from the values in the training period, rendering our model incapable of predicting future stock returns.

On the other hand, only 7% of the firm characteristics have no missing values. We tackle this problem by imputing the missing values month by month instead of discarding them directly. We first normalize the monthly data (except for indicator variables) excluding missing values by an individual feature, and plug the missing values with 0 in each month. By doing so, we complete data imputation and normalization.

In addition, we split the 38 years of data into training period of 25 years (1980/1 - 2004/12), the validation period of 6 years (2005/1 - 2010/12), and the out-of-sample period of the remaining 7 years (2011/1 - 2017/12).

To investigate multicollinearity, it is found in Figure 1 that there are 8 firm characteristics whose VIFs are larger than 10 (about 9%) and most of absolute correlations $|\rho_{ij}|$ (about 93%) are below 0.2. This indicates that the multicollinearity problem exists in our data. This poses a problem for traditional methods but would potentially be solved by machine-learning-based methods.

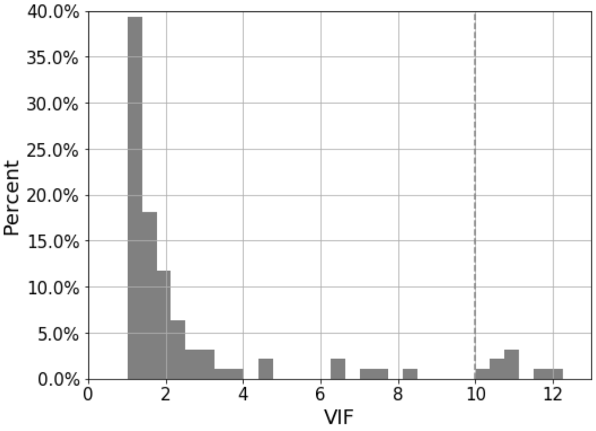

Fig. 1. Distribution of VIFs

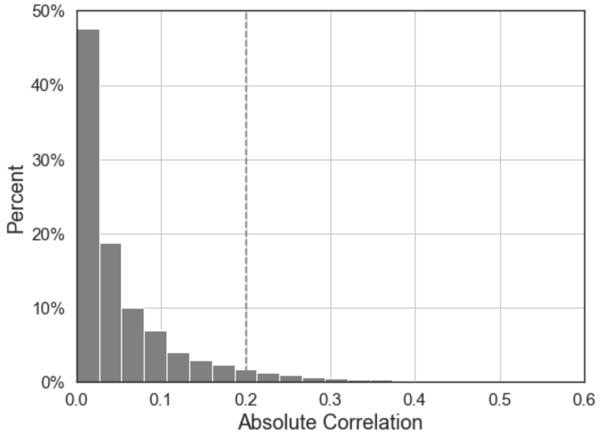

Fig. 2. Distribution of absolute value of correlations ($|\rho_{ij}|$)

Figure 3 shows the box plots of autocorrelation function (ACF) at lags 1, 2, and 6 for each firm characteristic. It clearly shows that most of average ACF of characteristics excesses 0.25, and 80% of them are significantly different from 0 at the 5% level with Ljung-Box test. This indicates that serial correlations exists in most of the characteristics. Serial correlation will also cause estimation errors in traditional methods.

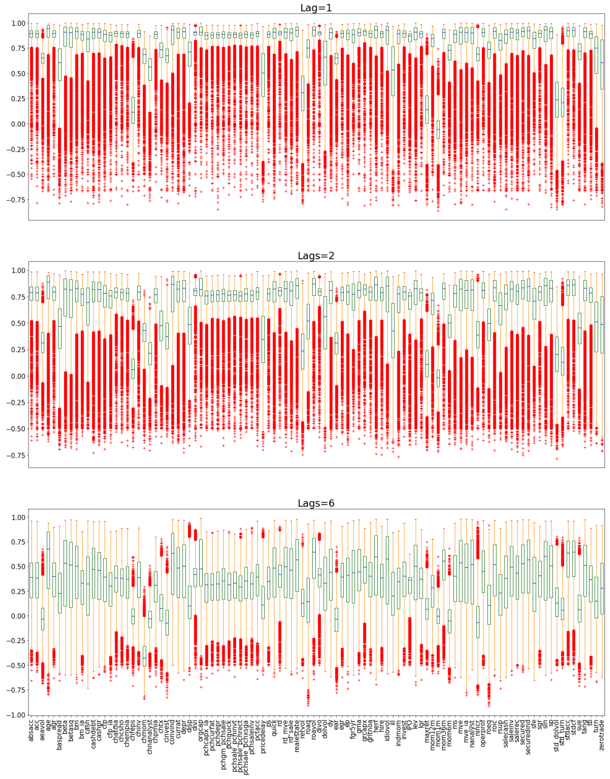

Fig. 3. ACF at lags 1,2,6 for firm characteristics.

