Ca' Foscari University of Venice

amU
Faculté d'économie et de gestion
Aix Marseille Université

Master's Degree in

Economia e Finanza

Double Degree Programme in Financial Risks and Sustainable Finance in cooperation with Aix-Marseille Université

Final Thesis

ASSET PRICING AND MACHINE LEARNING

Supervisor
Billio Monica

Student
Fabris Mattia
889576

Academic Year
2025 / 2026

Abstract *3*

Chapter 1 — Introduction *4*

1.1 Research Question and Motivation 4

1.2 Empirical Design and Contribution 5

Chapter 2 — Literature Review *6*

2.1 Overview 6

2.2 Traditional Asset Pricing and the "Factor Zoo" 6

2.3 Penalized Linear Models 7

2.4 Tree-Based Methods 7

2.5 Deep Learning and Sequence Models 7

2.6 Thesis Positioning 8

Chapter 3 — Data and Variables *10*

3.1 Data Sources 10

3.2 Stock Universe and Sample Construction 10

3.3 Firm-Level Predictive Characteristics 11

3.4 Data Preprocessing 12

3.5 Sample Split 13

Chapter 4 — Methodology *15*

4.1 Empirical Strategy 15

4.2 The Out-of-Sample R^2 Metric 15

4.3 Linear Models 16

4.3.1 Ordinary Least Squares 16

4.3.2 LASSO 16

4.3.3 Ridge Regression 16

4.3.4 Elastic Net 17

4.4 Random Forest 17

4.4.1 Model Description 17

4.4.2 Variable Importance: Mean Decrease in Impurity 18

4.4.3 Expanding Window Estimation 19

4.5 Gradient Boosting Machines 19

4.5.1 Model Description 19

4.5.2 Variable Importance and Estimation 20

4.6 Neural Networks 20

4.6.1 Architecture 20

4.6.2 Training Procedure and Regularization 21

4.6.3 Expanding Window Scheme 22

4.6.4 Variable Importance: Sample Sensitivity Derivative 22

4.7 Long Short-Term Memory (LSTM) Networks **23**
4.7.1 Architecture and Gating Mechanisms 23
4.7.2 Training, Regularization, and Interpretability 23
4.8 Summary of Model Specifications **23**
Chapter 5 — Empirical Results ***25***
5.1 Overview **25**
5.2 Linear Models **25**
5.2.1 OLS 25
5.2.2 Penalized Linear Models 26
5.3 Random Forest **26**
5.3.1 Static Model 26
5.3.2 Expanding Window 26
5.3.3 Variable Importance 27
5.4 Gradient Boosting Machine (GBM) **27**
5.4.1 Out-of-Sample Performance 27
5.4.2 Variable Importance: Gain-Based 27
5.5 Neural Networks **28**
5.5.1 Out-of-Sample Performance 28
5.5.2 LSTM 28
5.5.3 Variable Importance: SSD 28
5.6 Summary of Results **29**
Chapter 6 — Discussion ***30***
6.1 Overview **30**
6.2 The Role of Regularization **30**
6.3 The Contribution of Nonlinearity **30**
6.4 Neural Networks: Performance and the Sample Constraint **31**
6.5 LSTM: Sequential Dynamics and Temporal Persistence **31**
6.6 Comparison with the Original Paper **32**
Chapter 7 — Conclusion ***33***
Bibliography **35**

Abstract

This thesis investigates whether machine learning methods can outperform traditional asset pricing approaches in predicting individual stock returns. The analysis is motivated by the growing empirical evidence that financial markets exhibit nonlinear relationships, high-dimensional predictor sets, and time-varying dynamics that standard linear models are not designed to capture. To address this question, a structured model comparison is conducted on a panel of 1,000 U.S. stocks over the period 1957–2021, following the framework of Gu, Kelly, and Xiu (2020).

Ten models of increasing complexity are estimated and compared across a test period spanning 1987 to 2021: ordinary least squares, LASSO, Ridge regression, Elastic Net, a Random Forest, a Gradient Boosting Machine, three feed-forward neural network architectures of increasing depth (NN1, NN2, NN3), and a Long Short-Term Memory (LSTM) network. All models are trained on the 1957–1974 period, tuned on a validation set covering 1975–1986, and evaluated on the test set without any look-ahead. Performance is measured using the out-of-sample R^2_{OOS} defined in Gu, Kelly, and Xiu (2020), which benchmarks model forecasts against the naive prediction of zero excess return for every stock.

The analysis was first validated on a preliminary sample of 100 stocks, which confirmed the directional coherence of the results at reduced scale before the full analysis was conducted on 1,000 stocks. The qualitative ordering of model performance is broadly consistent with the original paper, while the quantitative differences are explained by the greater homogeneity of the stock universe used in this thesis relative to the full CRSP database. Overall, the findings support the view that machine learning adds value in asset pricing primarily through regularization and nonlinear interactions, and that the choice between model classes should be informed by the available data rather than by complexity alone.

Chapter 1 — Introduction

Asset pricing is a central field in financial economics because it seeks to explain the forces that determine expected returns across assets. The literature has developed a wide range of models to describe the cross-section and time variation of returns, from classical linear factor specifications to more flexible statistical approaches. At the same time, the rapid growth of financial data and computational power has made machine learning an increasingly attractive tool for asset pricing applications, opening a debate on whether data-driven flexibility can substitute or complement the economic discipline of traditional models.

Traditional asset pricing models have the advantage of being economically interpretable and relatively easy to estimate. The problem is that their simplifying assumptions, like linearity, can limit their ability to capture complex non linear relationships, interactions among variables and time varying pattern in financial data. On the other hand, machine learning methods are more capable to handle high dimensional predictor sets creating more predictive structures at the cost of reducing the transparency. This trade off between flexibility and interpretability is one of the main motivations that lead to this thesis.

1.1 Research Question and Motivation

The purpose of this thesis is to examine whether machine learning methods can improve the prediction of individual stock returns relative to more traditional approaches. In particular, the analysis compares conventional linear regressions with a set of machine learning algorithms that differ in the degree of nonlinearity and flexibility they allow. The goal is not simply to identify which model produces the highest out-of-sample R^2, but to understand why more flexible specifications may add value and under what conditions the additional complexity is economically justified.

This question is particularly relevant in the present data environment. A large number of firm-level characteristics have been documented to predict future returns in the cross-section, but the relationship between these predictors and expected returns is likely nonlinear and may depend on complex interactions that standard linear regressions cannot represent. Financial data are noisy, so predicting stock returns is difficult and overfitting is a real risk. The machine learning methods used in this thesis help address this problem because they are more flexible than linear models and include tools such as regularization in linear models, bootstrap-based tree methods, and early stopping in neural networks to control estimation error.

1.2 Empirical Design and Contribution

The empirical analysis follows the framework established by Gu, Kelly, and Xiu (2020), one of the most influential recent papers on machine learning in asset pricing. The analysis is applied to a panel of 1,000 U.S. stocks drawn from the CRSP database, covering the period from January 1957 to December 2021. Every stock is characterized each month by 95 firm-level predictive characteristics spanning momentum, value, investment, profitability, trading frictions, and intangibles. Excess returns are constructed by subtracting the monthly risk-free rate from Welch and Goyal (2008).

All models are trained on data ending in 1974, tuned on a validation period covering 1975–1986, and evaluated on a test period from 1987 to 2021. This strictly out-of-sample design ensures that no future information contaminated the model estimation at any stage. The nonlinear models, the Random Forest, the Gradient Boosting Machine, and the neural networks (including an LSTM architecture), are additionally estimated using a recursive expanding window scheme in which the model is re-estimated and re-tuned annually, allowing the parameters to adapt to changes in the data generating process over the 35-year evaluation horizon.

