Machine learning in systematic equity allocation: A model comparison

Tony Guida and Guillaume Coqueret

Abstract

In this paper, we compare popular Machine Learning (ML) approaches (random forest, boosted trees and neural networks) to build diversified equity portfolio, using different weighting scheme and cash neutral long-short portfolios. We demonstrate that using ML models on a large number of features gives an average error rate of 39% for predicting the 12-month sector neutral outperformance of a stock.
We find that, irrespective of the weighting scheme, boosted trees and neural networks outperform a long only factor investing type of global equity portfolio. Boosted trees shows the best risk/return profile from the set of ML algorithms used. It generates on average an excess return of 2.1% per annum, compared to a simple multifactor portfolio on a long-short basis and 1.1% above multi-factor for the only strategies across all weighting scheme.

1. Introduction

Because of their supposedly superior forecasting abilities, Machine Learning (ML) algorithms have become popular in financial economics. Sadly, as is underlined in De Prado (2018), ML is far from a magical panacea for investors. The degrees of freedom when building an automated decision process are infinite: investment universe, liquidity and turnover constraints, forecasting tool(s), input features, hyper-parameter management, factor exposure imperatives, etc. In this article, we focus on the choice of the forecasting tool. It is well known that supervised learning encompasses a large spectrum of methods. A non-exhaustive list includes: generalized linear models (possibly penalized), hierarchical models (e.g., trees and their extensions), Bayesian classifiers, neural networks, and support vector machines. It can be a daunting task to choose one in particular and our aim here is to shed some light on the relative performance of these techniques.

The literature is obviously a good place to start looking. With regard to learning tools, many studies have already been carried out. We group them by algorithm:[1]

- Bayesian inference (Bodnar et al. (2017)),
- regularized quadratic optimization (Goto & Xu (2015), Ban et al. (2016)),
- penalized predictive regressions (Rapach et al. (2013)),
- flag pattern recognition (Arévalo et al. (2017)),
- support vector machines (Cao and Tay (2003), Matias and Reboredo (2012), Dunis et al. (2013)),
- trees and their extensions (Ballings et al. (2015), Patel et al. (2015), Moritz and Zimmermann (2016), and Krauss et al. (2017), Gu et al. (2018)),
- other clustering methods (Raffinot (2018)),
- neural networks and their extensions (Dunis et al. (2008), Adeodato et al. (2011), Ballings et al. (2015), Patel et al. (2015), Heaton et al. (2017), Krauss et al. (2017), Fisher and Krauss (2018), Gu et al. (2018))
- mixes of several of the above (Dunis et al. (2013), Nair et al. (2017)).

Obviously, the input features are also crucial.[2] Some studies, e.g., Krauss et al. (2017) and Dunis et al. (2013) rely solely on price data. Nonetheless, there is somewhat of a consensus in the academic financial literature that additional factors are likely to drive future returns.

This strand of research has been thriving since the seminal study by Fama and French (1992). They showed that book-to-market ratios and market capitalizations structure future returns. For instance, they

[1] We only mention the papers related to asset management and portfolio selection. Other applications of Machine Learning in finance include: fraud detection, credit scoring, and derivative pricing.

[2] Computer scientists often mention the saying « *garbage in, garbage out* » and in our context, it is obvious that the accuracy of predictions will heavily depend on the quality and relevance of predictors.

report that small firms tend to yield returns that are higher than those of large firms.[3]

Hundreds of similar studies have been published in the last two decades. They propose new characteristics and innovative methodologies, and sometimes confirm or contradict previous findings. The literature on these so-called *anomalies* is thus rather vast and has its own meta-studies (see e.g., Subrahmanyam (2010), Green et al. (2013), Harvey et al. (2016), McLean and Pontiff (2016), and Feng et al. (2017)).

The major industrial impact of this blossoming academic field is the rapid growth of smart-beta indices. In this scheme, funds allocate, possibly automatically, according to particular firm attributes. Often, these funds offer higher performance for a lower cost, compared to more traditionally managed funds. This explains why the asset management industry is increasing its demand in smart-beta strategies (Kahn and Lemmon (2016)).[4]

This phenomenon is likely to increase the dependence between firm features and future returns. Consequently, we will work both with a large spectrum of firms and a wide range of companies' characteristics. This makes sense because it is typically the purpose of ML algorithms to make sense of large datasets. This nonetheless presupposes that these algorithms are properly supervised. Nonlinear models such as decision trees and neural networks are expected to make the most of several dozens or several hundreds of explanatory variables.

Our main findings are the following:

- Boosted trees and neural network models can be used in global stocks selection models with an effective degree of accuracy.
- Without being "taught" asset pricing theories, the ML algorithms find most of the factor investing metrics as important variables,

[3] This is usually referred to as the size premium. This stream of literature was in fact initiated by Banz (1981) and is reviewed in Van Dijk (2011).

[4] For a more detailed view of the intertwining between factor investing and asset management, we refer to the monographs of Ilmanen (2011) and Ang (2014).

even if the feature dataset is large (350 characteristics). This represents a promising step towards an “empirical asset pricing” approach in quantitative equity (see, e.g., Gu et al. (2018)).

- Multi-factor signals based on trees boosted and neural networks outperformed the signals based on simple factor-investing type:
  - This result holds for decile performance and L/S performance analysis using theoretical equal weight portfolios.
  - This result holds also for long-only portfolios consisting of top decile stocks, irrespective of the weighting scheme used and net of simple transactions costs.

This article is organized as follows. In Section 2, we introduce the data and methods we implement. In Section 3, we build the models and assess their quality. Experiments and results for testing the models are in Section 4. Finally, in Section 5 we conclude.

2. Data and methodology

2.1. Building blocks in supervised learning

Our study focuses on two families of ML tools: decision trees and neural networks. The classical reference for both techniques is Friedman et al. (2009).