TABLE I
MONTHLY OUT-OF-SAMPLE STOCK-LEVEL PREDICTION PERFORMANCE (PERCENTAGE R^2)

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

III. RESULTS

A. Individual-level measures

The most straightforward way to measure the performance of a model is to consider the accuracy of individual predictions $\{z_{i,t}, g(z_{i,t};\Theta)\}_{i\in I_t, t\in\tau}$.

We use out-of-sample R^2 as a measure of accuracy. More specifically, let τ be the months in testing data (2011/1 - 2017/12), I_t be the indices of all firms at time t, I_{top} be the indices of top 400 stocks (by market value) at time t, and I_{bottom} be the indices of bottom 400 stocks (by market value) at time t.

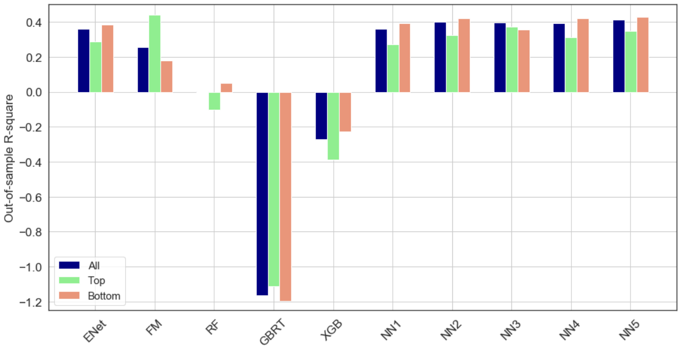

Fig. 4. R^2_{oos} for all firms, top 400 firms and bottom 400 firms.

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

The OLS model produces an R^2_{oos} of -9.41%. This indicates that OLS fails to predict returns. Then, if we use Huber loss function to avoid the influence of the extreme outcome values in OLS, R^2_{oos} improves from -9.41% to -3.11%. Moreover, R^2_{oos} raises to 0.36% and 0.26% for ENet and FM, respectively. Indeed, ENet chooses a smaller set of variables but produces higher R^2_{oos}. This indicates that including all the predictors does not guarantee better prediction.

Tree-Based methods (random forest, boosted trees, and XGB) seem to fail to forecast returns. R^2_{oos} of tree-based models are negative: 0.00%, -1.16%, and -0.27%. This is possibly because that the prediction outcomes are computed by the average $r_{i,t+1}$ of observation corresponding to leaves (terminal nodes) in training periods. Note that we only consider shallow trees. The number of leaves is possibly not large enough to predict returns.

Neural networks outperform all methods in predicting stock returns: R^2_{oos} is 0.36% for NN1 and reaches its peaks at 0.41% for NN5. However, we can observe that the marginal benefits from the deeper neural network become smaller. The results show that there is limitation for a deep neural network to improve the forecasts.

The second and third rows of Table I report the R^2_{oos} for large and small stocks (the top and bottom 400 by market values in each month). Note that we use the parameters estimated from the full data to forecasts returns for the two subsamples. We observe that FM dominates all of models among large stocks, with R^2_{oos} 0.44%.

B. Portfolio-level measures

To measure the performance of models, we consider not just how good the model predicts the individual returns, but also how good a portfolio we can construct if we are to use the models as a basis for building portfolios. We create 10 bottom-up portfolios for each prediction in the following procedure:

Given a prediction $\{r_{i,\hat{t+1}} := g(z_{i,t},\Theta)\}_{i\in I_t,t\in\tau}$, we consider a strategy of building a portfolio: At each time t, let $Q_{p,t}$ be the $p-$th quantile of $\{r_{i,\hat{t+1}} := g(z_{i,t};\Theta)\}_{i\in I_t,t\in\tau}$.

We consider 10 portfolios with weights at time $t+1$, $w_{p,t+1}$, which are N_t-shaped vector,

$$w_{p,i,t+1} = \begin{cases} 1/N_{p,t} \text{ if } r_{i,\hat{t+1}} \in Q_{p,t} \\ 0 \text{ if } r_{i,\hat{t+1}} \notin Q_{p,t} \end{cases}$$

The portfolio consisting of stocks from the 1^{st} quadrant is called the "Low portfolio" and the one consisting of stocks from the 10^{th} quadrant is called the "High portfolio". We also consider an extra portfolio called "long-short", which longs (give positive weights) to the "High portfolio" and shorts (give negative weights) to the "Low portfolio". Finally, the "equal-weight portfolio" refers to giving N_t stocks equal weights, which serves as a benchmark.

