

Cryptocurrency Risk Factors via Machine Learning

Li-Han Chang¹, Yeh-Ting Chang^{*2}, Hung-Wen Cheng², Pei-Jie Hsiao³

¹ Tamkang University, ² Soochow University, ³ National Sun Yat-sen University

Abstract

This paper conducts a comparative analysis of machine learning methods for the canonical problem of empirical asset pricing, forecasting cross-sectional returns. Using a large panel of cryptocurrencies and market-based predictors, we find that machine learning models, particularly gradient boosting methods, deliver significantly higher out-of-sample predictive accuracy and investment performance than traditional linear benchmarks. In value-weighted portfolio strategies, the best-performing model (XGBoost) more than triples the Sharpe ratio of the OLS benchmark. We attribute these gains to the model's ability to capture complex nonlinear interactions among predictors, which are missed by simpler methods. While neural networks and regularized regressions offer moderate improvements, tree-based methods consistently outperform. Unlike the equity literature, we find little evidence of momentum effects; instead, the most informative features are related to volatility, liquidity, and extreme return behavior.

Keywords: Machine learning, gradient boosting, (deep) neural networks, cryptocurrency, risk factor.

1 Introduction

The application of machine learning in financial prediction has expanded rapidly, particularly in cryptocurrency markets marked by high volatility, low transparency, and rapid innovation. Unlike equity markets with structured data and established factors, traditional models struggle to capture crypto dynamics. While some view crypto prices as purely speculative, others argue that data-driven methods can reveal hidden structures. Supporting this view, this study systematically compares machine learning models for predicting cross-sectional excess returns using market-based features such as price, volume, and volatility. By contrasting linear and nonlinear algorithms, we aim to uncover empirical regularities and assess the effectiveness of modern predictive techniques tailored to the distinct characteristics of cryptocurrency markets.

Machine learning techniques have been increasingly incorporated into the asset pricing literature. Early contributions applied regularization and neural network methods to specific prediction problems, such as option pricing and return forecasting by Hutchinson, Lo and Poggio (1994), Yao, Li and Tan (2000), and Rapach, Strauss and Zhou (2013), while Heaton, Polson and Witte (2016) developed a neural network framework for portfolio selection. These early works demonstrate the flexibility of machine learning in financial prediction tasks. In the broader asset pricing literature,

traditional strategies such as size and momentum remain central. Banz (1981) was the first to document the size effect in equity markets, while Fama and French (1992) established size and value as key factors in explaining the cross-section of expected returns. Momentum has been widely studied in subsequent research by Jegadeesh and Titman (1993), Moskowitz and Grinblatt (1999), Moskowitz, Ooi and Pedersen (2012), and Asness, Moskowitz and Pedersen (2013). Within the machine learning domain, recent studies have addressed methodological advances in asset pricing. Dimensionality reduction methods have been used to estimate factor models by Giglio and Xiu (2016) and Kelly, Pruitt and Su (2017), while shrinkage and selection techniques have been employed to approximate stochastic discount factors and nonlinear mappings of expected returns by Kozak, Nagel and Santosh (2020) and Freyberger, Neuhierl and Weber (2020). The present study builds on this literature by systematically exploring the behavior of various machine learning methods in predicting cross-sectional stock returns, with an emphasis on comparative performance across models.

Parallel to the development in equities, the cryptocurrency literature has grown rapidly. Yermack (2015) was among the first to draw academic attention to the topic. Subsequent studies have proposed structural models tailored to crypto markets, including Biais, Bisière, Bouvard and Casamatta (2019) and Routledge and Zetlin-Jones (2021), while others such as Stoffels (2017), Hubrich (2017), Liu and Tsyvinski

(2020), and Li and Yi (2019) have documented empirical characteristics of crypto investing. These findings underscore the need for flexible, data-driven approaches in environments marked by high volatility, low signal-to-noise ratios, and shifting predictive structures.

Our work extends the empirical literature on return prediction, particularly within the first strand that focuses on explaining cross-sectional differences in expected returns using asset-level characteristics. In the traditional equity setting, this approach is exemplified by Fama and French (2008) and Lewellen (2015), who regress future returns on a limited set of lagged firm characteristics. We adapt this framework to the cryptocurrency market by constructing a comprehensive set of market-based predictors, including measures of price, volatility, liquidity, and trading activity, to forecast returns across a broad cross-section of tokens. This stands in contrast to the second major strand of the literature, which emphasizes time-series prediction of aggregate returns using macroeconomic variables, as in Welch and Goyal (2008), Koijen and Nieuwerburgh (2011), and Rapach and Zhou (2013). Such a distinction is particularly relevant in cryptocurrencies, where well-established macroeconomic predictors are largely absent and asset pricing is not yet anchored to a stable set of aggregate economic fundamentals.

To address this gap, we incorporate machine learning techniques into the cross-sectional asset pricing framework, enabling flexible modeling of nonlinearities and

high-dimensional interactions using observable token-level data. The decentralized and speculative nature of the crypto market further reinforces the need for approaches that rely less on fixed economic structures and more on empirical patterns. Building on these insights, recent research has applied machine learning to cryptocurrency return prediction. Falcon and Lyu (2023) show that tree-based and neural network models significantly improve directional accuracy in major tokens. Lee and Wang (2024) demonstrate the predictive value of high-frequency volatility components, including realized skewness and kurtosis. Liu, Tsyvinski and Xiwu (2022) identify risk factors that are distinct from those found in equities, suggesting that crypto markets require alternative modeling frameworks. Bali, Beckmeyer, Mörke and Weigert (2023) further show that big data inputs improve the performance of option return models. Gu, Kelly and Xiu (2020) provide a comprehensive benchmark of machine learning methods in equity return prediction, which serves as an empirical foundation extended in this study to the structurally unique cryptocurrency domain. Other studies on crypto forecasting remain narrower in scope, often focusing on a small set of tokens or time-series models, such as Borri and Shakhnov (2022) and Trimborn, Li and Härdle (2020).

In contrast, this study comprehensively evaluates machine learning models for cross-sectional return prediction in cryptocurrency markets. We compare linear, regularized, ensemble, and neural network approaches within a unified framework

using 21 market-based features across more than 1,000 tokens. Beyond statistical accuracy, we examine portfolio-level implications, offering new evidence on the performance of flexible algorithms in an environment characterized by limited structure and extreme noise.

The concept of machine learning is context-dependent but can be characterized by three core elements: (a) high-dimensional predictive models, (b) regularization techniques for improving generalization, and (c) efficient algorithms for exploring complex model spaces. The flexibility of high-dimensional models enables the capture of nonlinear and rapidly evolving patterns in cryptocurrency returns, though it also raises overfitting concerns. To mitigate these risks, we employ regularization tools such as cross-validation and penalty-based methods, alongside optimization algorithms that efficiently navigate vast parameter spaces. Cryptocurrency return prediction aligns naturally with this framework. Since each token's cross-sectional risk premium represents an expected excess return, the task is inherently predictive. Unlike traditional finance, where theoretical structures guide model design, the decentralized and fast-changing nature of crypto markets favors flexible, data-driven approaches. Machine learning, emphasizing prediction and out-of-sample generalization, thus provides an ideal foundation for analyzing this setting.

A central challenge in cryptocurrency return prediction lies in the high dimensionality and complexity of the feature space. Our dataset contains 21 market-based predictors derived from prices, volumes, momentum, and volatility, many of which are correlated and nonlinear. As in traditional asset pricing, the signal-to-noise ratio is low, and with limited theoretical guidance, the true functional relationship between predictors and expected returns is uncertain. Machine learning addresses these issues through variable selection, dimension reduction, and flexible functional forms, enabling the extraction of stable signals from noisy, high-dimensional data. Moreover, the diversity of algorithms, from penalized regressions to tree-based models and deep neural networks, facilitates exploration of nonlinearities and interactions often missed by rigid parametric methods. Regularization and out-of-sample validation further control overfitting, allowing these flexible models to generalize effectively in the volatile and complex crypto environment.

To address the empirical challenges outlined above, we implement a comprehensive suite of machine learning models drawn from key branches of the predictive modeling literature. Our selection includes linear regression, penalized generalized linear models, dimension reduction techniques such as principal component regression (PCR) and partial least squares (PLS), ensemble tree-based methods including random forests, XGBoost, LightGBM, and CatBoost, as well as feedforward

neural networks of varying depth. While some of these algorithms, such as CatBoost and LightGBM, were originally designed for classification tasks, they support regression objectives and have demonstrated strong performance on tabular data with complex nonlinearities. In particular, we include CatBoost not only for its competitive structure but also as a test case for assessing whether a classification-oriented algorithm can remain effective in a return prediction setting.

The key finding of this study is that machine learning models, especially tree based approaches such as XGBoost, demonstrate strong effectiveness in extracting predictive signals from noisy and high dimensional cryptocurrency data. Compared to linear baselines, these models generate consistently higher out-of-sample R^2 and yield significantly improved risk-adjusted investment performance. The results demonstrate that machine learning is not only valuable for understanding return-generating processes in decentralized markets, but also holds clear practical relevance for tasks such as token screening, portfolio construction, and algorithmic trading, thereby supporting its growing role in the financial technology ecosystem.

To benchmark the economic value of our predictive models, we evaluate risk-adjusted returns from value-weighted long–short decile strategies. The results show that machine learning models, particularly XGBoost, achieve substantially higher out-of-

sample Sharpe ratios than traditional linear benchmarks. For example, in the decile sort excluding COCOS-BCX, XGBoost reaches a Sharpe ratio of 0.726 compared to 0.2367 for OLS. The superior portfolio-level performance arises from aggregation, which reduces idiosyncratic noise in volatile and illiquid tokens, thereby improving the signal-to-noise ratio. Moreover, value-weighted portfolios generate stronger economic gains than equal-weighted ones, as the latter are diluted by small-cap tokens with high volatility. These findings underscore both the importance of portfolio construction and the practical advantage of machine learning in value-weighted strategies that more accurately reflect real-world capital allocation.

Our feature importance analysis shows that the most predictive signals for cryptocurrency returns differ sharply from those in equity markets. While momentum dominates stocks, volatility-related features such as return skewness (RETSKEW), volume volatility (STDPRCOL), return kurtosis (RETKURT), and maximum daily return (MAXRET), consistently rank highest across models in cryptocurrency. Token price (PRC) and trading volume (VOL) are also important, whereas market capitalization (MCAP) has moderate but stable predictive power, particularly in linear models. Momentum variables receive little weight, suggesting that risk-related and liquidity characteristics, rather than past trends, drive crypto returns. These findings

highlight the ability of machine learning to uncover structural signals distinct from traditional finance.

Our results underscore the potential of machine learning for improving risk premium measurement in cryptocurrency markets. By approximating the conditional expectation of future excess returns from observable characteristics, these methods capture nonlinear patterns and mitigate overfitting in high-dimensional settings, delivering superior predictive performance relative to traditional benchmarks. However, these improvements are primarily statistical rather than structural, as machine learning enhances forecasting without directly explaining the economic mechanisms behind asset prices. Interpreting such predictions within asset pricing theory remains an important challenge. Recent work combining machine learning with equilibrium models (Kelly, Pruitt, and Su, 2019; Gu, Kelly, and Xiu, 2019) points to a promising direction for future research.

The paper is organized as follows. Section I describes the data. Section II examines the cross-sectional return predictors. Section III builds the cryptocurrency factor models. Section IV investigates potential mechanisms behind the cryptocurrency size and momentum effects. Section V provides additional results. Finally, Section VI concludes.

2 Methodology

This section presents the machine learning framework for predicting cross-sectional cryptocurrency returns. Each method is described through three components. First, we define a statistical model that maps token characteristics such as price, volume, and volatility to one-week-ahead excess returns in a potentially nonlinear manner. Second, we specify the estimation objective: all models are trained to minimize out-of-sample prediction errors, primarily using the mean squared error (MSE) as the loss function.

To mitigate overfitting in high-dimensional settings, we apply regularization strategies that penalize model complexity and enhance robustness to outliers, enabling better generalization to unseen data. Third, we describe the learning algorithms used to explore the vast space of potential model specifications. Nonlinear transformations and interactions can rapidly expand this space, particularly in our setting with complex, time-varying relationships across assets. Each algorithm is therefore designed to efficiently search for and identify high-performing models within this large, dynamic feature space.

To ensure transparency and reproducibility, we report all parameter settings used in our machine learning models, including hyperparameters (e.g., learning rate,

regularization strength, number of trees), training procedures (validation design and data splits), and implementation details (optimization method, early stopping, random seed). Because these parameters strongly affect predictive performance, even minor adjustments can alter out-of-sample results. For fairness across methods, we employ a consistent rolling-validation framework for hyperparameter tuning, as described in Section 3.5. The complete model specifications are documented in Section 4.2 to facilitate replication and future research extensions.

At the heart of our prediction framework is a simple additive model that expresses the weekly excess return of a cryptocurrency as the sum of its conditional expectation and an error

$$r_{i,t+1} = \mathbb{E}_t(r_{i,t+1}) + \varepsilon_{i,t+1}, \quad (1)$$

where

$$\mathbb{E}_t(r_{i,t+1}) = g^*(z_{i,t}). \quad (2)$$

In this formulation, $r_{i,t+1}$ represents the return of cryptocurrency i during week $t+1$, while $z_{i,t} \in \mathbb{R}^P$ denotes a P -dimensional vector of observable features at time t . The unknown function $g^*(\cdot)$ captures the relationship between these features and the

expected return. Our goal is to use data-driven techniques to estimate this function and generate reliable out-of-sample forecasts.

Model stability and interpretability are ensured through two key constraints. First, the prediction function is assumed invariant across both time and cryptocurrencies, meaning the same functional form applies to all tokens and all weeks. This design enables the model to pool information and learn consistent patterns across the market. Second, the information set is limited to current-period features only, meaning predictions rely solely on $z_{i,t}$ without lagged variables or peer token data. This restriction prevents the use of unavailable or future information and preserves a strictly out-of-sample forecasting structure.

2-1 Rolling Forecasting, Sample Splitting

To evaluate the out-of-sample predictive performance of each model in a manner consistent with practical implementation, we adopt a rolling-origin forecasting framework with expanding training windows. This approach simulates how predictive models would operate in real world settings, using only information available at the time of prediction and updating as new data becomes available. We generate annual rolling forecasts from 2019 to 2022. For each forecast year t , the model is trained on all available data from 2014 up to year t , validated on data from the immediately

preceding year $t - 1$, and tested on year t . This results in a moving validation window and an expanding training window, ensuring that hyperparameters are tuned on recent but non-overlapping data, while test performance is evaluated strictly out-of-sample.

This design preserves the temporal ordering of financial data and follows the standard three-way split widely used in machine learning (Gu, Kelly, and Xiu, 2020). The training set estimates model parameters for given hyperparameter values, while the validation set tunes those hyperparameters by simulating out-of-sample prediction. For each candidate configuration, forecasts are generated on the validation set, and the one minimizing validation loss is selected. The test set, never used during estimation or tuning, provides the final benchmark for truly out-of-sample performance.

This framework is essential because most machine learning models depend on hyperparameters such as the number of trees, learning rate, and regularization strength that govern model complexity and affect overfitting. Given limited theoretical guidance, data-driven validation is the standard approach. However, since the validation set informs model selection, only the test set offers a genuinely out-of-sample evaluation.

Although our validation window is shorter than that used in some traditional equity-based studies (e.g., 12 years in Gu et al.), this modification reflects the relatively limited history of the cryptocurrency market. Nevertheless, we maintain the same core logic:

hyperparameters are tuned on recent data, but performance is evaluated exclusively on unseen future observations. This ensures a forward-looking, unbiased assessment of model generalization.

Figure 1 illustrates our sample-splitting structure in detail. For instance, when forecasting returns in 2020, the model is trained on historical data from 2014 to 2020, validated on 2019, and tested on 2020. This process is repeated for each year from 2019 to 2022, with the training window progressively expanding over time to incorporate newly available data. This design allows the model to continuously adapt to updated market conditions while ensuring strict out-of-sample testing for each forecast year.

This methodology is widely adopted in empirical finance research, particularly for return prediction tasks involving financial time-series data (e.g., Gu, Kelly, and Xiu, 2020). While our validation window is shorter than theirs due to the relatively limited time span of the cryptocurrency market, the core principles remain the same: only past data is used for training, hyperparameters are tuned on the most recent period, and model performance is evaluated on truly unseen future data. Such a structure ensures that our performance assessments are free from look-ahead bias and reflective of realistic forecasting constraints.

< Figure 1 >

2-2 Simple Linear Regression

We begin our modeling approach with the most basic method in our analysis: the simple linear predictive regression model estimated via Ordinary Least Squares (OLS). While this model is not expected to perform well in a high-dimensional cryptocurrency setting, it serves as a useful benchmark for highlighting the strengths of more advanced machine learning methods.

The simple linear model assumes that the conditional expectation function $g^*(\cdot)$ can be approximated by a linear combination of raw feature inputs and a parameter vector θ , such that:

$$g(z_{i,t}; \theta) = z'_{i,t} \theta. \quad (3)$$

This formulation implies a linear specification and does not capture any nonlinear effects or interactions between features.

Our baseline estimation of the simple linear model uses a traditional least squares objective function, often referred to as the “ l_2 ” loss:

$$\mathcal{L}(\theta) = \frac{1}{NT} \sum_{i=1}^N \sum_{t=1}^T \left(r_{i,t+1} - g(z_{i,t}; \theta) \right)^2. \quad (4)$$

Minimizing $\mathcal{L}(\theta)$ yields the pooled OLS estimator. The main advantage of the l_2

objective is its closed-form analytical solution, which avoids complex optimization procedures and serves as a computationally efficient starting point when modeling cryptocurrency return predictions.

2-2-1 Extension: Robust objective functions

In certain cases, replacing Equation (4) with a weighted least squares objective can enhance predictive performance. A general form of the weighted objective function is:

$$\mathcal{L}_W(\theta) = \frac{1}{NT} \sum_{i=1}^N \sum_{t=1}^T w_{i,t} \left(r_{i,t+1} - g(z_{i,t}; \theta) \right)^2. \quad (5)$$

This weighting allows the econometrician to place greater emphasis on observations deemed more economically or statistically informative. For example, one weighting scheme sets $w_{i,t}$ inversely proportional to the number of available assets in period t , ensuring that each time period contributes equally to model estimation. This effectively applies equal weighting across all observations in each time slice, regardless of market coverage.

Another variant sets $w_{i,t}$ proportional to the market capitalization of asset i at time t , thereby assigning greater influence to larger assets. This is motivated by the fact that although smaller cryptocurrencies account for the majority of asset count, they represent a very small fraction of total market capitalization. In contrast, large-cap

cryptocurrencies, such as Bitcoin or Ethereum, dominate total traded value and thus warrant heavier weighting under this rationale.

Heavy tails are a well-documented feature of asset returns and predictor variables, particularly in the context of cryptocurrencies. Because the standard least squares objective emphasizes large errors due to its quadratic form, the presence of outliers can distort estimation and degrade model stability.

To address this, the statistics and machine learning literature has developed robust variants of the least squares loss. One commonly used alternative is the **Huber loss**, which blends the squared loss for small residuals and the absolute loss for large residuals. The robust Huber objective function is defined as:

$$\mathcal{L}_H(\theta) = \frac{1}{NT} \sum_{i=1}^N \sum_{t=1}^T H(r_{i,t+1} - g(z_{i,t}; \theta), \xi), \quad (6)$$

with the Huber loss function $H(x; \xi)$ defined as:

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

The tuning parameter ξ determines the threshold at which the loss transitions from quadratic to linear, and can be adaptively selected based on the data.

While this section introduces robust objective functions in the context of simple linear models, they are readily applicable across the broader machine learning methods

considered in this paper. In our empirical analysis, we evaluate the predictive improvements offered by robust loss functions in settings prone to outliers and volatility, such as cryptocurrency markets.

2-3 Penalized linear

In the context of cryptocurrency return prediction, the simple linear model often fails when dealing with a large set of predictive signals. As the number of input features P approaches the number of time-series observations T , the model becomes prone to overfitting noise rather than capturing true signal. This issue is particularly pronounced in the cryptocurrency space, where the signal-to-noise ratio is notoriously low due to market volatility, speculative behavior, and limited fundamentals.

To mitigate overfitting, a key strategy is to constrain the number of effective parameters in the model. A common approach in machine learning is to introduce a penalty term to the objective function to enforce model simplicity—this technique is known as *regularization*. Although regularization typically reduces in-sample fit, its purpose is to improve out-of-sample performance by preventing the model from capturing random noise in the training set.

The statistical formulation for our penalized linear model follows the same structure as the baseline linear model in Equation (3), using only the raw, untransformed predictors. The main difference is the inclusion of a penalty term applied to the loss function:

$$\mathcal{L}(\theta; \cdot) = \mathcal{L}(\theta) + \phi(\theta; \cdot). \quad (7)$$

Among various penalty choices, we adopt the widely used *elastic net* regularization, defined as:

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

The elastic net introduces two non-negative hyperparameters λ and ρ , and nests two popular regularization methods as special cases. When $\rho = 0$, the penalty reduces to the Lasso, which uses an l_1 - norm and promotes sparsity by setting some coefficient estimates exactly to zero, effectively acting as a variable selection method. When $\rho = 1$, the penalty corresponds to Ridge regression, which applies an l_2 - norm that shrinks all coefficients towards zero without eliminating them entirely, serving as a shrinkage method to control variance.

For intermediate values of ρ , the Elastic Net model combines the advantages of both Lasso and Ridge, promoting sparsity while maintaining estimation stability. This is particularly valuable in cryptocurrency return prediction, where predictive signals are

often noisy, correlated, and high-dimensional. The model simultaneously performs variable selection through the l_1 component and coefficient shrinkage via the l_2 penalty, enabling it to isolate a concise set of informative features while guarding against overfitting. This hybrid regularization structure makes Elastic Net especially suitable for capturing meaningful structure in volatile and complex crypto market data.

2-4 PCR and PLS

While penalized linear models address high dimensionality by shrinking or eliminating individual predictors, such methods may lead to suboptimal forecasting performance when the features are highly correlated, a common scenario in cryptocurrency data, where various price based and volume based indicators often exhibit collinearity. In such cases, selecting a subset of features (as in lasso regression) may be less effective than constructing aggregate representations of the predictors through averaging.

Rather than performing feature selection, dimension reduction approaches like principal component regression (PCR) and partial least squares (PLS) aim to project the original predictor space into a smaller number of uncorrelated linear combinations. These transformations help isolate signal from noise and reduce redundancy, which is particularly useful in the noisy and highly volatile crypto asset environment.

PCR applies principal component analysis (PCA) to the original predictor matrix to obtain a small set of orthogonal components that preserve most of the variance in the data. In the second step, a subset of these components is used in the predictive regression. However, a limitation of PCR is that it forms components based solely on predictor variance, without regard to their relevance for return prediction, potentially discarding useful predictive information.

In contrast, PLS explicitly considers the forecast target when constructing components. For each predictor j , a univariate OLS regression is run to estimate its marginal relationship to cryptocurrency returns. These individual coefficients ϕ_j are then used to form weighted averages of the predictors, emphasizing those with greater predictive power. This process continues iteratively by orthogonalizing both the predictors and the target with respect to previously constructed components, yielding a set of components that maximize covariance with the target variable.

Our implementation of PCR and PLS begins from the vectorized representation of the linear model in Equations (1) – (3). Specifically, we rewrite the predictive regression equation $r_{i,t+1} = z'_{i,t}\theta + \varepsilon_{i,t+1}$ as :

$$R = Z\theta + E, \quad (9)$$

where R denotes the $NT \times 1$ vector of cryptocurrency excess returns $r_{i,t+1}$, Z is the $NT \times P$ matrix of stacked predictive signals $z_{i,t}$, and E is the corresponding vector of residuals.

Both PCR and PLS adopt a common strategy for addressing high-dimensionality: they compress the original predictor matrix Z of dimension P into a lower-dimensional representation composed of K linear combinations. The forecasting model used for both methods is then:

$$R = (Z\Omega_k)\theta_K + \tilde{E}. \quad (10)$$

where Ω_k is a $P \times K$ matrix with columns $\omega_1, \omega_2, \dots, \omega_K$, representing the linear transformation weights used to construct the reduced components. The transformed matrix $Z\Omega_k$ serves as the dimension-reduced representation of the original crypto factor set, and θ_K becomes a $K \times 1$ coefficient vector in place of the full $P \times 1$ vector.

PCR determines the weights Ω_k recursively, with the j^{th} component calculated by solving:

$$\omega_j = \arg \max_{\omega} \text{Var}(Z_{\omega}), \quad \text{s.t. } \omega' \omega = 1, \text{Cov}(Z_{\omega}, Z_{\omega_l}) = 0, \quad l = 1, 2, \dots, j-1. \quad (11)$$

Conceptually, PCR seeks K linear combinations that best reproduce the structure of the full predictor space. Importantly, this approach does not incorporate any information

about the return target when selecting components. Instead, PCR emphasizes capturing the maximum variance within the feature space, regardless of whether such variation is predictive of future crypto returns. The weight matrix Ω_k is obtained via singular value decomposition (SVD), making PCR highly efficient from a computational perspective.

In contrast, PLS performs component extraction with the return prediction objective in mind. It searches for directions in the predictor space that are most strongly associated with the target R . Specifically, the j^{th} PLS component weight is calculated by solving:

$$\omega_j = \arg \max_{\omega} \text{Cov}^2(R, Z_{\omega}), \quad \text{s.t. } \omega' \omega = 1, \text{Cov}(Z_{\omega}, Z_{\omega_l}) = 0, \quad l = 1, 2, \dots, j-1. \quad (12)$$

This formulation reveals a key distinction between PCR and PLS: while PCR focuses solely on predictor variance, PLS prioritizes components with high forecast relevance. In doing so, PLS is willing to accept a less precise approximation of Z to obtain components that better explain returns. The optimization problem in Equation (12) is solvable through standard routines, notably the SIMPLS algorithm proposed by de Jong (1993). Once Ω_k is determined, the final prediction coefficients θ_K are estimated in both PCR and PLS via OLS regression of R on $Z\Omega_k$. The number of components K is

treated as a tuning parameter and is selected using out-of-sample validation to ensure optimal performance in cryptocurrency return prediction tasks.

2-5 Generalized linear

Linear models remain widely used in empirical finance due to their interpretability and simplicity, often serving as a first-order approximation to more complex data-generating processes. However, when the “true” underlying relationship between features and returns is nonlinear, as is often the case in cryptocurrency markets, restricting the functional form to linearity introduces approximation error stemming from model misspecification. Let $g^*(z_{i,t})$ denote the true model, and $g(z_{i,t}; \theta)$ the functional form specified by the econometrician. Let $g(z_{i,t}; \hat{\theta})$ and $\hat{r}_{i,t+1}$ denote the fitted model and its corresponding return forecast. The model’s forecast error can be decomposed as:

$$r_{i,t+1} - \hat{r}_{i,t+1} = g^*(z_{i,t}) - g(z_{i,t}; \theta) + \underbrace{g(z_{i,t}; \theta) - g(z_{i,t}; \hat{\theta})}_{\text{estimation error}} + \epsilon_{i,t+1}$$

approximation error
estimation error
intrinsic error

The intrinsic error represents the unpredictable component of returns, arising from exogenous shocks such as news events or regulatory shifts, and is inherently irreducible. Estimation error stems from sampling variation and can, in principle, be reduced by increasing the dataset size. Approximation error, which results from using an overly

rigid model, is within the econometrician's control and can be mitigated by adopting more flexible functional forms.

To address this, we introduce a class of nonparametric extensions to the linear model $g(\cdot)$, allowing for increased flexibility while applying regularization to guard against overfitting. The most direct of these extensions is the generalized linear model, which enhances linearity by incorporating nonlinear transformations of predictors as additive terms. This makes it the most comparable nonlinear analog to the OLS-based models in Sections 2.2 and 2.3.

Our generalized linear specification modifies the standard linear model by expanding each predictor using a spline basis. Specifically, we define:

$$g(\mathbf{z}; \theta, p(\cdot)) = \sum_{j=1}^P p(\mathbf{z}_j)' \theta_j, \quad (13)$$

where $p(\cdot) = (p_1(\cdot), p_2(\cdot), \dots, p_K(\cdot))'$ is a vector of spline basis functions, and $\theta = (\theta_1, \theta_2, \dots, \theta_N)$ is now a $K \times N$ matrix of coefficients. We use a second-order spline expansion of the form: $(1, \mathbf{z}, (\mathbf{z} - c_1)^2, \dots, (\mathbf{z} - c_{K-2})^2)$, where $c_1, c_2, \dots, c_{K-2}$ are pre-defined knots.

Due to the additive nature of higher-order terms, this generalized model can be estimated using the same least squares framework introduced in Section 2.2. We

optionally incorporate Huber’s robust loss function to reduce the influence of outliers, which are common in cryptocurrency returns. However, since the spline expansion substantially increases the number of parameters, regularization becomes essential to control overfitting and maintain generalizability.

To this end, we apply the group lasso penalty, which is well-suited for spline-based expansions. Its penalty function is given by:

$$\phi(\theta; \lambda, K) = \lambda \sum_{j=1}^p \left(\sum_{k=1}^K \theta_{j,k}^2 \right)^{\frac{1}{2}}. \quad (14)$$

The group lasso either retains or removes all K spline terms associated with a given predictor j , enabling structured sparsity. We incorporate this penalty within the general objective function defined in Equation (7). The model is optimized using an accelerated proximal gradient descent algorithm, similar to that used for elastic net. The two main hyperparameters λ and K are selected via cross-validation to balance model complexity and predictive accuracy.

2-6 Boosted regression trees and random forests

The generalized model in Equation (13) captures nonlinear effects of individual predictors, but does not accommodate interactions between them. One strategy to incorporate interactions is to augment the model with multivariate transformations of

predictors. However, while univariate expansions (e.g., spline basis) increase the number of parameters linearly with K , multivariate interactions induce combinatorial explosion in parameter size. Without strong prior knowledge, this quickly renders the generalized linear model computationally infeasible.