Early work on trees goes back to the 1960s, but a full treatment and many extensions only occurred in the 1990s (Hastie and Tibshirani (1990). The two notable improvements we work with are random forests and boosted trees. Random forests, first introduced in Ho (1995), consist in aggregating the outcomes of simple trees when the training sample is randomized. The premise of boosting (e.g., Schapire (1990) and Freund & Schapire (1997)) is the same: the idea is to combine trees to improve accuracy. The essential difference is that unlike random forests, boosted machines optimize each new tree that is added to the model. We refer to Friedman et al. (2000) for an in-depth treatment on this topic.

Neural networks are chains of differentiable, parameterized geometric functions[5] and their origins can be traced to the mid-XXth century. We refer to Goodfellow et al. (2016) for more details on the subject.

The recent development of open-source ML packages has made it easy to implement the aforementioned algorithms. We refer for instance to xgboost (Chen and Guestrin (2016) for tree methods and to Keras (Chollet (2017)) for neural networks. Both families of tools were used in our study.

2.2. Data

We collect monthly returns and monthly stocks' characteristics for the top 3500 global[6] stocks according to their market capitalization, free-float adjusted. The full dataset goes from December 2003 until December 2017. The universe of stocks consists of all common equities using Quandl premium equity packages.[7] The dataset does not suffer from survivorship bias. Prices are monthly discrete total return, taking into account stocks splits and dividend adjustments. Prices are expressed in dollars as all the others amounts. Conversion has been automated for the non-USD stocks using the FX spot at the date of computation.

The dataset encompasses approximately 600,000 instances, an instance consisting of the combination of one stock and one date. The variable y we want to predict is the probability of 1-year forward sector-neutral outperformance (see below). The explanatory variables in our model include a large set of 350 features based on traditional, financial, price and volume-based metrics.

[5] This definition was coined by François Chollet.

[6] We focus on global stocks in order to assess how efficiently ML-based signals and portfolios will adapt to different country, region and foreign exchange variables without explicitly including them as features in the models.

[7] See https://www.quandl.com/

2.3. Features and labels engineering

Henceforth, we use interchangeably the term "feature" or dependent variable to express a stock characteristic. In this section, we will explain the features transformation that has been performed to linearize each characteristic and to express them in the same unit.

A substantial portion of research in ML-based financial applications fails because of a lack of economic framing and unrealistic or ill-defined goals, such as finding the "best stocks". Instead, our purpose is more reasonable, as we seek to predict extreme behavior and single out the goods stocks from the worst ones within each sector and express this as a probability in order to rank the full cross section of stocks.

We "engineer" both labels (future returns) and features to enforce a structure that will provide the algorithm with a more causal representation of equity markets. Again, we shift away from the traditional approach that seeks to infer future performance from past prices or short-term returns. We set fundamental, risk, volume and momentum-based variables as our features. Each feature and label is expressed in z-scores and then translated into percentile to ease the comparison in the results analysis part.[8]

In the same fashion, we impose some structure in the labels by sequentially:

1. Resorting to one-year (1Y) performance, which is enough for having a certain degree of causality between the nature of the features in the datasets and the tenor of the labels.
2. Then, for each stock return, we subtract the average return on the corresponding sector.
3. The third step is getting rid of the outliers in the labels: stocks outside the [5th; 95th] percentile of their sector neutral performance are excluded for the training. Our goal here is to imply as much causality for the features with the labels. For instance, we are getting rid of stocks that have been acquired in an M&A, or stocks that have been in fraud accounting scandals, because we want the labels to be truly linked to the features.

[8] It is often recommended to feed neural networks with normalized data, usually ranging from 0 to 1.

4. Processing only the remaining top and bottom quintiles of the filtered stocks: we suppress the bulk of moderate returns because they do not carry much information. We want to approximate a function for the top and bottom parts of the cross section. By doing so, we hope to have a clear hierarchical representation of sector under/out-performing stocks.
5. We define Y_1^i as the probability of a stock i to outperform its sector S over a one year ahead period. Accordingly, $Y_0^i = 1 - Y_1^i$ will be the probability of a stock i to underperform its sector after one year. Y_1^i serves as the primary input of our classification task. The label we process in the algorithm is the following:

$$y^i = \begin{cases} 1 \; if \; Y_0^i \geq 0.5 \\ 0 \; if \; Y_0^i < 0.5 \end{cases}.$$

   Hence, this variable tracks whether or not the corresponding stock is likely to outperform (its sector).

In the next subsection, we focus on the explanatory variables that we rely on to predict y^i.

2.4. Variables/features used

In our models, we aim to predict, each month, the probability of a stock to outperform its sector using ML models. Since our exercise here is about showing how adaptive and outperforming ML-based models can be, we will use all features in our dataset without any prior financial/economic belief. Said differently, we do not resort to important feature discovery in the first phase, but rather leave it to the models to determine which features matter. In the case of ML prediction using trees or neural networks, highly correlated variables will not perturb the models. A large number of possibly correlated variables will give the algorithm more degrees of freedom to determine the added value of each single variable.

The features can be categorized in six groups and we summarize them in Table 1 below.

TABLE 1: **Summary and examples of features per family type**. According to hierarchical cluster six main family of features based on metrics' types. We provide some examples for each family.

Valuation	**Prof. / Qual**	**MoM./ technical**	**Risk**	**Estimates**	**Volume/ liquidity**
Earnings Yield	ROE	12-1 monthly returns	5-y bear vol.	EPS revision	Market cap
Book yield	FCF/Assets	6 Months RSI	3-year correlation	EY FY1	Volume
Sales yield	GrossProfit/Capital employed	12-1 M. returns / vol.	Specific risk residual from PCA	EPS growth FY1	Liquidity at Risk

3. Building the model

In this Section, we introduce the machine learning models' hyper-parameters that we found of interest using our data.