The thesis pursues two objectives. First, it provides an empirical replication and extension of the Gu, Kelly, and Xiu (2020) framework applied to a more recent but more limited stock universe, offering direct evidence on whether the patterns documented in the original paper persist in a different data environment. Second, it interprets the results by connecting the statistical performance of each model to its economic implications. The analysis was initially validated on a smaller preliminary dataset of 100 stocks to verify the correctness of the estimation pipeline before scaling to the full 1,000-stock universe, ensuring the reliability of the final results.

Chapter 2 — Literature Review

2.1 Overview

The empirical asset pricing literature has experienced a paradigm shift over the last decade, transitioning from traditional low-dimensional econometric models to advanced, high-dimensional machine learning frameworks. As highlighted by Bagnara (2024), the rapidly expanding literature on machine learning in asset pricing can be broadly categorized into distinct methodological families: penalized regressions, tree-based methods, and deep learning architectures. Following this, the chapter reviews the key contributions of these families.

2.2 Traditional Asset Pricing and the "Factor Zoo"

In the past, the central goal of empirical asset pricing has been to understand the cross-sectional determinants of expected returns. The early literature relied on the Capital Asset Pricing Model (CAPM), but its well-documented empirical failures led researchers to develop multi-factor models, most notably those of Fama and French (1993) and Carhart (1997).

The traditional approach to estimating these models relied on portfolio sorts, in which stocks are ranked by a given characteristic and grouped into long-short portfolios to isolate the associated risk premium. While intuitive, this method rests on restrictive assumptions and becomes impractical when many characteristics are considered simultaneously, as sorting stocks across multiple dimensions quickly leads to empty or poorly diversified portfolios.

As data availability expanded, researchers documented a growing number of firm characteristics with apparent predictive power over returns. Cochrane (2011) famously described the resulting proliferation as a "factor zoo", and subsequent work by Harvey et al. (2016) and Hou et al. (2015) raised serious concerns about which of these variables carry genuinely independent information. In this high-dimensional environment, standard linear models face well-known limitations: multicollinearity, omitted variable bias, and the inability to capture nonlinear interactions among predictors. These structural constraints directly motivated the adoption of machine learning methods in empirical asset pricing.

2.3 Penalized Linear Models

To navigate the "factor zoo," the literature has turned to penalized linear models. As noted by Bagnara (2024), these methods serve as a critical bridge between traditional econometrics and modern machine learning, providing a robust framework to address the high-dimensional nature of cross-sectional asset returns while preserving model interpretability.

In an environment where the number of potential predictors is large, traditional OLS estimators suffer from severe instability and overfitting. Penalized methods mitigate these issues by introducing a regularization term to the objective function. This constraint effectively acts as an economically motivated prior, assuming that the predictive signal is sparse or that individual factor weights should be balanced to prevent idiosyncratic noise from dominating the model.

By effectively managing multicollinearity and reducing estimator variance, these models provide a reliable baseline for return predictability. Their role in the literature can be summarized in two points: they demonstrate that high-dimensional equity data can yield genuine out-of-sample predictability when regularized, and they serve as an essential benchmark to measure the marginal contribution of more complex non-linear models.

2.4 Tree-Based Methods

Penalized regressions assume a linear relationship between firm characteristics and future returns. Financial markets, however, are characterized by complex, interactive, and non-linear dynamics. Tree-based methods, specifically Random Forests and Gradient Boosting Machines (GBM), relax the linearity assumption. By constructing ensembles of decision trees, these models natively capture intricate interactions among predictors without the need for manual specification. Gu et al. (2020) highlight that Random Forests consistently rank among the best-performing algorithms in asset pricing by effectively averaging out the noise of individual deep trees. Similarly, Gradient Boosting builds trees sequentially, with each new tree minimizing the residual error of the previous ensemble, offering a robust non-parametric alternative for stock return prediction.

2.5 Deep Learning and Sequence Models

The application of deep neural networks in asset pricing represents an ongoing evolution in the literature. While theoretical contributions suggest that deep architectures are uniquely capable of

approximating complex non-linear relationships and capturing broad macroeconomic interactions, their empirical implementation is subject to intense academic debate.

Dense feed-forward neural networks have been widely explored to predict stock returns. However, the literature highlights a significant performance trade-off between architectural depth and predictive stability. Avramov et al. (2023) observe that, in equity markets characterized by a low signal-to-noise ratio, dense networks may exhibit sensitivity to the training sample, necessitating rigorous regularization and economic constraints to ensure out-of-sample robustness. Instead of indicating a fundamental limitation of neural approaches, these findings emphasize that the success of deep learning in asset pricing is highly contingent upon the alignment between model complexity and the information structure of the underlying data.

To address these challenges, recent research has explored sequence-based models, such as Long Short-Term Memory (LSTM) networks. Differently to static feed-forward architectures, LSTMs incorporate gating mechanisms that regulate information flow, allowing the model to distinguish between persistent signals and transient market noise. Fischer and Krauss (2018) demonstrate that these temporal filtering properties are particularly effective for financial time series. By learning the state-dependent nature of return predictability, LSTMs provide an alternative framework that can offer distinct predictive advantages in complex, high-dimensional datasets.

2.6 Thesis Positioning

While this thesis replicates the foundational methodological pillars of Gu, Kelly, and Xiu (2020), specifically regarding the treatment of firm-level characteristics, it extends the analysis by imposing a uniform expanding-window estimation scheme across all models. This ensures a consistent benchmark for evaluating predictive stability across diverse functional forms. The positioning of this work is defined by three methodological pillars:

- Estimation Scheme: All models are estimated using a consistent expanding-window scheme. This ensures that the performance of models with different functional forms, from linear to non-linear, is assessed under the same information set, preventing discrepancies arising from different training periods.
- Model Complexity Comparison: The analysis contrasts standard feed-forward neural networks with LSTM architectures. This allows for a direct evaluation of whether a model's

ability to process sequential data improves predictive accuracy compared to static architectures in an environment with a low signal-to-noise ratio.

- Interpretability: The study employs post-estimation methods, including parametric coefficient analysis, Mean Decrease Impurity (MDI) for trees, Sum of Squared Derivatives (SSD) for dense networks, and gradient-based importance for LSTMs, to identify the firm-level characteristics that contribute to the predictive performance of each model.

  By applying these methods to the same dataset and estimation window, this work quantifies the empirical performance of different algorithms, identifying which methodologies effectively capture predictive signals and which are prone to overfitting in this specific asset pricing context.

Chapter 3 — Data and Variables

3.1 Data Sources

The empirical analysis relies on two primary data sources. Monthly individual stock returns are sourced from the Center for Research in Security Prices (CRSP), covering all firms listed on the three major U.S. exchanges: NYSE, AMEX, and NASDAQ. The dataset spans from January 1957 to December 2021, providing a long time series that captures multiple business cycles, periods of financial stress, and structural changes in the composition of the equity market. For each stock-month observation, the raw return series is combined with a set of firm-level predictive characteristics, which are obtained from the same CRSP-based data infrastructure used in Gu, Kelly, and Xiu (2020) and described in detail in Section 3.2.

The risk-free rate used to compute excess returns is taken from the macroeconomic dataset of Welch and Goyal (2008), which provides a monthly series of the Treasury bill rate (rfree) already expressed as a decimal monthly return. For each stock-month observation, the excess return is defined as the difference between the realized raw return and the risk-free rate:

$$r_{i,t}^{e} = r_{i,t} - r_{f,t}$$

This definition is standard in the asset pricing literature and ensures that the models are trained to predict the compensation above the risk-free rate rather than the total return level.

3.2 Stock Universe and Sample Construction

The complete dataset analyzed comprises 1,000 individual stocks, each associated with a CRSP permanent identifier (PERMNO) that prevents ambiguous tracking across time. For each stock, separate return and characteristics files are available; these two sources are merged at the stock-month level using the (PERMNO, date) pair as the composite join key.