As an alternative, we employ **ensemble tree-based models**, which handle nonlinearities and high-order predictor interactions flexibly and nonparametrically. Unlike linear or additive models, decision trees recursively partition the predictor space by thresholding on feature values to group similar observations. A tree is constructed through sequential "splits" on features that best separate the data according to a chosen loss criterion. Each leaf node represents a subset of the predictor space, with the outcome modeled as a constant (usually the sample mean) within that region. The prediction function of a single tree $\mathcal{T}$, with K leaves and depth L , can be written as:

$$g(z_{i,t}; \theta, K, L) = \sum_{k=1}^K \theta_k \mathbf{1}_{\{z_{i,t} \in C_k(L)\}}, \quad (15)$$

where $C_k(L)$ denotes the k^{th} region in the partition of the predictor space, defined by up to L nested indicator functions. The example in Figure 2 has the following prediction equation:

$$\begin{aligned} g(z_{i,t}; \theta, 3, 2) = & \theta_1 \mathbf{1}_{\{size_{i,t} < 0.5\}} \mathbf{1}_{\{b/m_{i,t} < 0.3\}} + \theta_2 \mathbf{1}_{\{size_{i,t} < 0.5\}} \mathbf{1}_{\{b/m_{i,t} \geq 0.3\}} \\ & + \theta_3 \mathbf{1}_{\{size_{i,t} \geq 0.5\}}. \end{aligned}$$

And This figure presents the diagrams of a regression tree (left) and its equivalent representation (right) in the space of two characteristics (size and value). The terminal nodes of the tree are colored in blue, yellow, and red. Based on their values of these two characteristics, the sample of individual stocks is divided into three categories.

< Figure 2 >

To grow a tree is to find partitions that best differentiate among the potential outcomes. The specific predictor variable on which a branch is split, and the corresponding threshold, are selected to minimize forecast error. Due to the vast number of potential tree configurations, exact optimization is infeasible. To address this, we adopt a widely used greedy algorithm introduced by Breiman et al. (1984). This method constructs the tree recursively by locally optimizing forecast error at each branching point. At each level, a predictor and split value are selected to maximize the difference in average outcomes across the resulting partitions.

The forecast error associated with a branch C is referred to as “impurity,” which measures the homogeneity of observations within the partition. We use the standard l_2 impurity metric:

$$H(\theta, C) = \frac{1}{|C|} \sum_{z_{i,t} \in C} (r_{i,t+1} - \theta)^2, \quad (16)$$

where $|C|$ is the number of observations in region C . The optimal prediction θ for the region is simply its sample mean: $\theta = \frac{1}{|C|} \sum_{z_{i,t} \in C} r_{i,t+1}$. This procedure is equivalent to choosing the branch C that locally minimizes the impurity. Splitting continues until a prespecified stopping rule is triggered, such as a maximum depth or minimum leaf size, which is selected using a validation set.

A key advantage of decision trees is their invariance to monotonic transformations of inputs. They can handle both categorical and numerical predictors in a unified framework, model complex nonlinear relationships, and approximate high-order interactions: a tree of depth L can capture up to $(L-1)$ - way interactions. However, this flexibility comes at the cost of high variance and overfitting risk. As such, trees must be regularized. In our empirical work, we apply two ensemble-based regularization techniques that aggregate predictions across multiple trees.

The first regularization approach is boosting, which recursively combines forecasts from many shallow trees. Individually, shallow trees serve as “weak learners” with limited predictive power. Boosting theory suggests that combining many weak learners can result in a more robust and accurate model, known as a “strong learner.”

The boosting procedure used in our analysis, implemented via XGBoost, LightGBM, and CatBoost, is based on gradient boosted regression trees (GBRT). The

process begins by fitting a shallow tree (e.g., with depth $L=1$) to the training data. Due to its simplicity, this tree typically exhibits high bias. A second tree, again of depth L , is then fit to the residuals from the first model. The predictions from the two trees are added to form the new model. To reduce the risk of overfitting, the contribution of the second tree is scaled by a shrinkage factor $v \in (0,1)$.

This process is repeated iteratively. At each step b , a new shallow tree is trained on the residuals from the model composed of the previous $b-1$ trees. Its prediction is added with shrinkage weight v . After B iterations, the final model is the sum of all trees, resulting in an additive ensemble governed by three hyperparameters: tree depth L , learning rate v , and number of boosting rounds B , all selected via validation.

Like boosting, random forest is an ensemble method that aggregates predictions across multiple trees. It is a variant of the broader technique known as bootstrap aggregation, or “bagging”. In this framework, B bootstrap samples are drawn from the data, and a separate regression tree is trained on each. The final prediction is the average of these individual tree forecasts. While trees fit on individual bootstrap samples may be deep and overfit their respective subsamples, aggregating their outputs reduces variance and stabilizes performance.

To further reduce correlation among trees, random forests use feature subsampling, often referred to as “dropout.” At each split, only a randomly selected subset of features is considered for splitting. This helps prevent dominant features (e.g., market capitalization) from driving early splits in most trees. As a result, trees become more diverse, and the ensemble achieves better variance reduction than standard bagging. The tuning parameters for the random forest, tree depth L , number of features per split, and number of bootstrap samples B , are determined via validation.

2-7 Neural networks

The final nonlinear modeling approach we explore is the artificial neural network, widely regarded as one of the most powerful tools in modern machine learning. Neural networks are supported by strong theoretical foundations, known as “universal approximators,” capable of modeling any smooth nonlinear mapping (Hornik et al., 1989; Cybenko, 1989). Their flexibility and expressive power have made them the method of choice in complex tasks such as computer vision, natural language processing, and strategic game playing.

In financial applications, particularly in the cryptocurrency market where predictor relationships may be highly nonlinear and noisy, neural networks offer a valuable modeling alternative. However, their complexity also makes them relatively opaque

and harder to interpret compared to simpler methods like linear models or tree-based approaches.

Our implementation focuses on the standard feed-forward neural network architecture. This comprises an input layer of raw features, one or more hidden layers that apply nonlinear transformations, and an output layer that generates the final prediction. Analogous to biological systems, neurons in different layers are connected by weighted links or “synapses” transmitting information forward from input to output. Figure 3 shows two illustrative examples.

In our illustrative example, the input layer has four features $(z_1, \dots, z_4)$. The simplest version, with no hidden layers, corresponds to a basic linear model. Each input feature is multiplied by a corresponding weight from the parameter vector θ , and the result is aggregated as: $\theta_0 + \sum_{k=1}^4 z_k \theta_k$; In this setup, the network behaves exactly like a linear regression model.

To capture more complex relationships, we incorporate one or more hidden layers. For example, adding a hidden layer with five neurons allows the model to form nonlinear combinations of the inputs. Each neuron computes a weighted sum of the inputs and passes it through a nonlinear activation function $f(\cdot)$. For the second neuron in the hidden layer, the transformation can be written as: $x_2^{(1)} = f(\theta_{2,0}^{(0)} + \sum_{j=1}^4 z_j \theta_{2,j}^{(0)})$.

The outputs from these hidden units are then linearly combined in the output layer to produce the final forecast: $g(z; \theta) = \theta_0^{(1)} + \sum_{j=1}^5 x_j^{(1)} \theta_j^{(1)}$. In this example, the total number of parameters is $31 = (4+1) \times 5 + 6$, accounting for both the hidden layer weights and the output layer weights.

Designing a neural network involves selecting the number of layers, number of neurons per layer, and connection patterns. Although the universal approximation theorem suggests that even a single hidden layer can approximate any smooth function, recent findings show that deeper networks can often match performance with fewer parameters. However, for smaller datasets, as is often the case in weekly cryptocurrency data, simpler networks often generalize better. Deeper architectures are harder to train due to non-convexity and sensitivity in gradient-based optimization, often suffering from issues like vanishing or exploding gradients during back-propagation.

Because identifying the best neural network architecture through exhaustive search is computationally infeasible, we instead predefine five architectures of increasing depth:

- **NN1:** One hidden layer with 32 neurons
- **NN2:** Two hidden layers with 32 and 16 neurons
- **NN3:** Three hidden layers with 32, 16, and 8 neurons

- **NN4:** Four hidden layers with 32, 16, 8, and 4 neurons
- **NN5:** Five hidden layers with 32, 16, 8, 4, and 2 neurons

All networks are fully connected, meaning each unit receives input from every neuron in the previous layer. The structure follows the geometric pyramid rule (Masters, 1993). By comparing performance across NN1–NN5, we evaluate how network depth impacts prediction accuracy in cryptocurrency return modeling.

Each neuron in the network applies the same nonlinear activation function, chosen from common options such as sigmoid, tanh, or softmax. In this study, we use the Rectified Linear Unit (ReLU), which is defined as:

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

ReLU is widely adopted for its computational efficiency and ability to promote sparsity in activations, thus speeding up training and reducing overfitting in high-dimensional neural architectures.

Our neural network model follows the general recursive structure. Let $K^{(l)}$ denote the number of neurons in layer $l = 1, \dots, L$. Define the output of neuron k in layer l as $x_k^{(l)}$. The vector of outputs from layer l , augmented to include a constant, is defined as $x^{(l)} = (1, x_1^{(l)}, \dots, x_{K^{(l)}}^{(l)})'$. The input layer is defined using the raw predictors

as $x^{(0)} = (1, z_1, \dots, z_N)'$. The recursive computation of the neural network at each neuron in layer $l > 0$ is:

$$x_k^{(l)} = \text{ReLU}\left(x^{(l-1)'} \theta_k^{(l-1)}\right), \quad (17)$$

with the final prediction output defined as:

$$g(z; \theta) = x^{(L-1)'} \theta^{(L-1)}. \quad (18)$$

The total number of weight parameters in each hidden layer l is $K^{(l)}(1 + K^{(l-1)})$, and the output layer adds $1 + K^{(L-1)}$ additional weights.

We estimate the neural network parameters by minimizing a penalized l_2 loss function, which evaluates prediction error. Unlike tree-based models that use greedy, stage-wise optimization, neural networks allow joint optimization of all parameters simultaneously, which provides better flexibility but also significantly increases computational complexity. To address this, we employ stochastic gradient descent (SGD) for model training.

Instead of computing gradients across the entire dataset at each iteration (as in full-batch gradient descent), SGD computes gradients from small, randomly selected mini-batches. While this introduces variance into each update, it substantially reduces computational burden and accelerates convergence, an important advantage when

training high capacity networks. For the reasons described above, namely the high nonlinearity and parameter complexity of neural networks, regularization plays an especially important role during training. In addition to the l_1 penalization of model weights, we apply four additional regularization techniques: learning rate shrinkage, early stopping, batch normalization, and ensembles.

Learning Rate Shrinkage.

A critical tuning parameter in stochastic gradient descent (SGD) is the learning rate, which determines the step size of weight updates. When gradients begin to vanish, a high learning rate can introduce noise, destabilizing the optimization process. To address this, we adopt the "learning rate shrinkage" algorithm of Kingma and Ba (2014).

Early Stopping.

This is a widely-used regularization method in machine learning. It begins with an initial parameter guess (e.g., all θ values near zero) and progressively updates parameters to reduce training error. After each update, predictions are evaluated on a validation set. When the validation error begins to increase, typically before the training error reaches its minimum, the optimization is stopped early. This approach helps shrink parameters back toward their initial values and is a popular alternative to l_2 -

penalization due to its lower computational cost. In this paper, we use early stopping in conjunction with l_1 - regularization.

Batch Normalization.

Proposed by Ioffe and Szegedy (2015), batch normalization is a technique used to normalize the input distribution of hidden layers during training. This helps address the issue of internal covariate shift, where the distribution of inputs varies between the training and validation sets. At each training step (or “batch”), the algorithm de-means and rescales inputs to stabilize learning. This is especially useful in deep neural networks with multiple layers and nonlinearities.

Ensemble Learning.

Following Hansen and Salamon (1990) and Dietterich (2000), we implement an ensemble strategy by training multiple neural networks using different random seeds. Each network produces independent predictions, and we aggregate them through averaging. This reduces prediction variance and improves robustness, particularly in noisy domains like cryptocurrency markets, where optimization randomness can lead to divergent outputs.

< Figure 3 >

2-8 Performance evaluation

To assess the predictive performance of our models for forecasting individual excess cryptocurrency returns, we compute the out-of-sample R^2 following the metric proposed by Gu, Kelly, and Xiu (2020):

$$R_{oos}^2 = \frac{\sum_{(i,t) \in \mathcal{T}_3} (r_{i,t+1} - \hat{r}_{i,t+1})^2}{\sum_{(i,t) \in \mathcal{T}_3} r_{i,t+1}^2}, \quad (19)$$

where $\mathcal{T}_3$ denotes the test set, which is strictly out-of-sample and does not participate in model training or hyperparameter tuning. The R_{oos}^2 aggregates prediction errors across all cryptocurrencies and time periods, providing a panel-level measure of overall model accuracy.

An important feature of this R^2 specification is that the denominator uses the total sum of squared excess returns without demeaning. This choice is especially relevant in the cryptocurrency market, where return distributions are highly volatile and often exhibit extreme values. In many empirical asset pricing studies, predictions are evaluated relative to historical average returns. However, this approach tends to understate the difficulty of forecasting asset-level returns in volatile environments like crypto markets. For most tokens, historical mean returns are noisy and unreliable, and using them as a benchmark can overstate model performance. To avoid this issue, we follow the convention of comparing model forecasts to a naive benchmark of zero. That

is, a model only earns a positive R_{oos}^2 if it produces lower squared errors than a naive forecast of zero excess return. This stricter evaluation standard helps ensure that any documented predictive power is not simply the result of fitting to noisy historical averages, but rather reflects true return forecasting ability in the cross-section of cryptocurrencies.

On the other hand, to evaluate whether the predictive accuracy of one model significantly exceeds that of another, we adopt a modified version of the Diebold and Mariano (1995) test for pairwise comparison of out-of-sample performance. While traditional applications of the Diebold-Mariano test assume weak time-series dependence, this assumption may not hold in the cross-sectional context of cryptocurrency markets, where return series across tokens may exhibit substantial contemporaneous correlation.

We follow the approach of Gu, Kelly, and Xiu (2020) and adapt the test to our setting by comparing the cross-sectional average of squared prediction errors across tokens, rather than testing prediction errors for each asset individually. Specifically, to assess whether method (1) outperforms method (2), we define the test statistic as:

$$DM_{12} = \frac{\bar{d}_{12}}{\hat{\sigma}_{\bar{d}_{12}}}, \quad (20)$$

where

$$d_{12,t+1} = \frac{1}{n_{3,t+1}} \sum_{i=1}^{n_{3,t+1}} ((\hat{e}_{i,t+1}^{(1)})^2 - (\hat{e}_{i,t+1}^{(2)})^2) \quad (21)$$

and $\hat{e}_{i,t+1}^{(1)}$, $\hat{e}_{i,t+1}^{(2)}$ denote the one-step-ahead prediction errors for cryptocurrency i at time t under model 1 and model 2, respectively. The quantity $n_{3,t+1}$ is the number of tokens in the test sample at time $t + 1$. The mean of $d_{12,t}$ over all testing weeks is denoted by $\bar{d}_{12}$, and the standard error $\hat{\sigma}_{\bar{d}_{12}}$ is computed using the Newey-West estimator to correct for any potential heteroscedasticity or autocorrelation. This adjusted test is more suitable for high-frequency, cross-sectional prediction settings like ours, where each $d_{12,t}$ is based on a large panel of tokens per period but little serial dependence across time. As such, this implementation satisfies mild regularity conditions and yields valid p -values for testing significant differences in forecast accuracy between competing models.

2-9 Variable importance and marginal relationships in Crypto Market

To enhance the interpretability of our forecasting models, we evaluate the relative importance of each input variable using feature attribution methods tailored to the structure of each model class. For tree-based models such as XGBoost and Random Forest, we rely on the built-in importance metrics based on impurity reduction or gain,

which summarize the frequency and quality of feature splits across the ensemble. These scores help identify which predictors most consistently contribute to lowering out-of-sample prediction error.

For neural network models, we adopt permutation importance as a model-agnostic approach to quantify variable relevance. Specifically, we follow the method introduced by Breiman (2001), in which each feature is randomly permuted across the test set while all others are held fixed. The resulting drop in model performance, measured by the decline in prediction accuracy or out of sample R^2 , serves as an estimate of the feature's marginal contribution. A larger drop indicates a more influential predictor. This technique is particularly suitable for complex and nonlinear models where traditional gradient-based methods may be less stable or interpretable.

In addition to importance rankings, we construct Partial Dependence Plots (PDPs) to visualize the marginal relationship between individual predictors and the model's predicted return. PDPs show the average model response as a function of one feature, while marginalizing over the distribution of all others. Although PDPs assume feature independence and may not fully capture interactions or joint effects, they remain a valuable first-order diagnostic tool. By visualizing the shape and direction of marginal effects, for example, a monotonically increasing return response to recent momentum,

we gain economic intuition about how the model utilizes each input. By combining variable importance scores with graphical interpretations such as PDPs, we aim to reduce the opacity of machine learning models and provide a more transparent view of the signals driving cryptocurrency return predictions. This layered interpretability helps bridge the gap between statistical performance and economic understanding.

3. Data

3.1 Data Source and Sample Selection

In this study, we adopt the empirical framework proposed by Liu and Tsyvinski (2020) for analyzing the cryptocurrency market and extend the sample period through April 2022. Our primary data source is CoinMarketCap.com, a widely recognized aggregator of cryptocurrency information. The platform collects and standardizes daily trading data from over 200 major exchanges, providing detailed metrics such as opening, closing, high, and low prices, as well as daily trading volume and market capitalization (in U.S. dollars) for the majority of actively traded cryptocurrencies.

To ensure data quality and consistency, CoinMarketCap enforces specific listing requirements. Eligible cryptocurrencies must be traded on a public exchange with an accessible API that reports real-time price and 24-hour trading volume. They must also maintain an operational website and block explorer, and demonstrate active trading

volume on at least one supported exchange. These listing standards not only protect investors but also mitigate survivorship bias by including both active and delisted tokens at the time of data collection. Furthermore, CoinMarketCap calculates each asset's price as the volume weighted average across exchanges, and its market capitalization is determined by multiplying price by circulating supply, analogous to how public float is used in traditional equity markets. CoinMarketCap's transparency, standardized structure, and widespread adoption in academic and industry research make it a highly reliable source for empirical studies on digital assets. Its extensive coverage allows researchers to capture the dynamic evolution of the cryptocurrency ecosystem over time, encompassing both large-cap coins like Bitcoin and Ethereum as well as smaller altcoins with emerging functionalities.

We construct our sample by extracting daily data for the top 1,000 cryptocurrencies ranked by market capitalization, while imposing a minimum market cap threshold of USD 1 million to ensure economic significance and exclude illiquid or highly speculative tokens. To further ensure data reliability, we exclude any observations with zero or missing values for market capitalization, trading volume, or price. After applying these filters, we retain a final panel of 996 unique cryptocurrencies.

The sample period spans from January 5, 2014, to April 4, 2022, covering a total of 8.25 years and 435 calendar weeks. We use daily closing prices to compute weekly returns, where each week runs from Monday to Sunday to maintain temporal consistency across assets. In addition, we obtain the 1-month Treasury bill rate from the Bank of Canada to serve as a proxy for the risk-free rate. This allows us to calculate weekly excess returns for each cryptocurrency by subtracting the corresponding risk-free rate from their raw returns.

The rationale behind our selection criteria is twofold. First, eliminating coins with negligible market value or incomplete trading data helps reduce statistical noise and limits the influence of non-substantive or inactive assets. Second, focusing on economically relevant and sufficiently liquid tokens ensures that our analysis reflects meaningful market dynamics and yields more reliable results. By curating a clean and representative sample, we enhance the robustness, interpretability, and generalizability of our empirical findings.

3.2 Summary Statistics of the Cryptocurrency Market

Summary statistics are presented in Table 1. Panel A reports the yearly evolution of sample characteristics from 2014 to 2022. The number of cryptocurrencies that pass our filtering criteria grows markedly from 24 in 2014 to 995 in 2022, consistent with

the rapid expansion of the digital asset market. Over the full sample, the average market capitalization across all coins is approximately 769.30 million, with a median market cap of 33.12 million, reflecting the presence of several large-cap coins alongside a long tail of smaller tokens.

Trading activity also increases significantly over time. The mean daily trading volume for the full sample is 100.88 million, while the median volume is 1.81 million, again showing high dispersion in liquidity levels across tokens. Notably, the average market cap rises from just \$ 306.58 million in 2014 to over \$ 2 billion by 2021 and 2022, while the average daily volume grows more than tenfold over the same period. We construct a value-weighted cryptocurrency market return index using all valid coins in the sample. The excess market return (CMKT) is computed as the difference between the market return and the risk-free rate, proxied by the 1-month Treasury bill rate obtained from the Bank of Canada.

Panel B of Table 1 presents descriptive statistics for the weekly returns of the overall coin market index and three major cryptocurrencies: Bitcoin, Ethereum, and Ripple (XRP). The average weekly return of the coin market index is 0.19%, with a standard deviation of 3.94%. Bitcoin yields a slightly higher average return of 0.21%, while Ethereum and XRP post significantly higher weekly returns of 0.50% and 0.35%,

respectively. However, they also exhibit greater volatility, with standard deviations of 6.27% for Ethereum and 7.42% for XRP.

Among the assets, XRP shows extreme positive skewness (6.30) and excess kurtosis (126.54), indicating large and infrequent positive jumps. By contrast, the coin market index displays negative skewness (−1.07) and moderate fat tails (kurtosis = 10.21), suggesting occasional sharp downturns. These statistics highlight the differing risk-return profiles across major digital assets.

These statistics reveal the highly skewed structure of the cryptocurrency market. A few large-cap tokens such as Bitcoin and Ethereum dominate trading volume and market capitalization, while most tokens remain small and thinly traded. This long-tailed distribution poses challenges for empirical modeling, including liquidity constraints, volatility clustering, and unstable return predictability. The wide gaps between mean and median values of market cap and volume further indicate extreme heterogeneity, where a handful of high-value tokens disproportionately drive market movements—making value-weighted returns particularly sensitive to their performance.

These characteristics reflect both the opportunities and risks inherent in the cryptocurrency market, and they justify our use of machine learning techniques, which

are capable of capturing nonlinear relationships and interacting effects across a wide cross-section of assets.

< Table 1 >

3.3 Portfolio Sorting and Market Return Construction

To evaluate the return-predictive power of each factor, we adopt the portfolio sorting methodology introduced by Liu, Tsyvinski, and Wu (2019, 2022). Specifically, at the beginning of each week, we compute the factor value for every cryptocurrency in the sample. Based on the cross-sectional distribution of each factor, we sort tokens into quintile portfolios from the lowest (Q1) to the highest (Q5). We then track the performance of each portfolio over the subsequent week and calculate the average weekly excess return for each quintile by subtracting the one-month Treasury bill rate (obtained from the Bank of Canada) from the raw return. The return differential between the top and bottom quintiles ($Q5 - Q1$) represents a zero-investment long-short strategy, which we use to assess the economic significance and directionality of each factor.

To construct an aggregate market benchmark, we follow Liu et al. (2022) and compute a Value-Weighted cryptocurrency market index, where the weight of each token is proportional to its market capitalization. The cryptocurrency market excess return (CMKT) is then defined as the difference between this market index return and the risk-free rate. This measure serves as a baseline for calculating systematic risk-adjusted performance.

In terms of factor selection, our study excludes beta-related variables and fundamental (accounting-based) factors for two reasons. First, the estimation of beta in crypto markets is notoriously unstable due to extreme volatility and a lack of reliable market benchmarks. Second, as noted by Liu et al. (2019), financial statements and accounting disclosures are either unavailable or irrelevant for the majority of cryptocurrencies, which are often not backed by corporate entities. Therefore, we restrict our attention to market-based factors, which are not only more observable and robust, but also better reflect the decentralized and data-driven nature of the cryptocurrency ecosystem. This design ensures that our factor framework remains consistent, transparent, and applicable across a wide range of tokens over the full sample period from 2014 to 2022.

3.4 Factor Construction for Cross-Sectional Return Analysis in Cryptocurrencies

We construct a comprehensive set of return predictors by extending the methodology by Liu, Tsyvinski, and Wu (2022). Their framework identifies a broad range of market based variables, derived solely from price, trading volume, and market capitalization, that are effective in explaining the cross section of cryptocurrency returns. This market-data-only approach is particularly appropriate in the cryptocurrency context, where standardized accounting information and firm fundamentals are typically unavailable or inapplicable.

Building on their design, we replicate most of the original factors while making several key adjustments. First, we exclude all beta-related variables and any factor requiring long-term correlation estimates, as such measures are unstable in highly volatile crypto markets and require extensive historical data. Instead, we restrict our factor set to observable, market-based variables that are robustly available throughout the sample. Second, we retain the 100-week momentum factor ($r_{100,0}$), which was excluded in some replications, believing it may capture persistent long-term trends. Meanwhile, we replace the original 50-week momentum factor with a 40-week version ($r_{40,0}$) to balance time horizon with data coverage.

In addition, we extend the factor set by introducing two new variables:

- **RETSKEW:** the skewness of weekly returns during the formation week.
- **RETKURT:** the kurtosis of weekly returns, capturing fat tails and asymmetric risk.

Together, our final dataset includes 21 risk factors, organized into four major categories:

- **Size:** MCAP, PRC, MAXDPRC, AGE
- **Momentum:** r 1,0, r 2,0, r 3,0, r 4,0, r 8,0, r 16,0, r 40,0, r 100,0
- **Volume:** VOL, PRCVOL, VOLSCALED
- **Volatility:** RETVOL, RETSKEW, RETKURT, MAXRET, STDPRCVOL, DAMIHUD

In the empirical analysis spanning January 2014 to April 2022, 13 of these 21 factors exhibit statistically significant explanatory power for weekly cross-sectional returns.

These include:

- **Size-related:** MCAP, PRC, MAXDPRC
- **Momentum-related:** r 1,0, r 2,0, r 3,0
- **Volume-related:** VOL, PRCVOL, VOLSCALED
- **Volatility-related:** RETVOL, RETSKEW, MAXRET, DAMIHUD

Our results reaffirm many findings in Liu et al. (2022) while also demonstrating the value of incorporating higher-order moment factors that reflect extreme return behavior and liquidity conditions. This factor structure serves as the foundation for both the portfolio-level analysis and machine learning prediction exercises presented in the following sections.

Table 2 reports the definitions of the 21 risk factors used in our analysis, grouped into four categories: size, momentum, volume, and volatility. The majority are based on the methodology of Liu, Tsyvinski, and Wu (2022), with some modifications including the replacement of the 50-week momentum factor with a 40-week variant, the retention of 100-week momentum, and the addition of two higher-moment volatility measures: skewness and kurtosis of weekly returns.

< Table 2 >

3.5 Crypto-Specific Feature Framework

All of the machine learning methods we consider are designed to approximate the overarching empirical model $\mathbb{E}_t(r_{i,t+1}) = g^*(z_{i,t})$ defined in Equation (2). In our paper, we do not incorporate macroeconomic variables into the construction of predictive features, as is commonly done in traditional equity return prediction (e.g., Welch and Goyal, 2008; Gu, Kelly, and Xiu, 2020). Unlike the stock market, the

cryptocurrency market lacks a consistent and well-defined set of macroeconomic state variables, and many widely used indicators (such as interest rates, term spreads, or equity issuance) are not meaningful or observable for digital assets. As a result, we focus exclusively on market-based characteristics that can be constructed directly at the cryptocurrency level.

Formally, we define the input vector $z_{i,t} \in \mathbb{R}^P$ as a **P -dimensional** vector of asset-specific features for cryptocurrency i observed at time t . ($P = 21$) These predictors include signals related to size (e.g., market capitalization, price), momentum (short- to medium-term return trends), volume (trading activity, liquidity), and volatility (return dispersion and risk). The features are computed on a weekly basis using public historical data from CoinMarketCap, and are selected based on their empirical relevance in prior literature (e.g., Liu, Tsyvinski, and Wu, 2022) as well as their interpretability within the cryptocurrency domain.

This design choice simplifies the feature space compared to macro-interacted models like $z_{i,t} = x_t \otimes c_{i,t}$, but is more appropriate for the decentralized and highly dynamic nature of crypto assets. It also avoids potential look-ahead issues or misalignment between global macro signals and individual token behavior. Consequently, all machine learning models in this study are trained to approximate the

general nonlinear function $g^*(z_{i,t})$, where predictions are solely driven by cryptocurrency-level market features.

4. Empirical analysis

4.1 Weekly Cross-sectional Prediction Performance

Table 3 summarizes the weekly out-of-sample predictive performance across fourteen different machine learning models for forecasting individual cryptocurrency excess returns. The models encompass traditional linear methods, regularized regressions, dimension reduction techniques, tree-based ensemble methods, and deep learning architectures. The performance metric used is the out-of-sample R^2_{oos} , aggregated across all test periods from 2019 to 2022.

Linear models show limited predictive power in the cryptocurrency context. The benchmark OLS model yields a negative R^2_{oos} of (-0.147), indicating substantial overfitting and poor generalization. Similarly, other linear approaches such as PLS (-0.013), PCR (-0.0009), and Enet (-0.0078) also fail to outperform a naive zero-return benchmark. The generalized linear model with Huber loss (GLM) shows some improvement, achieving a modest R^2_{oos} of 0.088, but remains outperformed by more flexible nonlinear models.

Among the tree-based ensemble methods, XGBoost delivers the strongest performance, with an R^2_{oos} of 0.248. This suggests that XGBoost's ability to model complex nonlinear interactions and perform implicit variable selection provides a considerable advantage in the highly volatile cryptocurrency market. Random Forest (RF) and CatBoost follow with R^2_{oos} values of 0.083 and 0.019, respectively. LightGBM also performs moderately well, reaching 0.059.

The performance of neural networks is more nuanced. Shallow architectures like NN1 and NN3 exhibit low predictive power, with R^2_{oos} values of 0.004 and 0.0001, respectively. Deeper networks such as NN2 and NN4 show better results, reaching 0.0145 and 0.0133. However, the deepest model NN5 performs similarly to shallow variants (0.002), indicating that increasing model depth does not consistently enhance performance. These results suggest that while neural networks are capable of capturing nonlinearities, their advantage in this high-noise, low-signal environment may be limited without more extensive tuning or structural innovations.

Overall, the models can be roughly ranked into three performance tiers. The top tier is occupied by XGBoost, which clearly outperforms all others. The second tier includes RF, GLM, CatBoost, and LightGBM, all of which produce low but positive

Roos² values. The third tier includes all linear models and most neural network variants, which provide marginal or negative out-of-sample explanatory power.