3.1. Hyper-parameters

There are many different hyper-parameters in ML models, covering them all is outside the scope of the paper. We will confine our introduction to the parameters that we have tested or used along this classification cexercise. The list is the following:

Boosted trees:

- The learning rate, η: it is the step size shrinkage used in update to prevents overfitting. After each boosting step, we can directly get the weights of new features and η actually shrinks the feature weights to make the boosting process more conservative.
- The maximum depth: it is the longest path (in terms of nodes) from the root to a leaf of the tree. Increasing this value will make the model more complex and more likely to be overfitting.
- Regression λ: it is the L^2 regularization term on weights and increasing this value will make model more conservative.
- Number of rounds: maximum number of iterations (trees).

Random forests:

- mtry: the number of features that are kept when generating a random tree.
- Number of trees: total number of random trees that are aggregated in the model

Neural networks:

- Units: dimension of the layer.
- Batch size: number of samples per gradient update.
- Activation: function applied to the output of one layer before the next layer processes it.
- Epoch: number of iterations of learning (backpropagation).
- Drop out: fraction of inputs randomly set to zero to reduce the risk of overfitting.

Hyperparameters tuning is always a daunting task in ML, we use existing literature guidelines for establishing models baseline parameters for random forest and gradient boosting models. Regarding neural networks we create 2 models, one very simple and the other with more complexity.

The list of the different models and their respective parameters are gathered in Table 2.

TABLE 2: List of parameters and structure for ML models

Model	Parameters and structure
RF	Classification: mtry = 19, mtrees = 100
XGB	Classification: eta = 0.1, max depth = 5, lambda = 1, Nb rounds = 1000
NN1	Classification: 1 hidden layer. No drop-out. ReLu activation. Layer structure: 32-16-1 units
NN3	Classification: 3 hidden layers. Drop-out 40%. ReLu activation. Layer structure: 100-100-50-50-1 units

Random forest is the model with the lowest number of parameters to adjust. Consistent with Bernard et al. (2009), we use mtry = $\sqrt{F}$ F being the total number of features in the dataset. To define the number of trees we base our choice on Probst et al. (2017) which shows that most of the gain is obtained with 100 trees.

Concerning gradient boosted trees, we refer to Guida et al. (2018) that use grid search and cross validation to find the best vector of parameters in equity dataset. The hyperparameters for neural networks are chosen to define one rather shallow (NN1) and one deeper (NN3) network. The structures are comparable to those of Abe et al. (2018) and the orders of magnitudes are also similar to those of Gu et al. (2018).

3.2. Assessing the quality of the models

In the process of assessing the quality of the model, many different evaluation metrics are available. In this sub-section, we introduce the metrics that assess a machine learning model's quality. The classical metrics that assess the quality of models are:

Precision: **Tp / (Tp + Fp)**
Precision could be defined as a rate of successful prediction for sector neutral outperforming stocks.

Recall: **Tp / (Tp + Fn)**
Recall could be defined as a true rate, since we include the instances that have been wrongly classified in negative.

Accuracy: **(Tp + Tf) / (Tp+Tn+Fp+Fn)**
This is the accuracy level used in the cross-validation part.

F1 score: **2 * Precision * Recall / (Precision + Recall)**

With: Fp (false positive), Fn (false negative), Tp (true positive), Tn (true negative). Among the selected models, the outcome for the different evaluation metrics is the following:

TABLE 3: ML model's average quality metrics (in sample).

Model	Accuracy	Precision	Recall	F1
RF	82.58%	81.40%	80.50%	81.72%
XGB	84.19%	83.21%	83.61%	83.40%
NN3	82.19%	81.21%	81.61%	81.40%
NN1	80.40%	79.51%	79.07%	79.30%

Because we train on the tails (extreme quintiles) of the cross-sectional distribution, there is very little imbalance in the class: average metrics for recall, precision, F1 and accuracy are therefore very close.
All models exhibit a high level of quality in the training part, which is expected since we are training on the tails where the causal effect between characteristics and relative performance is clearer. The true question is whether or not these high scores will translate in out-of-sample profitability.

3.3. Variable importance: in-sample analysis

One common criticism against machine learning is the so-called "black-box"[9] nature of the prediction, as if it was impossible to understand or trace which feature or combination of features are responsible for the forecast. Ensemble learning with trees has a very nice feature that easily rules out this criticism: variable importance. In order to assess the variable importance for neural networks, which are mathematically and operationally more complicated, we use Local Interpretable Model agnostic Explanation (LIME). We encourage the interested reader to check the details in Ribeiro et al. (2016).

In tables 4, 5, 6 and 7, we display the top 30 score for variable importance (averaged each year) of our models that we trained and used for prediction every month from December 2005 until December 2017.[10] Each month, we keep the variable importance from each trained model. There are many metrics related to variable importance. In our

[9] Such criticism is not always justified, even for the most complex models. For instance, Neural Networks can be "white boxed" with 20 lines of Python.

[10] We provide on example to clarify the protocol: for the prediction of the end of November 2017, we rely on the features as of the beginning of Nov 2017 and use the model that has been trained using a 24 months dataset that ranged from early Nov 2014 until early Nov 2016.

exercise, we use the gain metric which is equal to the relative contribution (in terms of accuracy) to the model from the corresponding features. One can summarize the gain metric as a prediction usefulness indicator. All gain measures across features have been rescaled for visualization purposes and sum to 1.