The resulting panel contains approximately 258,000 stock-month observations distributed over 780 monthly periods from January 1957 to December 2021. The number of observations per month is not constant, reflecting the entry and exit of firms from the sample over time due to delistings, mergers, and new listings.

Preliminary Sample: 100 Stocks

Prior to running the full analysis on 1,000 stocks, a reduced sample of 100 randomly selected stocks was used to validate the data pipeline, test the preprocessing routines, and verify the correctness of all model implementations.

The 100-stock sample was insufficient for the machine learning models to learn stable and generalizable patterns from the data. In particular, regularized models and neural networks require a sufficiently large cross-section at each point in time to estimate meaningful relationships between firm characteristics and returns. The results obtained on the reduced sample confirmed that the predictive signal was weak, and that out-of-sample performance was substantially below what would be expected from a larger dataset, motivating the transition to the 1,000-stock universe for the main analysis.

3.3 Firm-Level Predictive Characteristics

Each stock-month observation is described by a set of 95 firm-level characteristics that serve as predictors in all models. These characteristics are drawn from the empirical asset pricing literature and cover a broad range of economic dimensions known to be associated with expected return variation in the cross-section of stocks. Following the classification used in Gu, Kelly, and Xiu (2020), the predictors can be grouped into six main categories:

- Momentum: variables capturing past return patterns at different horizons, including momentum mom1m, mom6m, mom12m, mom36m, mom60m, and cumulative abnormal returns around earnings announcements (abr).
- Value-versus-growth: variables related to valuation ratios, including book-to-market equity (bm), earnings-to-price (ep), cash-flow-to-price (cfp), sales-to-price (sp), and dividend yield (dy).
- Investment: variables measuring firm investment and financing activity, including asset growth (agr), operating accruals (acc), and net stock issues (ni).
- Profitability: variables capturing earnings quality and operating efficiency, including return on equity (roe), return on assets (Roa1), gross profitability (gma), and operating profitability (op).

- Trading frictions: variables related to liquidity and market microstructure, including market equity (me), illiquidity (ill), bid-ask spread (baspread), dollar trading volume (dolvol), share turnover (turn), and maximum daily return (maxret).
- Intangibles: variables related to R&D, advertising, and other off-balance-sheet items, including R&D-to-market (rdm), advertising expense-to-market (adm), and employee growth rate (hire).

The final predictor set also includes sic2, the two-digit Standard Industrial Classification code, which captures broad industry membership.

The update frequency of these characteristics varies: some are available at monthly frequency, others at quarterly or annual frequency. To avoid any form of look-ahead bias in the construction of the predictor matrix, a publication lag is applied following the convention in Gu, Kelly, and Xiu (2020): monthly characteristics are lagged by one-month, quarterly releases by four months, and annual accounting variables by six months. This ensures that at each prediction date, only information that would have been available to an investor in real time is used as input to the models.

3.4 Data Preprocessing

The preprocessing of the panel follows four sequential steps designed to ensure that the data are clean, comparably scaled, and free of distortions due to outliers or structural missing patterns.

Missing value imputation. A substantial number of observations contains missing values for some characteristics, due to the limited availability of accounting data for some firms in the early part of the sample. Missing values are imputed using the cross-sectional median of each characteristic within each month. For months where no valid observation is available, a global median computed over the full time series is used as a fallback. After imputation, no residual missing values remain in the characteristic matrix.

Rank normalization. Following Gu, Kelly, and Xiu (2020), each characteristic is rank normalized into the interval $[-1, +1]$ within each cross-section at each point in time. The normalized value for stock i in month t is:

$$\tilde{z}_{i,t} = 2 \cdot \frac{\text{rank}(z_{i,t}) - 1}{N_t - 1} - 1$$

where N_t denotes the number of stocks with non-missing observations in month t. This transformation removes the influence of extreme values in the raw characteristic distributions,

ensures comparability across variables with different units and scales, and prevents any single predictor from dominating the estimation due to its numerical range.

Winsorization. The excess return series is winsorized at the 1st and 99th percentiles to mitigate the influence of extreme observations. After this adjustment, excess returns range from approximately −25.6% to +31.8% on a monthly basis. It should be noted that in the present implementation, the winsorization thresholds are computed on the full panel rather than exclusively on the training sample, which may introduce a mild form of look-ahead bias. The practical impact of this choice is low given the large number of observations, but it is acknowledged as a methodological limitation.

Removal of observations with missing returns. After all preprocessing steps, observations for which the excess return remains missing, primarily due to the absence of a valid raw return in the CRSP files, are dropped from the sample. This results in a final clean panel of approximately 257,991 usable stock-month observations.

3.5 Sample Split

Following the design used in Gu, Kelly, and Xiu (2020), the clean panel is initially divided into three baseline sub-periods that serve distinct roles in the initial setup and evaluation of the models:

Period	Years	Observations	Role
Training	1957–1974	~24,217	Parameter estimation
Validation	1975–1986	~28,263	Hyperparameter selection
Test	1987–2021	~205,718	Out-of-sample evaluation

While this static split establishes the baseline temporal structure, the final out-of-sample evaluation employs a recursive expanding-window scheme to ensure a robust evaluation of predictive performance. For each year in the test period, the models are re-estimated using all available historical data up to that point. This approach ensures that the information set available to the models at each prediction date is strictly limited to the past, preventing any look-ahead bias and allowing the parameters to adapt to evolving market regimes.

This recursive expanding-window design is preferred over a pure rolling-window approach to ensure that machine learning algorithms, in particular deep learning architecture that need a huge number of data, have access to a sufficiently large training sample to maintain parameter stability.

Concurrently, to ensure that model tuning remains relevant to the prevailing market environment, the hyperparameter optimization (validation) is restricted to a rolling 12-year window immediately preceding each test year, effectively balancing the need for long-term historical learning with the necessity of recent market adaptability.

Chapter 4 — Methodology

4.1 Empirical Strategy

The empirical analysis follows the framework proposed by Gu, Kelly, and Xiu (2020) and is organized as a model comparison exercise. The main objective is to assess whether increasingly flexible methods improve return prediction relative to simple linear benchmarks, using a similar set of firm characteristics and the same out-of-sample evaluation rule. The prediction problem is defined at the individual stock level: at each month, the goal is to forecast the one-month-ahead excess return of each stock using a set of firm-level characteristics observed at time t. In general terms, the prediction equation can be written as

$$r_{i,t+1}^{e} = f(z_{i,t}) + \varepsilon_{i,t+1}$$

where $r_{i,t+1}^{e}$ denotes the excess return of stock i in month $t+1$, $z_{i,t}$ is the vector of characteristics observed at time t, $f(\cdot)$ is a prediction function estimated from the data, and $\varepsilon_{i,t+1}$ is a zero-mean error term. The models differ in the functional form assumed for $f(\cdot)$: linear models impose a linear relationship, while tree-based methods and neural networks allow $f(\cdot)$ to be a flexible nonlinear function of the predictors.

4.2 The Out-of-Sample R² Metric

The primary performance metric used to compare all models is the out-of-sample R^2, denoted R_{OOS}^2, following the definition in Equation 19 of Gu, Kelly, and Xiu (2020). For a set of realized excess returns $\{r_{i,t}\}$ and model forecasts $\{\hat{r}_{i,t}\}$, this measure is defined as:

$$R_{OOS}^2 = 1 - \frac{\sum_{i,t}\ (r_{i,t} - \hat{r}_{i,t})^2}{\sum_{i,t}\ r_{i,t}^2}$$

The denominator uses the sum of squared realized excess returns rather than the sum of squared deviations from the mean. This reflects the natural benchmark for return prediction: a model with no predictive ability would simply forecast a return of zero for every stock at every period. A positive R_{OOS}^2 therefore indicates that the model improves upon this naive benchmark, while a negative value indicates that the model's forecasts are worse than simply predicting zero for all stocks.