Compared to stock-level forecasts in traditional financial markets, the R_{oos}^2 values observed here are generally smaller, which may reflect greater market inefficiency, noise, or the dominance of speculative trading in the cryptocurrency space. Nevertheless, the superior performance of ensemble learning methods like XGBoost underscores the value of capturing complex nonlinear relationships in return prediction tasks.

< Table 3 >

4.2 Hyperparameter Settings and Tuning Strategy

To ensure that our model comparisons are fair, robust, and reflect the true out-of-sample predictive ability of each algorithm, it is essential to carefully tune the hyperparameters associated with each method. In machine learning, model performance can vary significantly depending on how these parameters are selected, especially in high noise environments like the cryptocurrency market. Poorly chosen settings may lead to overfitting, where the model captures noise instead of signal, or underfitting, where the model is too simplistic to detect meaningful patterns.

For example, a tree-based model with overly deep trees may memorize specific data paths rather than generalize across patterns, while a neural network with too few hidden units may fail to approximate the complex nonlinearity in return structures. Similarly, insufficient regularization in linear models may cause the algorithm to overweight irrelevant predictors, leading to large in-sample fit but poor out-of-sample performance.

To address these challenges, we employ validation-based hyperparameter tuning strategies, where parameters are selected based on their performance on a separate validation set prior to testing. This helps balance model complexity and generalization. The following table summarizes the key hyperparameter settings used for each model in our analysis, including tree depth, learning rate, regularization penalties, and network architecture choices.

For benchmarking purposes, we include three traditional linear models: ordinary least squares (OLS), partial least squares (PLS), and principal component regression (PCR). All models are implemented using standard configurations. For PLS and PCR, the number of components is set to one to reduce complexity and ensure comparability. These baseline models help highlight the potential performance gains offered by more flexible machine learning methods.

Elastic Net (ENet) combines both l_1 (lasso) and l_2 (ridge) penalties to perform variable selection and shrinkage simultaneously. In our implementation, we set $\alpha = 0.5$, implying equal weight between the l_1 and l_2 penalties, and chose a small l_1 ratio = $1e-4$ to favor ridge-like regularization. The maximum number of iterations was set to 10,000 to ensure convergence. This configuration is designed to control model complexity and mitigate overfitting in high-dimensional predictor spaces typical of cryptocurrency return data.

Generalized Linear Model (GLM) with group lasso regularization is employed using $\text{reg_lambda} = 0.1$ to impose sparsity and reduce the influence of noisy variables. A fixed $\text{random_state} = 1$ is used for reproducibility. The use of group-wise shrinkage allows the model to retain meaningful predictor blocks while discarding irrelevant ones, which is particularly suitable for factor-based return prediction settings.

To enhance the performance of tree-based models in our forecasting task, we carefully tune hyperparameters for Random Forest, XGBoost, CatBoost, and LightGBM to balance model complexity and generalization. These models are known for their strong predictive capacity and built-in feature selection capabilities, but without proper regularization, they can easily overfit noisy financial data—particularly in the cryptocurrency market, where volatility and nonstationarity are pronounced.

For Random Forest, we set `n_estimators = 900` to ensure a stable ensemble size while limiting `max_depth = 6` to avoid overly deep trees that capture noise. The large values for `min_samples_split = 1000` and `min_samples_leaf = 1000` act as a regularizer by forcing each split to be supported by a substantial number of observations, thereby improving generalization. The `max_features = 'sqrt'` strategy randomly selects a subset of features at each split to reduce correlation between trees, and `random_state = 42` ensures replicability.

For XGBoost, the model is tuned more aggressively for control and precision. We set `n_estimators = 5000` and use a relatively shallow tree depth (`max_depth = 5`) combined with a very small `learning_rate=0.015` to slow down learning and improve convergence stability. We further control overfitting using `gamma = 0.2` (minimum loss reduction), `reg_lambda = 1` (l_2 regularization), and `max_delta_step = 5` to limit update steps. The histogram-based tree method (`tree_method = 'hist'`) improves training speed and memory efficiency.

The CatBoost model is configured with `iterations = 110` and `depth = 7`, while setting a small `learning_rate = 0.0017` to improve generalization. The `l2_leaf_reg = 3` introduces l_2 regularization on leaf values, acting as a shrinkage term. CatBoost's

native handling of categorical variables is not used in our case, but its powerful regularization and fast convergence still make it a strong candidate for tabular data.

Finally, LightGBM is set to `n_estimators = 99`, with `max_depth = 5` and an extremely low `learning_rate = 0.00085`, making it the most conservative learner among the tree models. The `boosting = 'goss'` (Gradient-based One-Side Sampling) algorithm is selected to retain large-gradient samples for faster convergence. The `num_leaves = 10` and `min_data_in_leaf = 23` settings help control the complexity of the tree and avoid overfitting to small fluctuations.

These hyperparameter settings are informed by validation-based tuning, aiming to strike a balance between underfitting and overfitting. By ensuring that tree models are neither too simple to capture relevant signals nor too complex to memorize noise, we enhance their robustness and forecasting accuracy in a high-dimensional and volatile environment like cryptocurrencies.

As part of our model suite, we design five variants of feedforward neural networks (NN1 through NN5) to explore the predictive potential of deep learning in the cryptocurrency market. These models vary primarily in depth and internal architecture, allowing us to investigate the trade-offs between complexity and performance. Each network is trained using stochastic gradient descent (SGD) with a learning rate of 0.001

and a clip value of 0.5 to stabilize gradients. We employ mean squared error (MSE) as the loss function and apply early stopping based on validation loss. The batch size is set to 10,000, and training is limited to 10 epochs to prevent overfitting.

The NN1 architecture consists of 20 repeated layers with 32 units, followed by batch normalization and a single output layer, forming a relatively shallow baseline model. Building on this, NN2 includes an additional hidden layer with 16 ReLU-activated units and l_1 regularization, encouraging sparsity and reducing noise. NN3 expands further by adding an extra hidden layer of 8 ReLU units, offering greater capacity while preserving regularization. In NN4, the depth increases with three successive hidden layers (8, 4, and 2 units), forming a funnel-like architecture to extract condensed hierarchical features. NN5 maintains a similar deep structure as NN4 but omits the base loop, providing a non-repetitive layer stacking approach for comparison. All hyperparameter settings for the fourteen models discussed above are summarized in Table 4.

< Table 4 >

4.3 Model Comparison via Diebold-Mariano Tests

To evaluate whether differences in predictive accuracy across models are statistically significant, we conduct pairwise comparisons using the Diebold-Mariano (DM) test.

This test assesses whether one model significantly outperforms another in terms of out-of-sample forecast accuracy by comparing their prediction errors across the test period. Specifically, we adopt a cross-sectional adaptation of the DM test, where the test statistic is constructed from the average loss differential (squared error) across all cryptocurrencies at each time point, following the method proposed by Gu, Kelly, and Xiu (2020).

Table 5 reports the resulting DM test statistics for all pairwise comparisons among the 14 models. Each entry corresponds to a comparison between the model in the column and the model in the row. A positive value indicates that the column model outperforms the row model, and vice versa. One asterisk (*) denotes statistical significance at the 5% level, while two asterisks (**) indicate significance under a Bonferroni - adjusted critical value** that accounts for multiple comparisons.

Given that we compare 14 models, a total of 91 unique pairwise tests are conducted. To control for family-wise error rate due to multiple testing, we apply a Bonferroni correction, dividing the nominal significance level ($\alpha = 0.05$) by the number of comparisons (91), resulting in an adjusted one-sided critical value of approximately 2.64. Therefore, only test statistics exceeding this threshold in absolute value are flagged with "***" in the table.

The results show that tree-based models (such as XGBoost, CatBoost, and LGBM) consistently outperform traditional linear models (e.g., OLS, PCR, PLS), with many of the differences statistically significant even under the Bonferroni correction. Furthermore, neural networks (NN1–NN5) demonstrate the strongest and most consistent performance gains, yielding statistically significant improvements over nearly all linear and tree-based models. Notably, NN3 and NN4 achieve the most frequent and robust outperformance, including several comparisons that remain significant at the more stringent Bonferroni level. This provides strong evidence that deep learning architectures are especially effective in capturing the nonlinear return dynamics present in the cryptocurrency market.

< Table 5 >

4.4 Assessing the Importance of Predictors Across Models

To better understand how individual covariates contribute to return predictability in the cryptocurrency market, we examine the variable importance results from all fourteen models considered in this study. Unlike prior literature that uses squared partial derivatives (SSD) to assess feature influence, we instead adopt model-specific feature importance algorithms tailored to each method. For tree-based models such as Random Forest, XGBoost, CatBoost, and LightGBM, we extract built-in importance metrics

based on the reduction in mean squared error or split gain. For neural networks, we apply permutation importance, which measures the marginal increase in MSE after permuting each input variable.

Figure 4 to Figure 17 display the normalized feature importance scores for each of the fourteen models, highlighting the top 21 covariates used in return prediction. In linear models such as OLS, PCR, PLS, and ElasticNet, volatility-based variables such as STDPRCOL, VOL, and RETSKEW consistently dominate, reflecting their direct relationship with short-term price risk. Generalized linear models (GLM) show a similar pattern but with an even more concentrated reliance on a handful of variables.

Tree-based models, particularly CatBoost and LightGBM, identify broader sets of influential features, but commonly rank PRC, RETSKEW, RETURT, and MAXRET among the top predictors. XGBoost displays a more concentrated attribution pattern, placing almost all weight on RETSKEW and STDPRCOL. Random Forest distributes weights more evenly across key volatility, return, and skewness-based metrics. Neural networks exhibit richer and more diverse attributions across predictors. For instance, NN1 and NN2 highlight RETVOL, MAXRET, and STDPRCOL as dominant inputs, while deeper networks like NN4 and NN5 suggest moderate importance dispersion

among 10 to 15 different variables, including momentum-based factors like MOM4, MOM8, and MOM100.

To facilitate cross-model comparison, we construct a Heatmap aggregating normalized importance scores across all models. Figure 18 reveals clusters of influential variables consistently ranked highly across methods. RETSKEW, STDPRCOL, PRC, and RETURT emerge as key predictors, with high salience in both traditional models and deep learning architectures. Conversely, certain variables such as MOM16, MOM100, and DAMIHU appear only intermittently and tend to exhibit relatively limited influence across models.

These findings suggest that while each algorithm has its own mechanism for extracting information from predictors, there is broad consensus on the importance of volatility, skewness, and price-related signals in predicting cryptocurrency returns. Deep learning models show greater flexibility in incorporating nonlinear interactions, but many of the fundamental signals identified overlap with those from simpler models, reinforcing the robustness of the identified predictors.

< Figures 4 - Figures 17 >

< Figure 18 >

To further investigate the driving forces behind model predictions, we focus on a subset of 13 predictor variables that were previously identified as statistically

significant in our univariate screening tests. These variables, selected based on their explanatory power for cryptocurrency returns, span dimensions such as volatility (e.g., STDPRCOL, RETVOL), liquidity (PRCVOL, DAMIHU), valuation (PRC, MAXDPRC), momentum (MOM1, MOM3, MOM100), market characteristics (MCAP, VOL), and distributional shape of returns (RETSKEW, RETKURT, MAXRET, MARKET).

Figure 19 and Table 6 present the normalized variable importance across models for this selected subset. Notably, nonlinear models, particularly tree based algorithms like XGBoost, CatBoost, and Random Forest, allocate disproportionate importance to specific variables such as RETSKEW, PRC, and RETKURT, sometimes concentrating heavily on a single dominant signal. For instance, RETSKEW alone contributes over 70% of the total importance weight in XGBoost. In contrast, linear models such as OLS and PCR distribute importance more evenly across a wider set of predictors, suggesting that they rely on broader but weaker signals. Neural networks present a middle ground. While some models (e.g., NN2 and NN3) highlight MAXRET, RETKURT, and RETVOL as key features, others (e.g., NN5) downplay momentum signals in favor of volatility and skewness-related variables. This variation highlights the flexibility of neural architectures in adapting to different predictive patterns within the data.

A noteworthy observation arises from the GLM (Generalized Linear Model) results, which assign zero importance to a substantial number of variables. This phenomenon can be primarily attributed to the strong regularization applied through the group lasso penalty. Unlike traditional linear models, the GLM used in our setup is designed to perform automatic variable selection by shrinking the coefficients of irrelevant or weakly predictive variables exactly to zero. As a result, only a small subset of features survives the penalization process and is retained in the final model.

The sparsity is particularly evident in Figure 4 to Figure 17, where the GLM model highlights only a few dominant predictors, such as STDPRCOL, RETSKEW, and RETKURT, while entirely excluding others like MOM, MCAP, and DAMIHU. While this outcome improves model interpretability and may help reduce overfitting, it also suggests that GLM's structure may underexploit nonlinear or interaction-based signals that are present in the cryptocurrency market. This contrasts with tree-based and neural network models, which tend to capture predictive patterns across a broader set of variables. In sum, the prevalence of zero coefficients in GLM reflects the inherent trade-off of using heavy regularization: enhanced simplicity and robustness at the potential cost of predictive nuance.

From a financial interpretation standpoint, the prominence of variables like RETSKEW and RETKURT across many nonlinear models suggests that tail risks and asymmetries in return distributions play a critical role in cryptocurrency return predictability. Similarly, the importance of PRC, VOL, and MCAP echoes common findings in market microstructure literature, where price levels and market size often correlate with liquidity and investor attention. Lastly, the persistent appearance of short term and intermediate momentum variables (MOM1, MOM3, MOM100) supports the notion that technical trading signals, while noisy, still contribute meaningfully in certain modeling setups.

< Figure 19 > < Table 6 >

4.5 Marginal Association between Characteristics and Expected Returns

To investigate how individual characteristic influence predicted returns, we construct Partial Dependence Plots (PDPs) for each of the 13 statistically significant factors identified earlier. The goal of PDP analysis is to isolate the marginal effect of a single predictor on the model's output, expected weekly return, while holding all other variables fixed at their median values. This approach provides intuitive, visual insights into whether the model captures theoretically consistent return patterns, such as the size premium or momentum continuation.

For each PDP, the corresponding predictor is rescaled to the interval $[-1,1]$, allowing for comparability across variables with different scales. All other features are set to zero (i.e., their cross-sectional median after normalization), ensuring that the model's prediction reflects only the change in the selected variable. The resulting PDP curve then traces the predicted change in weekly return as the chosen characteristic varies across its range. To guide interpretation, we rely on the sign of the t-statistics from our return-sorting tests (Appendix). A positive t-statistic suggests that higher values of the characteristic should be associated with higher expected returns, implying that the PDP should be upward sloping. Conversely, a negative t-statistic indicates a downward-sloping pattern. By comparing PDP slopes to the expected sign, we can assess whether the machine learning models have correctly learned and internalized these cross-sectional return signals. This PDP framework allows us to bridge the gap between statistical significance and model interpretability, offering an interpretable view of how complex, possibly non-linear models translate input characteristics into return predictions.

4.5.1 Size Characteristics and Return Patterns

We begin our analysis by examining the predictive role of size-related variables, specifically market capitalization (MCAP), price level (PRC), and the maximum price

over the past week (MAXDPRC). These variables are commonly associated with the size anomaly observed in financial markets, where smaller assets (e.g., low market cap or low price coins) tend to yield higher returns than larger ones. This phenomenon, initially documented in equity markets by Banz (1981), has also been found to persist in the cryptocurrency market according to Liu, Tsyvinski, and Xiwu (2022).

To investigate whether our models capture this size effect, we rely on both cross-sectional portfolio sorting (5-1 t-statistics) and Partial Dependence Plots (PDPs). A significant negative t-statistic in 5-1 tests suggests that lower-sized assets outperform larger ones, implying a downward-sloping PDP curve is expected for effective models. Detailed insights can be found in the Partial Dependence Plots (PDP) presented in Figures 20 ~ 22.

For MCAP, our 5-1 t-statistic is significantly negative ($t = -3.08$), indicating a strong size premium. Correspondingly, the PDPs of models such as PLS, XGB, RF, and ENet exhibit clear downward trends, suggesting that these models correctly associate lower market capitalization with higher predicted returns. Similarly, PRC also yields a significantly negative 5-1 t-statistic ($t = -2.61$), supporting the notion that lower-priced cryptocurrencies yield higher returns. The downward-sloping PDPs of ENet, PLS, NN1, and NN3 reinforce this finding, showing that the models successfully

learn the inverse relationship between price and expected return. For MAXDPRC, which measures recent extreme price levels, we again observe a negative and significant 5-1 t-statistic ($t = -2.60$). PDPs from PLS, CatBoost, and XGB further support this conclusion, as their negative slopes suggest that assets with high recent price peaks tend to have lower future returns, possibly reflecting a mean reversion tendency among overbought cryptocurrencies.

In sum, the consistent evidence across MCAP, PRC, and MAXDPRC affirms that our models effectively capture the negative association between size and return in the cryptocurrency market, mirroring well-established findings in equity markets. This demonstrates the models' ability to identify the size premium phenomenon under a machine learning framework. In other words, cryptocurrencies with smaller market capitalization, lower price levels, and lower recent highs tend to deliver higher future returns, and our machine learning models are able to learn and replicate this relationship. The ability of multiple model classes, both linear and nonlinear, to uncover this empirical regularity demonstrates the effectiveness of machine learning in detecting size related anomalies in the cross section of cryptocurrency returns.

4.5.2 Momentum Characteristics and Return Patterns

We next examine the predictive roles of momentum-related variables, specifically MOM1 (last week return), MOM2 (return from 2 weeks ago), and MOM3 (return from 3 weeks ago). These variables are designed to capture two types of empirical regularities widely discussed in the literature: the momentum effect, where past winners continue to outperform, and the reversal effect, where short-term winners tend to underperform in the future. In traditional equity markets, Jegadeesh and Titman (1993) find a short-term reversal in the first month followed by a medium-term momentum effect spanning months 2 to 12. In contrast, Liu, Tsyvinski, and Xiwu (2022) report that in the cryptocurrency market, both reversal and momentum effects tend to appear earlier, typically within a 1 to 4 week window, likely due to high volatility and investor behavioral factors unique to crypto.

To investigate these patterns, we analyze both 5-1 portfolio t-statistics and Partial Dependence Plots (PDPs) for each momentum variable. A negative 5-1 t-statistic and a downward-sloping PDP indicate the presence of a short-term reversal, while a positive t-statistic and upward-sloping PDP suggest a momentum continuation. Figures 23 ~ 25 display the Partial Dependence Plots (PDP), which provide further details on the results.

For MOM1, we observe a significantly positive 5-1 t-statistic ($t = 2.13$), yet most models (e.g., ENET, NN1, NN5) display downward-sloping PDPs. This contradiction implies that while the raw portfolio sorting suggests a momentum effect, the models' nonlinear structure more strongly captures a reversal effect, meaning that coins with strong returns last week tend to revert in the following week. In MOM2, the t-statistic remains positive ($t = 2.58$), and PDPs for neural networks such as NN1 and NN2 and NN3 begin to show upward trends, suggesting the emergence of short-term momentum beyond the first lag. However, ENET still show negative PDP slopes, indicating mixed evidence across methods. For MOM3, the t-statistic is the strongest among the three ($t = 3.42$), and most models, including NN1, NN3, and NN4, exhibit upward-sloping PDPs. This provides more consistent support for a momentum effect, where past winners in week 3 continue to yield higher returns.

Taken together, our results suggest that machine learning models successfully identify a reversal effect in the most recent week (MOM1) and momentum continuation from weeks 2–3 (MOM2–MOM3). These findings are consistent with the earlier timing of momentum effects documented in cryptocurrency markets and highlight the nonlinear learning capacity of neural networks and tree-based models in capturing return dynamics.

4.5.3 Volume Characteristics and Return Patterns

We next examine the predictive power of volume-related variables, including average dollar trading volume (VOL), price-volume ratio (PRCVOL), and dollar trading volume scaled by market capitalization (VOL SCALED). These proxies relate to market liquidity, and prior studies (e.g., Chordia et al., 2001; Liu et al., 2022) suggest a negative relationship between trading volume and future returns, that is, assets with lower trading activity tend to deliver higher returns, possibly due to liquidity premiums or investor inattention.

Starting with VOL, the 5-1 t-statistic is significantly negative ($t = -1.67$), implying that low-volume coins earn higher weekly returns. Consistently, PDP curves from models like PLS and NN5 exhibit clear downward slopes, indicating that lower volume is associated with higher predicted returns. This pattern aligns with the traditional illiquidity premium documented in both equity and cryptocurrency markets. For VOLSCALED, we observe a similar relationship. The 5-1 t-statistic is also negative ($t = -2.27$), and PDPs from PLS and NN1 show decreasing trends, reinforcing the inverse link between liquidity and expected return. This confirms that even after accounting for size differences, scaled trading activity continues to predict lower future returns.

In contrast, the results for PRCVOL are more mixed. Although the 5-1 t-statistic remains negative ($t = -1.91$), only a subset of models (e.g., RF, XGB, NN5) show the expected downward trend in PDPs. Others (e.g., NN1, NN2, NN3) show flat or even increasing slopes, suggesting heterogeneous effects across model types. These inconsistencies imply that PRCVOL may be a weaker or noisier proxy for liquidity in the cryptocurrency context.

Taken together, the findings from VOL and VOL SCALED provide strong support for the existence of a volume-related return premium in the crypto market. The consistent negative association between trading activity and returns is successfully captured by several machine learning models, particularly PLS and certain neural networks. However, PRCVOL appears less robust, potentially due to its construction or sensitivity to price volatility. For a more comprehensive understanding of the results, see the Partial Dependence Plots (PDP) in Figures 26 ~ 28.

We infer this phenomenon based on two observations. First, a significantly negative 5-1 portfolio t-value indicates that assets in the low-volume group (Group 1) earn significantly higher returns than those in the high-volume group (Group 5). This pattern supports the notion that lower trading volume is associated with higher returns, consistent with the liquidity premium or investor inattention hypotheses. Second, the

Partial Dependence Plot (PDP) shows that predicted returns decline as trading volume increases. Holding all other variables constant, the models predict lower returns for more actively traded assets, aligning with the view that less traded assets tend to deliver higher expected returns.

4.5.4 Volatility Characteristics and Return Patterns

We further examine whether our models successfully capture well-known volatility-related return patterns by analyzing four characteristics: return skewness (RETSKEW), the standard deviation of daily returns in portfolio formation week (RETVOL), maximum daily return (MAXRET), and the Amihud illiquidity measure (DAMIHUD). Prior studies have noted that, unlike equity markets where volatility is often associated with lower future returns due to risk aversion and limits to arbitrage, the cryptocurrency market may exhibit a different dynamic. In particular, assets with higher volatility, more extreme daily returns, or greater illiquidity tend to yield higher expected returns, consistent with the notion of a risk premium or compensation for limited arbitrage opportunities (Liu et al., 2022).

To assess whether these phenomena are reflected in our machine learning models, we evaluate both the 5-1 portfolio t-statistics and Partial Dependence Plots (PDPs). A significantly positive 5-1 t-statistic indicates that assets ranked high on a given

volatility-related measure outperform those ranked low, while an upward-sloping PDP suggests that the model expects higher returns for assets with higher volatility or illiquidity. The Partial Dependence Plots (PDP) in Figures 29 ~ 31 offer a detailed breakdown of the model's behavior.

Our results generally support the existence of such effects. RETVOL and MAXRET both exhibit significantly positive 5-1 t-values, and their corresponding PDPs from models such as ENET, PLS, show clear upward slopes, indicating that assets with larger price fluctuations or extreme daily returns are associated with higher predicted returns. This aligns with the idea that the cryptocurrency market rewards risk-taking behavior, possibly due to its speculative nature and lack of strong arbitrage mechanisms. As for DAMIHU, the results are less aligned with traditional illiquidity-based theories. While the 5-1 t-statistic is significantly positive ($t = 1.71$), indicating that illiquid assets (with higher Amihud values) tend to earn higher returns, most models fail to reflect this relationship clearly in the PDP plots. In fact, only the RF model exhibits an upward sloping PDP, consistent with the illiquidity premium hypothesis, that is, assets with higher trading frictions are compensated with higher expected returns. However, other models (e.g., PLS, XGB, NN4, NN5) show flat or slightly negative PDP trends, suggesting that DAMIHU's predictive signal may be weak or inconsistently captured across different algorithms

Among the four volatility-related predictors, RETSKEW stands out as the most consistently and strongly captured characteristic across models. The 5-1 portfolio sorting based on RETSKEW yields a highly significant positive t-statistic ($t = 7.15$), indicating that assets with higher return asymmetry tend to generate greater returns. This finding is further reinforced by the upward-sloping PDPs across nearly all models, including ENET, PLS, NN1, NN2, NN3, RF, and XGB. The consistency of both statistical and visual evidence suggests that RETSKEW contains strong predictive information in the cryptocurrency market.

In conclusion, our analysis reveals that volatility related characteristics, particularly return skewness, play a significant role in predicting cryptocurrency returns. While traditional equity market research has documented a puzzling negative relationship between certain types of volatility and future returns, known as the low volatility anomaly, this pattern does not appear to hold in the cryptocurrency domain. For instance, Ang, Hodrick, Xing, and Zhang (2006) show that stocks with high idiosyncratic volatility tend to yield significantly lower future returns, a finding echoed by Bali and Cakici (2008). These results challenge the classical notion of risk-reward tradeoff and are often attributed to limits to arbitrage and behavioral biases, where investors are drawn to speculative assets despite their poor performance.

However, it is important to distinguish between types of volatility. While systematic risk (e.g., market beta) is expected to be rewarded in classical asset pricing models like the CAPM, other forms such as return skewness or maximum daily returns may appeal to investors with lottery-like preferences, potentially leading to a positive return premium. Recent studies, such as Liu, Tsyvinski, and Xiwu (2022), find that these effects are even more pronounced in the cryptocurrency market, where speculation and market inefficiencies are widespread.

Our results are consistent with these newer findings. We observe significantly positive 5-1 t-values and upward-sloping PDPs for several volatility-related predictors, especially RETSKEW, RETVOL, and MAXRET, indicating that assets with greater volatility or return asymmetry are associated with higher expected returns. These findings suggest that, in contrast to equity markets, the crypto space may reward risk-taking behavior rather than penalize it, possibly due to investor speculation, limited arbitrage, and distinct market microstructure.

5. Machine Learning Portfolios

To evaluate the practical investment implications of machine learning return forecasts, we construct a series of long-short portfolio performance tables. These tables allow us to examine whether the models can effectively rank assets in a way that translates into

profitable trading strategies. By comparing realized returns and Sharpe ratios across decile or quintile portfolios, we assess how well the predicted return rankings align with actual performance.

For each model, we sort cryptocurrencies at the end of each week based on their predicted one-week-ahead return. The sorted assets are then grouped into either 5 (quintile) or 10 (decile) portfolios. We form long-short portfolios by buying the top group and shorting the bottom group. These portfolios are constructed under two weighting schemes, value weighted and equal weighted, and evaluated while excluding COCOS-BCX to ensure robustness. All results are reported as out-of-sample average returns and Sharpe ratios.

5.1 Motivation for Data Separation: The COCOS-BCX Outlier

Why do we make such a separation in our data? Extreme Return Outlier and Its Timeline: The Case of COCOS-BCX. One major outlier in our dataset was caused by the coin COCOS-BCX, which underwent a significant token redenomination process in January 2021. The official announcement declared that the total supply would be reduced by a factor of 1000:1, effective from January 19, 2021 (UTC+8). As a result, 1000 old tokens were exchanged for 1 new token, mechanically increasing the token's unit price by 1000 times.

Prior to this event, on January 15–16, market participants began speculating on the news, leading to a sharp price pump. The closing price surged from 0.0007 on January 15 to 0.84 on January 21, representing a nominal gain of over +100,000%. However, this surge was purely mechanical, not reflective of any real change in value. In addition, major exchanges such as Binance suspended trading and token transfers from January 18 onward to facilitate the redenomination. During this freeze, trading activity became highly irregular, with reduced liquidity and sporadic price quotes. This contributed to extreme weekly returns and volatility, particularly around January 17, which coincided with our portfolio formation date.

In our portfolio analysis, this resulted in our pre-study which Decile 8 showing a standard deviation (SD) greater than 2.4 and abnormally high predicted returns (Y values exceeding +1400%). Because these effects are driven by a mechanical redenomination and not actual market dynamics, the inclusion of COCOS-BCX significantly distorted both model training and performance evaluation. Therefore, to validate the robustness of our findings and ensure that model performance is not overly influenced by a single outlier, we repeated all forecasting and portfolio analyses with and without COCOS-BCX. This dual-scenario approach provides a clearer understanding of the model's true predictive power under normal versus anomalous conditions.

5-2 Risk-Adjusted Performance of Machine Learning Portfolios under Different Grouping and Weighting Schemes

5.2.1 Risk-Adjusted Performance of Machine Learning Portfolios without Cocos-BCX

- **Value-weighted ,10, without Cocos-BCX**

Table 7 presents the out-of-sample results of Value-Weighted decile portfolios constructed without the inclusion of COCOS-BCX. Cryptocurrencies are ranked weekly by predicted one-week-ahead returns, and portfolios are rebalanced accordingly. The H–L return (Decile 10 minus Decile 1) and its corresponding Sharpe ratio are reported as key metrics.

XGBoost achieves the best overall performance, with a Sharpe ratio of 0.726 and a weekly H–L return of 0.118%, confirming its leading status even after removing the previously dominant outlier. GLM and Elastic Net also perform exceptionally well (Sharpe = 0.639 and 0.564, respectively), showing substantial improvement compared to their performance when COCOS-BCX was included. PLS, LGBM, and CatBoost exhibit moderate to strong performance, all achieving Sharpe ratios between 0.45 and 0.50. These results suggest that without extreme-return distortion, a broader range of models are capable of generating effective cross-sectional predictions. In contrast,

neural networks remain inconsistent. While NN5 delivers a respectable Sharpe ratio of 0.498, earlier variants such as NN2 and NN3 underperform, with weak or negative H–L returns, highlighting their sensitivity to grouping schemes and volatility.

Overall, excluding COCOS-BCX leads to more stable and higher-quality signals across most models, especially for tree-based and regularized linear methods. XGBoost continues to be the most robust and accurate predictor, with traditional models showing greater competitiveness when freed from the influence of extreme-return tokens.