TABLE 4. Average yearly variable importance for XGB model

Name	2006	2007	2008	2009	2010	2011	2012	2013	2014	2015	2016
adv12m	4.9%	2.5%	0.3%	3.3%	9.1%	10.8%	10.9%	10.6%	8.8%	6.8%	4.1%
mktcap	6.9%	2.9%	0.8%	0.9%	3.0%	6.9%	6.3%	5.2%	7.0%	8.1%	11.4%
cfy_fy1	7.3%	6.5%	5.7%	4.2%	5.2%	4.1%	3.3%	2.3%	0.8%	1.8%	1.3%
comp_val_sn	5.6%	3.7%	2.1%	3.1%	3.1%	3.6%	5.5%	5.7%	4.5%	3.2%	2.8%
cps	0.2%	0.3%	1.4%	1.3%	2.1%	5.5%	6.5%	7.3%	5.0%	4.0%	2.9%
share_turn	0.2%	4.6%	9.9%	7.7%	5.2%	3.7%	1.3%	0.9%	0.9%	0.5%	0.4%
ratio_bull_bear_3y	0.3%	0.2%	1.2%	2.2%	0.2%	0.5%	2.6%	5.2%	7.7%	6.6%	8.4%
adv60	0.2%	1.7%	7.1%	6.6%	3.1%	0.0%	0.1%	0.3%	0.3%	0.3%	1.0%
fundam_disp_pos_2y	4.1%	2.8%	3.2%	2.6%	1.6%	1.1%	1.4%	2.9%	2.5%	2.0%	0.8%
roe_estim	0.0%	1.2%	4.8%	2.3%	2.3%	3.2%	2.8%	0.7%	1.6%	0.7%	0.2%
comp_liq	5.1%	3.0%	2.7%	1.1%	3.1%	2.1%	0.0%	0.2%	0.1%	0.1%	0.2%
ey_fy2	0.6%	1.6%	1.4%	0.4%	2.1%	2.7%	2.4%	2.3%	1.7%	0.5%	0.1%
vol_termstruct	0.3%	0.7%	0.6%	0.2%	1.2%	0.4%	0.7%	2.2%	3.4%	3.0%	2.6%
roa_estim	0.0%	0.7%	1.2%	2.2%	0.5%	0.3%	1.9%	2.2%	2.0%	1.4%	2.1%
dy_fy1	0.0%	1.1%	1.5%	2.2%	0.6%	0.9%	1.3%	1.7%	2.0%	1.5%	1.7%
vol_disp_vol_5y	1.4%	0.5%	0.5%	0.8%	1.7%	1.9%	1.9%	1.5%	1.1%	0.5%	0.4%
pb	3.9%	1.3%	1.3%	0.9%	0.4%	0.8%	1.2%	0.8%	0.0%	0.7%	0.4%
correl_5y	0.0%	0.2%	0.9%	4.9%	2.3%	0.8%	0.2%	0.5%	0.3%	0.4%	0.6%
divyld	0.0%	0.0%	0.1%	0.7%	1.8%	2.1%	2.3%	1.7%	0.7%	0.1%	0.0%
pe	0.4%	1.6%	1.8%	0.1%	1.6%	1.1%	1.0%	0.8%	0.2%	0.4%	0.2%
noa	0.1%	0.8%	1.8%	0.5%	0.6%	0.9%	0.5%	1.7%	0.8%	0.4%	0.9%
dy_fy2	0.7%	1.8%	1.0%	1.4%	0.5%	0.2%	0.6%	1.1%	0.4%	0.7%	0.5%
ebitda _capex_interest	1.6%	2.1%	3.2%	1.2%	0.1%	0.1%	0.1%	0.1%	0.1%	0.0%	0.0%
gp_bv	2.6%	0.8%	0.4%	0.8%	0.8%	0.1%	0.0%	0.4%	0.1%	1.3%	0.8%
fcfyld	0.0%	0.0%	0.2%	1.6%	1.2%	0.7%	0.0%	0.4%	0.7%	1.4%	1.9%

TABLE 5. Average yearly variable importance for RF model

Name	2006	2007	2008	2009	2010	2011	2012	2013	2014	2015	2016
share_turn	0.4%	1.9%	4.0%	1.3%	0.4%	0.4%	0.4%	0.3%	0.3%	1.3%	0.8%
at_fixed_assets	0.7%	0.8%	0.8%	0.5%	0.3%	0.4%	0.6%	1.1%	3.5%	1.7%	1.0%
adv12m	1.0%	0.7%	0.8%	1.4%	2.2%	1.7%	0.8%	0.6%	0.5%	0.6%	0.4%
fundam_disp_pos_2y	2.1%	1.1%	0.9%	0.6%	0.5%	0.6%	0.8%	0.8%	0.9%	0.6%	0.5%
r11m	0.2%	0.2%	0.2%	0.6%	5.1%	1.6%	0.2%	0.2%	0.2%	0.2%	0.2%
mktcap	1.1%	0.6%	0.4%	0.5%	1.1%	2.0%	0.9%	0.5%	0.5%	0.5%	0.5%
bidask	0.2%	0.5%	0.3%	0.2%	0.3%	0.6%	0.5%	0.4%	1.6%	2.9%	1.0%
adv60	0.7%	0.7%	1.1%	1.5%	1.2%	1.0%	0.6%	0.4%	0.4%	0.3%	0.3%
comp_lvol_sn	0.3%	0.3%	0.3%	1.0%	0.7%	0.3%	0.6%	1.8%	0.4%	0.3%	1.9%
cps	0.4%	0.7%	1.3%	0.8%	0.4%	1.0%	0.9%	0.5%	0.4%	0.5%	0.4%
capture_bear_3y	0.9%	0.4%	0.4%	0.4%	0.4%	2.6%	0.4%	0.5%	0.3%	0.3%	0.7%
vol_disp_vol_5y	0.7%	0.6%	0.6%	0.7%	0.5%	0.4%	0.4%	0.5%	0.9%	0.9%	0.6%
vol5y	0.5%	0.4%	0.4%	0.7%	0.4%	1.0%	0.4%	0.7%	0.7%	0.5%	0.8%
effective_tax_rate	0.5%	1.0%	1.5%	0.4%	0.4%	0.4%	0.5%	0.4%	0.4%	0.4%	0.4%
return_totassets	0.4%	0.5%	0.5%	0.7%	0.9%	0.7%	0.7%	0.4%	0.5%	0.6%	0.7%
volstfx	0.3%	0.3%	0.3%	0.5%	0.5%	0.5%	0.3%	2.5%	0.7%	0.3%	0.3%
bb_yld	0.4%	0.4%	0.3%	0.3%	0.2%	0.2%	0.3%	0.6%	1.4%	1.7%	0.5%
noa	0.6%	0.7%	0.8%	0.4%	0.3%	0.6%	0.8%	0.6%	0.6%	0.5%	0.4%
vol3y_bear	0.4%	0.4%	0.4%	0.4%	0.6%	2.0%	0.4%	0.5%	0.4%	0.4%	0.4%
comp_lvol_5	0.2%	0.2%	0.2%	0.7%	0.5%	0.2%	0.6%	1.7%	0.3%	0.2%	1.1%
dy_fy1	0.4%	0.9%	1.2%	0.5%	0.3%	0.4%	0.5%	0.5%	0.5%	0.5%	0.4%
adv20	0.6%	0.6%	0.7%	1.1%	0.8%	0.6%	0.5%	0.3%	0.3%	0.3%	0.2%
correl_5y	0.6%	0.4%	0.5%	1.6%	0.4%	0.3%	0.3%	0.4%	0.4%	0.5%	0.4%
participation_bull_bear_5y	1.7%	1.3%	0.4%	0.3%	0.3%	0.2%	0.3%	0.4%	0.3%	0.2%	0.2%
volltfx	0.3%	0.3%	0.3%	0.3%	0.5%	1.3%	0.3%	1.1%	0.3%	0.3%	0.4%