4.3 Linear Models

4.3.1 Ordinary Least Squares

The first model estimated is ordinary least squares (OLS), which assumes a linear and additive relationship between the characteristics and the expected excess return:

$$\hat{r}^{e}_{i,t+1} = \hat{\beta}_0 + \sum_{j=1}^{p} \hat{\beta}_j z_{j,i,t}$$

where $p = 95$ is the number of firm-level characteristics. The OLS estimator minimizes the sum of squared prediction errors on the training sample and is estimated without regularization.

In this context, OLS serves primarily as a baseline benchmark. To ensure a rigorous and uniform evaluation, all linear models are estimated under two distinct schemes: a baseline static configuration (trained once on the initial 1957–1974 sample) and a recursive expanding-window configuration identical to the one applied to the non-linear algorithms. This dual approach isolates the marginal benefit of non-linear functional forms from the performance gains attributable to continuous temporal adaptation.

4.3.2 LASSO

Using the formula of Gu, Kelly, and Xiu (2020) on the elastic net penalty (Eq. 8 of the paper). Setting ρ = 0 recovers the LASSO (L1 penalty only):

$$\varphi(\theta;\, \lambda, \rho) \;=\; \lambda(1-\rho) \sum_{j=1}^{P} |\theta_j|$$

The term $\lambda \sum_j |\beta_j|$ is the L1 penalty, which shrinks the estimated coefficients toward zero and, importantly, sets some of them exactly to zero. This means that LASSO simultaneously estimates the model and selects the most relevant predictors, producing a sparse specification where only a subset of the 95 characteristics receives a non-zero coefficient

4.3.3 Ridge Regression

Ridge regression applies a different form of regularization. Ridge regression corresponds to setting ρ = 1 in the elastic net penalty of Equation (8) of Gu, Kelly, and Xiu (2020), yielding a pure L2 penalty:

$$\varphi(\theta;\ \lambda, \rho)\ =\ \frac{1}{2}\lambda\rho \sum_{j=1}^{P} \theta_j^2$$

Unlike LASSO, ridge regression shrinks all coefficients continuously toward zero but never forces any of them to be exactly zero. This means that all 95 characteristics always contribute to the prediction, albeit with reduced weights when λ is large. Ridge regression is particularly well-suited to settings where many predictors carry weak but non-negligible predictive information, as it distributes the shrinkage smoothly across all variables rather than concentrating it on a selected subset.

The tuning procedure is identical to that described for LASSO: λ is selected from the same logarithmic grid using the validation set R^2_{OOS} as the selection criterion.

4.3.4 Elastic Net

The elastic net penalty of Equation (8) of Gu, Kelly, and Xiu (2020) combines L1 and L2 regularization through the mixing parameter ρ ∈ [0,1]:

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

The mixing parameter p controls the relative weight of the two penalties. In the present implementation, p is fixed at 0.5, which assigns equal importance to the sparsity-inducing L1 component and the smoothing L2 component. This choice follows the convention adopted in Gu, Kelly, and Xiu (2020) and provides a balanced compromise between the variable-selection behavior of LASSO and the stability of ridge regression. The penalty parameter λ is again selected by validation-set performance over the same grid.

4.4 Random Forest

4.4.1 Model Description

Random Forest is an ensemble method that builds many regression trees and combines their predictions by averaging. The idea is that a single decision tree, although able to capture nonlinear patterns and interactions between variables, tends to be highly sensitive to the specific observations used for training and can therefore produce unstable and poorly generalizing forecasts. By averaging many trees estimated on different bootstrap samples of the data, Random Forest reduces this instability while preserving the ability to model complex relationships.

For return prediction, a regression tree partitions the set of stocks into groups based on a sequence of binary splits on the characteristic values. In every internal node, the algorithm selects the predictor and split point that minimizes the prediction error within the resulting subgroups. The stocks that end up in the same terminal node are then assigned the same predicted return, equal to the average realized excess return of the training observations in that node. This mechanism allows the model to capture the idea that the relationship between a firm's characteristics and its expected return is not linear: for example, momentum may predict returns differently depending on whether the firm is also large-cap or small-cap.

The Random Forest implementation follows the specification of Gu, Kelly, and Xiu (2020). Two hyperparameters are tuned using the validation set:

- **Maximum tree depth** (L): the maximum number of sequential splits allowed along any branch of a tree, explored over the grid $\{1,2,3,4,5,6\}$. Deeper trees capture more complex interactions but are more prone to overfitting.
- **Number of randomly sampled predictors at each split** (mtry): at each split, only a random subset of the 95 characteristics is considered, which reduces the correlation between individual trees and improves the diversity of the ensemble. Three values are explored: approximately $p/3$, $p/5$, and $p/10$, corresponding to roughly 32, 19, and 10 characteristics per split.

The number of trees is fixed at 300 for the final model (200 are used during the tuning phase to reduce computation time). Bootstrap sampling is used to construct each tree, as in the original specification of Breiman (2001).

4.4.2 Variable Importance: Mean Decrease in Impurity

Beyond its forecasting role, the Random Forest model provides a measure of variable importance through the Mean Decrease in Impurity (MDI). For every characteristic, the MDI accumulates, across all trees and all splits where that variable is used, the reduction in prediction error attributable to the split. A higher MDI score indicates that the characteristic plays a more central role in the model's predictions.

In the implementation, the MDI scores are normalized so that they sum to one, allowing them to be interpreted as relative contributions. This measure is reported for the top 20 characteristics and

provides an economically interpretable picture of which firm attributes drive return predictability in the nonlinear model.

4.4.3 Expanding Window Estimation

In addition to a static model trained once on the full training sample (1957–1974), the Random Forest is also estimated using a recursive expanding window scheme designed to more closely replicate the out-of-sample conditions of an actual investor. For each year t in the test period (1987–2021), the procedure works as follows:

1. The training set includes all observations from 1957 up to year $t - 13$.
2. The validation set covers the 12-year window immediately before year t, i.e., from year $t - 12$ to year $t - 1$.
3. The model is re-tuned annually using this rolling validation window, so that the selected hyperparameters (L and mtry) can adapt to changes in the data.
4. The re-estimated model then generates forecasts for year t, which are stored and never used again.

This procedure is applied for every test year, and the resulting forecasts are concatenated to produce a single time series of out-of-sample predictions that is used to compute the overall R^2_{OOS}. Years for which the training or validation set contains fewer than 500 or 50 observations respectively are skipped to avoid numerical instability. This expanding window approach is more computationally demanding than the static model but better reflects the temporal dynamics of the data and avoids any form of look-ahead bias in the hyperparameter selection.

4.5 Gradient Boosting Machines

4.5.1 Model Description

Gradient Boosting Machines (GBM) represent an alternative tree-based ensemble method that builds regressions trees sequentially rather than in parallel. While Random Forest relies on bagging to reduce estimator variance, GBM focuses on bias reduction by training each new tree to predict the residuals (errors) of the cumulative ensemble that preceded it. This sequential optimization allows the model to continuously refine its predictions in regions of the feature space where previous specifications performed poorly.

The implementation follows an extreme gradient boosting framework (XGBoost). Similar to the Random Forest setup, the hyperparameter of the number of boosting rounds (nrounds) is optimized annually using the rolling validation window to prevent overfitting and adapt to structural changes in return dynamics.