- **Value-Weighted, 5, without Cocos-BCX**

Table 8 presents the out-of-sample performance of Value-Weighted quintile portfolios constructed without COCOS-BCX. Cryptocurrencies are sorted weekly into five groups based on predicted one-week-ahead returns, and long-short returns (Q5–Q1) are calculated. The Sharpe ratio of the long-short portfolio is used to evaluate risk-adjusted performance.

XGBoost emerges as the best-performing model, delivering the highest Sharpe ratio of 0.658 and the largest H–L return of 0.0801%. Regularized linear models also show strong results, with Elastic Net and GLM achieving Sharpe ratios of 0.609 and 0.588, respectively. Notably, PLS achieves an H–L return of 0.0707% with a Sharpe ratio of 0.584, outperforming traditional OLS and PCR models. In contrast, models

such as CatBoost and Random Forest provide more modest performance, with Sharpe ratios below 0.45. Meanwhile, neural network models, especially NN1 through NN3, exhibit negative or near zero H to L returns, indicating their lack of robustness in this setting even after excluding extreme tokens.

Overall, removing COCOS-BCX leads to more stable and consistent performance among tree-based and regularized linear models. XGBoost continues to be the most effective model, while Elastic Net, GLM, and PLS also demonstrate strong predictive validity under coarser quintile groupings.

- **Equal-Weighted, 10, without Cocos-BCX**

Table 9 reports the out-of-sample performance of equal-weighted decile portfolios constructed without COCOS-BCX. Each week, cryptocurrencies are sorted into ten groups based on predicted one-week-ahead returns. The difference in average returns between the top and bottom deciles (Q10–Q1) and its Sharpe ratio are used as the evaluation metrics.

Among all models, XGBoost delivers the strongest overall performance, with an H–L Sharpe ratio of 0.394 and the highest average return of 0.411%. LightGBM follows closely, achieving a Sharpe ratio of 0.307 and an H–L return of 0.320%. These results highlight the consistency and predictive strength of tree-based ensemble models

under equal-weighted schemes. CatBoost and Elastic Net also produce competitive results, with Sharpe ratios around 0.30, while GLM achieves a Sharpe of 0.282, demonstrating that regularized and linear models can also perform well when outlier noise is reduced. In contrast, many neural network models and traditional regressions offer lower Sharpe ratios or inconsistent H–L returns and are therefore omitted from detailed discussion.

In sum, removing COCOS-BCX substantially improves performance stability and model comparability. XGBoost remains the top-performing model, while LightGBM, CatBoost, and Elastic Net also exhibit strong risk-adjusted return characteristics under equal-weighted decile formation.

- **Equal-Weighted, 5, without Cocos-BCX**

Table 10 reports the out-of-sample performance of equal-weighted quintile portfolios constructed without the token COCOS-BCX. Cryptocurrencies are sorted weekly into five groups based on their predicted one-week-ahead returns. The performance of the long-short (Q5–Q1) strategy is evaluated by its average return and Sharpe ratio.

LightGBM emerges as the top-performing model, delivering the highest Sharpe ratio of 0.609 and a stable H–L return of 0.103%, suggesting its strong ability to generate consistent risk-adjusted returns under coarse grouping. In contrast, XGBoost

yields the highest raw return at 0.249%, paired with a slightly lower Sharpe ratio of 0.477, indicating its attractiveness for return-seeking investors willing to tolerate more volatility. Among neural networks, only NN4 shows strong and consistent performance, attaining a Sharpe ratio of 0.311 and an H–L return of 0.160%. Other architectures such as NN1, NN2, and NN5 yield moderate results, while NN3 performs poorly (Sharpe = 0.041, H–L = 0.021%), indicating weak and unstable predictive capacity in this configuration.

In summary, the exclusion of COCOS-BCX improves the robustness of portfolio strategies. LightGBM leads in Sharpe ratio, XGBoost leads in raw return, and both outperform deep neural networks in both consistency and efficiency under Equal-Weighted quintile formation.

5.2.4 Comparison across Portfolio Configurations (without COCOS-BCX)

Regularized models like Elastic Net and GLM consistently benefit from the exclusion of extreme tokens, with Sharpe ratios frequently exceeding 0.33 across all four configurations. Their performance improves particularly under 5-group settings, where coarser partitioning enhances return stability.

Tree-based models remain dominant. XGBoost delivers the best performance overall—achieving a Sharpe ratio of 0.726 and a return of 0.118% in the VW10

configuration—while also ranking among the top models in all other settings. LightGBM leads under EW5, with the highest Sharpe ratio (0.609) in that group, emphasizing its strength in more conservative, equal-weighted portfolios. This level of consistency and accuracy, perhaps unsurprisingly, outperforms OLS by a wide and unquestionable margin.

Among neural networks, only NN4 consistently delivers Sharpe ratios above 0.30, especially under EW5 and VW5. Other architectures particularly NN3 struggle to generalize, often producing unstable or weak long-short returns, regardless of grouping or weighting.

In summary, removing COCOS-BCX improves risk-adjusted performance and predictive clarity across all designs. XGBoost remains the most robust model overall, followed by LightGBM in low volatility configurations. Regularized linear models also emerge as viable alternatives under stable market conditions, while neural networks continue to show mixed results.

5.3 Cumulative Long-Short Portfolio Returns from Machine Learning Models

We examine the cumulative log returns of long and short portfolios constructed by various machine learning models in order to evaluate their performance over time. These visualizations help reveal not only how each model performs under real market

conditions, but also the relative stability and divergence of different strategies across changing regimes.

In equity market literature, such return plots are commonly benchmarked against the S&P 500 index, which serves as a widely accepted reference for passive investing. However, the cryptocurrency market lacks a comparable baseline with similar liquidity, representativeness, or institutional credibility. While indices such as Bitwise 10 or CMC Crypto 200 are available, they often suffer from rebalancing challenges, survivorship bias, or limited transparency. As a practical and model-free alternative, we adopt the 1/N strategy as our baseline. This approach equally weights all tradable cryptocurrencies in the sample and rebalances weekly. It represents the outcome of a naïve, passive diversification strategy. Its cumulative log return is plotted as a black solid line in each figure, providing a consistent and neutral reference point for evaluating the value added by model-driven strategies.

For each model, we compute the weekly long and short portfolio returns, apply a log transformation via `np.log1p`, and calculate their cumulative sums to reflect compounding performance. Key market events such as the COVID crash (March 2020), institutional adoption surge (January 2021), and retail-driven rallies (May 2021) are highlighted with shaded regions to contextualize performance shifts. This visualization

framework offers an intuitive and informative comparison between model-driven portfolios and the 1/N baseline across different periods of market stress and growth. And then I retain COCOS-BCX in this section because the cumulative return plots are designed to illustrate model behavior during major market events. Since these episodes reflect systemic dynamics, we choose not to exclude any individual asset from the investment universe. Furthermore, we focus on the VW10 and EW10 configurations for consistency with prior literature particularly Gu et al. (2020), which also visualizes only these two portfolio structures in the stock market context. This approach facilitates direct comparison across asset classes while preserving the full set of investable cryptocurrencies.

< Figure 33 >

Figure 33 presents the cumulative return plot, which provides further details. Among all models, XGBoost achieves the highest Sharpe ratio (0.618), reflecting the most efficient risk-adjusted performance based on long-short return spreads between decile 10 and decile 1. Nevertheless, its cumulative return curve does not reach the highest level by the end of the period. Models like PLS and GLM deliver higher raw cumulative returns, but also exhibit greater volatility, which slightly reduces their Sharpe scores. This apparent inconsistency highlights a key distinction: Sharpe ratio emphasizes return-to-risk efficiency, not absolute performance. XGBoost's steady and

well-separated return distribution results in superior Sharpe ratios, whereas PLS's and GLM's curve compounds more aggressively, but at the cost of increased variation.

Importantly, all machine learning models outperform the $1/N$ baseline strategy, indicating that data-driven signals provide substantial value beyond naïve diversification. In contrast, the OLS model performs worse than the baseline, both in cumulative return and Sharpe ratio. This outcome clearly demonstrates the advantage of nonlinear and regularized learning methods over traditional linear forecasting models.

The shaded regions in Figure 33 correspond to three major macro events that had notable impacts on the cryptocurrency market:

- **March 2020: COVID-19 Global Panic Selloff**
→ Market impact: Significant decline
- **January 2021: Institutional Inflows and Euphoria Phase**
→ Market impact: Strong rally
- **May 2021: Meme Coin Mania and Subsequent Reversal**
→ Market impact: Short-term spike followed by sharp reversal

These episodes are highlighted to contextualize model behavior during periods of extreme volatility and regime shifts. Observing how long-short portfolios respond

during these specific windows offers insight into their robustness under real world stress.

< Figure 34 >

Figure 34 displays the cumulative log returns of long and short portfolios under the Equal-Weighted, 10-group configuration. In contrast to the VW10 setting, here we observe that nearly all models “including OLS” consistently outperform the 1/N baseline strategy, indicating broad-based improvement in portfolio robustness under this setup.

The most outstanding performer in this configuration is XGBoost, which not only achieves the highest cumulative log return among all models but also maintains a stable and consistent growth trajectory throughout the sample period. Its dominance further confirms the model's capacity to capture predictive signals across a wide range of crypto assets, even when no value-based weighting is applied.

One plausible explanation lies in the nature of equal-weighted portfolios. Compared to value-weighted designs, EW schemes distribute capital more evenly across assets, placing relatively greater emphasis on mid- and small-cap coins. This reduces concentration in dominant assets (such as BTC or ETH), mitigates the risk of outlier distortion (e.g., COCOS-BCX), and allows the models' signals—particularly

those sensitive to cross-sectional variation—to have broader portfolio impact. This diversification benefit may explain why even simpler models like OLS and PCR, which typically underperform in VW settings, exhibit noticeably stronger returns in the EW10 configuration. In sum, equal-weighted designs appear to enhance the effectiveness of model-driven portfolio construction, with XGBoost once again standing out as the most consistent and powerful performer.

The shaded areas in Figure 35 mark three important episodes that shaped the cryptocurrency market during 2020–2022. Each period represents a unique combination of technological innovation, institutional behavior, and investor sentiment shifts that had meaningful impacts on portfolio dynamics:

- **September–October 2020: DeFi Expansion and Uniswap Launch**
→ Market impact: Sustained uptrend
- **January 2021: Bitcoin Breaks ATH, Institutions Enter Aggressively**
→ Market impact: Strong bullish breakout
- **December 2021 – January 2022: Final Rally and Regime Shift**
→ Market impact: Initial surge followed by soft reversal

These events are shaded to provide context for evaluating how different machine learning models respond to growth phases, institutional shifts, and speculative tops.

Their inclusion supports interpretation of cumulative return trajectories under diverse market conditions.

6. Conclusion

Using the cryptocurrency market as an empirical setting, we conduct a comprehensive comparison of machine learning methods for predicting cross-sectional returns. Our findings demonstrate that machine learning can substantially improve predictive accuracy relative to traditional linear models, especially in environments characterized by high dimensionality and structural uncertainty. Among the methods considered, XGBoost consistently delivers the best out-of-sample performance, outperforming both deep neural networks and penalized regressions. We trace this advantage to its ability to capture complex, nonlinear interactions among predictors—features that are particularly valuable in the decentralized and volatile context of crypto assets.

Interestingly, we find that shallow or moderately complex models outperform deeper learning architectures, a result that contrasts with findings in fields such as computer vision. This likely reflects the relatively limited sample size and low signal-to-noise ratio in crypto asset pricing, which constrain the effective depth of learnable structure. Machine learning methods are most effective in forecasting the returns of higher-cap tokens, where liquidity and trading volume help stabilize signals. Lastly, we

find that models consistently rely on a core set of predictors related to volatility (RETSKEW, RETKURT, MAXRET), volume dispersion (STDPRCOL), and size (MCAP). In contrast to equity studies, momentum signals contribute little to predictive accuracy in the cryptocurrency setting.

The strong performance of machine learning methods in predicting cryptocurrency returns holds important implications for both academic research and practical investment management. By providing more accurate estimates of risk premiums in a market where traditional pricing models fail to gain traction, machine learning helps reduce the noise and estimation error that often obscure return dynamics. This improved measurement lays a stronger empirical foundation for future work aiming to uncover the underlying economic mechanisms of digital asset pricing.

In addition, our findings support the growing role of machine learning in the infrastructure of modern financial technology. Accurate prediction of token-level returns facilitates data-driven portfolio construction, systematic asset selection, and algorithmic trading, especially in decentralized and rapidly evolving markets such as cryptocurrencies. As the Fintech industry continues to expand into alternative asset classes, our study highlights the practical value of integrating machine learning tools into the architecture of crypto-focused investment platforms.

References

- Ang, A., Hodrick, R. J., Xing, Y., & Zhang, X. (2006). The cross-section of volatility and expected returns. *The Journal of Finance*, 61(1), 259-299.
- Asness, C. S., Moskowitz, T. J., & Pedersen, L. H. (2013). Value and momentum everywhere. *The Journal of Finance*, 68(3), 929-985.
- Bali, T. G., Beckmeyer, H., Moerke, M., & Weigert, F. (2023). Option return predictability with machine learning and big data. *The Review of Financial Studies*, 36(9), 3548-3602.
- Bali, T. G., & Cakici, N. (2008). Idiosyncratic volatility and the cross section of expected returns. *Journal of Financial and Quantitative Analysis*, 43(1), 29-58.
- Banz, R. W. (1981). The relationship between return and market value of common stocks. *Journal of Financial economics*, 9(1), 3-18.
- Biais, B., Bisiere, C., Bouvard, M., & Casamatta, C. (2019). The blockchain folk theorem. *The Review of Financial Studies*, 32(5), 1662-1715.
- Borri, N., & Shakhnov, K. (2022). The cross-section of cryptocurrency returns. *The Review of Asset Pricing Studies*, 12(3), 667-705.
- Breiman, L. (2001). Random forests. *Machine learning*, 45(1), 5-32.

Breiman, L., Friedman, J., Olshen, R. A., & Stone, C. J. (2017). *Classification and regression trees*. Chapman and Hall/CRC.

Chordia, T., Roll, R., & Subrahmanyam, A. (2001). Market liquidity and trading activity. *The Journal of Finance*, 56(2), 501-530.

Cybenko, G. (1989). Approximation by superpositions of a sigmoidal function. *Mathematics of control, signals and systems*, 2(4), 303-314.

De Jong, S. (1993). SIMPLS: an alternative approach to partial least squares regression. *Chemometrics and intelligent laboratory systems*, 18(3), 251-263.

Diebold, F. X. (2015). Comparing predictive accuracy, twenty years later: A personal perspective on the use and abuse of Diebold–Mariano tests. *Journal of Business & Economic Statistics*, 33(1), 1-1.

Diebold, F. X., & Mariano, R. S. (2002). Comparing predictive accuracy. *Journal of Business & economic statistics*, 20(1), 134-144.

Dietterich, T. G. (2000, June). Ensemble methods in machine learning. In *International workshop on multiple classifier systems* (pp. 1-15). Berlin, Heidelberg: Springer Berlin Heidelberg.

Efficiency, S. M. (1993). Returns to buying winners and selling losers: Implications for. *The Journal of Finance*, 48(1), 65-91.

Falcon, A., & Lyu, T. (2021). Daily cryptocurrency returns forecasting and trading via machine learning. *Journal of Student Research*, 10(4).

Fama, E. F., & French, K. R. (1992). The cross-section of expected stock returns. *the Journal of Finance*, 47(2), 427-465.

Fama, E. F., & French, K. R. (2008). Dissecting anomalies. *The Journal of Finance*, 63(4), 1653-1678.

Feng, G., Giglio, S., & Xiu, D. (2020). Taming the factor zoo: A test of new factors. *The Journal of Finance*, 75(3), 1327-1370.

Freyberger, J., Neuhierl, A., & Weber, M. (2020). Dissecting characteristics nonparametrically. *The Review of Financial Studies*, 33(5), 2326-2377.

Giglio, S., & Xiu, D. (2017). *Inference on risk premia in the presence of omitted factors* (No. w23527). National Bureau of Economic Research.

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

Hansen, L. K., & Salamon, P. (2002). Neural network ensembles. *IEEE transactions on pattern analysis and machine intelligence*, 12(10), 993-1001.

Heaton, J. B., Polson, N. G., & Witte, J. H. (2016). Deep learning in finance. *arXiv preprint arXiv:1602.06561*.

Hornik, K., Stinchcombe, M., & White, H. (1989). Multilayer feedforward networks are universal approximators. *Neural networks*, 2(5), 359-366.

Hubrich, S. (2017). 'Know When to Hodl'Em, Know When to Fodl'Em': An Investigation of Factor Based Investing in the Cryptocurrency Space. *Know When to Fodl'Em': An Investigation of Factor Based Investing in the Cryptocurrency Space (October 28, 2017)*.

Hutchinson, J. M., Lo, A. W., & Poggio, T. (1994). A nonparametric approach to pricing and hedging derivative securities via learning networks. *The Journal of Finance*, 49(3), 851-889.

Ioffe, S., & Szegedy, C. (2015, June). Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning* (pp. 448-456). pmlr.

Kelly, B. T., Pruitt, S., & Su, Y. (2017). Some characteristics are risk exposures, and the rest are irrelevant. *Unpublished Manuscript, University of Chicago*, 76.

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

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

Koijen, R. S., & Van Nieuwerburgh, S. (2011). Predictability of returns and cash flows. *Annu. Rev. Financ. Econ.*, 3(1), 467-491.

Kozak, S., Nagel, S., & Santosh, S. (2020). Shrinking the cross-section. *Journal of Financial Economics*, 135(2), 271-292.

Lee, S. S., & Wang, M. (2024). Variance decomposition and cryptocurrency return prediction. *Journal of Financial and Quantitative Analysis*, 1-32.

Lewellen, J. (2014). The cross section of expected stock returns. *Forthcoming in Critical Finance Review, Tuck School of Business Working Paper*, (2511246).

Li, J., & Yi, G. (2018). Toward a factor structure in crypto asset returns. *Available at SSRN 3258231*.

Liu, Y., & Tsyvinski, A. (2021). Risks and returns of cryptocurrency. *The Review of Financial Studies*, 34(6), 2689-2727.

Liu, Y., Tsyvinski, A., & Wu, X. (2022). Common risk factors in cryptocurrency. *The Journal of Finance*, 77(2), 1133-1177.

Masters, T. (1993). *Practical neural network recipes in C++*. Morgan Kaufmann.

Moskowitz, T. J., & Grinblatt, M. (1999). Do industries explain momentum?. *The Journal of Finance*, 54(4), 1249-1290.

Moskowitz, T. J., Ooi, Y. H., & Pedersen, L. H. (2012). Time series momentum. *Journal of Financial economics*, 104(2), 228-250.

Rapach, D. E., Strauss, J. K., & Zhou, G. (2013). International stock return predictability: What is the role of the United States?. *The Journal of Finance*, 68(4), 1633-1662.

Routledge, B., & Zetlin-Jones, A. (2022). Currency stability using blockchain technology. *Journal of Economic Dynamics and Control*, 142, 104155.

Soska, K., Dong, J. D., Khodaverdian, A., Zetlin-Jones, A., Routledge, B., & Christin, N. (2021, April). Towards understanding cryptocurrency derivatives: A case study

of BitMEX. In *Proceedings of the Web Conference 2021* (pp. 45-57).

Stoffels, J. (2017). Asset pricing of cryptocurrencies and momentum based patterns. *Erasmus School of Economics working paper*.

Trimborn, S., Li, M., & Härdle, W. K. (2020). Investing with cryptocurrencies—A liquidity constrained investment approach. *Journal of Financial Econometrics*, 18(2), 280-306.

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

Yao, J., Li, Y., & Tan, C. L. (2000). Option price forecasting using neural networks. *Omega*, 28(4), 455-466.

Yermack, D. (2024). Is Bitcoin a real currency? An economic appraisal. In *Handbook of digital currency* (pp. 29-40). Academic Press.

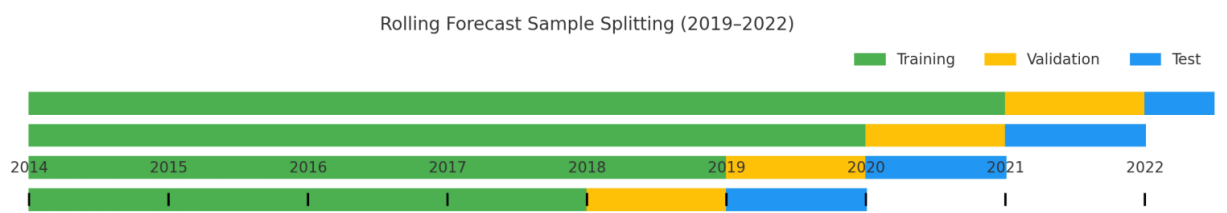


Figure 1

Rolling Forecast Sample Splitting (2019–2022)