TABLE 6. Average yearly variable importance for NN1 model

Name	2006	2007	2008	2009	2010	2011	2012	2013	2014	2015	2016
at_fixed_assets	2.4%	2.5%	2.5%	1.4%	0.9%	1.3%	2.4%	3.9%	8.0%	4.4%	2.8%
fundam_disp_pos_2y	5.6%	3.7%	2.9%	2.3%	2.0%	2.4%	2.6%	2.2%	2.6%	1.5%	2.1%
r11m	0.3%	0.2%	0.3%	2.3%	9.2%	5.0%	0.4%	0.3%	0.4%	0.2%	0.7%
share_turn	1.1%	4.3%	9.6%	4.1%	1.2%	0.8%	1.0%	0.6%	0.6%	1.8%	2.5%
adv12m	2.6%	1.7%	1.3%	3.0%	5.8%	4.1%	2.6%	2.1%	1.4%	0.7%	0.7%
mktcap	3.5%	1.9%	1.1%	1.2%	2.7%	4.9%	3.5%	1.5%	1.7%	1.5%	1.8%
cps	1.5%	2.4%	3.8%	3.0%	1.5%	2.9%	3.3%	1.8%	0.7%	1.1%	0.9%
vol_disp_vol_5y	2.9%	1.6%	1.6%	2.2%	1.2%	1.0%	1.2%	1.5%	3.3%	3.4%	2.0%
return_totassets	1.1%	1.5%	1.5%	2.8%	2.7%	2.5%	2.4%	1.1%	1.6%	1.5%	2.5%
vol_disp_vol_3y	2.2%	2.5%	1.9%	1.9%	1.2%	0.8%	1.4%	1.6%	2.3%	2.5%	2.0%
capture_bear_3y	2.2%	0.9%	0.8%	1.3%	1.5%	7.6%	1.3%	1.2%	0.6%	0.5%	1.5%
bidask	0.6%	1.0%	0.4%	0.2%	0.5%	1.6%	1.7%	1.0%	2.6%	5.5%	2.3%
effective_tax_rate	1.8%	3.1%	3.9%	1.0%	0.8%	1.0%	1.4%	1.3%	0.9%	1.0%	1.0%
volstfx	0.6%	0.7%	0.5%	1.5%	1.7%	0.6%	0.9%	8.0%	1.8%	0.4%	0.7%
noa	1.2%	1.8%	1.5%	1.0%	0.7%	1.3%	2.1%	1.5%	1.4%	2.1%	1.3%
vol3y_bear	1.5%	0.9%	0.9%	1.1%	1.5%	5.0%	0.9%	0.6%	0.8%	1.0%	1.3%
vol5y	1.1%	1.2%	1.3%	2.2%	1.1%	1.2%	1.0%	1.0%	2.0%	1.6%	2.4%
vol_fundam_sa_bs_pl	1.8%	2.4%	3.0%	1.0%	0.9%	1.1%	1.0%	1.2%	1.0%	0.9%	1.3%
correl_5y	1.4%	1.1%	1.3%	4.9%	1.1%	0.5%	0.8%	0.9%	0.7%	1.3%	1.4%
dy_fy2	0.6%	2.1%	3.2%	1.2%	0.7%	1.0%	1.6%	0.8%	1.0%	1.1%	1.3%
tot_liabilities_tot_assets	1.0%	1.1%	0.9%	1.1%	1.3%	1.2%	1.0%	1.4%	2.1%	2.0%	1.1%
comp_lvol_sn	0.5%	0.3%	0.4%	2.7%	1.8%	0.4%	0.8%	1.2%	0.8%	0.7%	4.4%
capture_bull_3y	0.7%	1.2%	2.0%	1.6%	1.1%	1.0%	1.3%	0.9%	0.9%	1.0%	1.2%
bb_yld	1.2%	0.8%	0.6%	0.5%	0.3%	0.5%	0.5%	1.5%	2.9%	2.9%	1.2%
pb	2.8%	1.2%	0.7%	0.6%	0.5%	0.5%	0.7%	0.8%	2.1%	1.8%	1.0%