4.5.2 Variable Importance and Estimation

To maintain methodological symmetry with the Random Forest framework, the relative contribution of each firm characteristic within the GBM model is evaluated via fractional contribution to total gain. These importance scores are normalized cross-sectionally to sum to one. Crucially, to ensure a balanced comparison, the GBM framework is evaluated under both the baseline static split and the recursive expanding-window estimation scheme.

4.6 Neural Networks

4.6.1 Architecture

Neural networks represent the most flexible class of models included in the analysis. Following Gu, Kelly, and Xiu (2020), three architectures of increasing depth are considered:

Model	Hidden Layers	Neurons per Layer
NN1	1	32
NN2	2	32 → 16
NN3	3	32 → 16 → 8

Each architecture takes the 95 cross-sectionally rank-normalized firm characteristics as inputs and processes them to yield a single scalar output, the predicted excess return for each stock.

To introduce non-linear relationships into the expected return function, a Rectified Linear Unit (ReLU) activation function ($ReLU(x)=max(0,x)$) is applied between each pair of consecutive layers. The ReLU function has become a standard choice in the machine learning literature due to its computational efficiency and its ability to encourage sparsity by mapping negative values directly to zero. Furthermore, each hidden layer incorporates batch normalization. By rescaling and stabilizing the variance of layer inputs at each training step, this technique effectively mitigates internal covariate shift, thereby smoothing the optimization landscape and accelerating gradient convergence.

A direct comparison among NN1, NN2, and NN3 allows this study to evaluate whether incremental architectural depth yields substantial out-of-sample predictive gains, or if a more constrained model specification is favored for return forecasting.

4.6.2 Training Procedure and Regularization

The network weights and parameters are estimated by minimizing a penalized mean squared error (MSE) loss function that embeds an L1 regularization term:

$$\mathcal{L}(\theta) = \frac{1}{N}\sum_{i,t}\left(r^e_{i,t+1} - \hat{r}^e_{i,t+1}(\theta)\right)^2 + \lambda\sum_{k}|\theta_k|$$

where θ denotes all the network weights, N is the number of training observations, and the second term is an L1 penalty on the absolute values of the weights, analogous to the LASSO penalty in the linear case. This regularization term discourages the network from assigning excessive importance to any individual weight and helps prevent overfitting.

The optimization is performed using the Adam algorithm, an efficient variant of stochastic gradient descent that automatically adjusts the learning rate for each parameter based on estimates of the first and second moments of the gradient. The learning rate starts at 10^{-3} and is reduced by a factor of 0.5 whenever the validation loss does not improve for three consecutive epochs. Training is stopped early, before the maximum of 100 epochs, if the validation loss fails to improve for five consecutive epochs, a standard technique to prevent overfitting by halting the optimization before the model starts to memorize the training data.

The penalty parameter λ is tuned on the validation set for each year in the test period. Three candidate values are considered: $\{10^{-5}, 10^{-4}, 10^{-3}\}$. For the tuning phase, a faster ensemble of three randomly initialized networks is used to select the optimal λ, after which the final model for that year is estimated as an ensemble of 10 independently trained networks. The final prediction for each stock-year is obtained by averaging the forecasts of all 10 networks. This ensemble approach reduces the sensitivity of the results to the random initialization of the network weights and produces more stable out-of-sample forecasts, following the same logic as averaging trees in a Random Forest.

The network training uses mini-batches of 10,000 observations drawn randomly from the training set at each optimization step, which reduces the computational cost of each gradient update compared to using the full sample at once.

4.6.3 Expanding Window Scheme

The same recursive expanding window scheme described for the Random Forest in Section 4.4.3 is applied to all three neural network architectures. For each year t in the test period, the network is trained on all observations from the beginning of the sample to year and used to forecast returns for year t. The procedure is repeated for every test year, and all forecasts are collected to compute a single R^2_{OOS} value for each architecture.

This yearly re-estimation approach means that the neural networks can adapt to structural changes in the return-generating process over time.

4.6.4 Variable Importance: Sample Sensitivity Derivative

For the deepest architecture (NN3), a measure of variable importance is computed following the method described in Gu, Kelly, and Xiu (2020). This measure, referred to as the Sample Sensitivity Derivative (SSD), quantifies the average sensitivity of the network's output to small perturbations in each input characteristic. Formally, for characteristic j, the SSD is computed as:

$$SSD_j = \frac{1}{N}\sum_{i,t}\left(\frac{\partial \hat{r}^e_{i,t+1}}{\partial z_{j,i,t}}\right)^2$$

A large SSD for characteristic j indicates that small changes in that variable produce large changes in the model's prediction, meaning that the model assigns high informational content to that characteristic. The SSD values are normalized to sum to one so that they can be compared directly with the MDI scores from the Random Forest, providing a consistent picture of predictor importance across the two families of nonlinear models. For the SSD calculation, the NN3 model is re-trained on the combined training and validation set (1957–1986) with the final $\lambda = 10^{-5}$, and the gradients are computed analytically using automatic differentiation.

4.7 Long Short-Term Memory (LSTM) Networks

4.7.1 Architecture and Gating Mechanisms

To manage the limitations of standard dense feed-forward networks in processing financial time series, the analysis implements a recurrent neural network sequence architecture, specifically a Long Short-Term Memory (LSTM) network. Standard networks treat stock-month observations as independent cross-sectional points; conversely, the LSTM architecture natively processes data as temporal sequences, allowing the model to capture persistent dependencies and conditional dynamics over time.

The implemented module consists of one LSTM layer containing 32 hidden units, mapped to a linear dense output layer that generates the scalar one-month-ahead predicted excess return. The internal architecture utilizes structural gating mechanisms, comprising an input gate, a forget gate, and an output gate, to regulate the retention and decay of historical information, effectively smoothing out high-frequency market noise.

4.7.2 Training, Regularization, and Interpretability

The training protocol for the LSTM matches the neural network pipeline described in Section 4.6.2, incorporating Adam optimization, learning rate reduction on plateaus, L1 weight penalization, and a 10-model ensemble averaging routine to stabilize predictions. Due to its architecture, the model is estimated exclusively under the recursive expanding-window scheme.

For model interpretability, feature importance is derived via a gradient-based approach analogous to the Sum of Squared Derivatives (SSD) used for the dense networks. Gradients of the network output with respect to the input characteristics are calculated analytically via automatic differentiation. The squared gradients are averaged over the temporal dimension and normalized to sum to one, providing a direct, non-parametric mapping of characteristic importance in the recurrent framework.

4.8 Summary of Model Specifications

The table below summarizes the main characteristics of each model estimated in the analysis.

Model	Linearity	Regularization	Hyperparameters	Estimation Schemes	Interpretability Metric
OLS	Linear	None	—	Static & Expanding Window	Average Coefficients (beta)
LASSO	Linear	L1	lambda	Static & Expanding Window	Sparsity / Non-zero beta
Ridge	Linear	L2	lambda	Static & Expanding Window	Magnitude of beta
Elastic Net	Linear	L1 + L2	lambda (rho = 0.5)	Static & Expanding Window	Absolute betaRanking
Random Forest	Nonlinear	Tree Constraints	Depth, mtry	Static & Expanding Window	MDI Importance
GBM	Nonlinear	Shrinkage / Depth	Learning Rate, Depth	Static & Expanding Window	Fractional Gain
NN1	Nonlinear	L1 Weights	lambda, Early Stopping	Expanding Window Only	—
NN2	Nonlinear	L1 Weights	lambda, Early Stopping	Expanding Window Only	—
NN3	Nonlinear	L1 Weights	lambda, Early Stopping	Expanding Window Only	Sample Sensitivity (SSD)
LSTM	Recurrent	L1 Weights	lambda, Gating Units	Expanding Window Only	Gradient-based Importance

All models are trained on the 1957–1974 period (static models) and expanding windows starting from 1957, and all hyperparameters are selected using only the validation data. The test period covers 1987–2021 and is used exclusively for the final out-of-sample evaluation reported in Chapter 5.