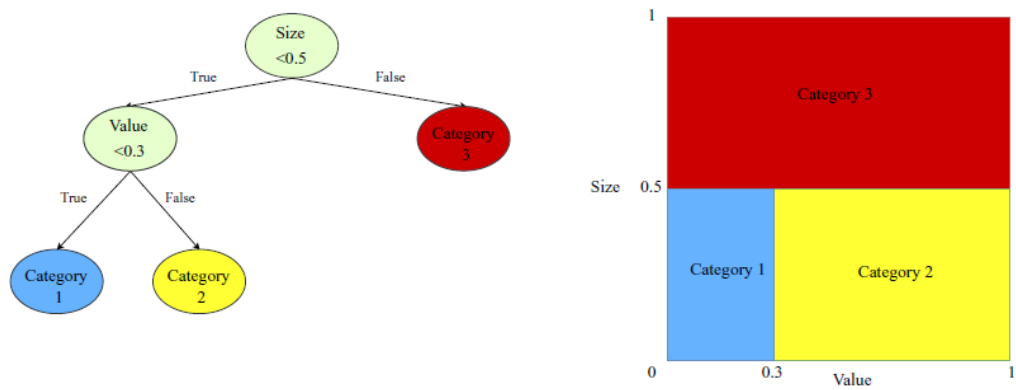


Figure 2

Regression tree example

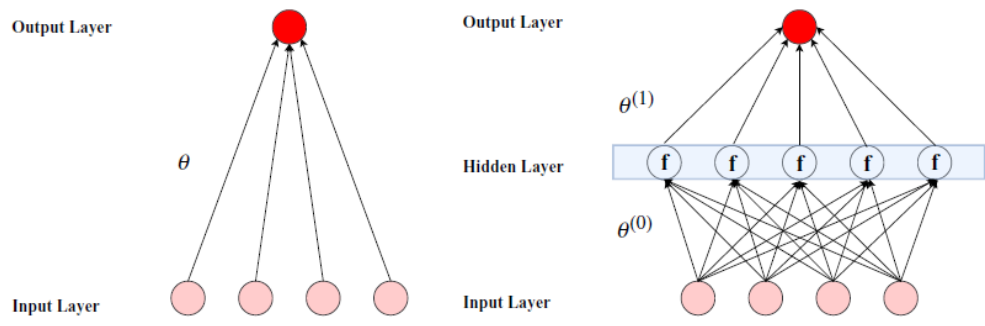


Figure 3

Neural networks

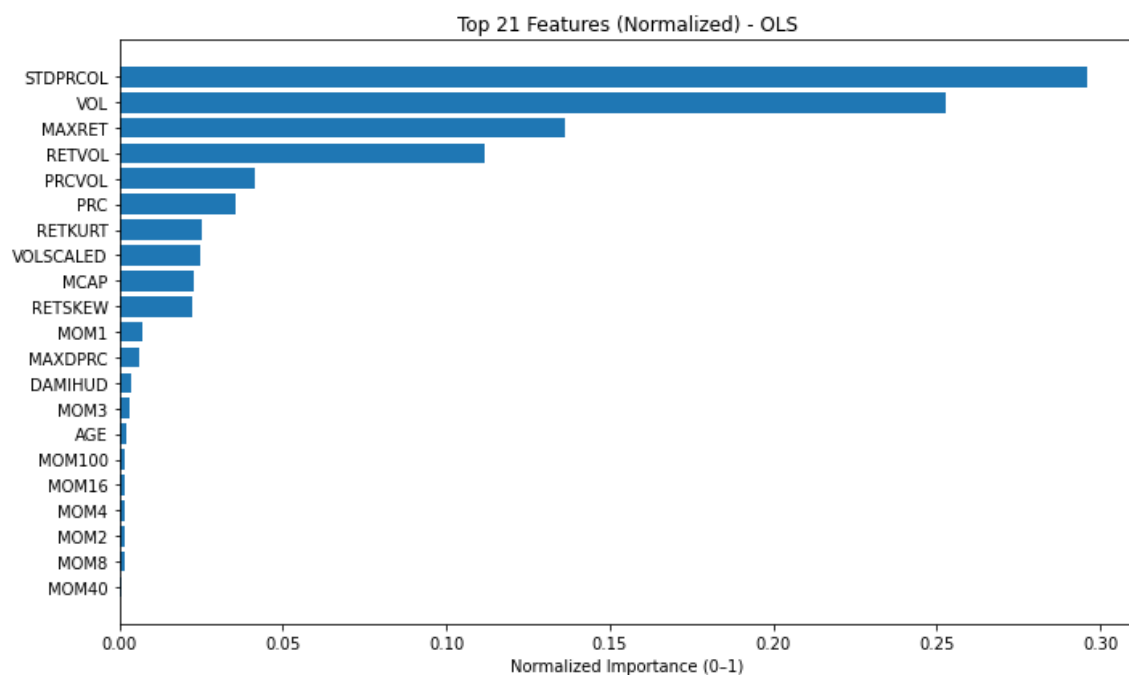


Figure 4

Feature Importance OLS

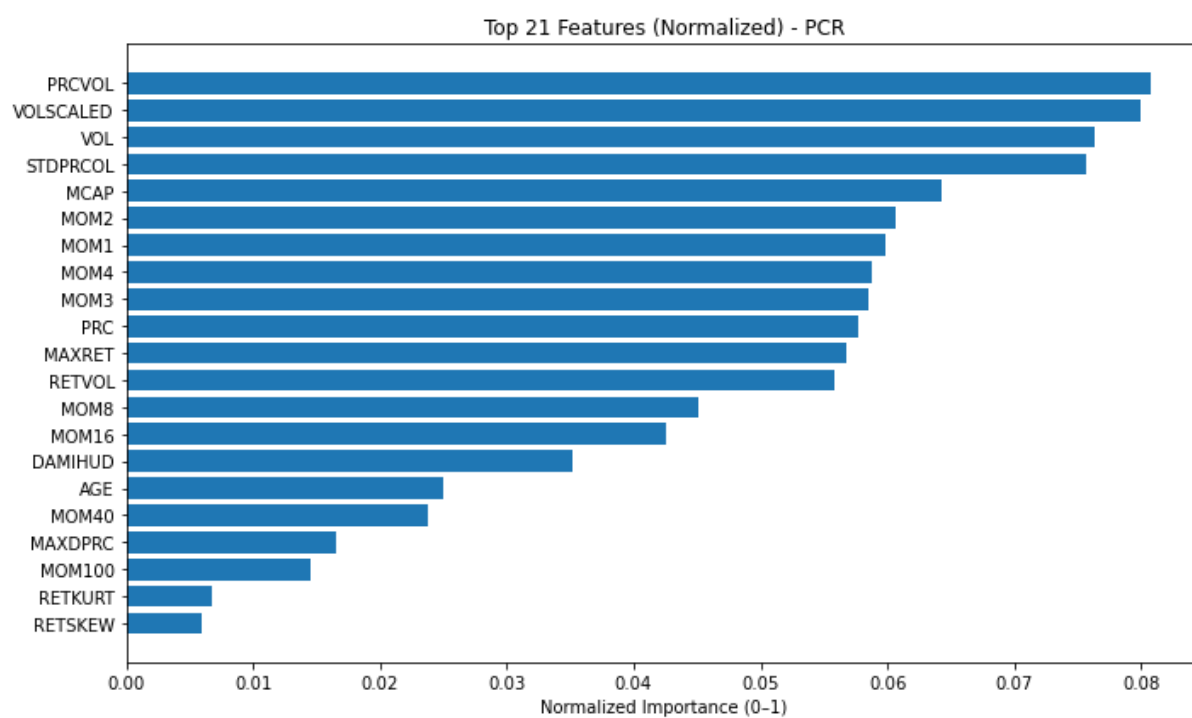


Figure 5

Feature Importance PCR

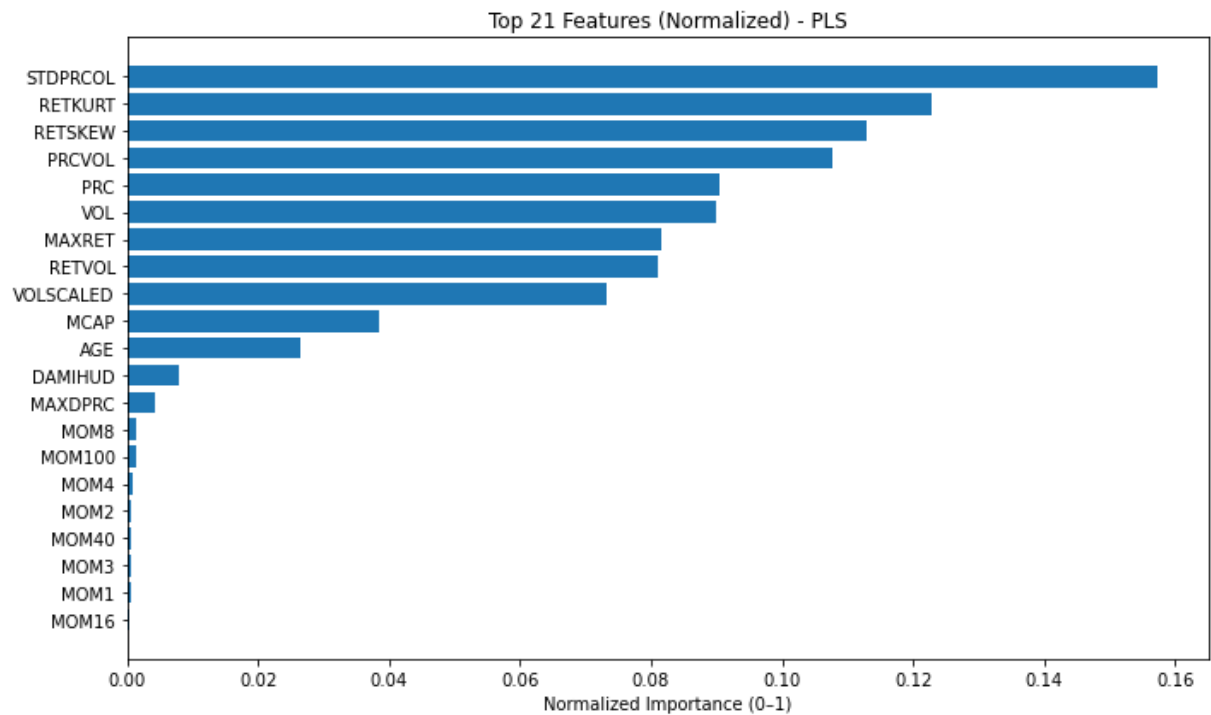


Figure 6

Feature Importance PLS

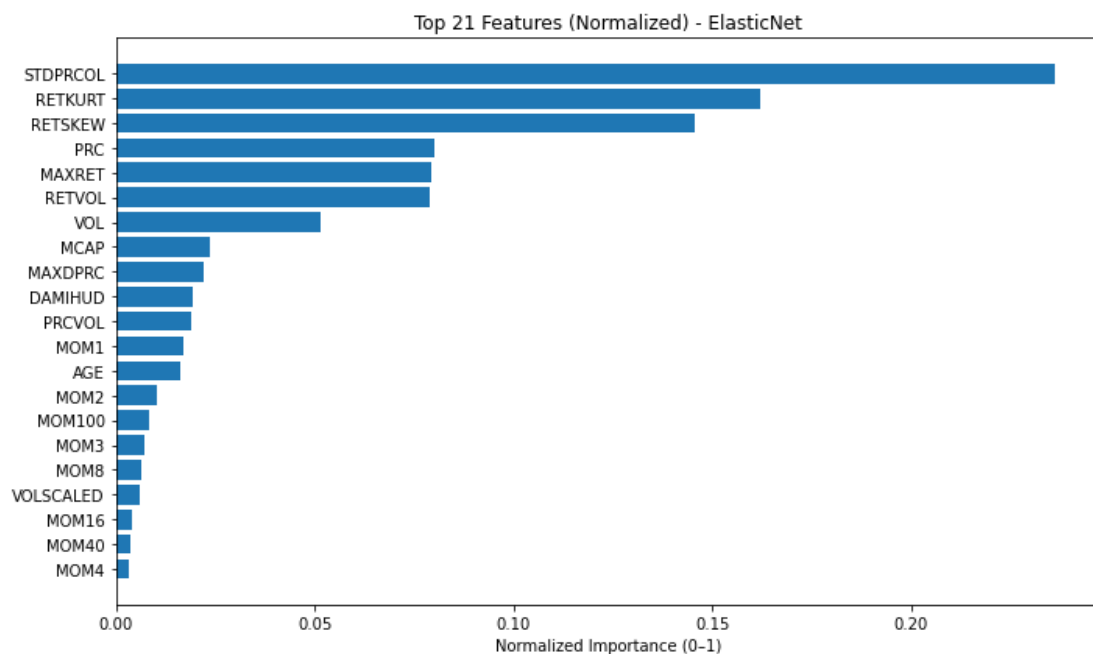


Figure 7

Feature Importance ENET

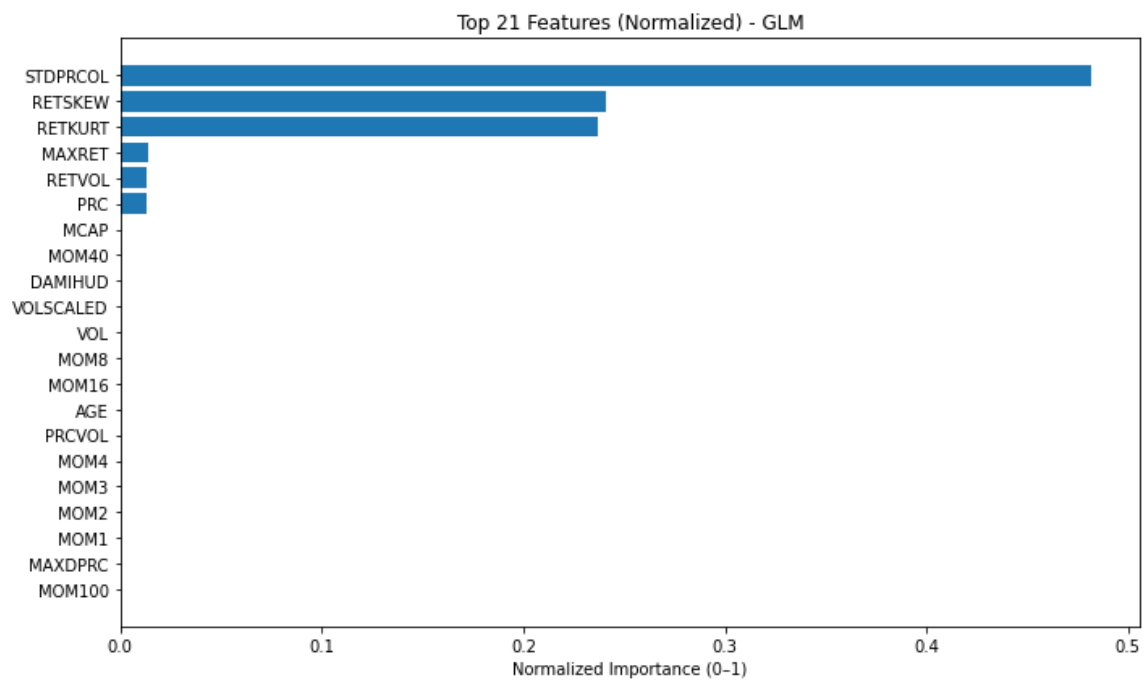


Figure 8

Feature Importance GLM

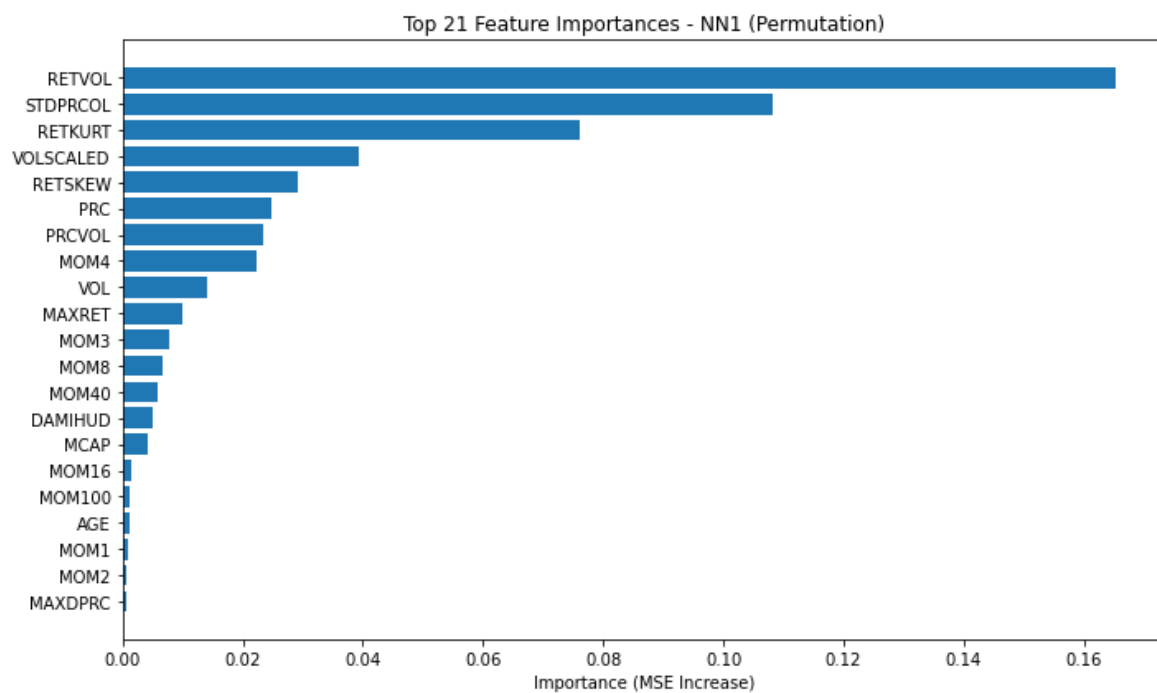


Figure 9

Feature Importance NN1

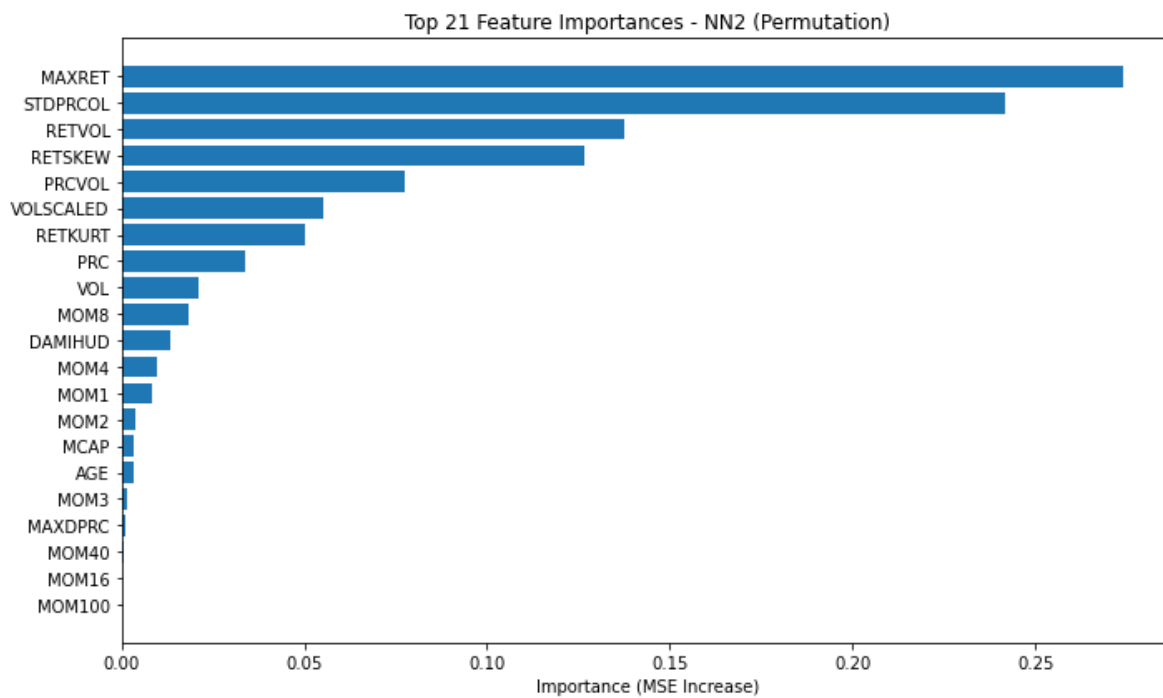


Figure 10

Feature Importance NN2

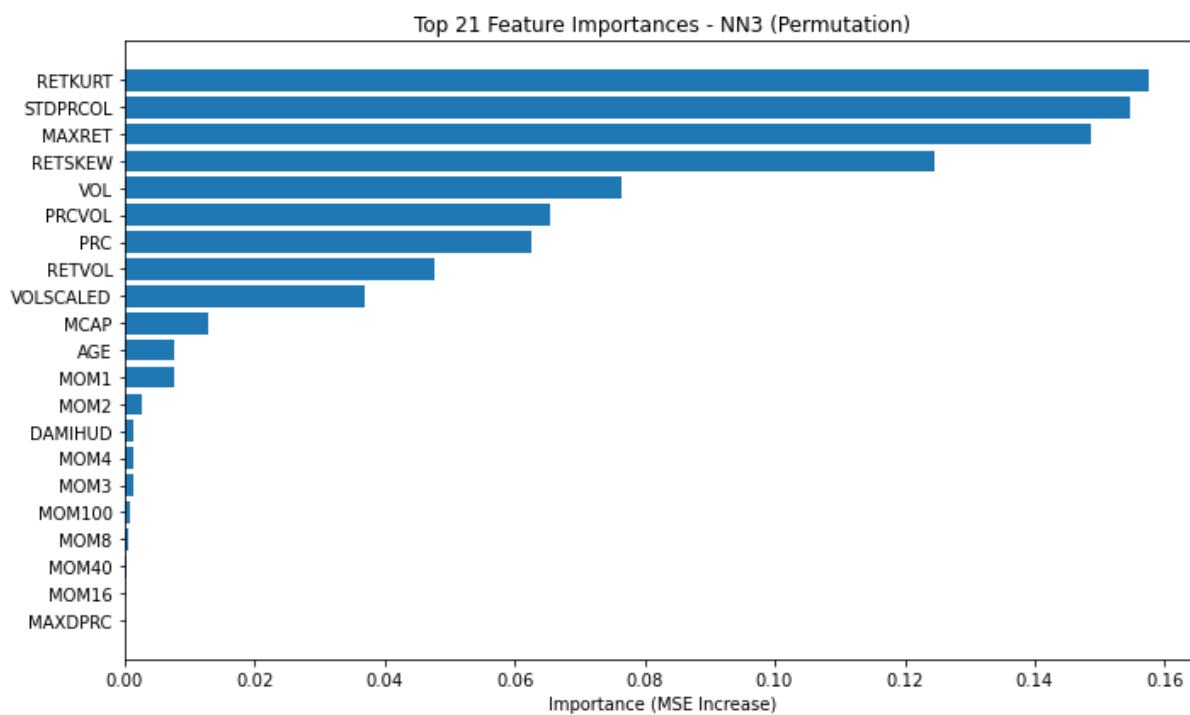


Figure 11

Feature Importance NN3

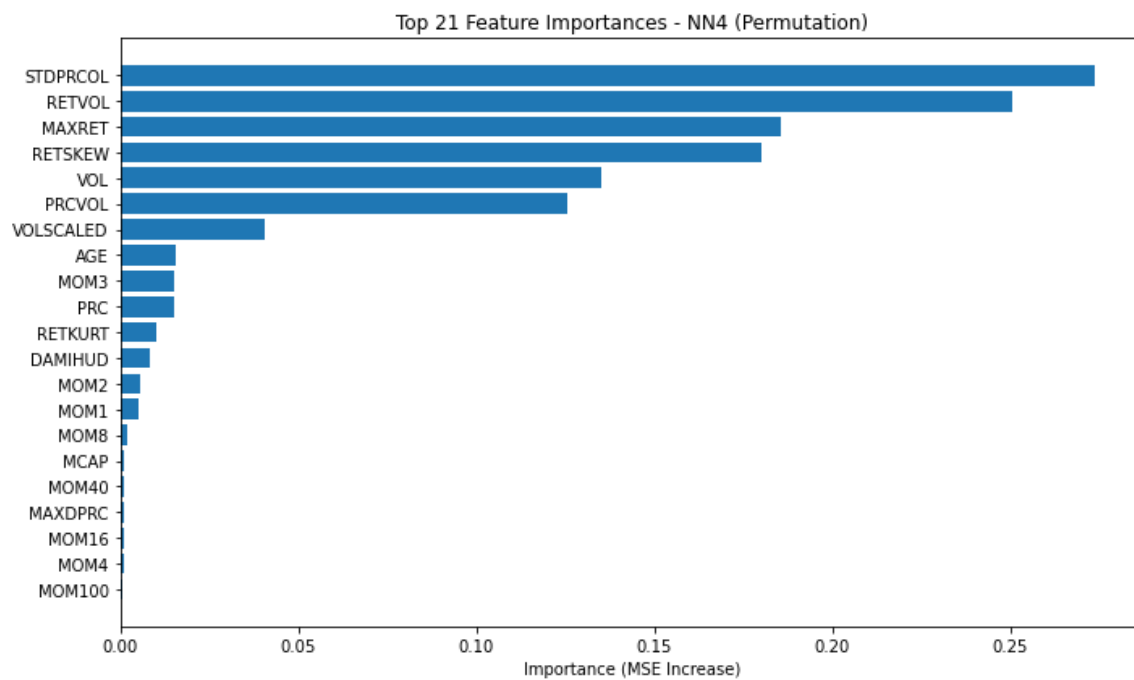


Figure 12

Feature Importance NN4

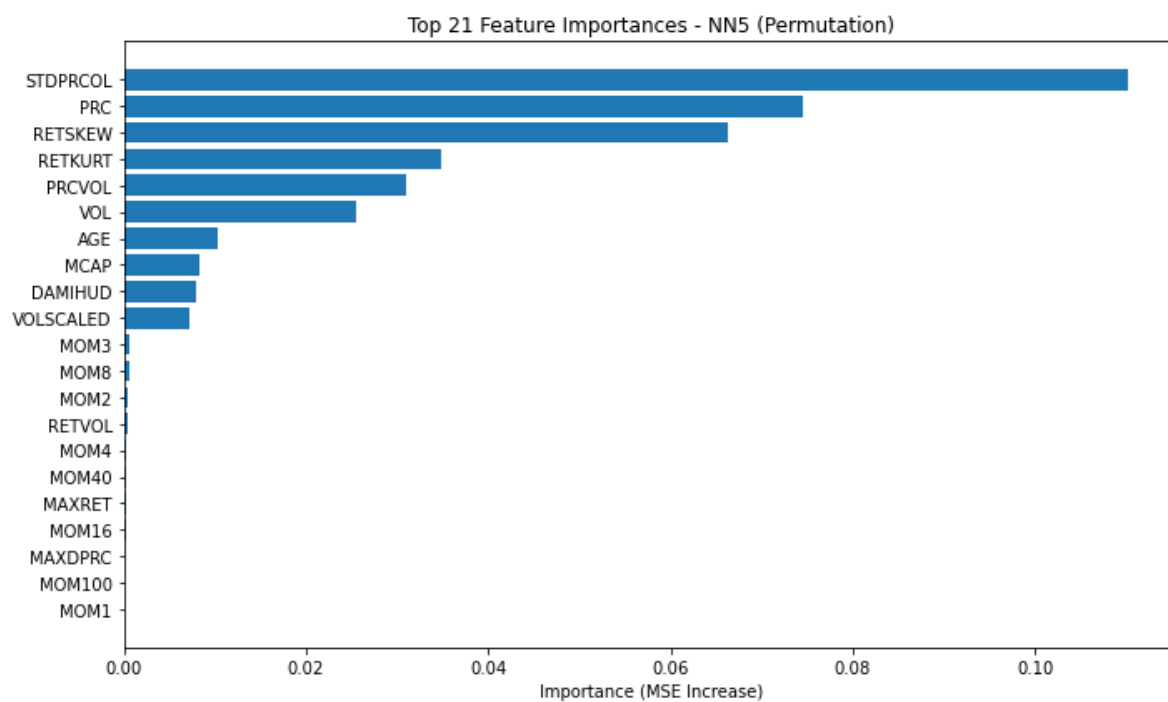


Figure 13

Feature Importance NN5

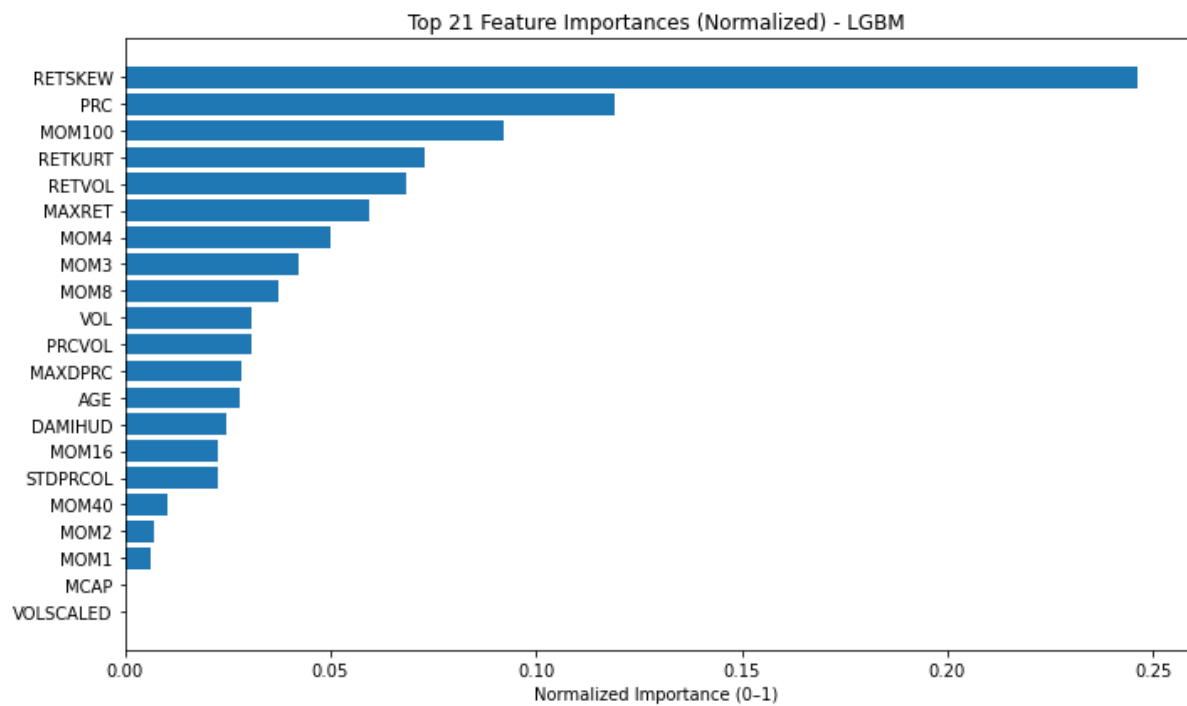


Figure 14

Feature Importance LGBM

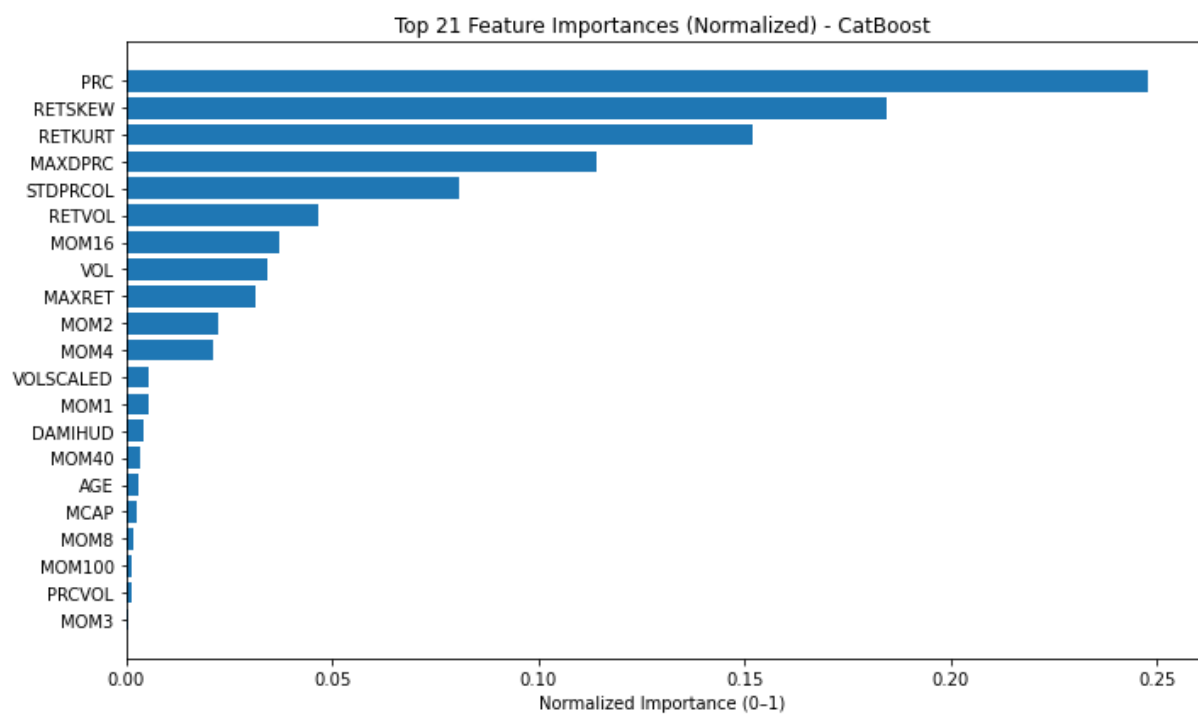


Figure 15

Feature Importance CatBoost

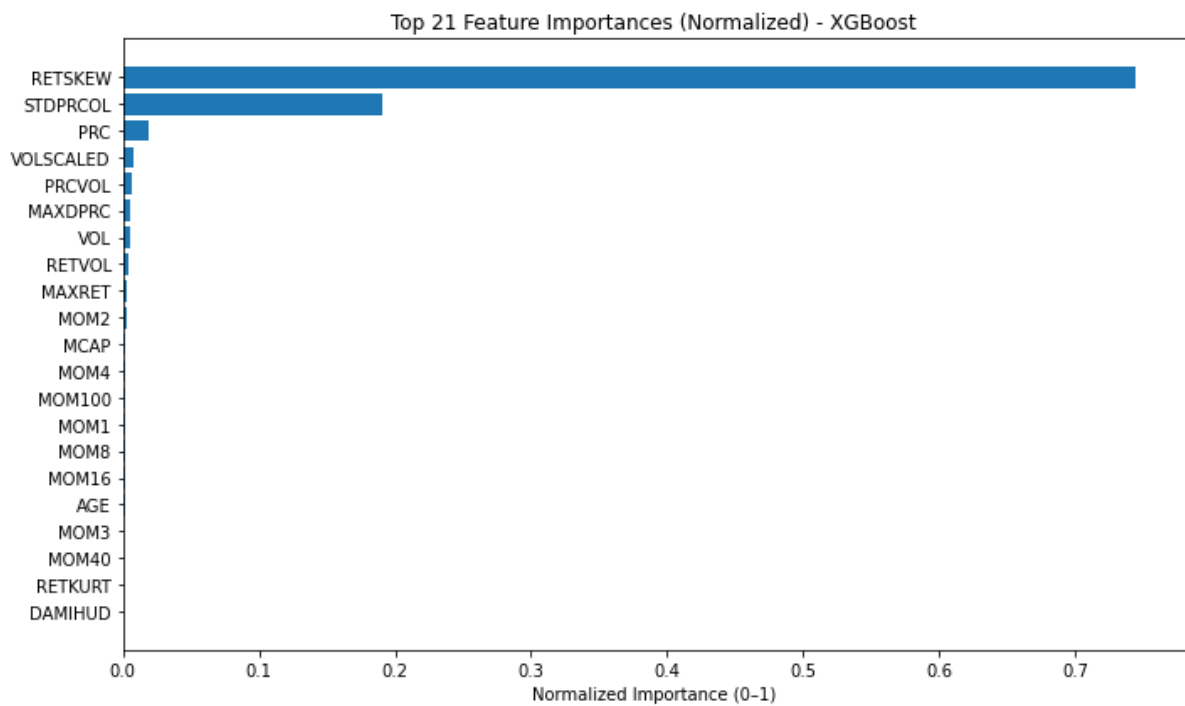


Figure 16

Feature Importance XGB

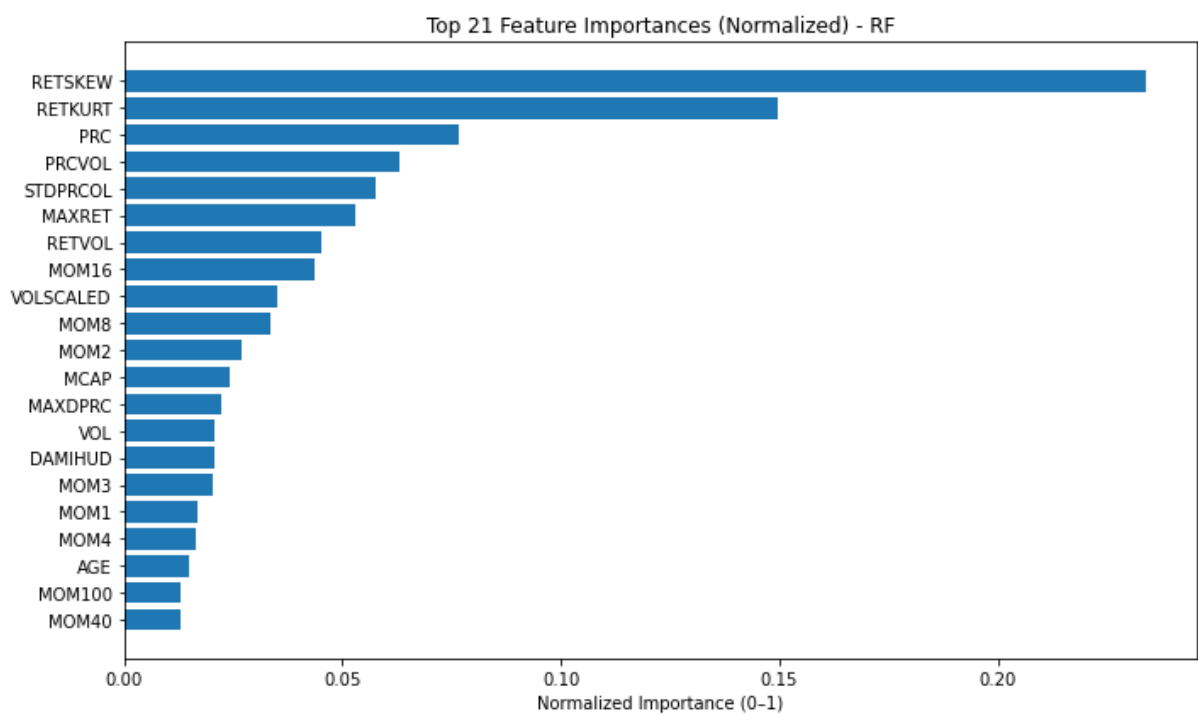


Figure 17

Feature Importance RF

Feature importance for each model

This set of figures illustrates the top 21 predictor importance scores, normalized within each model. The visualizations enable comparison of how different models prioritize cryptocurrency return predictors.