TABLE 7. Average yearly variable importance for NN3 model

Name	2006	2007	2008	2009	2010	2011	2012	2013	2014	2015	2016
fundam_disp_pos_2y	2.8%	2.7%	2.5%	2.1%	2.4%	2.4%	2.1%	1.8%	1.9%	1.9%	2.0%
vol_disp_vol_5y	2.5%	1.8%	2.0%	2.1%	1.7%	1.8%	1.6%	2.0%	2.5%	2.4%	2.2%
mktcap	2.4%	1.8%	1.5%	1.6%	2.3%	2.5%	2.5%	2.1%	1.9%	1.7%	1.8%
cps	1.8%	2.1%	2.3%	2.0%	2.1%	2.5%	2.3%	2.0%	1.4%	1.4%	1.5%
return_totassets	1.5%	1.9%	1.9%	1.9%	2.0%	2.2%	1.9%	1.9%	1.9%	1.8%	2.0%
effective_tax_rate	1.8%	2.1%	2.0%	1.6%	1.5%	1.5%	1.6%	1.7%	1.7%	1.5%	1.6%
at_fixed_assets	1.4%	1.5%	1.7%	1.5%	1.2%	1.7%	1.9%	1.8%	1.9%	1.8%	1.7%
capture_bull_3y	1.3%	1.7%	1.9%	1.8%	1.5%	1.3%	1.4%	1.2%	1.4%	1.4%	1.5%
adv12m	1.8%	1.4%	1.1%	1.2%	2.0%	2.0%	1.5%	1.8%	1.4%	1.2%	1.0%
workingcap_totassets	1.6%	1.5%	1.7%	1.2%	1.2%	1.4%	1.5%	1.3%	1.3%	1.3%	1.5%
share_turn	1.3%	1.4%	1.8%	1.8%	1.6%	1.4%	1.2%	1.2%	1.0%	1.3%	1.4%
correl_1y	1.4%	1.2%	1.1%	1.9%	1.6%	1.1%	1.2%	1.4%	1.7%	1.6%	1.1%
eps_growth_12m_disp	0.4%	1.3%	1.3%	1.5%	1.1%	1.4%	1.3%	1.7%	1.9%	1.6%	1.7%
intangible_assets_revenue	1.3%	1.3%	1.2%	1.4%	1.2%	1.2%	1.6%	1.4%	1.5%	1.3%	1.5%
correl_5y	1.5%	1.5%	1.5%	1.9%	1.2%	0.8%	0.9%	1.2%	1.0%	1.5%	1.7%
payables_receivables	1.2%	1.3%	1.2%	1.4%	1.3%	1.3%	1.4%	1.4%	1.4%	1.2%	1.4%
tot_liabilities_tot_assets	1.0%	1.2%	1.2%	1.1%	1.4%	1.4%	1.2%	1.3%	1.5%	1.7%	1.2%
capture_bear_3y	1.5%	1.1%	1.1%	1.3%	1.8%	2.4%	1.3%	1.2%	0.8%	0.9%	0.9%
vol_termstruct	1.3%	1.1%	1.0%	1.1%	1.6%	1.6%	1.2%	1.1%	1.1%	1.2%	1.3%
pb	1.7%	1.4%	1.0%	1.0%	1.0%	1.0%	1.0%	1.2%	1.5%	1.4%	1.2%
psales	1.2%	1.4%	1.1%	1.1%	1.0%	1.4%	1.5%	1.2%	1.2%	1.1%	1.0%
net debt	1.1%	1.1%	1.2%	1.4%	1.5%	1.2%	1.0%	1.1%	1.1%	1.1%	1.0%
grprofitgrowth	1.0%	0.9%	1.0%	1.0%	1.1%	1.5%	1.7%	1.3%	1.2%	1.2%	1.1%
risk_spe_5y	1.1%	1.0%	1.4%	1.4%	1.6%	1.2%	0.9%	0.9%	0.9%	1.1%	1.1%
vol3y_bear	1.4%	0.9%	0.9%	1.3%	1.3%	1.5%	0.9%	0.9%	1.0%	1.3%	1.2%

First, we observe that on average there is not a high concentration in the features explaining more than a 1/3rd of the total prediction's importance. Then, looking at the type of features one can note that:

- We do have features coming from the 6 different metrics families gathered for all our models.
- We acknowledge some very common, well-known and over-researched characteristics mentioned in the asset pricing literature (book-yield for value, market cap for size, profitability on assets for quality, price volatility for low volatility anomaly and 12-1 month momentum).
- Despite our resorting to very different ML models, we find a lot of common features among the top 30. Characteristics such as market cap, average daily volume in amount, cash flow yield, momentum 12-1 month and others are common in all top 30 variable importance. 15 metrics are in common in 3 models.

4. Results and discussion

We now proceed to a use case. Our use case will test our ML-based signals as a base for constructing equally-weighted portfolios. We process our probability of sector outperformance just like any other signal. We normalize it, express it in percentile and assess the performance of monthly-rebalanced decile portfolios.[11] Each month, the model is trained and the current features (characteristics) are fed to the model so that it forecasts the probability of sectorial out-performance for each stock. Then, we form 10 portfolios accordingly: the first (*resp.*, last) portfolio consists of the 10% of stocks that are most (*resp*., least) likely to outperform their sector. The weighting in portfolios is uniform (equal weights for all stocks).

As benchmark, we construct a basic multi-factor signal using the following characteristics:

1. **Value**: earnings yield, book yield, EV/EBITDA.
2. **Quality**: return on equity, debt/equity.
3. **Momentum**: 12-1 total return performance.
4. **Low volatility**: 3 years and 1-year price volatility.
5. **Size**: market capitalization.

[11] Such portfolio sorting procedures are commonplace since the seminal work of Fama and French (1992).

4.1. Decile performance analysis

In Table 8, we compute the annual return in USD, gross of transaction costs, of equal-weighted portfolios using our ML signals versus the multi-factor benchmark (right column). In addition, we compute the hypothetical cash neutral long-short performance as another metric of comparison as the simple subtraction of D10 minus D1.

TABLE 8: Decile performance analysis for all signals

	XGB	RF	NN1	NN3	Mfactor
D1	0.8%	3.1%	0.2%	0.5%	0.2%
D2	1.8%	4.1%	2.8%	2.0%	1.3%
D3	2.5%	2.7%	2.4%	1.7%	1.9%
D4	4.1%	3.1%	2.3%	2.4%	2.9%
D5	4.0%	4.9%	3.9%	3.4%	3.6%
D6	3.9%	3.2%	4.7%	6.0%	3.3%
D7	4.1%	3.9%	5.9%	5.9%	5.1%
D8	5.5%	3.7%	5.2%	4.9%	4.9%
D9	5.3%	4.6%	5.7%	6.8%	4.9%
D10	7.3%	5.4%	6.1%	5.5%	5.1%
L/S	6.5%	2.3%	5.9%	5.0%	4.9%

We report that the monotonic effect is present for most of the signals and less pronounced at the decile level for the Random Forest model. On the long-short side, the XGB model dominates, followed by the shallow NN1 model and the Mfactor. On the short side (decile 1) the NN1 and the Mfactor signal managed to accurately identify underperforming stocks while on the long side (decile 10) the XGB (by far) and the NN1 have the strongest annualized performance.