Chapter 5 — Empirical Results

5.1 Overview

This chapter presents the out-of-sample predictive performance of all models estimated in the analysis. The evaluation covers the test period from 1987 to 2021, for a total of 35 years and approximately 205,000 stock-month observations. Performance is measured exclusively using the out-of-sample R^2_{OOS} defined in Gu, Kelly, and Xiu (2020), which benchmarks each model against the naive forecast of zero excess return for every stock at every point in time. A positive R^2_{OOS} therefore indicates predictive ability relative to this benchmark, while a negative value implies that the model does worse than simply predicting zero.

The models are organised into three families of increasing complexity: linear benchmarks (OLS, LASSO, Ridge, and Elastic Net), tree-based methods (Random Forest and Gradient Boosting Machine), and neural networks (NN1, NN2, NN3, and LSTM). All nonlinear models are estimated under a recursive expanding-window scheme that re-estimates the model annually, so that predictions at each test year are based exclusively on information available up to that point. For each family, results are reported separately for the static and rolling-window estimation schemes where applicable, and a synthesis of all models is presented in Table 5.1.

5.2 Linear Models

5.2.1 OLS

The static OLS model produces a strongly negative R^2_{OOS} on the test set (−13.30%), confirming the well-known tendency of unrestricted least squares to overfit when the number of predictors is large relative to the training sample. With 95 characteristics and only roughly 24,000 training observations, the coefficient estimates are highly unstable, and the forecasts deteriorate substantially when applied to the test period. The rolling-window version, by contrast, recovers positive predictability at +0.45%, which suggests that time-varying estimation partially compensates for the lack of regularization by limiting the accumulation of estimation error over time.

This result is consistent with Gu, Kelly, and Xiu (2020), who exclude OLS from their main comparison precisely because of its instability in high-dimensional settings. The difference between the static and rolling-window variants provides a useful illustration of why estimation scheme matters as much as functional form when the signal-to-noise ratio is low.

5.2.2 Penalized Linear Models

LASSO, Ridge, and Elastic Net all achieve positive and quantitatively similar out-of-sample performance, both in the static and rolling-window configurations. In the static scheme the three models produce R^2_{OOS} values of +0.95% (LASSO), +0.98% (Ridge), and +0.97% (Elastic Net). LASSO selects 13 characteristics out of 95 at the optimal regularisation parameter λ = 0.000132, while Ridge retains all predictors with smooth shrinkage controlled by λ = 0.011498.

Under the rolling-window scheme all three penalized models improve noticeably, reaching values in the range of +2.00–2.01%. This improvement is explained by two factors: the rolling validation window keeps the penalty parameter relevant to recent market conditions, and the accumulation of training data over time improves coefficient stability. The three models are nearly indistinguishable in terms of R^2_{OOS}, suggesting that in this dataset the specific form of the regularisation penalty matters less than the act of regularizing itself. This finding is aligned with Bagnara (2024), who notes that penalised linear models tend to converge in performance when the predictor set is broad and the signal is diffuse.

5.3 Random Forest

5.3.1 Static Model

The static Random Forest is tuned on the 1975–1986 validation set over a grid of maximum tree depth (1 to 6) and feature subsampling rates (mtry ∈ {10, 19, 32}). The optimal hyperparameters are depth = 1 and mtry = 32, with a validation R^2_{OOS} of +1.01%. The shallow optimal depth is notable: it indicates that the forest best captures return predictability through simple, single-split trees that effectively model pairwise interactions between characteristics and returns without overfitting to higher-order structure. The static test R^2_{OOS} is +0.88%, slightly below the penalised linear models estimated under the rolling-window scheme, reflecting the well-known degradation of any static model applied to a 35-year evaluation horizon.

5.3.2 Expanding Window

Under the recursive expanding-window scheme the Random Forest improves substantially to +2.03%, the highest value among all tree-based and linear models. The improvement relative to the static variant confirms that re-estimating the model annually allows the forest to adapt its split structure to changing cross-sectional patterns, particularly those associated with momentum cycles and liquidity regime changes observed in the latter part of the test period.

5.3.3 Variable Importance

The normalized MDI scores from the static Random Forest are highly concentrated on a small number of characteristics. The top five contributors: mom12m (25.6%), mom1m (23.6%), indmom (20.3%), std_turn (11.4%), and mom6m (10.7%), together account for more than 90% of total feature importance. This extreme concentration suggests that at depth 1 each tree is effectively a single split on one of these momentum or turnover variables, and the forest aggregates over these univariate signals rather than learning complex interactions. The dominance of momentum characteristics is consistent with the broader asset pricing literature, where price-continuation effects represent one of the most robust and persistent cross-sectional predictors.

5.4 Gradient Boosting Machine (GBM)

5.4.1 Out-of-Sample Performance

The Gradient Boosting Machine (GBM), implemented via XGBoost with a learning rate of 0.01, maximum depth of 6, subsample ratio of 0.5, and early stopping based on validation loss, achieves a test $R^2_{OOS}$of +1.89% in the static configuration and +1.97% in the rolling-window scheme. These values place GBM among the best-performing models in this study, above the static Random Forest and in line with the rolling Random Forest.

The good performance of GBM is consistent with the broader machine learning in finance literature. As highlighted by Bagnara (2024), sequential boosting algorithms are particularly effective when the cross-section contains heterogeneous predictive signals that cannot be efficiently captured by averaging independent trees. By iteratively fitting residuals, GBM progressively sharpens the signal encoded in characteristics related to profitability, investment, and sales growth.

5.4.2 Variable Importance: Gain-Based

The normalized gain-based importance scores from the static GBM model confirm that the most influential characteristics partially overlap with those identified by the Random Forest, but the distribution is less concentrated. While momentum variables remain important, volatility and liquidity-related characteristics also receive substantial weight alongside the momentum family, with the remaining gain spread across size, profitability, and trading activity variables, suggesting that GBM is able to exploit a richer set of signals than the shallow Random Forest. This feature utilization reflects the iterative nature of boosting, which sequentially corrects residuals and can therefore identify weaker signals that would be marginalized in a single-pass ensemble.

5.5 Neural Networks

5.5.1 Out-of-Sample Performance

The three feed-forward neural network architectures, NN1 (one hidden layer of 32 units), NN2 (hidden layers of 32 and 16 units), and NN3 (hidden layers of 32, 16, and 8 units), are all estimated under the recursive expanding-window scheme with an ensemble of 10 independently initialized networks. The R^2_{OOS}values are +0.06% for NN1, +0.74% for NN2, and +1.08% for NN3. The monotonic increase with depth confirms that deeper architectures are better able to capture nonlinear interactions in the predictor space, consistent with Gu, Kelly, and Xiu (2020).

Interestingly, the results for NN2 and NN3 exceed those reported in the original paper (NN1: +0.33%, NN2: +0.39%, NN3: +0.40%), while NN1 is the only architecture that underperforms its paper counterpart. This result is slightly unexpected, given that this analysis operates on a considerably smaller stock universe. A plausible interpretation is that the 1,000-stock sample, despite being more limited in size, turned out to be more predictable than the full CRSP universe: a more homogeneous cross-section of predominantly larger and more liquid firms may carry a cleaner and more persistent signal in the characteristic space, allowing the networks to learn more stable mappings even with a smaller training set.

5.5.2 LSTM

The Long Short-Term Memory (LSTM) network achieves the highest R^2_{OOS} among all models in this study, at +2.07%. This result is particularly notable because the LSTM is the only architecture that explicitly accounts for the temporal structure of the data: through its recurrent mechanisms it can distinguish between persistent signals and temporary noise in the time series of firm characteristics, rather than treating each stock-month observation as independent. The improvement over NN3 (+1.08%) suggests that incorporating sequential dynamics adds genuine predictive value beyond what is captured by static cross-sectional mappings, consistent with Fischer and Krauss (2018).