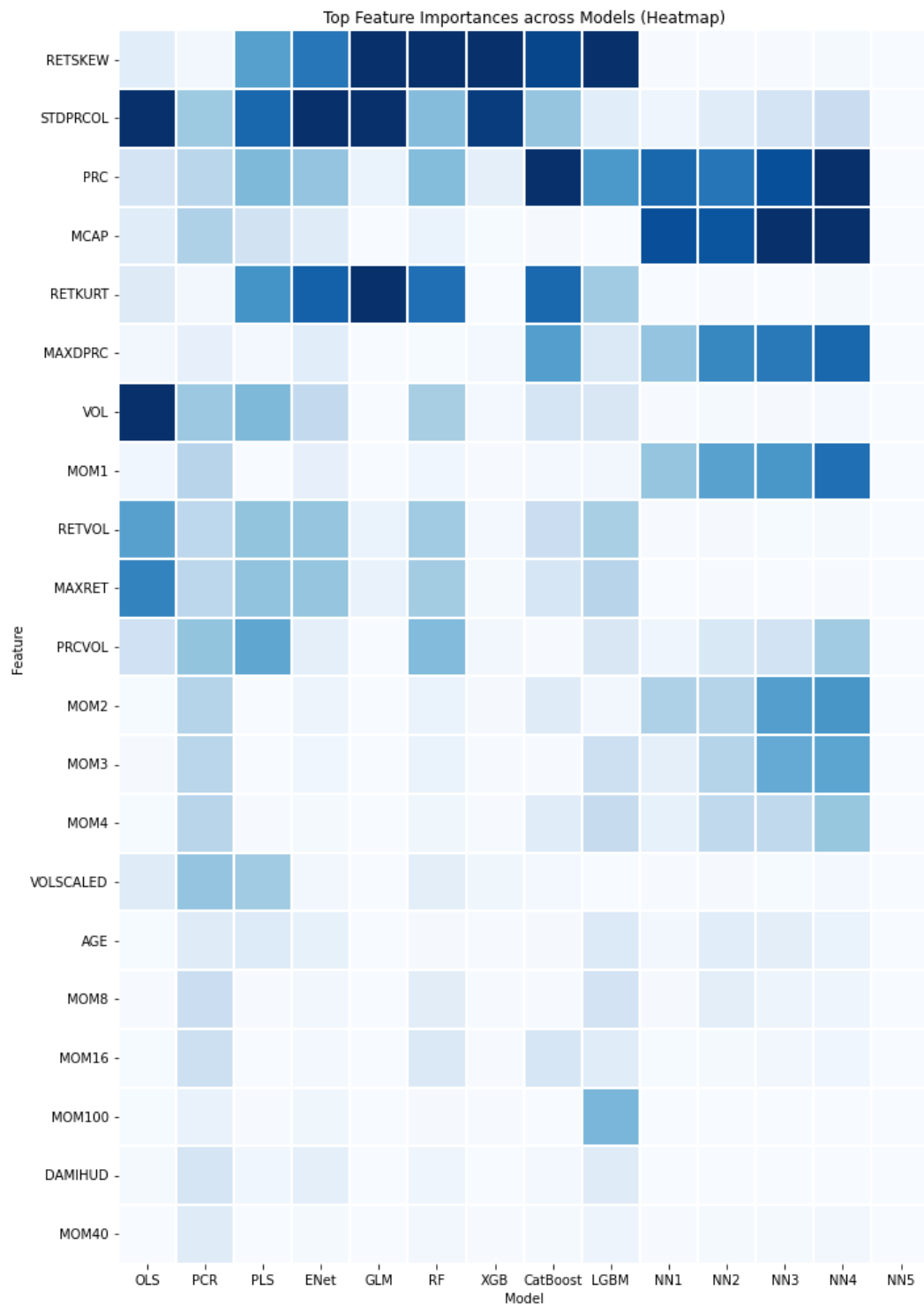


Figure 18

Heatmap of top feature importance across all models.

This heatmap summarizes and compares the relative importance of all 21 predictors across the 14 models. Darker colors indicate higher importance within each model.

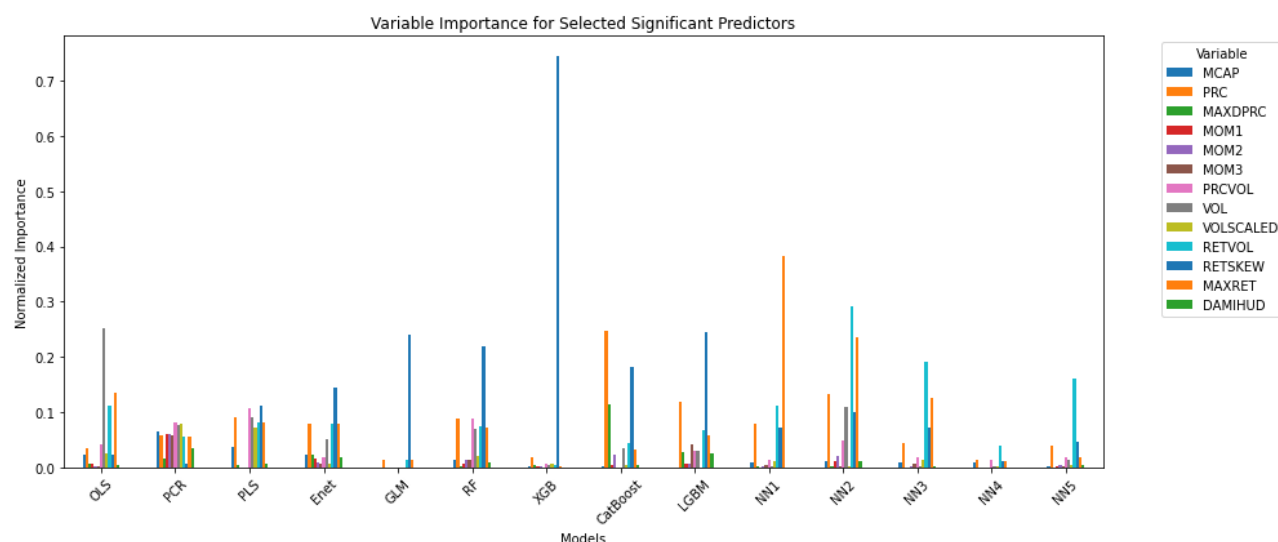


Figure 19

Bar plot of normalized feature importance across models for the 13 statistically significant predictors.

This figure illustrates the relative importance (normalized to sum to one within each model) of 13 selected predictors that were previously identified as statistically significant. It highlights how different models prioritize these variables differently.

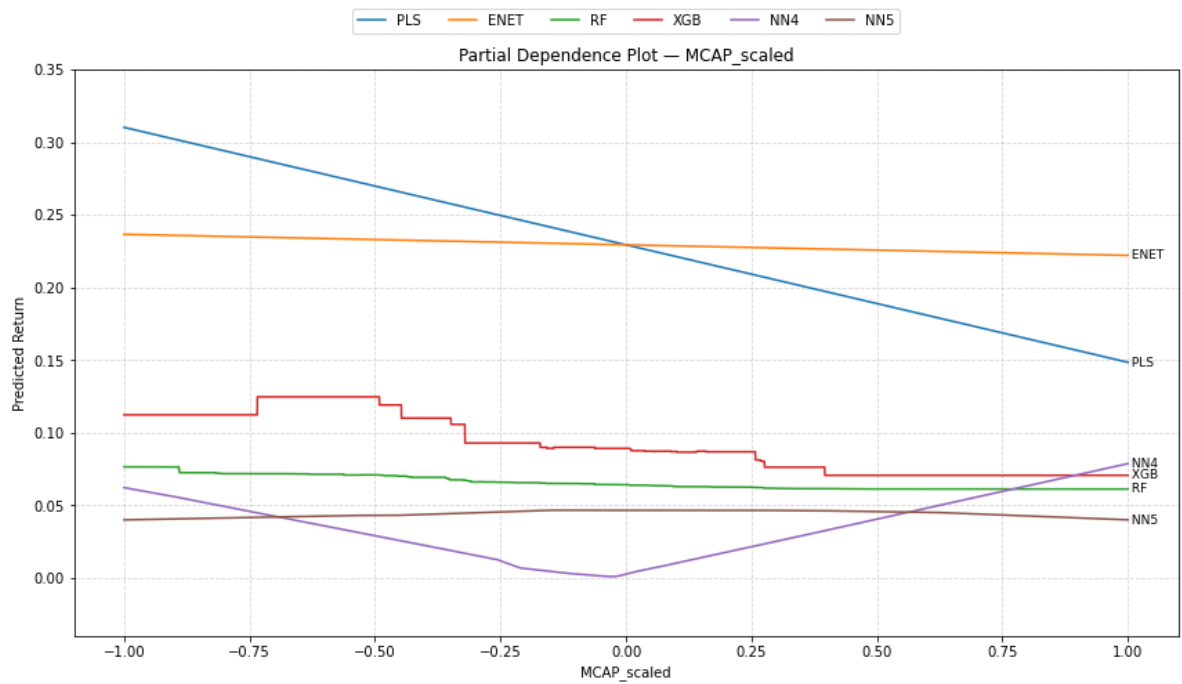


Figure 20

Partial Dependence Plot of MCAP

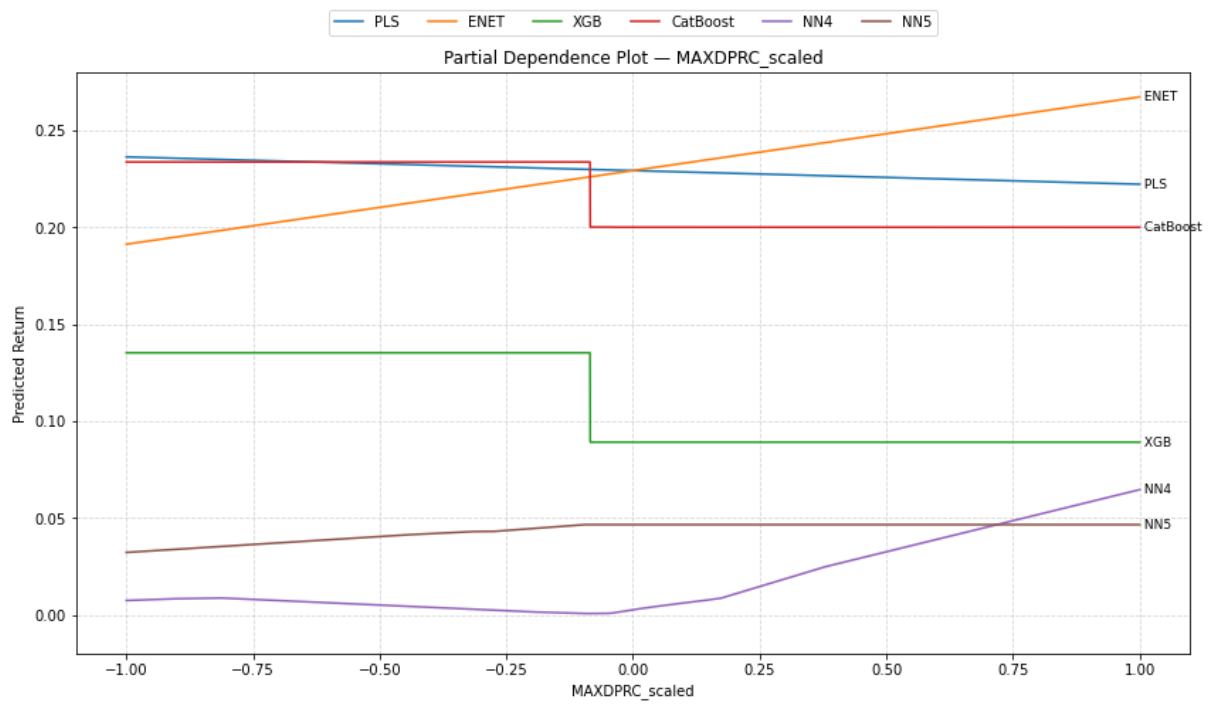


Figure 21

Partial Dependence Plot of MAXDPRC

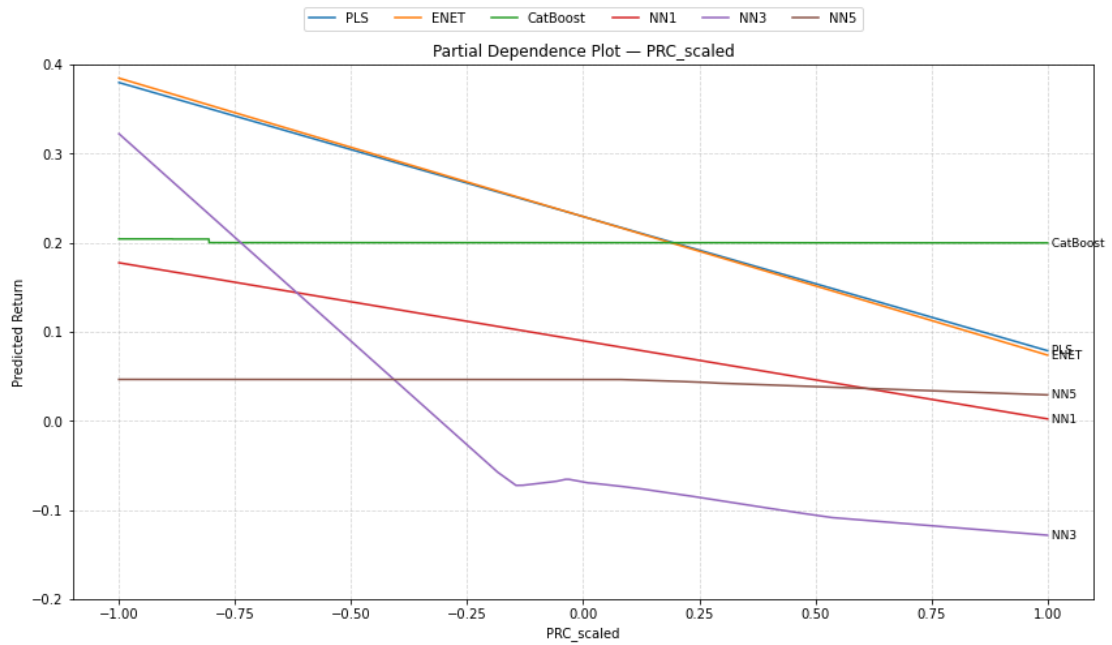


Figure 22

Partial Dependence Plot of PRC

Partial Dependence Plot of Size factor

This figure presents the 5-1 value-weighted return sorting results and the corresponding Partial Dependence Plots (PDPs) for selected Size predictors. Significant t-values in the sorting tables indicate return predictability, while the slope direction of PDPs reveals how each predictor affects expected returns across models.

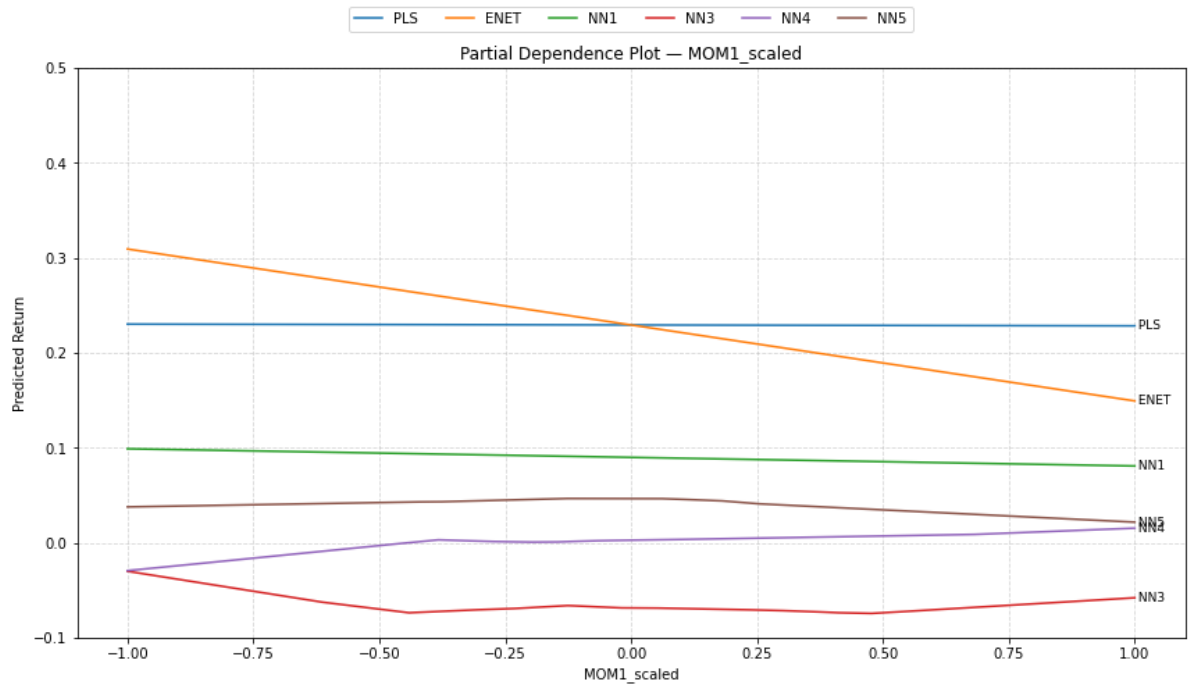


Figure 23

Partial Dependence Plot of MOM1

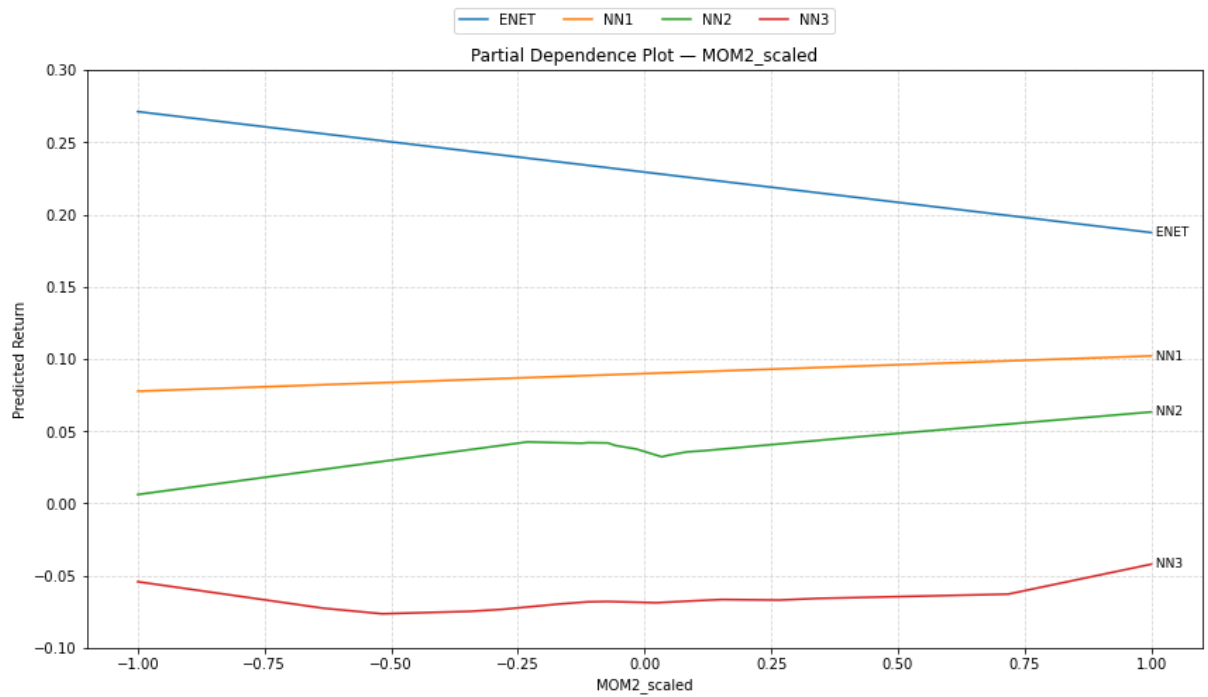


Figure 24

Partial Dependence Plot of MOM2

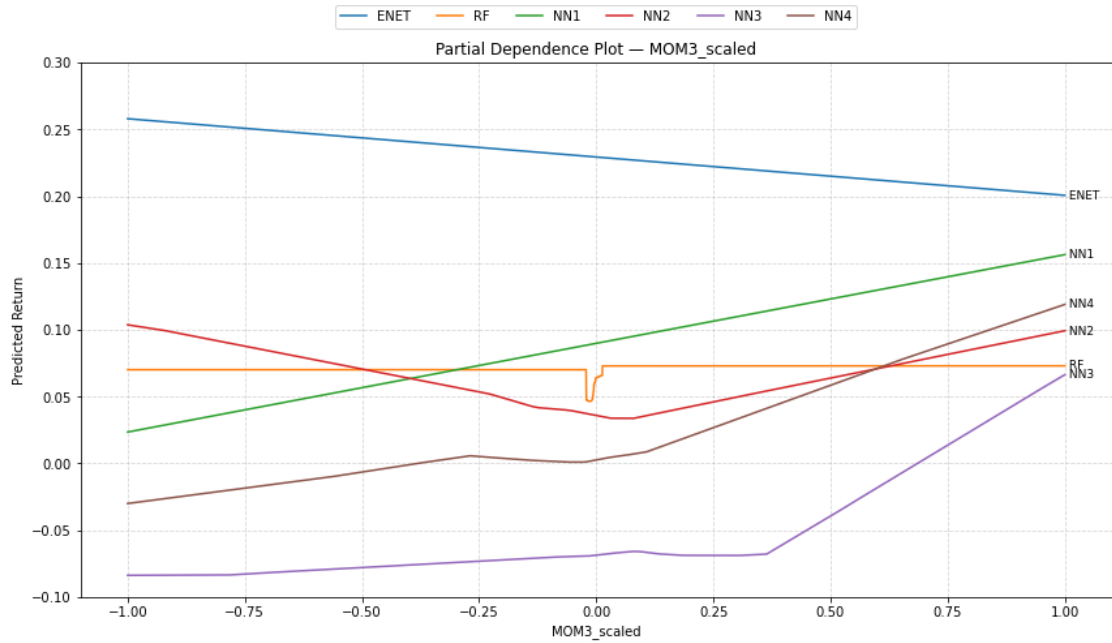


Figure 25

Partial Dependence Plot of MOM3

Partial Dependence Plot of Momentum factor

This figure presents the 5-1 value-weighted return sorting results and the corresponding Partial Dependence Plots (PDPs) for selected Momentum predictors. Significant t-values in the sorting tables indicate return predictability, while the slope direction of PDPs reveals how each predictor affects expected returns across models.

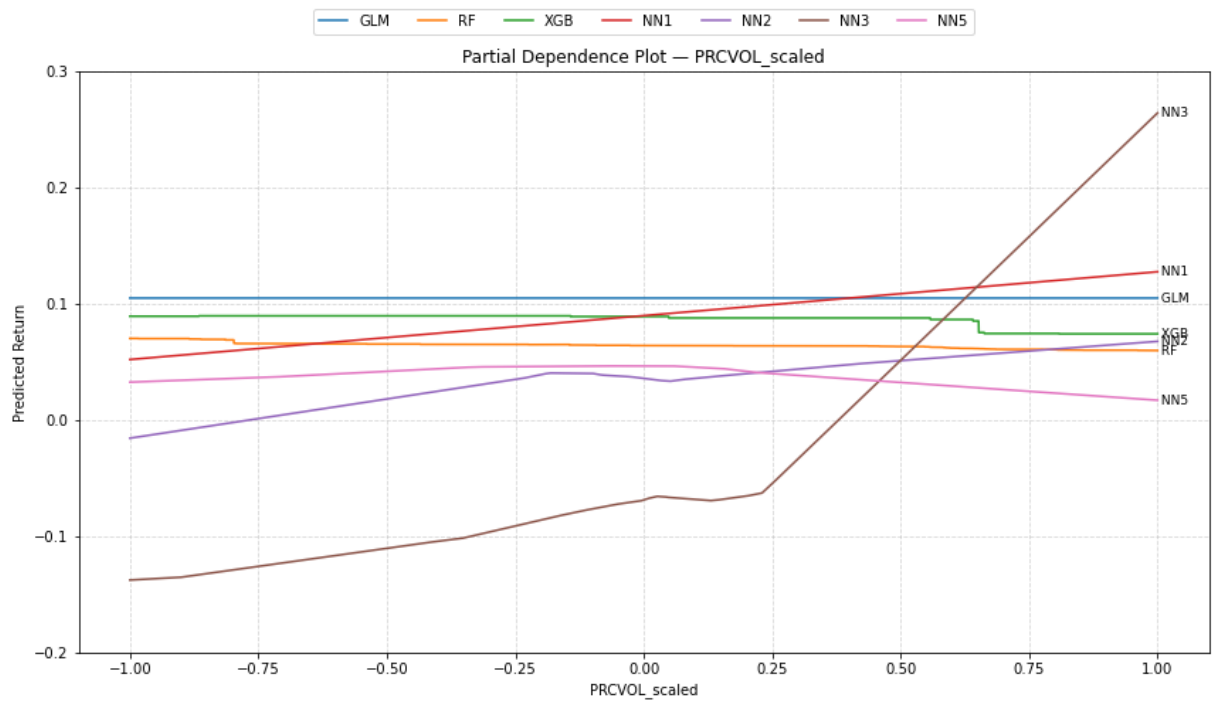


Figure 26

Partial Dependence Plot of PRCVOL

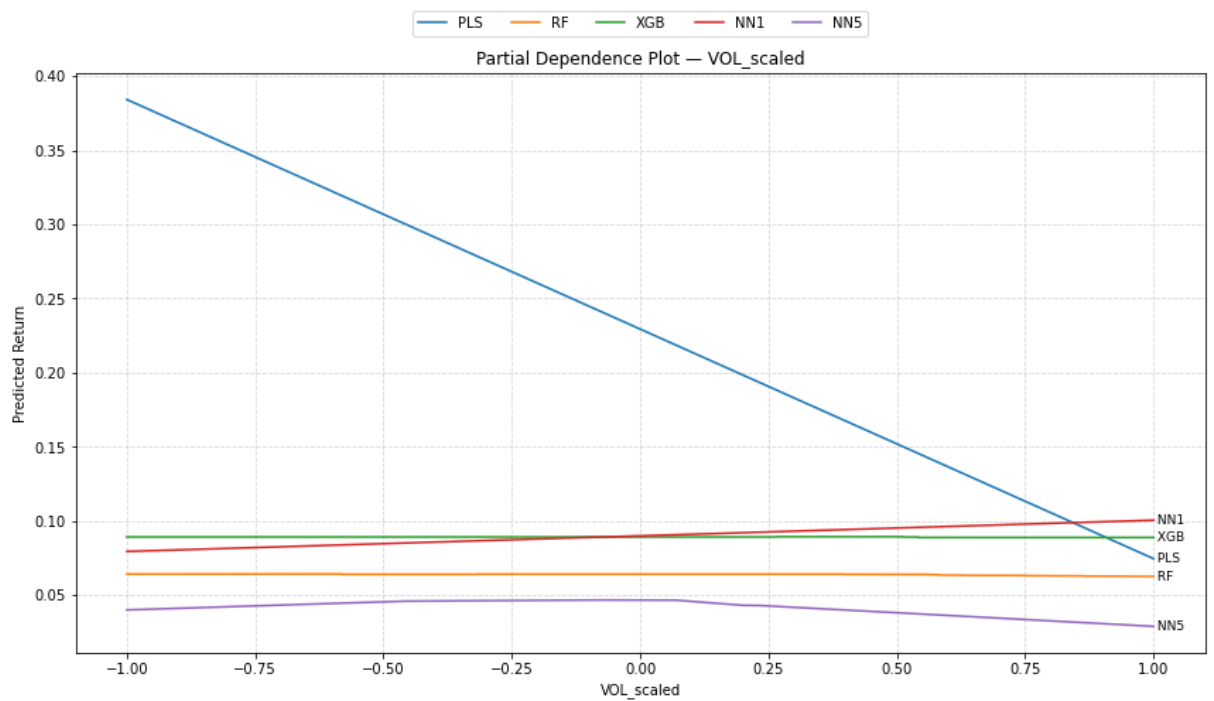


Figure 27

Partial Dependence Plot of VOL

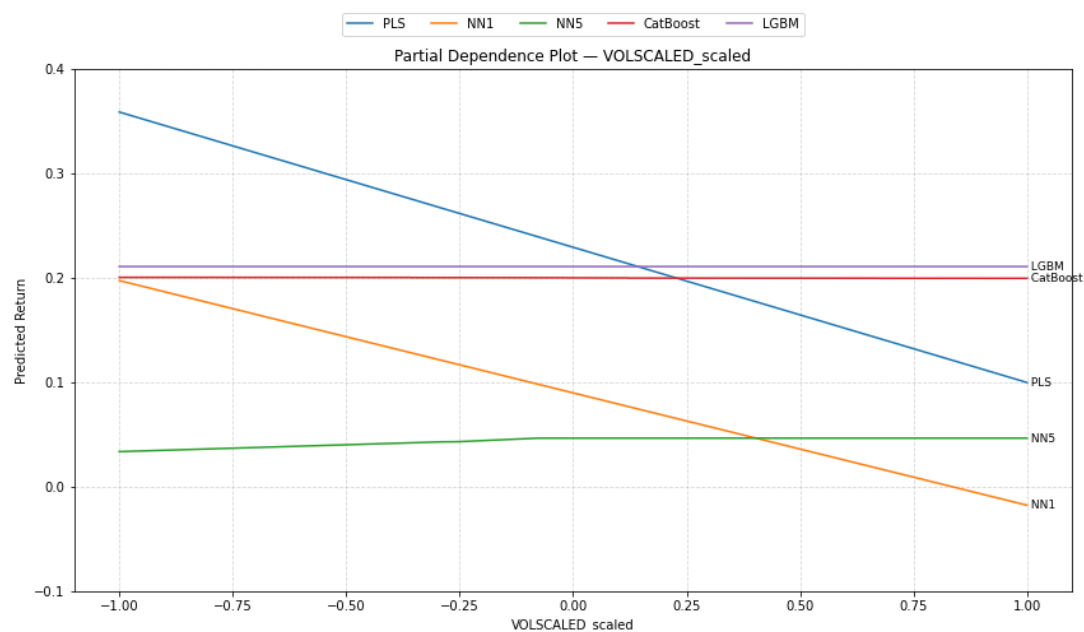


Figure 28

Partial Dependence Plot of VOLSCALED

Partial Dependence Plot of Volume factor

This figure presents the 5-1 value-weighted return sorting results and the corresponding Partial Dependence Plots (PDPs) for selected Volume predictors. Significant t-values in the sorting tables indicate return predictability, while the slope direction of PDPs reveals how each predictor affects expected returns across models.