4.2. Long only portfolios analysis

In this analysis we continue our comparison by constructing more realistic long-only portfolio and looking at some classic portfolio performance analytics. We use our 4 different ML signals across 3 different weighting schemes (capitalization weighting, equal weighting and risk weighting) to construct 12 portfolios rebalanced monthly. We also construct 3 different benchmarks using the multi-factor signal, one for each weighting scheme.

We seek to make the exercise as close as possible to the practices of asset management industry. Hence, we impose some classic constraints on each of the portfolios. The maximum weight per position is 3% while minimum weight is 10 basis points. We exclude all stocks below 100 millions USD market capitalization and we constrain the sector exposure to a maximum of 40%.

In Table 9, we can see different analytics comparing the 15 different portfolios.

TABLE 9: **Strategies comparison**

Strategies	Return p.a.	volatility	perf/risk ratio	Net Return p.a.	Net perf/risk ratio	Maximum Drawdown	Historical VaR (95%)	Historical ES (95%)	Turnover
XGB_CW	**3.9%**	18.0%	**0.21**	2.51%	0.14	60.4%	-6.8%	-12.7%	273%
NN1_CW	2.7%	18.9%	0.14	1.41%	0.07	62.9%	-7.4%	-14.8%	262%
RF_CW	3.2%	18.7%	0.17	1.91%	0.10	58.9%	-7.5%	-13.8%	264%
NN3_CW	3.9%	18.6%	0.21	**2.68%**	**0.14**	59.3%	-8.7%	-14.0%	241%
Mfactor_CW	2.8%	13.8%	0.20	2.22%	0.16	50.4%	-7.7%	-9.7%	116%
XGB_EW	**7.3%**	19.5%	**0.38**	**5.84%**	**0.30**	60.6%	-7.7%	-14.4%	296%
NN1_EW	6.1%	19.0%	0.32	4.91%	0.26	58.5%	-8.4%	-13.7%	242%
RF_EW	5.4%	18.7%	0.29	4.15%	0.22	57.4%	-7.6%	-14.1%	248%
NN3_EW	5.5%	19.5%	0.28	4.11%	0.21	59.9%	-8.3%	-14.5%	280%
Mfactor_EW	5.1%	15.6%	0.32	4.34%	0.28	55.0%	-7.7%	-10.6%	143%
XGB_RW	**7.3%**	17.4%	**0.42**	**5.81%**	**0.33**	57.3%	-6.9%	-12.9%	250%
NN1_RW	6.4%	17.1%	0.37	5.42%	0.32	55.7%	-7.5%	-12.5%	188%
RF_RW	5.4%	17.4%	0.31	4.50%	0.26	55.5%	-7.2%	-13.2%	189%
NN3_RW	5.5%	17.5%	0.31	4.51%	0.26	56.2%	-7.5%	-13.1%	193%
Mfactor_RW	4.7%	13.7%	0.34	4.22%	0.31	51.2%	-6.8%	-9.6%	101%

In the cross-section of models, XGB dominates the others on all weighting schemes - on a pure performance standpoint. On a net performance basis, when subtracting transaction costs, XGB outperformed the other model on 2 out 3 weighting schemes. There is no real difference in terms of volatility profile for the ML strategies. Irrespective of the weighting scheme, they exhibit higher level of volatility than the Mfactor strategies. These volatilities are in line with market benchmark volatility levels that are close to 17% for the last 11 years.[12] The levels of annual turnover are comparable between the different strategies and are more driven by the weighting scheme. This explains why risk-weights generate the smallest levels of turnover.

[12] See, e.g., the MSCI AC World Index Total return - in USD.

Nevertheless, these annual turnover levels remain in the range of active equity strategies (between 150-300% turnover per year).

5. Conclusion

In this paper, we apply well-known ML algorithms to systematic equity investing. Our methodology requires a critical stage of feature and label engineering. This step helps the algorithms uncover hidden structures in the equity market space. Only then can a modern quantitative approach make accurate long-term predictions. Our insightful findings contradict recent criticisms that ML-based approaches were only suitable when predicting very short-term prices movements.

We find that a naïve equally-weighted long short portfolio using boosted tree algorithm with 350 features generates an average outperformance of 1.6% (net of transaction costs), compared to a classic multi-factor portfolio. Our results also suggest that neural networks of modest sizes show promising results and we leave such explorations for future research.

A more industry-focused use case further confirms the potential of ML algorithms implemented in long only portfolios. The strategies we propose outperform classic multi-factor allocations across 3 different common weighting schemes, such as cap weighting, equal weighting and risk weighting. Nonetheless, discrepancies between forecasting tools highlight the need for further investigation in these directions.

In a context of constant whistleblowing about potential crowding in systematic risk premia equity, ML algorithms emerge as one, if not *the* solution against the risk of commoditization for equity multi-factor portfolios. We believe this field, at the crossroad of finance, data science and statistical learning, will serve as playground for academics and practitioners alike in the decades to come.

Bibliography

Abe M., Nakayama H. (2018) Deep Learning for Forecasting Stock Returns in the Cross-Section. In: Phung D., Tseng V., Webb G., Ho B., Ganji M., Rashidi L. (eds) *Advances in Knowledge Discovery and Data Mining. PAKDD 2018. Lecture Notes in Computer Science*, vol 10937. Springer, Cham

Adeodato, P. J., Arnaud, A. L., Vasconcelos, G. C., Cunha, R. C., & Monteiro, D. S. (2011). MLP ensembles improve long term prediction accuracy over single networks. *International Journal of Forecasting*, 27(3), 661-671.