We can then define the predicted portfolio returns

$$r_{p,\hat{t+1}} = \sum_{i=1}^{N_t} w_{p,i,t+1} r_{i,\hat{t+1}}$$

and the realized portfolio returns

$$r_{p,t+1} = \sum_{i=1}^{N_t} w_{p,i,t+1} r_{i,t+1}.$$

The portfolio-level out-of-sample R^2 can then be defined as

$$R^2_{oos,p,\tau} = 1 - \frac{\sum_{p\in 1\sim 10, t\in\tau}(r_{p,t+1} - r_{p,\hat{t+1}})^2}{\sum_{p\in 1\sim 10, t\in\tau}(r_{p,t+1}))^2}.$$

The following shows the R^2_{oos} for 10 portfolios with each model. The result is similar to the individual-level R^2_{oos}.

TABLE II
MONTHLY PORTFOLIO-LEVEL OUT-OF-SAMPLE PREDICTIVE PERFORMANCE (PERCENTAGE R^2)

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

Other than the accuracy-based ones, we also consider the following metrics for the portfolios that reveals other aspects of the portfolios. First, the Last value is

$$P^p_T = 100 \times \prod_{t=1}^{T} e^{r_{p,t}}.$$

The Compound annual growth rate (CAGR) is

$$\mathrm{CAGR}_p = \frac{1}{T/12}(\log P^p_T - \log P^p_0).$$

A good model is expected to have significant difference of last value and CAGR between the high portfolio and the low one.

On the other hand, the following metrics measures the risk of the portfolios. The standard deviation (SD) is

$$\mathrm{SD}_p = \sqrt{E_t\left[(r_{p,t} - E_t[r_{p,t}]^2\right] \times 12}.$$

To consider both returns and risk, the Sharpe ratio (SR) is

$$\mathrm{SR}_p = \frac{\mathrm{CAGR}_p - r_f}{\mathrm{SD}_p}.$$

The turnover rate (TO) is

$$\mathrm{TO}_p = \frac{1}{T}\sum_{t=2}^{T}\Big(\sum_{i=1}^{N_t}|w_{i,p,t} - w_{i,p,t-1}|.$$

Last, another commonly used risk measure is the maximum drawdown (MaxDD)

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

Several interesting observations in Table III (next page) and Figure 5 are pointed out below. First, machine learning technique are able capable to distinguish high and low portfolios. In fact, for all methods except RF, the CAGR of portfolios increases monotonically as the group rank increases, hinting to the ability of prediction portfolio-level returns. Specifically, XGBoost and NN4 have the largest difference of high and low portfolio in last values (about 320).

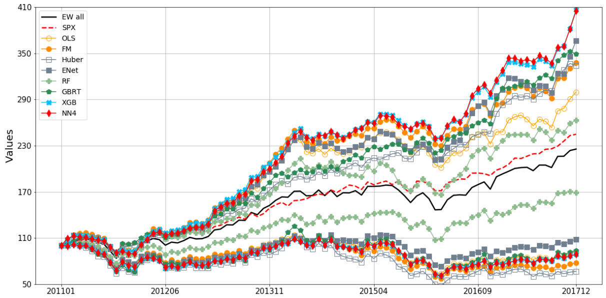

Fig. 5. Cumulative Returns of High Portfolio and Low Portfolio.