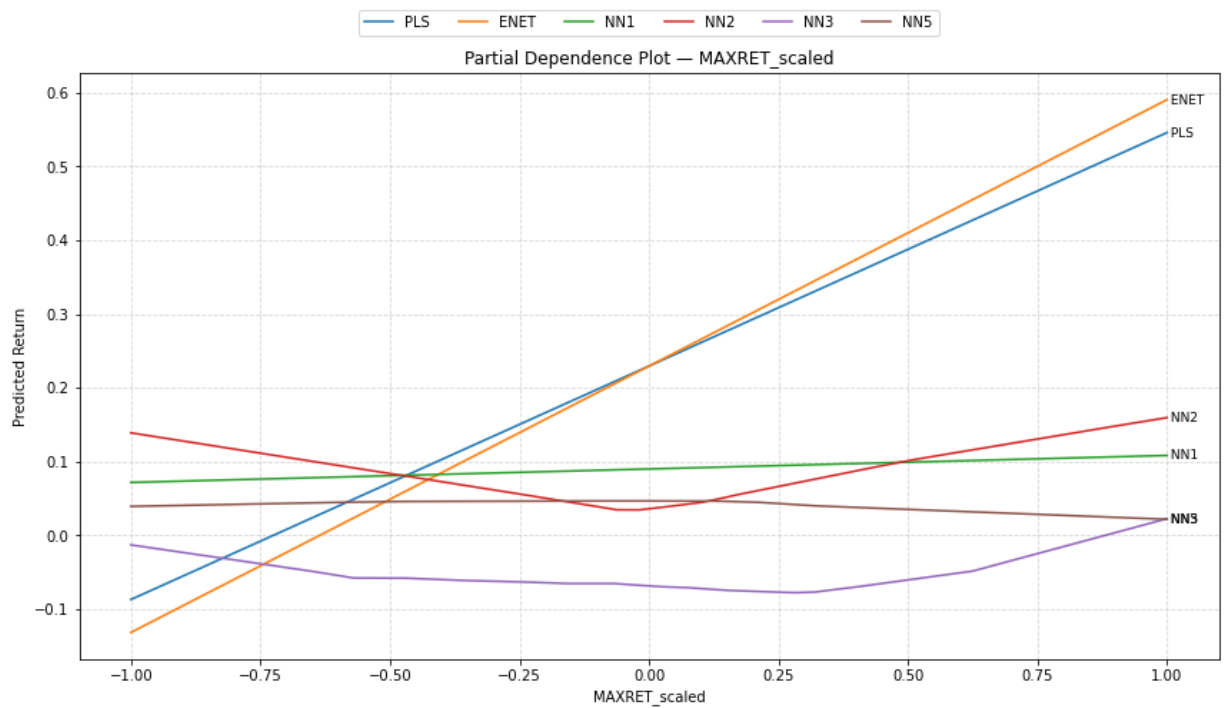


Figure 29

Partial Dependence Plot of MAXRET

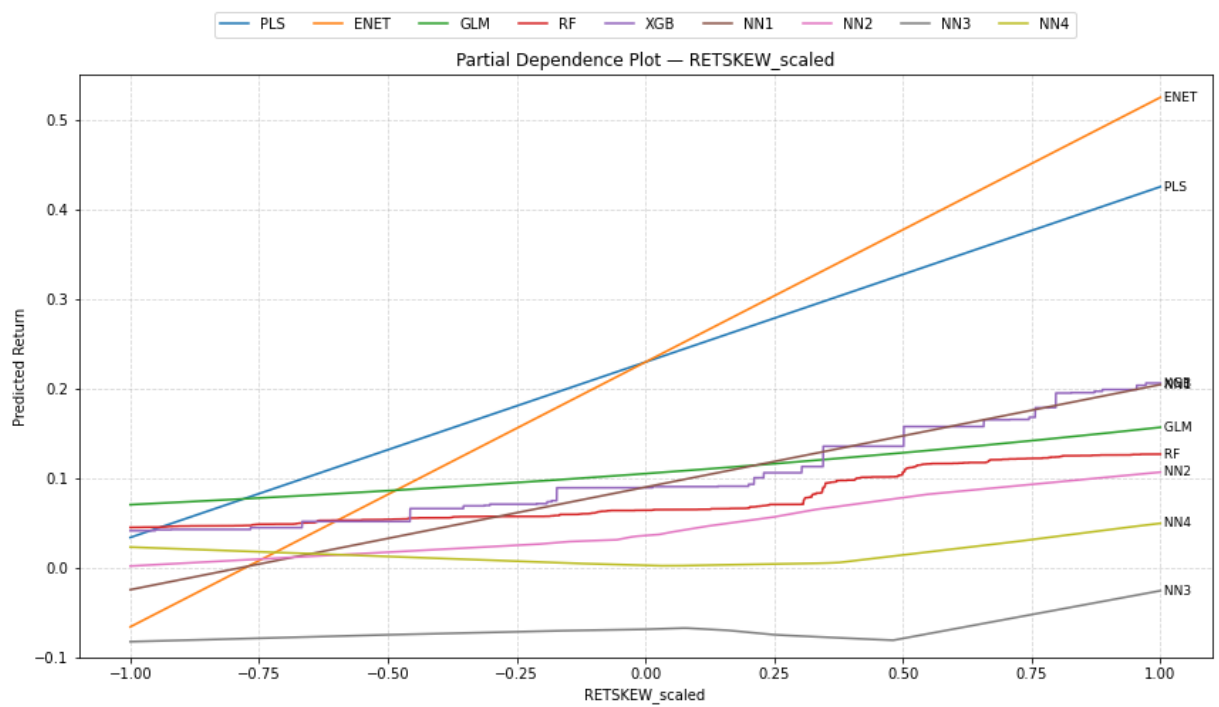


Figure 30

Partial Dependence Plot of RETSKEW

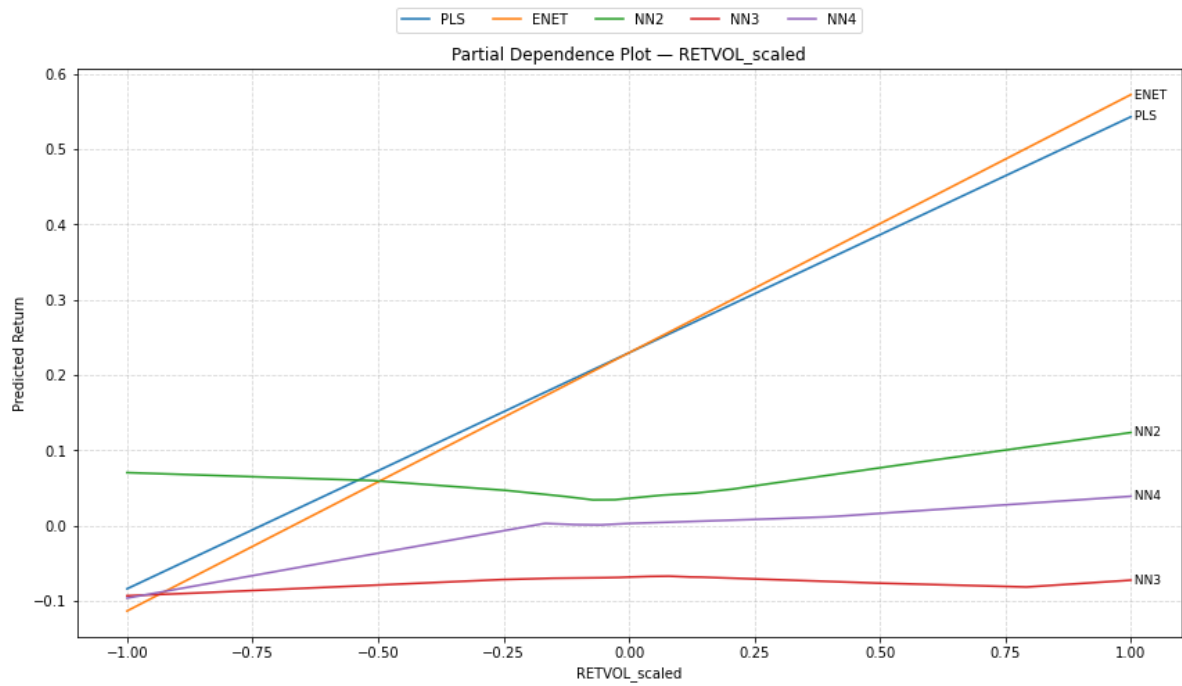


Figure 31

Partial Dependence Plot of RETVOL

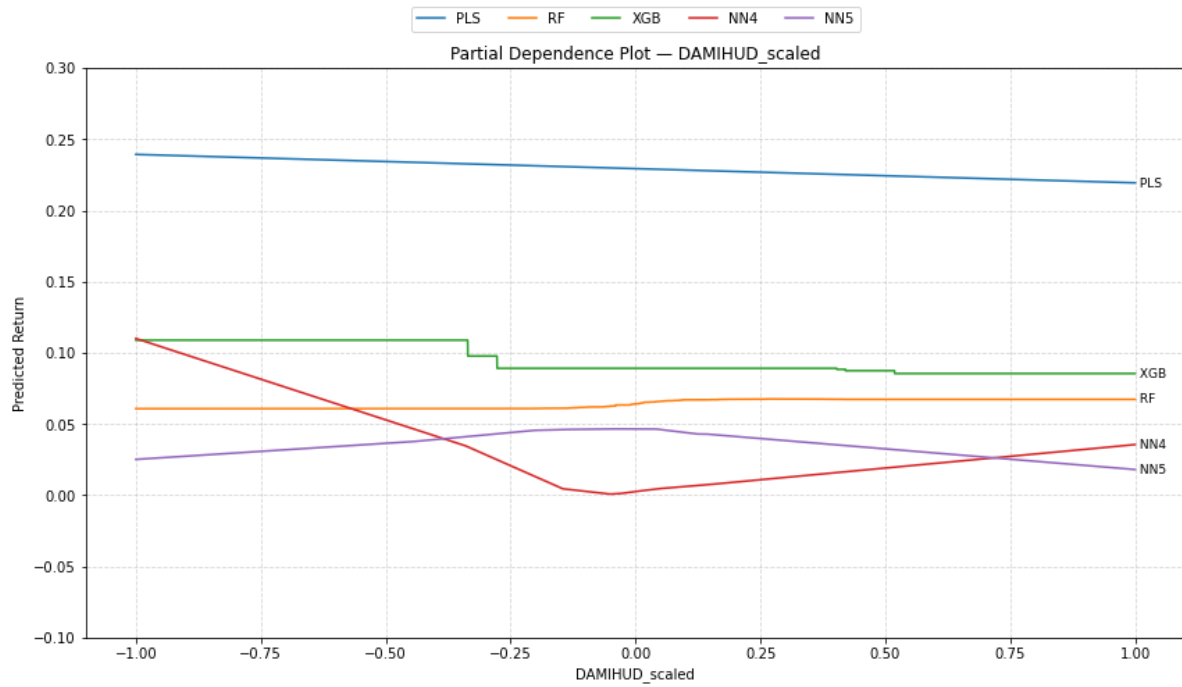


Figure 32

Partial Dependence Plot of DAMIHU

Partial Dependence Plot of Volatility factor

This figure presents the 5-1 value-weighted return sorting results and the corresponding Partial Dependence Plots (PDPs) for selected Volatility predictors. Significant t-values in the sorting tables indicate return predictability, while the slope direction of PDPs reveals how each predictor affects expected returns across models.

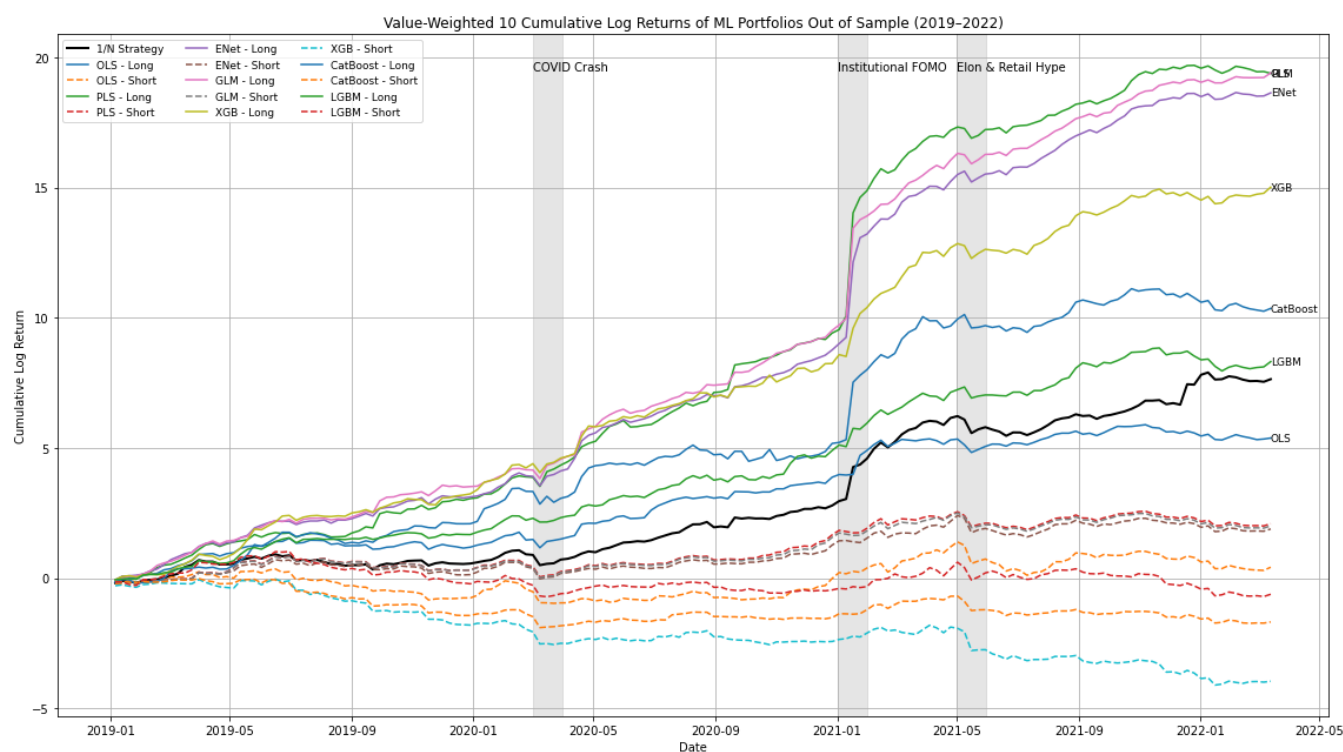


Figure 33

Out-of-Sample Cumulative Log Returns for ML Portfolios and 1/N Benchmark

(Value-Weighted, 10-group)

This figure visualizes the cumulative log returns of long and short portfolios constructed by various machine learning models under the Value-Weighted, 10-group configuration, including COCOS-BCX. The black line represents the baseline 1/N strategy, which equally weights all tradable assets each week. Long portfolios are shown as solid lines, short portfolios as dashed lines.

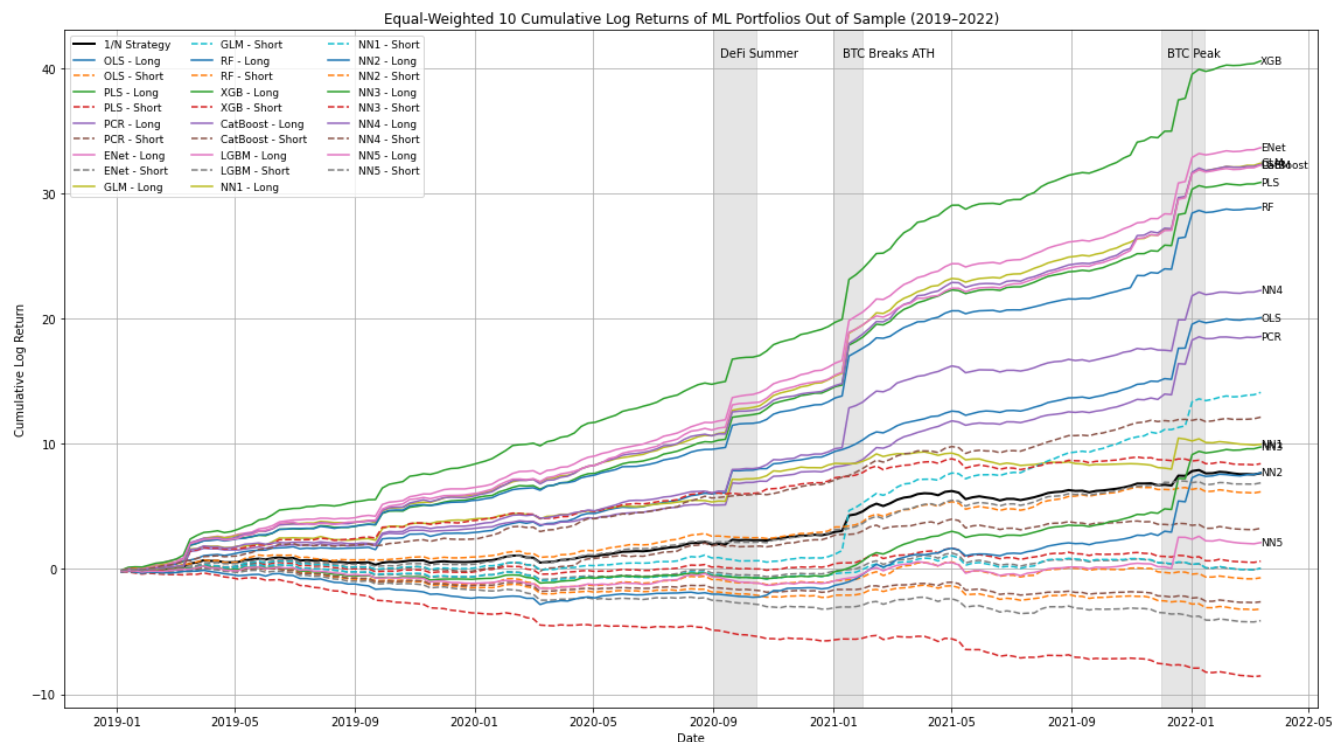


Figure 34

Out-of-Sample Cumulative Log Returns for ML Portfolios and 1/N Benchmark

(Equal-Weighted, 10-group)

This figure visualizes the cumulative log returns of long and short portfolios constructed by various machine learning models under the Equal-Weighted, 10-group configuration, including COCOS-BCX. The black line represents the baseline 1/N strategy, which equally weights all tradable assets each week. Long portfolios are shown as solid lines, short portfolios as dashed lines.

Table 1**Summary Statistics**

Panel A reports the number of coins, the mean and median market cap, and the mean and median daily dollar volume. Panel B reports the characteristics of Coin Market Index returns, Bitcoin returns, Ethereum returns, and XRP returns.

Panel A. Characteristics by Year					
Year	Num of Coins	Market Cap (mil)		Volume (thous)	
		Mean	Median	Mean	Median
2014	24	306.58	3.02	1431.44	9.98
2015	32	140.18	0.91	1235.41	3.13
2016	50	214.21	5.12	2492.02	26.59
2017	142	1015.05	62.33	48672.8	1214.26
2018	272	1084.17	30.38	64072.6	932.61
2019	406	518.045	9.48	134831	660.75
2020	682	760.29	10.43	182169	771.74
2021	977	2001.83	55.72	271487	3046.43
2022	995	2008.02	61.65	174681	2440.54
Full	996	769.3	33.12	100876	1806.06
Panel B. Return Characteristics					
	Mean	Median	SD	Skewness	Kurtosis
Coin Market Return	0.0019	0.0025	0.03944	-1.0706	10.2147
Bitcoin Return	0.0021	0.0017	0.039	-0.1441	7.135
Ethereum Return	0.005	0.0008	0.0628	0.1352	14.5872
XRP Return	0.0035	-0.0022	0.0742	6.3034	126.544

Table 2
Factor Definitions

Table 2: Factor Definitions

Category	Predictor	Definition	Reference
Size	MCAP	Log last-day market capitalization in the portfolio formation week	Banz (1981)
Size	PRC	Log last-day price in the portfolio formation week	Miller and Scholes (1982)
Size	MAXDPRC	Maximum price of the portfolio formation week	George and Hwang (2004)
Size	AGE	Number of weeks since listed on ComMarketCap.com	Barry and Brown (1984)
Momentum	r 1,0	One-week cumulative return (momentum)	Jegadeesh and Titman (1993)
Momentum	r 2,0	Two-week cumulative return (momentum)	Jegadeesh and Titman (1993)
Momentum	r 3,0	Three-week cumulative return (momentum)	Jegadeesh and Titman (1993)
Momentum	r 4,0	Four-week cumulative return (momentum)	Jegadeesh and Titman (1993)
Momentum	r 8,0	Eight-week cumulative return (momentum)	Jegadeesh and Titman (1993)
Momentum	r 16,0	Sixteen-week cumulative return (momentum)	Jegadeesh and Titman (1993)
Momentum	r 40,0	Forty-week cumulative return (momentum)	Jegadeesh and Titman (1993)
Momentum	r 100,0	One-hundred-week cumulative return (momentum)	De Bondt and Thaler (1985)
Volume	VOL	Log average daily volume during the portfolio formation week	De Bondt and Thaler (1985)
Volume	PRCVOL	Log average daily volume times price	Chordia, Subrahmanyam, and Anshuman (2001)
Volume	VOLSCALED	Log average daily volume times price scaled by market capitalization in the portfolio formation week	Chordia, Subrahmanyam, and Anshuman (2001)
Volatility	RETVOL	Standard deviation of daily returns during the formation week	Chordia, Subrahmanyam, and Anshuman (2001)
Volatility	RETSKEW	Skewness of weekly returns during the formation week	Ang et al. (2006)
Volatility	RETKURT	Kurtosis of weekly returns during the formation week	newly added
Volatility	MAXRET	Maximum daily return during the formation week	newly added
Volatility	STDPRCVOL	Log standard deviation of dollar volume in the portfolio formation week	Bali, Cakici, and Whitelaw (2011)
Volatility	DAMIHUD	Amihud illiquidity: average absolute daily return divided by dollar volume	Anshuman (2001)
			Amihud (2002)

Table 3
Weekly out-of-sample prediction performance

Linear Models					
Model	OLS	PCR	PLS	Enet	GLM
	-0.146877	-0.000898	-0.01254	-0.00783	0.0082449
Neural Networks					
Model	NN1	NN2	NN3	NN4	NN5
	0.0040438	0.0145	0.0001159	0.0133099	0.0019572
Tree-based Models					
Model	RF	XGBoost	CatBoost	LGBM	
	0.0031223	0.247374	0.0103532	0.0584337	

This table summarizes the out-of-sample R^2_{oos} values for 14 models across four categories: linear models (OLS, PLS, PCR, Enet, GLM), tree-based ensemble models (RF, XGBoost, LightGBM, CatBoost), and fully connected neural networks (NN1–NN5). The performance is assessed over weekly prediction horizons using an expanding-window rolling forecast from 2019 to 2022.

Table 4
Model Hyperparameters Summary

Model	Key Hyperparameters
OLS	No hyperparameters. Baseline linear regression using all predictors.
PLS	n_components = 1. Extracts one latent factor to capture covariance between predictors and target.
PCR	n_components = 1. Principal components extracted from standardized predictors before regression.
Elastic Net	alpha = 0.5, l1_ratio = 1e-4, max_iter = 10,000. Balances l_1/l_2 regularization with a small L1 penalty.
GLM	reg_lambda = 0.1, random_state = 1. Penalizes grouped spline features to control model complexity.
Random Forest	n_estimators = 900, max_depth = 6, min_samples_split = 1000, min_samples_leaf = 1000, max_features = 'sqrt', random_state = 42, n_jobs = -1. Shallow trees with large leaf size to reduce overfitting.
XGBoost	n_estimators = 5000, max_depth = 5, learning_rate = 0.015, subsample = 1, colsample_bytree = 1, tree_method = 'hist', gamma = 0.2, reg_lambda = 1, max_delta_step = 5, random_state = 42, n_jobs = -1. Fine-tuned for stability and conservative learning.

CatBoost	iterations = 110, depth = 7, learning_rate = 0.0017, l_2 _leaf_reg = 3, random_seed = 42, verbose = 0. Regularized deep boosting model.
LightGBM	n_estimators = 99, max_depth = 5, learning_rate = 0.00085, num_leaves = 10, min_data_in_leaf = 23, boosting = 'goss', random_state = 42, n_jobs = -1. Optimized with conservative leaf-wise growth.
NN1	20×Dense(32) + BatchNorm → Dense(1). Batch size = 10,000, epochs = 10, with early stopping. Optimizer: SGD (lr = 0.001, clipvalue = 0.5), loss = MSE.
NN2	20×Dense(32) + BatchNorm → Dense(16, ReLU, L1=1e-5) → Dense(1). Batch size = 10,000, epochs = 10, with early stopping. Same optimizer and loss as NN1.
NN3	20×Dense(32) + BatchNorm → Dense(16, ReLU, L1=1e-5) → Dense(8) → Dense(1). Batch size = 10,000, epochs = 10, with early stopping. Same optimizer and loss as NN1.
NN4	20×Dense(32) + BatchNorm → Dense(16, ReLU, L1=1e-5) → Dense(8→4, ReLU) → Dense(1). Batch size = 10,000, epochs = 10, with early stopping. Same optimizer and loss as NN1.
NN5	20×Dense(32) + BatchNorm → Dense(16→8→4→2, ReLU, L1=1e-5) → Dense(1). Batch size = 10,000, epochs = 10, with early stopping. Same optimizer and loss as NN1.

Table 5**Diebold-Mariano Test Statistics for Pairwise Model Comparisons**

	OLS	PCR	PLS	Enet	GLM	RF	XGB	CatBoost	LGBM	NN1	NN2	NN3	NN4	NN5
OLS	-	11.16**	14.63**	15.21**	14.07**	11.99**	4.27**	11.67**	12.21**	11.89**	11.18**	12.03**	12.39**	11.58**
PCR		-	(-3.02*)	(-2.32*)	0.36	(-0.58)	1.63	1.31	1.64	(-0.25)	6.03**	1.98*	1.84	3.22*
PLS			-	8.81**	7.38**	3.37*	2.00*	3.62**	3.51*	3.59**	3.19*	4.11**	4.60**	3.61**
Enet				-	5.42**	2.52*	1.94	2.96*	3.21*	2.72*	2.49*	3.21*	3.61**	2.83*
GLM					-	(-0.93)	1.66	0.72	1.74	(-0.57)	(-0.02)	0.30	0.57	0.16
RF						-	1.71	2.04*	2.06*	0.41	1.07	1.65	1.92	1.39
XGB							-	(-1.61)	(-1.60)	(-1.64)	(-1.61)	(-1.60)	(-1.60)	(-1.60)
CatBoost								-	1.52	(-1.31)	(-0.94)	(-0.68)	(-0.52)	(-0.77)
LGBM									-	(-1.69)	(-1.53)	(-1.51)	(-1.49)	(-1.50)
NN1										-	1.24	6.09**	5.89**	2.95*
NN2											-	0.79	0.96	1.03
NN3												-	1.39	(-0.52)
NN5													-	(-0.89)

This table reports Diebold-Mariano (DM) test statistics for pairwise comparisons of out-of-sample forecast performance between the 14 models evaluated. Each entry compares the model in the column against the model in the row, where a positive statistic indicates that the column model achieves lower forecast error than the row model. The test is based on the cross-sectional average of squared error differences across all cryptocurrencies at each weekly forecast horizon.

We follow a cross-sectional DM testing framework, which is more appropriate in our setting due to the high cross-sectional dimension of tokens. DM statistics are assumed to follow a standard normal distribution under the null hypothesis of equal predictive performance.

- A single asterisk (*) indicates statistical significance at the 5% level ($|DM| > 1.96$).

A double asterisk (**) indicates significance under a Bonferroni correction**, accounting for 91 pairwise comparisons. The adjusted critical value is approximately 2.64, controlling the family-wise error rate at 5%.

Table 6
Variable importance for significant predictors

Table 6. Variable importance for significant predictors													
	MCAP	PRC	MAXDPRC	MOM1	MOM2	MOM3	PRCVOL	VOL	VOLSCALED	RETVOL	RETSKEW	MAXRET	DAMIHUDD
OLS	0.0228	0.0355	0.0059	0.0072	0.0017	0.0029	0.0414	0.2529	0.0249	0.1119	0.0225	0.1362	0.0038
PCR	0.0642	0.0577	0.0165	0.0598	0.0607	0.0585	0.0807	0.0764	0.0799	0.0558	0.0059	0.0568	0.0352
PLS	0.0385	0.0904	0.0043	0.0006	0.0007	0.0006	0.1077	0.0899	0.0731	0.0811	0.1128	0.0814	0.0079
Enet	0.0235	0.08	0.0223	0.0169	0.0104	0.0071	0.0189	0.0517	0.0061	0.079	0.1455	0.0794	0.0193
GLM	0	0.0132	0	0	0	0	0	0	0	0.0133	0.2409	0.0141	0
RF	0.0137	0.0878	0.002	0.0076	0.0132	0.014	0.0886	0.0689	0.0199	0.074	0.22	0.0719	0.0101
XGB	0.0018	0.0193	0.0052	0.0012	0.0026	0.001	0.0061	0.0051	0.0079	0.0042	0.7437	0.0034	0.0005
CatBoost	0.0025	0.2471	0.1146	0.0048	0.024	0.0011	0.001	0.0343	0.0049	0.045	0.1829	0.0325	0.0048
LGBM	0.0003	0.119	0.0283	0.0065	0.007	0.0424	0.0309	0.0309	0	0.0685	0.2461	0.0595	0.0247
NN1	0.0089	0.08	0.0019	0.0001	0.0012	0.0054	0.0141	0.0029	0.0107	0.1109	0.073	0.3836	0.0002
NN2	0.0105	0.1333	0.0015	0.012	0.0204	0.0028	0.0481	0.1089	0.0025	0.2914	0.101	0.2346	0.0124
NN3	0.0085	0.0447	0.0001	0.0008	0.0023	0.0079	0.0187	0.0022	0.015	0.1912	0.0717	0.1259	0.0031
NN4	0.0099	0.0149	8.21e-0.6	0.0005	3.21e-0.5	0.0011	0.0151	0.0017	0.0035	0.0398	0.0116	0.0121	0.0005
NN5	0.0013	0.04	1.98e-0.5	0.0014	0.0037	0.0014	0.018	0.0143	0.0039	0.1612	0.046	0.018	0.0048

Table 7

Risk-Adjusted Returns of Value-Weighted 10-Group Portfolios by Model

(without COCOS-BCX)

In this table, we report the performance of prediction-sorted portfolios over the out-of-sample testing period. All cryptocurrencies are sorted into deciles based on their predicted one-week-ahead returns. Columns “Pred,” “Avg,” “SD,” and “SR” provide the predicted weekly returns for each decile, the average realized weekly returns, their standard deviations, and Sharpe ratios, respectively. All portfolios are Value-weighted-10. The sample excludes the coin COCOS-BCX.