5.5.3 Variable Importance: SSD

The Sample Sensitivity Derivative (SSD) analysis for NN3 identifies the top predictors as pchsale_pchinvt (2.92%), herf (2.41%), sp (2.26%), grltnoa (2.18%), and roaq (2.11%). The list is more diversified than the MDI rankings from the Random Forest, reflecting the neural network's ability to process a broader set of characteristics simultaneously. Profitability variables (roaq, cashdebt, cfp_ia), sales-based signals (pchsale_pchinvt, rsup), and industry concentration (herf) together

account for a substantial share of total sensitivity. The limited role of pure momentum characteristics in the SSD scores, compared to their dominance in the MDI rankings, suggests that neural networks are exploiting a different and complementary portion of the information set relative to tree-based methods.

5.6 Summary of Results

Table 5.1 reports the R^2_{OOS} for all models under both estimation schemes. The key takeaway is that the ranking of models by complexity generally corresponds to the ranking by predictive performance, with the LSTM at the top and static OLS at the bottom. However, the differences between model families are not always large, and the rolling-window versions of penalised linear models are competitive with several nonlinear alternatives.

Model	Static (%)	Rolling Window (%)
OLS	−13.30	+0.45
LASSO	+0.95	+2.01
Ridge	+0.98	+1.99
Elastic Net	+0.97	+2.01
Random Forest	+0.88	+2.03
GBM	+1.89	+1.97
NN1	—	+0.06
NN2	—	+0.74
NN3	—	+1.08
LSTM	—	+2.07

Table 5.1 — R^2_{OOS} Test Set: All Models. Values in percentage points. The benchmark is the zero-return forecast. Neural networks and LSTM are estimated exclusively under the rolling-window scheme.

Chapter 6 — Discussion

6.1 Overview

The results presented in Chapter 5 raise several interpretive questions that go beyond a simple ranking of models by R^2_{OOS}. This chapter discusses the main themes that emerge from the empirical comparison: the role of regularisation, the contribution of nonlinearity, the performance of neural networks relative to their complexity, the LSTM's advantage, and the relationship between these findings and the original results in Gu, Kelly, and Xiu (2020). The goal is not to declare a winner among the methods, but to understand the conditions under which each approach adds value and what the results collectively imply for the use of machine learning in empirical asset pricing.

6.2 The Role of Regularization

Probably the most important finding of this analysis is the contrast between OLS and the penalized linear models. Static OLS produces an R^2_{OOS} of −13.30%, while LASSO, Ridge, and Elastic Net estimated on the same training window achieve values near +0.97%. This difference is entirely attributable to regularisation: the penalty terms prevent the model from fitting noise in the training data, producing out-of-sample forecasts that are more stable and economically coherent.

This finding is consistent with the emphasis placed on regularization in Bagnara (2024), who concludes that the primary contribution of penalized regressions in asset pricing is variance reduction rather than model selection per se. The fact that LASSO, Ridge, and Elastic Net perform nearly identically supports this interpretation: the specific form of the penalty appears to be a secondary consideration relative to the act of constraining the coefficient magnitudes. The LASSO's sparse solution (13 active predictors) and Ridge's dense but shrunk solution produce quantitatively indistinguishable performance, suggesting that the predictive information in the full characteristic set is distributed across many weakly informative predictors rather than concentrated in a small subset.

6.3 The Contribution of Nonlinearity

Moving from penalized linear models to tree-based methods and neural networks reveals an additional layer of performance improvement, though the magnitude depends heavily on the estimation scheme. In the rolling-window configuration, the Random Forest (+2.03%) and GBM (+1.97%) are approximately comparable to the penalized linear models (+2.00–2.01%), while NN3

(+1.08%) and LSTM (+2.07%) sit at or above the linear benchmarks. This suggests that nonlinearity adds modest but genuine value over and above regularization alone.

The most interesting comparison is between the static and rolling-window versions of the Random Forest. The static model achieves +0.88%, while the rolling-window version reaches +2.03%, a gap attributable entirely to the recursive re-estimation. This illustrates a general principle in return prediction: the value of nonlinear functional forms is difficult to distinguish from the value of temporal adaptability when both vary across models. In this dataset the expanding window contributes at least as much to performance as the nonlinear split structure of the Random Forest.

6.4 Neural Networks: Performance and the Sample Constraint

The neural networks produce a more mixed picture than the tree-based methods. NN1 barely outperforms the zero benchmark (+0.06%), while NN2 and NN3 achieve +0.74% and +1.08% respectively, both below the rolling Random Forest and GBM. This stands in contrast to the results in Gu, Kelly, and Xiu (2020), where neural networks consistently outperform tree-based methods and rank at the top of the model comparison.

The explanation lies probably in the size of the cross-section. The full CRSP universe used in the original paper includes several thousand active stocks at any point in time, providing neural networks with substantially richer training signals than the more restricted panel used here. This difference in scale, probably, constrains the networks' ability to learn stable nonlinear representations, as the statistical mass available for estimating thousands of parameters is considerably smaller throughout the entire test period. The monotonic improvement from NN1 to NN3 confirms that depth helps when enough data is available, consistent with the universal approximation argument that underpins deep learning in asset pricing (Gu, Kelly, and Xiu, 2020). However, depth alone is not sufficient when the dataset is small.

6.5 LSTM: Sequential Dynamics and Temporal Persistence

The LSTM achieves the highest R^2_{OOS} in the study (+2.07%), marginally outperforming the rolling Random Forest and penalised linear models. Its advantage lies not in greater complexity, but in a different design: rather than relying solely on current characteristic values, the LSTM incorporates their recent evolution as an additional source of predictive information.

This advantage can be interpreted in light of the SSD analysis of NN3. Variables such as roaq, cashdebt, and sp are highly persistent over time, meaning that their recent trajectory carries additional information beyond the current level alone. A static model can only observe where a characteristic stands today, while the LSTM also conditions on where it has been, a distinction that becomes particularly valuable during periods of trend acceleration or reversal. This is consistent with Fischer and Krauss (2018).

6.6 Comparison with the Original Paper

The qualitative ordering of the results in this study is broadly consistent with Gu, Kelly, and Xiu (2020): penalized linear models dominate OLS, tree-based methods are competitive with or slightly better than linear models, and the best neural architectures sit near the top of the ranking. The quantitative values, however, differ in important ways.

In Gu, Kelly, and Xiu (2020) the neural networks achieve substantially higher R^2_{OOS} values, while the penalised linear models produce values below +0.40%. In this study the ranking is partially reversed: not only the penalized models under the rolling-window scheme reach +2.01%, but tree-based methods such as the rolling Random Forest (+2.03%) and GBM (+1.97%) also outperform all neural network architectures, with NN3 achieving only +1.08%. This inversion is explained by two combined factors. First, the 1,000-stock universe is more homogeneous than the full CRSP universe, compressing the cross-sectional dispersion in expected returns. Second, the rolling-window scheme applied uniformly to all models, including the linear ones, provides the penalized regressions with an adaptation mechanism that is absent in the original paper's static linear benchmarks, partially closing the gap with the nonlinear models.

A further point of comparison concerns the test period. The original paper evaluates models on data through 2016, while this analysis extends through 2021, incorporating the COVID-19 shock of 2020 and the subsequent recovery. The post-2016 period is characterized by elevated market volatility and structural breaks in factor exposures, which may differentially affect the stability of neural networks, something that the original paper could not have observed in its sample.