TABLE III: Performance of Bottom-Up Portfolios of Each Models

	Last Value	Pred.	CAGR	SD	SR	TO	Last Value	Pred.	CAGR	SD	SR	TO
			Equal-Weight						SPY			
Benchmark	225.60	-	0.12	0.14	0.81	0.02	244.87	-	0.13	0.11	1.18	-
			OLS						FM			
Low (L)	92	0.42	-0.01	0.21	-0.06	0.57	77	-0.01	-0.04	0.21	-0.17	0.61
2	158	0.55	0.07	0.16	0.40	1.09	162	0.00	0.07	0.16	0.43	1.12
3	198	0.60	0.10	0.14	0.68	1.28	226	0.01	0.12	0.14	0.81	1.29
4	243	0.64	0.13	0.14	0.90	1.36	236	0.01	0.12	0.14	0.89	1.37
5	248	0.67	0.13	0.14	0.95	1.38	247	0.02	0.13	0.13	0.96	1.40
6	255	0.69	0.13	0.14	0.96	1.37	255	0.02	0.13	0.14	0.95	1.39
7	291	0.72	0.15	0.14	1.12	1.34	267	0.02	0.14	0.14	1.02	1.35
8	277	0.76	0.15	0.14	1.07	1.25	323	0.02	0.17	0.14	1.18	1.28
9	319	0.80	0.17	0.14	1.16	1.09	281	0.03	0.15	0.14	1.05	1.10
High (H)	300	0.94	0.16	0.15	1.04	0.66	338	0.04	0.17	0.14	1.20	0.67
H-L	326	0.52	0.17	0.11	1.50	-	436	0.05	0.21	0.12	1.69	-
			Huber						ENet			
Low (L)	67	-0.07	-0.06	0.26	-0.23	0.43	108	0.15	0.01	0.19	0.06	0.24
2	153	0.21	0.06	0.21	0.30	0.90	200	0.16	0.10	0.17	0.58	0.51
3	189	0.33	0.09	0.17	0.55	1.10	205	0.16	0.10	0.15	0.66	0.71
4	248	0.40	0.13	0.15	0.86	1.23	232	0.16	0.12	0.14	0.83	0.82
5	260	0.45	0.14	0.14	0.97	1.30	260	0.16	0.14	0.14	0.96	0.85
6	290	0.49	0.15	0.13	1.19	1.34	228	0.16	0.12	0.15	0.80	0.84
7	288	0.52	0.15	0.12	1.23	1.34	271	0.16	0.14	0.14	0.98	0.79
8	314	0.56	0.16	0.12	1.35	1.28	250	0.17	0.13	0.15	0.90	0.70
9	316	0.59	0.16	0.12	1.42	1.14	227	0.17	0.12	0.13	0.92	0.60
High (H)	334	0.65	0.17	0.11	1.59	0.69	367	0.17	0.19	0.14	1.28	0.33
H-L	501	0.73	0.23	0.20	1.17	-	339	0.03	0.17	0.12	1.50	-
			RF						GBRT			
Low (L)	169	0.20	0.07	0.17	0.45	0.48	94	0.11	-0.01	0.26	-0.04	0.47
2	237	0.21	0.12	0.15	0.82	0.81	145	0.24	0.05	0.20	0.27	1.03
3	287	0.21	0.15	0.14	1.04	0.87	189	0.30	0.09	0.16	0.57	1.18
4	219	0.21	0.11	0.15	0.75	0.88	220	0.33	0.11	0.14	0.82	1.21
5	227	0.22	0.12	0.14	0.84	0.94	253	0.36	0.13	0.13	1.04	1.25
6	199	0.23	0.10	0.15	0.66	0.97	261	0.37	0.14	0.13	1.06	1.26
7	224	0.23	0.12	0.16	0.73	0.87	289	0.38	0.15	0.12	1.23	1.26
8	217	0.24	0.11	0.15	0.75	0.73	280	0.40	0.15	0.13	1.15	1.22
9	234	0.26	0.12	0.15	0.84	0.62	327	0.41	0.17	0.13	1.28	1.09
High (H)	263	0.28	0.14	0.16	0.89	0.34	349	0.43	0.18	0.13	1.39	0.69
H-L	156	0.08	0.06	0.11	0.60	-	373	0.31	0.19	0.19	0.97	-
			XGBoost						NN1			
Low (L)	89	0.23	-0.02	0.21	-0.08	0.35	117	0.09	-0.01	0.20	0.11	0.29
2	170	0.26	0.08	0.17	0.44	0.75	186	0.14	0.08	0.17	0.51	0.63
3	222	0.27	0.11	0.15	0.74	0.96	213	0.15	0.11	0.15	0.70	0.81
4	245	0.27	0.13	0.15	0.86	1.07	252	0.16	0.13	0.15	0.90	0.89
5	255	0.28	0.13	0.15	0.90	1.10	262	0.17	0.13	0.15	0.93	0.90
6	269	0.28	0.14	0.14	1.03	1.09	243	0.17	0.15	0.14	0.92	0.86
7	261	0.29	0.14	0.14	0.96	1.04	259	0.18	0.14	0.14	1.00	0.79
8	238	0.29	0.12	0.14	0.88	0.94	235	0.18	0.13	0.14	0.85	0.68
9	243	0.30	0.13	0.12	1.01	0.78	232	0.19	0.12	0.13	0.91	0.60
High (H)	407	0.33	0.20	0.13	1.55	0.40	326	0.23	0.18	0.15	1.10	0.31
H-L	458	0.10	0.22	0.13	1.68	-	278	0.14	0.19	0.14	1.04	-
			NN2						NN3			
Low (L)	95	0.13	0.02	0.21	-0.03	0.29	85	0.11	-0.02	0.22	-0.11	0.44
2	179	0.15	0.09	0.18	0.45	0.66	173	0.14	0.09	0.17	0.47	0.90
3	215	0.16	0.11	0.15	0.73	0.84	205	0.16	0.11	0.15	0.69	1.09
4	245	0.16	0.13	0.14	0.88	0.90	238	0.16	0.12	0.14	0.89	1.16
5	255	0.17	0.14	0.14	0.97	0.93	281	0.17	0.14	0.13	1.10	1.19
6	282	0.17	0.13	0.13	1.10	0.91	267	0.17	0.13	0.13	1.08	1.17
7	260	0.17	0.14	0.14	0.96	0.83	280	0.17	0.13	0.14	1.08	1.11
8	248	0.18	0.12	0.14	0.90	0.71	253	0.17	0.13	0.14	0.95	1.01
9	230	0.19	0.12	0.12	0.96	0.62	289	0.19	0.14	0.13	1.12	0.85
High (H)	359	0.21	0.17	0.16	1.17	0.33	314	0.21	0.20	0.16	0.99	0.45
H-L	378	0.08	0.15	0.14	1.39	-	371	0.08	0.22	0.13	1.49	-
			NN4						NN5			
Low (L)	89	0.11	-0.02	0.21	-0.08	0.27	88	0.03	-0.03	0.22	-0.08	0.37
2	181	0.14	0.07	0.18	0.48	0.58	159	0.12	0.09	0.20	0.34	0.81
3	212	0.16	0.09	0.16	0.66	0.75	183	0.14	0.12	0.17	0.51	1.03
4	225	0.16	0.11	0.15	0.78	0.81	212	0.15	0.12	0.16	0.67	1.14
5	258	0.17	0.14	0.15	0.92	0.83	263	0.16	0.13	0.14	0.98	1.22
6	255	0.17	0.15	0.15	0.90	0.81	277	0.18	0.13	0.13	1.12	1.24
7	241	0.17	0.14	0.14	0.91	0.76	267	0.19	0.14	0.13	1.10	1.20
8	254	0.17	0.17	0.13	1.00	0.69	319	0.21	0.13	0.12	1.34	1.09
9	276	0.18	0.15	0.12	1.21	0.61	278	0.23	0.15	0.11	1.31	0.91
High (H)	405	0.19	0.19	0.13	1.50	0.32	368	0.27	0.18	0.13	1.44	0.53
H-L	457	0.08	0.20	0.14	1.50	-	418	0.24	0.21	0.14	1.48	-