Table 11. Performance of the machine learning portfolios (Without COCOS-BCX Value-Weighted 10)

	OLS				PCR				PLS			
	Pred	Avg	SD	SR	Pred	Avg	SD	SR	Pred	Avg	SD	SR
Low(L)	-3.8085	0.0101	0.1203	0.0841	-0.0042	0.0190	0.0979	0.1942	-1.9656	0.0174	0.0969	0.1800
2	-2.2248	0.0137	0.1192	0.1151	0.1311	0.0124	0.1131	0.1096	-1.0126	0.0127	0.1131	0.1124
3	-1.4005	0.0190	0.1092	0.1741	0.1836	0.0157	0.1125	0.1397	-0.6319	0.0264	0.1222	0.2163
4	-0.7349	0.0132	0.1110	0.1189	0.2212	0.0208	0.1128	0.1847	-0.3398	0.0154	0.1163	0.1321
5	-0.1215	0.0163	0.1198	0.1363	0.2552	0.0163	0.1135	0.1438	-0.0788	0.0282	0.1324	0.2127
6	0.4895	0.0141	0.1104	0.1280	0.2880	0.0223	0.1201	0.1854	0.1813	0.0275	0.1125	0.2444
7	1.1590	0.0231	0.1223	0.1888	0.3229	0.0204	0.1125	0.1817	0.4546	0.0363	0.1354	0.2684
8	1.9484	0.0290	0.1105	0.2624	0.3633	0.0111	0.1107	0.1004	0.7653	0.0338	0.1138	0.2969
9	2.9838	0.0380	0.1170	0.3249	0.4112	0.0207	0.1161	0.1785	1.1694	0.0771	0.1704	0.4526
High(H)	5.2157	0.0401	0.1343	0.2986	0.5430	0.0387	0.1920	0.2015	2.1526	0.1139	0.2047	0.5562
H-L	9.0242	0.0300	0.1267	0.2367	0.5472	0.0197	0.1774	0.1110	4.1182	0.0964	0.1922	0.5018
	ENET				GLM				RF			
	Pred	Avg	SD	SR	Pred	Avg	SD	SR	Pred	Avg	SD	SR
Low(L)	-1.4081	0.0165	0.0994	0.1663	0.0138	0.0169	0.0961	0.1754	-0.0129	-0.0022	0.1123	-0.0192
2	-0.7961	0.0149	0.1134	0.1314	0.0238	0.0192	0.1196	0.1601	-0.0013	0.0122	0.1263	0.0968
3	-0.4946	0.0154	0.1072	0.1436	0.0324	0.0222	0.1219	0.1818	0.0068	0.0044	0.1155	0.0385
4	-0.2450	0.0158	0.1171	0.1352	0.0420	0.0147	0.1138	0.1293	0.0152	0.0239	0.1238	0.1929
5	-0.0078	0.0143	0.1087	0.1313	0.0538	0.0307	0.1200	0.2557	0.0251	0.0209	0.1223	0.1706
6	0.2333	0.0217	0.1261	0.1721	0.0695	0.0222	0.1235	0.1800	0.0378	0.0275	0.1151	0.2388
7	0.5033	0.0365	0.1308	0.2787	0.0919	0.0289	0.1188	0.2436	0.0606	0.0322	0.1347	0.2388
8	0.8345	0.0353	0.1316	0.2679	0.1297	0.0477	0.1355	0.3517	0.1504	0.0395	0.1514	0.2611
9	1.3062	0.0563	0.1242	0.4537	0.2066	0.0657	0.1432	0.4588	0.4499	0.0309	0.1461	0.2119
High(H)	2.3501	0.1139	0.1892	0.6020	0.9164	0.1126	0.1683	0.6693	2.0668	0.0272	0.0685	0.3973
H-L	3.7582	0.0974	0.1726	0.5640	0.9025	0.0957	0.1499	0.6387	2.0797	0.0294	0.0988	0.2973
	XGB				CATBOOST				LGBM			
	Pred	Avg	SD	SR	Pred	Avg	SD	SR	Pred	Avg	SD	SR
Low(L)	-0.0828	-0.0170	0.1076	-0.1575	0.2845	-0.0064	0.0818	-0.0782	0.3123	0.0025	0.1093	0.0225
2	-0.0377	-0.0040	0.1112	-0.0358	0.2853	0.0127	0.1128	0.1129	0.3127	-0.0024	0.1106	-0.0221
3	-0.0199	-0.0041	0.1052	-0.0387	0.2858	0.0194	0.1186	0.1636	0.3127	0.0057	0.1224	0.0470
4	-0.0061	0.0166	0.1219	0.1360	0.2865	0.0109	0.1138	0.0956	0.3133	0.0089	0.1240	0.0719
5	0.0068	0.0059	0.1054	0.0560	0.2871	0.0128	0.1121	0.1142	0.3143	0.0102	0.1362	0.0749
6	0.0200	0.0226	0.1203	0.1879	0.2877	0.0370	0.1489	0.2487	0.3155	0.0183	0.1260	0.1456
7	0.0353	0.0390	0.1224	0.3185	0.2884	0.0406	0.1274	0.3182	0.3172	0.0208	0.1147	0.1815
8	0.0552	0.0484	0.1231	0.3932	0.2895	0.0416	0.1379	0.3015	0.3189	0.0348	0.1046	0.3329
9	0.0870	0.0599	0.1465	0.4087	0.2908	0.0498	0.1601	0.3110	0.3198	0.0523	0.1401	0.3735
High(H)	0.4317	0.1014	0.1840	0.5508	0.3473	0.0661	0.1861	0.3554	0.5533	0.0546	0.1289	0.4234
H-L	0.5145	0.1183	0.1629	0.7263	0.0628	0.0725	0.1616	0.4489	0.2410	0.0521	0.1105	0.4717
	NN1				NN2				NN3			
	Pred	Avg	SD	SR	Pred	Avg	SD	SR	Pred	Avg	SD	SR
Low(L)	-0.1229	0.0178	0.1382	0.1286	-0.1339	0.0224	0.1230	0.1819	-0.1190	0.0243	0.1211	0.2011
2	-0.0575	0.0077	0.1216	0.0633	-0.0431	0.0216	0.1200	0.1802	-0.0557	0.0323	0.1189	0.2718
3	-0.0273	0.0062	0.1124	0.0548	-0.0025	0.0119	0.1115	0.1068	-0.0250	0.0249	0.1204	0.2071
4	0.0000	0.0095	0.1201	0.0789	0.0298	0.0146	0.1285	0.1140	-0.0018	0.0289	0.1237	0.2339
5	0.0279	0.0163	0.1319	0.1234	0.0590	0.0210	0.1212	0.1732	0.0187	0.0207	0.1279	0.1619
6	0.0582	0.0217	0.1159	0.1876	0.0876	0.0098	0.1146	0.0854	0.0416	0.0141	0.1229	0.1150
7	0.0941	0.0141	0.1296	0.1085	0.1205	0.0147	0.1061	0.1385	0.0700	0.0096	0.1134	0.0845
8	0.1404	0.0190	0.1160	0.1634	0.1686	0.0090	0.1117	0.0807	0.1094	0.0019	0.1060	0.0184
9	0.2072	0.0188	0.1119	0.1682	0.2608	0.0225	0.1172	0.1923	0.1715	0.0187	0.1311	0.1453
High(H)	0.3971	0.0260	0.1053	0.2465	0.5551	0.0280	0.1141	0.2450	0.4829	0.0173	0.0971	0.1785
H-L	0.5200	0.0082	0.1238	0.0662	0.6890	0.0056	0.1205	0.0546	0.6018	-0.0070	0.1023	-0.0685
	NN4				NN5							
	Pred	Avg	SD	SR	Pred	Avg	SD	SR				
Low(L)	-0.0193	0.0059	0.1062	0.0559	-0.0013	0.0096	0.1162	0.0828				
2	-0.0130	0.0047	0.1091	0.0432	0.0007	0.0151	0.1087	0.1388				
3	-0.0061	0.0109	0.1115	0.0975	0.0038	0.0211	0.1129	0.1870				
4	0.0025	0.0055	0.1110	0.0500	0.0071	0.0176	0.1235	0.1423				
5	0.0140	0.0121	0.1144	0.1054	0.0107	0.0169	0.1100	0.1540				
6	0.0315	0.0133	0.1101	0.1207	0.0149	0.0167	0.1140	0.1466				
7	0.0659	0.0232	0.1124	0.2061	0.0201	0.0215	0.1158	0.1858				
8	0.1418	0.0223	0.1208	0.1849	0.0283	0.0163	0.1143	0.1428				
9	0.2918	0.0476	0.1423	0.3347	0.0466	0.0265	0.1085	0.2444				
High(H)	0.7135	0.0217	0.0985	0.2207	0.1686	0.0268	0.0961	0.2783				
H-L	0.7328	0.0158	0.0784	0.2016	0.1699	0.0785	0.1577	0.4975				

Table 8

Risk-Adjusted Returns of Value-Weighted 5-Group Portfolios by Model

(without COCOS-BCX)

Table 12. Performance of the machine learning portfolios (Without COCOS-BCX Value-Weighted 5)												
	OLS				PCR				PLS			
	Pred	Avg	SD	SR	Pred	Avg	SD	SR	Pred	Avg	SD	SR
Low(L)	-3.0202	0.0154	0.1111	0.1385	0.0631	0.0178	0.0967	0.1841	-1.4912	0.0172	0.0966	0.1777
2	-1.0670	0.0159	0.1038	0.1530	0.2024	0.0181	0.1076	0.1686	-0.4856	0.0219	0.1131	0.1940
3	0.1832	0.0155	0.1094	0.1415	0.2716	0.0181	0.1086	0.1665	0.0509	0.0292	0.1153	0.2532
4	1.5539	0.0256	0.1089	0.2345	0.3431	0.0166	0.1081	0.1536	0.6100	0.0355	0.1221	0.2912
High(H)	4.1051	0.0327	0.1062	0.3076	0.4774	0.0249	0.1201	0.2073	1.6633	0.0879	0.1500	0.5863
H-L	7.1253	0.0173	0.0781	0.2212	0.4142	0.0071	0.0879	0.0809	3.1546	0.0708	0.1213	0.5836
	ENET				GLM				RF			
	Pred	Avg	SD	SR	Pred	Avg	SD	SR	Pred	Avg	SD	SR
Low(L)	-1.1035	0.0173	0.1004	0.1726	0.0188	0.0174	0.0971	0.1796	-0.0071	0.0084	0.1168	0.0722
2	-0.3696	0.0160	0.1033	0.1551	0.0372	0.0206	0.1080	0.1906	0.0110	0.0130	0.1176	0.1108
3	0.1124	0.0182	0.1133	0.1610	0.0616	0.0285	0.1164	0.2450	0.0315	0.0250	0.1140	0.2189
4	0.6690	0.0386	0.1199	0.3219	0.1108	0.0404	0.1207	0.3346	0.1055	0.0331	0.1291	0.2566
High(H)	1.8306	0.0769	0.1283	0.5993	0.5632	0.0822	0.1329	0.6188	1.2622	0.0167	0.0736	0.2268
H-L	2.9341	0.0595	0.0978	0.6086	0.5444	0.0648	0.1102	0.5876	1.2693	0.0083	0.0802	0.1029
	XGB				CATBOOST				LGBM			
	Pred	Avg	SD	SR	Pred	Avg	SD	SR	Pred	Avg	SD	SR
Low(L)	-0.0603	-0.0061	0.1060	-0.0573	0.2849	0.0017	0.0889	0.0191	0.3121	0.0032	0.1009	0.0315
2	-0.0130	0.0050	0.1021	0.0486	0.2862	0.0187	0.1109	0.1688	0.3256	0.0287	0.1177	0.2438
3	0.0134	0.0109	0.1009	0.1076	0.2874	0.0214	0.1127	0.1895	0.3604	0.0366	0.1026	0.3567
4	0.0452	0.0402	0.1104	0.3643	0.2890	0.0437	0.1282	0.3407	0.4039	0.0682	0.1233	0.5529
High(H)	0.2602	0.0741	0.1449	0.5112	0.3192	0.0552	0.1574	0.3506	0.4182	0.0947	0.1727	0.5485
H-L	0.3205	0.0801	0.1218	0.6577	0.0343	0.0535	0.1215	0.4402	0.1061	0.0054	0.1697	0.0320
	NN1				NN2				NN3			
	Pred	Avg	SD	SR	Pred	Avg	SD	SR	Pred	Avg	SD	SR
Low(L)	-0.0903	0.0184	0.1198	0.1532	-0.0887	0.0228	0.1152	0.1979	-0.0875	0.0308	0.1092	0.2817
2	-0.0136	0.0078	0.1072	0.0730	0.0137	0.0158	0.1187	0.1331	-0.0134	0.0263	0.1160	0.2267
3	0.0430	0.0213	0.1176	0.1810	0.0733	0.0174	0.1052	0.1652	0.0301	0.0183	0.1215	0.1510
4	0.1173	0.0164	0.1189	0.1382	0.1446	0.0106	0.0986	0.1073	0.0897	0.0028	0.1004	0.0279
High(H)	0.3026	0.0166	0.0990	0.1679	0.4087	0.0177	0.1005	0.1765	0.3279	0.0178	0.0966	0.1843
H-L	0.3929	-0.0017	0.0898	-0.0194	0.4974	-0.0051	0.0753	-0.0673	0.4154	-0.0129	0.0772	-0.1678
	NN4				NN5							
	Pred	Avg	SD	SR	Pred	Avg	SD	SR				
Low(L)	-0.0161	0.0056	0.1039	0.0543	-0.0007	0.0105	0.1115	0.0945				
2	-0.0018	0.0067	0.1068	0.0631	0.0049	0.0163	0.1090	0.1497				
3	0.0227	0.0148	0.1082	0.1372	0.0121	0.0171	0.1045	0.1632				
4	0.1039	0.0219	0.1090	0.2009	0.0230	0.0199	0.1127	0.1763				
High(H)	0.5037	0.0184	0.0974	0.1893	0.0971	0.0240	0.0915	0.2630				
H-L	0.5198	0.0128	0.0683	0.1873	0.0978	0.0126	0.0811	0.1549				

In this table, we report the performance of prediction-sorted portfolios over the out-of-sample testing period. All cryptocurrencies are sorted into deciles based on their predicted one-week-ahead returns. Columns “Pred,” “Avg,” “SD,” and “SR” provide the predicted weekly returns for each decile, the average realized weekly returns, their standard deviations, and Sharpe ratios, respectively. All portfolios are Value-weighted-5. The sample excludes the coin COCOS-BCX.

Table 9

Risk-Adjusted Returns of Equal-Weighted 10-Group Portfolios by Model

(without COCOS-BCX)

In this table, we report the performance of prediction-sorted portfolios over the out-of-sample testing period. All cryptocurrencies are sorted into deciles based on their predicted one-week-ahead returns. Columns “Pred,” “Avg,” “SD,” and “SR” provide the predicted weekly returns for each decile, the average realized weekly returns, their standard deviations, and Sharpe ratios, respectively. All portfolios are Equal-weighted-10. The sample excludes the coin COCOS-BCX.

Table 13. Performance of the machine learning portfolios (Without COCOS-BCX Equal-Weighted 10)

	OLS				PCR				PLS			
	Pred	Avg	SD	SR	Pred	Avg	SD	SR	Pred	Avg	SD	SR
Low(L)	-3.8085	0.0029	0.1185	0.0241	-0.0042	0.0275	0.1233	0.2226	-1.9656	0.0097	0.1066	0.0906
2	-2.2248	0.0242	0.1602	0.1510	0.1311	0.0242	0.1182	0.2047	-1.0126	0.0040	0.1126	0.0354
3	-1.4005	0.0141	0.1121	0.1260	0.1836	0.0260	0.1216	0.2140	-0.6319	0.0129	0.1214	0.1061
4	-0.7349	0.0208	0.1240	0.1678	0.2212	0.0256	0.1227	0.2083	-0.3398	0.0175	0.1244	0.1406
5	-0.1215	0.0295	0.1232	0.2396	0.2552	0.0269	0.1221	0.2204	-0.0788	0.0131	0.1188	0.1099
6	0.4895	0.0314	0.1205	0.2603	0.2880	0.0305	0.1238	0.2465	0.1813	0.0221	0.1196	0.1847
7	1.1590	0.0368	0.1234	0.2982	0.3229	0.0253	0.1144	0.2207	0.4546	0.0255	0.1117	0.2283
8	1.9484	0.0538	0.2262	0.2379	0.3633	0.0293	0.1148	0.2548	0.7653	0.0359	0.1159	0.3097
9	2.9838	0.0546	0.1184	0.4611	0.4112	0.0530	0.1629	0.3253	1.1694	0.0634	0.1610	0.3938
High(H)	5.2157	0.2249	1.0209	0.2203	0.5430	0.2245	1.0416	0.2155	2.1526	0.2883	1.0406	0.2770
H-L	9.0242	0.2221	1.0183	0.2181	0.5472	0.1970	1.0362	0.1901	4.1182	0.2786	1.0336	0.2696
	ENET				GLM				RF			
	Pred	Avg	SD	SR	Pred	Avg	SD	SR	Pred	Avg	SD	SR
Low(L)	-1.4081	0.0066	0.1103	0.0594	0.0138	0.0071	0.1158	0.0615	-0.0129	-0.0139	0.0975	-0.1426
2	-0.7961	0.0062	0.1203	0.0518	0.0238	0.0048	0.1198	0.0404	-0.0013	-0.0039	0.1074	-0.0361
3	-0.4946	0.0113	0.1187	0.0955	0.0324	0.0121	0.1239	0.0973	0.0068	0.0067	0.1124	0.0593
4	-0.2450	0.0145	0.1225	0.1188	0.0420	0.0097	0.1194	0.0809	0.0152	0.0144	0.1163	0.1237
5	-0.0078	0.0101	0.1123	0.0900	0.0538	0.0154	0.1156	0.1329	0.0251	0.0230	0.1161	0.1978
6	0.2333	0.0174	0.1127	0.1548	0.0695	0.0205	0.1151	0.1778	0.0378	0.0393	0.1268	0.3099
7	0.5033	0.0211	0.1144	0.1843	0.0919	0.0305	0.1610	0.1893	0.0606	0.0428	0.1307	0.3272
8	0.8345	0.0406	0.1600	0.2537	0.1297	0.0372	0.1121	0.3314	0.1504	0.0542	0.1288	0.4208
9	1.3062	0.0567	0.1238	0.4577	0.2066	0.0563	0.1181	0.4766	0.4499	0.0550	0.1449	0.3797
High(H)	2.3501	0.3074	1.0400	0.2956	0.9164	0.2988	1.0414	0.2869	2.0668	0.2753	1.0425	0.2640
H-L	3.7582	0.3009	1.0323	0.2914	0.9025	0.2917	1.0338	0.2821	2.0797	0.2892	1.0400	0.2781
	XGB				CATBOOST				LGBM			
	Pred	Avg	SD	SR	Pred	Avg	SD	SR	Pred	Avg	SD	SR
Low(L)	-0.0828	-0.0442	0.0976	-0.4530	0.2845	-0.0112	0.0899	-0.1247	0.3123	-0.0187	0.1042	-0.1793
2	-0.0377	-0.0150	0.1000	-0.1495	0.2853	-0.0009	0.1046	-0.0090	0.3127	-0.0059	0.1060	-0.0561
3	-0.0199	-0.0051	0.1019	-0.0498	0.2858	0.0000	0.1045	-0.0005	0.3127	0.0014	0.1100	0.0123
4	-0.0061	0.0035	0.1062	0.0329	0.2865	0.0099	0.1142	0.0869	0.3133	0.0005	0.1138	0.0041
5	0.0068	0.0121	0.1117	0.1087	0.2871	0.0147	0.1127	0.1301	0.3143	0.0160	0.1150	0.1393
6	0.0200	0.0200	0.1156	0.1728	0.2877	0.0236	0.1161	0.2029	0.3155	0.0251	0.1293	0.1940
7	0.0353	0.0328	0.1180	0.2779	0.2884	0.0360	0.1184	0.3036	0.3172	0.0415	0.1246	0.3332
8	0.0552	0.0502	0.1238	0.4053	0.2895	0.0508	0.1292	0.3934	0.3189	0.0553	0.1227	0.4510
9	0.0870	0.0714	0.1287	0.5548	0.2908	0.0650	0.1379	0.4712	0.3198	0.0764	0.1348	0.5669
High(H)	0.4317	0.3666	1.0532	0.3481	0.3473	0.3048	1.0487	0.2906	0.5533	0.3011	1.0437	0.2885
H-L	0.5145	0.4108	1.0451	0.3931	0.0628	0.3160	1.0406	0.3036	0.2410	0.3197	1.0415	0.3070
	NN1				NN2				NN3			
	Pred	Avg	SD	SR	Pred	Avg	SD	SR	Pred	Avg	SD	SR
Low(L)	-0.1229	0.0324	0.2294	0.1411	-0.1339	0.0606	0.2353	0.2575	-0.1190	0.0678	0.2287	0.2966
2	-0.0575	0.0130	0.1154	0.1130	-0.0431	0.0328	0.1203	0.2728	-0.0557	0.0411	0.1120	0.3671
3	-0.0273	0.0187	0.1214	0.1538	-0.0025	0.0189	0.1137	0.1660	-0.0250	0.0369	0.1119	0.3301
4	0.0000	0.0122	0.1146	0.1065	0.0298	0.0178	0.1152	0.1548	-0.0018	0.0336	0.1130	0.2976
5	0.0279	0.0106	0.1090	0.0972	0.0590	0.0115	0.1114	0.1030	0.0187	0.0293	0.1120	0.2615
6	0.0582	0.0181	0.1110	0.1632	0.0876	0.0134	0.1129	0.1190	0.0416	0.0701	0.4041	0.1735
7	0.0941	0.0234	0.1099	0.2125	0.1205	0.0161	0.1143	0.1408	0.0700	0.0282	0.1180	0.2395
8	0.1404	0.0527	0.2076	0.2538	0.1686	0.0168	0.1147	0.1468	0.1094	0.0355	0.1316	0.2696
9	0.2072	0.0486	0.1265	0.3844	0.2608	0.0513	0.1986	0.2583	0.1715	0.0250	0.1269	0.1974
High(H)	0.3971	0.2620	1.0242	0.2558	0.5551	0.2524	1.0231	0.2467	0.4829	0.1251	0.9615	0.1302
H-L	0.5200	0.2297	1.0432	0.2201	0.6890	0.1918	1.0442	0.1838	0.6018	0.0573	0.9801	0.0585
	NN4				NN5							
	Pred	Avg	SD	SR	Pred	Avg	SD	SR				
Low(L)	-0.0193	0.0170	0.1157	0.1474	-0.0013	0.0163	0.1219	0.1340				
2	-0.0130	0.0130	0.1139	0.1142	0.0007	0.0333	0.1542	0.2160				
3	-0.0061	0.0133	0.1136	0.1175	0.0038	0.0264	0.1154	0.2289				
4	0.0025	0.0076	0.1119	0.0677	0.0071	0.0295	0.1206	0.2449				
5	0.0140	0.0252	0.2265	0.1110	0.0107	0.0318	0.1212	0.2623				
6	0.0315	0.0173	0.1600	0.1081	0.0149	0.0316	0.1211	0.2607				
7	0.0659	0.0219	0.1235	0.1775	0.0201	0.0307	0.1213	0.2533				
8	0.1418	0.0280	0.1152	0.2431	0.0283	0.0460	0.2282	0.2017				
9	0.2918	0.0622	0.1292	0.4813	0.0466	0.0763	0.3943	0.1936				
High(H)	0.7135	0.2865	1.0247	0.2796	0.1686	0.2654	1.1537	0.2301				
H-L	0.7328	0.2694	1.0238	0.2632	0.1699	0.2399	0.9885	0.2427				

Table 10

Risk-Adjusted Returns of Equal-Weighted 5-Group Portfolios by Model

(without COCOS-BCX)

Table 14. Performance of the machine learning portfolios (Without COCOS-BCX Equal-Weighted 5)												
	OLS				PCR				PLS			
	Pred	Avg	SD	SR	Pred	Avg	SD	SR	Pred	Avg	SD	SR
Low(L)	-3.0202	0.0135	0.1270	0.1060	0.0631	0.0258	0.1163	0.2221	-1.4912	0.0068	0.1080	0.0633
2	-1.0670	0.0175	0.1148	0.1521	0.2024	0.0258	0.1186	0.2172	-0.4856	0.0152	0.1207	0.1259
3	0.1832	0.0304	0.1188	0.2562	0.2716	0.0287	0.1203	0.2390	0.0509	0.0176	0.1173	0.1500
4	1.5539	0.0453	0.1516	0.2987	0.3431	0.0273	0.1124	0.2424	0.6100	0.0307	0.1106	0.2774
High(H)	4.1051	0.1402	0.5192	0.2700	0.4774	0.1392	0.5336	0.2608	1.6633	0.1764	0.5334	0.3308
H-L	7.1253	0.1267	0.5153	0.2460	0.4142	0.1133	0.5241	0.2162	3.1546	0.1696	0.5222	0.3248
	ENET				GLM				RF			
	Pred	Avg	SD	SR	Pred	Avg	SD	SR	Pred	Avg	SD	SR
Low(L)	-1.1035	0.0064	0.1130	0.0567	0.0188	0.0060	0.1156	0.0518	-0.0071	-0.0089	0.1015	-0.0880
2	-0.3696	0.0129	0.1188	0.1090	0.0372	0.0109	0.1201	0.0904	0.0110	0.0105	0.1132	0.0929
3	0.1124	0.0138	0.1105	0.1247	0.0616	0.0179	0.1123	0.1598	0.0315	0.0311	0.1194	0.2604
4	0.6690	0.0308	0.1250	0.2469	0.1108	0.0338	0.1230	0.2747	0.1055	0.0485	0.1269	0.3824
High(H)	1.8306	0.1827	0.5302	0.3446	0.5632	0.1782	0.5314	0.3353	1.2622	0.1657	0.5361	0.3091
H-L	2.9341	0.1763	0.5185	0.3401	0.5444	0.1722	0.5199	0.3312	1.2693	0.1746	0.5272	0.3313
	XGB				CATBOOST				LGBM			
	Pred	Avg	SD	SR	Pred	Avg	SD	SR	Pred	Avg	SD	SR
Low(L)	-0.0603	-0.0297	0.0972	-0.3052	0.2849	-0.0061	0.0959	-0.0636	0.3121	-0.0060	0.1035	-0.0581
2	-0.0130	-0.0008	0.1031	-0.0076	0.2862	0.0049	0.1082	0.0456	0.3256	0.0280	0.1087	0.2574
3	0.0134	0.0160	0.1121	0.1431	0.2874	0.0191	0.1129	0.1691	0.3604	0.1281	0.5581	0.2295
4	0.0452	0.0415	0.1185	0.3501	0.2890	0.0434	0.1209	0.3585	0.4039	0.3550	1.5841	0.2241
High(H)	0.2602	0.2198	0.5381	0.4084	0.3192	0.1855	0.5379	0.3448	0.4182	0.1990	0.1553	1.2810
H-L	0.3205	0.2494	0.5234	0.4766	0.0343	0.1916	0.5246	0.3652	0.1061	0.1032	0.1693	0.6094
	NN1				NN2				NN3			
	Pred	Avg	SD	SR	Pred	Avg	SD	SR	Pred	Avg	SD	SR
Low(L)	-0.0903	0.0228	0.1540	0.1484	-0.0887	0.0469	0.1588	0.2958	-0.0875	0.0546	0.1514	0.3608
2	-0.0136	0.0154	0.1129	0.1366	0.0137	0.0184	0.1115	0.1646	-0.0134	0.0353	0.1095	0.3221
3	0.0430	0.0144	0.1078	0.1333	0.0733	0.0125	0.1098	0.1136	0.0301	0.0498	0.2218	0.2243
4	0.1173	0.0381	0.1431	0.2661	0.1446	0.0164	0.1111	0.1479	0.0897	0.0319	0.1204	0.2649
High(H)	0.3026	0.1559	0.5230	0.2981	0.4087	0.1524	0.5275	0.2888	0.3279	0.0752	0.4952	0.1519
H-L	0.3929	0.1330	0.5266	0.2527	0.4974	0.1055	0.5310	0.1987	0.4154	0.0206	0.4966	0.0414
	NN4				NN5							
	Pred	Avg	SD	SR	Pred	Avg	SD	SR				
Low(L)	-0.0161	0.0150	0.1127	0.1334	-0.0007	0.0247	0.1343	0.1836				
2	-0.0018	0.0105	0.1107	0.0945	0.0049	0.0276	0.1137	0.2429				
3	0.0227	0.0212	0.1598	0.1326	0.0121	0.0341	0.1177	0.2898				
4	0.1039	0.0249	0.1161	0.2148	0.0230	0.0325	0.1175	0.2769				
High(H)	0.5037	0.1750	0.5227	0.3348	0.0971	0.1415	0.5398	0.2621				
H-L	0.5198	0.1600	0.5148	0.3107	0.0978	0.1097	0.5289	0.2075				

In this table, we report the performance of prediction-sorted portfolios over the out-of-sample testing period. All cryptocurrencies are sorted into deciles based on their predicted one-week-ahead returns. Columns “Pred,” “Avg,” “SD,” and “SR” provide the predicted weekly returns for each decile, the average realized weekly returns, their standard deviations, and Sharpe ratios, respectively. All portfolios are Equal-weighted-5. The sample excludes the coin COCOS-BCX.