Chapter 7 — Conclusion

This thesis set out to examine whether machine learning methods can improve the prediction of individual stock returns relative to traditional approaches, replicating and extending the framework of Gu, Kelly, and Xiu (2020) on a panel of 1,000 U.S. stocks over the period 1957–2021. The analysis estimates ten models of increasing complexity (OLS, LASSO, Ridge, Elastic Net, Random Forest, Gradient Boosting Machine, three feed-forward neural networks (NN1, NN2, NN3), and an LSTM) and evaluates them on a strictly out-of-sample test period from 1987 to 2021 using the R^2_{OOS} metric of Gu, Kelly, and Xiu (2020).

The main empirical finding is that machine learning does add value in return prediction, but the nature of the improvement depends critically on two factors that are often combined: regularization and temporal adaptability. Regularization, implemented through penalty terms in linear models or through ensemble aggregation in tree-based methods, is probably the most important driver of out-of-sample performance. The difference between static OLS (−13.30%) and penalized linear models (+0.95–0.98%) makes this point very clear. Temporal adaptability, implemented through the recursive expanding-window scheme, provides an additional and quantitatively large improvement for all model classes, bringing penalized linear models and tree-based methods to similar performance levels in the range of +1.97–2.03%.

The neural networks present a more heterogeneous picture. NN3 achieves +1.08% and the LSTM reaches +2.07%, the highest value in the study, but NN1 barely outperforms the zero benchmark at +0.06%. The gap between the NN1, NN2, and NN3 and the LSTM suggests that the temporal structure of the data, in particular the persistence of momentum and profitability signals, contains predictive information that static architectures cannot exploit. The underperformance of NN1 relative to the original paper is explained by the limited size of the cross-section: with 1,000 stocks, neural networks face a more constrained estimation environment than in the full CRSP universe, and regularization through L1 penalization and early stopping can only partially compensate for this.

The variable importance analysis provides a complementary perspective. The Random Forest concentrates nearly all its predictive weight on momentum variables, while NN3's SSD scores distribute importance more broadly across profitability, sales-based, and industry-level signals. This difference suggests that the two families of nonlinear models are exploiting different parts of the information set.

The choice of estimation scheme, the size of the cross-section, and the architecture of the nonlinear model all matter, and the conditions under which complexity pays off need to be assessed carefully in each specific data environment. The results confirm that the broad patterns documented in Gu, Kelly, and Xiu (2020) persist in an updated and more restricted sample: machine learning methods consistently outperform unregularized OLS, and the predictive role of momentum and profitability characteristics remains central across all methods. The post-2016 extension of the test period does not change these conclusions, but it adds evidence that the performance of neural networks can be sensitive to structural breaks during periods of elevated market stress. In particular, no single architecture emerges as a clear winner across all conditions. Rather than focusing on a single method, a wider comparison across multiple model families is more likely to provide a reliable picture of what works and under what conditions. Testing a larger range of algorithms on different stock universes and time periods would help establish which findings are robust and which are specific to a particular data environment.

Bibliography

- Avramov, D., Cheng, S. and Metzker, L. (2023). Machine learning performance and equity risk premia: A path-dependent analysis. *Journal of Financial Economics*, 148(3)
- Bagnara, M. (2024). Asset pricing and machine learning: A critical review. *Journal of Economic Surveys*, *38*(1).
- Breiman, L. (2001). Random forests. *Machine Learning*, *45*(1).
- Carhart, M. M. (1997). On persistence in mutual fund performance. *The Journal of Finance*, *52*(1).
- Cochrane, J. H. (2011). Presidential address: Discount rates. *The Journal of Finance*, 66(4)
- Fama, E. F., & French, K. R. (1993). Common risk factors in the returns on stocks and bonds. *Journal of Financial Economics*, *33*(1).
- Fischer, T., & Krauss, C. (2018). Deep learning with long short-term memory networks for financial market predictions. *European Journal of Operational Research*, *270*(2).
- Gu, S., Kelly, B., & Xiu, D. (2020). Empirical asset pricing via machine learning. *Review of Financial Studies*, *33*(5).
- Gu, S., Kelly, B., & Xiu, D. (2020). Empirical asset pricing via machine learning: Online technical appendix.
- Gu, S., Kelly, B., & Xiu, D. (2021). Autoencoder asset pricing models. *Journal of Econometrics*, *222*(1).
- Harvey, C. R., Liu, Y., & Zhu, H. (2016). …and the cross-section of expected returns. *Review of Financial Studies*, *29*(1).
- Hou, K., Xue, C., & Zhang, L. (2015). Digesting anomalies: An investment approach. *The Review of Financial Studies*, 28(3)
- Kelly, B. T., & Xiu, D. (2023). Financial machine learning. *Foundations and Trends in Finance*, *13*(3–4).
- Masini, R. P., Medeiros, M. C., & Mendes, E. F. (2023). Machine learning advances for time series forecasting. *Journal of Economic Surveys*, *37*(1).
- Welch, I., & Goyal, A. (2008). A comprehensive look at the empirical performance of equity premium prediction. *Review of Financial Studies*, *21*(4).

Dichiarazione del ricorso a strumenti di Intelligenza Artificiale (IA)

Il/La sottoscritto/a *Mattia Fabris*, matricola numero *889576*, iscritto/a presso Ca' Foscari al corso di Laurea/Laurea magistrale in *Economia e Finanza*, autore/autrice dell'elaborato finale/tesi di laurea *Asset Pricing and Machine Learning*, relatore/relatrice *Monica Billio*, consapevole delle sanzioni previste dall'art. 76 del Testo unico, D.P.R. 28/12/2000 n.445, e della decadenza dei benefici prevista dall'art. 75 del medesimo Testo unico in caso di dichiarazioni false o mendaci, sotto la propria personale responsabilità ai sensi degli artt. 46 e 47 del D.P.R. 28/12/2000 n. 445,

DICHIARA

- ☐ di NON aver fatto ricorso a tecnologie di intelligenza artificiale generativa per supportare vari aspetti della ricerca e della scrittura del suo elaborato finale/della sua tesi di laurea;
- ☒ di aver fatto ricorso a tecnologie di intelligenza artificiale generativa per supportare vari aspetti della ricerca e della scrittura del suo elaborato finale/della mia tesi di laurea, in particolare *(per ogni punto scrivere per esteso quanto viene richiesto)*:
  - *indicare il nome dello strumento (es. "ChatGPT" se applicazione o "GPT" se modello) con riferimento alla versione specifica (ad esempio, "ChatGPT-4, versione rilasciata a marzo 2024")*; *Gemini (giugno 2026), Perplexity (giugno 2026)*
  - *se sono stati utilizzati più strumenti, specificare ciascuno di essi indicandone nome e versione;*
  - *fornire informazioni su come è stato utilizzato lo strumento:*
    - ☒ *revisione e ottimizzazione dei testi per migliorare chiarezza e coerenza;*
    - ☐ *generazione di grafici e diagrammi per la presentazione dei dati;*
    - ☒ *supporto alla scrittura automatica di frasi;*
    - ☐ *produzione di immagini illustrative per integrare la parte metodologica;*
    - ☐ *altro (specificare)*;
  - *indicare le sezioni/i capitoli interessate/interessati; da Abstract a Capitolo 7*
  - *indicare come è stata verificata la correttezza, la qualità e l'attendibilità delle informazioni o dei testi eventualmente generati*. *I risultati generati dai modelli AI sono stati riletti al fine di confermare la coerenza e la correttezza. L'uso è stato fatto soprattutto per prendere spunto nella formulazione delle frasi in inglese e nell'organizzazione del discorso.*

DICHIARA INOLTRE

che tutto il materiale generato dall'IA è stato attentamente verificato, rivisto e rielaborato dall'autore/dall'autrice, che si assume la piena responsabilità per la qualità, l'accuratezza e l'integrità scientifica del lavoro.