Second, Sharpe ratio of the long-short portfolio under FM and XGBoost is 1.69 and 1.68, respectively, which is substantially larger than equal-weight (0.81) and SPX (1.02). Third, turnover rate of XGBoost and NN4 are lower than those of traditional methods, which is surprising given that they yield larger Sharpe ratios. Finally, Table IV shows that XGBoost and NN4 yield larger Sharpe ratios without increasing significant maximum drawdown.

TABLE IV
MAXIMUM DRAWDOWN OF EACH METHOD

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

IV. CONCLUSION

Our empirical analysis suggests benefits of applying machine learning methods in the field of empirical asset pricing. First of all, our analysis shows that machine learning methods can improve the predictability in stock returns. Machine learning methods can also estimate risk premia with less errors, in the sense of creating portfolios with higher Sharpe ratios. Moreover, adding penalty and regularization terms, such as ENet and XGBoost, substantially improves the predicted results of each model. Interestingly, for neural networks, we find that the marginal improvement decreases as the neural network gets ”deeper” since their R2 only changes a little.

Second, the success of machine learning methods in returns prediction allows investors to form a portfolio for practical purposes. Indeed, the difference among portfolios constructed based on the predicted stock returns using traditional and machine learning methods becomes even larger. That is, machine learning methods outperform traditional methods in distinguishing the stocks with higher predicted returns.

Our machine learning approach provides ways to capture the useful information through using a large set of predictors and estimate risk premia with less approximation errors. Our results suggest that machine learning has been an important role in empirical asset pricing fields.

ACKNOWLEDGMENT

We benefited from discussions with San-Lin Chung, Cheng-Few Lee, and Han-Hsin Lee in the seminars of Department of Information Management and Finance at National Chiao Tung University.

REFERENCES

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

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

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

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

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

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

Gu, S., B. T. Kelly, and D. Xiu (2019). Autoencoder asset pricing models. *Journal of Econometrics, forthcoming*.

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

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

Li, Y.-H. (2020). An empirical comparison between traditional methods and machine learning in asset pricign models. Master's thesis, National Chiao Tung University.

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

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

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