Ang, A. (2014). *Asset management: A systematic approach to factor investing*. Oxford University Press.

Arévalo, R., García, J., Guijarro, F., & Peris, A. (2017). A dynamic trading rule based on filtered flag pattern recognition for stock market price forecasting. *Expert Systems with Applications*, *81*, 177-192.

Ballings, M., Van den Poel, D., Hespeels, N., & Gryp, R. (2015). Evaluating multiple classifiers for stock price direction prediction. *Expert Systems with Applications*, *42*(20), 7046-7056.

Ban, G. Y., El Karoui, N., & Lim, A. E. (2016). Machine learning and portfolio optimization. *Management Science*, 64(3), 1136-1154.

Banz, R. W. (1981). The relationship between return and market value of common stocks. *Journal of Financial Economics*, *9*(1), 3-18.

S. Bernard, L. Heutte, and S. Adam (2009). Influence of hyperparameters on random forest accuracy. In MCS, volume 5519 of *Lecture Notes in Computer Science*, Springer, 171–180.

Bodnar, T., Mazur, S., & Okhrin, Y. (2017). Bayesian estimation of the global minimum variance portfolio. *European Journal of Operational Research*, *256*(1), 292-307.

Cao, L. J., & Tay, F. E. H. (2003). Support vector machine with adaptive parameters in financial time series forecasting. *IEEE Transactions on neural networks*, 14(6), 1506-1518.

Chen, T., & Guestrin, C. (2016). XGBoost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (pp. 785-794). ACM.

Chollet, F. (2017). *Deep learning with Python*. Manning Publications Co.

Daniel, K., & Titman, S. (1997). Evidence on the characteristics of cross-sectional variation in stock returns. *Journal of Finance*, *52*(1), 1-33.

De Prado, M. L. (2018). The 10 Reasons Most Machine Learning Funds Fail. *Journal of Portfolio Management*, 44(6), 120-133.

Dunis, C. L., Laws, J., & Evans, B. (2008). Trading futures spread portfolios: applications of higher order and recurrent networks. *European Journal of Finance*, 14(6), 503-521.

Dunis, C. L., Likothanassis, S. D., Karathanasopoulos, A. S., Sermpinis, G. S., & Theofilatos, K. A. (2013). A hybrid genetic algorithm–support vector machine approach in the task of forecasting and trading. *Journal of Asset Management*, 14(1), 52-71.

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

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

Feng, G., Giglio, S., & Xiu, D. (2017). Taming the factor zoo. SSRN *Working Paper*.

Fischer, T., & Krauss, C. (2018). Deep learning with long short-term memory networks for financial market predictions. *European Journal of Operational Research*.

Freund, Y., & Schapire, R. E. (1997). A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, *55*(1), 119-139.

Friedman, J., Hastie, T., & Tibshirani, R. (2000). Additive logistic regression: a statistical view of boosting (with discussion and a rejoinder by the authors). *Annals of Statistics*, *28*(2), 337-407.

Friedman, J. (2001). Greedy function approximation: a gradient boosting machine. *Annals of Statistics*, 1189-1232.

Friedman, J., Hastie, T., & Tibshirani, R. (2009). *The Elements of Statistical Learning* (2nd Edition). Springer.

Goodfellow, I., Bengio, Y., Courville, A., & Bengio, Y. (2016). *Deep Learning.* Volume 1. MIT Press.

Goto, S., & Xu, Y. (2015). Improving mean variance optimization through sparse hedging restrictions. *Journal of Financial and Quantitative Analysis*, 50(6), 1415-1441.

Green, J., Hand, J. R., & Zhang, X. F. (2013). The supraview of return predictive signals. *Review of Accounting Studies*, *18*(3), 692-730.

Gu, S., Kelly, B. T., & Xiu, D. (2018). Empirical Asset Pricing Via Machine Learning. *SSRN Working Paper*.

Guida, T. (2018). *Big Data and Machine Learning in Quantitative Investment*. Wiley.

Hastie, T. J., & Tibshirani, R. J. (1990). *Generalized additive models*, Volume 43 of Monographs on Statistics and Applied Probability. Chapman & Hall.

Harvey, C. R., Liu, Y., & Zhu, H. (2016). … and the cross-section of expected returns. *Review of Financial Studies*, *29*(1), 5-68.

Heaton, J. B., Polson, N. G., & Witte, J. H. (2017). Deep learning for finance: deep portfolios. *Applied Stochastic Models in Business and Industry*, 33(1), 3-12.

Ho, T. K. (1995, August). Random decision forests. In *Document analysis and recognition, 1995., proceedings of the third international conference on Document Analysis and Recognition* (Vol. 1, pp. 278-282). IEEE.

Ilmanen, A. (2011). *Expected returns: An investor's guide to harvesting market rewards*. John Wiley & Sons.

Jegadeesh, N., & Titman, S. (1993). Returns to buying winners and selling losers: Implications for stock market efficiency. *Journal of Finance*, *48*(1), 65-91.

Jegadeesh, N., & Titman, S. (2001). Profitability of momentum strategies: An evaluation of alternative explanations. *Journal of Finance*, *56*(2), 699-720.

Kahn, R. N., & Lemmon, M. (2016). The asset manager's dilemma: How smart beta is disrupting the investment management industry. *Financial Analysts Journal*, *72*(1), 15-20.

Krauss, C., Do, X. A., & Huck, N. (2017). Deep neural networks, gradient-boosted trees, random forests: Statistical arbitrage on the S&P 500. *European Journal of Operational Research*, *259*(2), 689-702.

Matías, J. M., & Reboredo, J. C. (2012). Forecasting performance of nonlinear models for intraday stock returns. *Journal of Forecasting*, 31(2), 172-188.

McLean, R. D., & Pontiff, J. (2016). Does academic research destroy stock return predictability? *Journal of Finance*, *71*(1), 5-32.

Moritz, B., & Zimmermann, T. (2016). Tree-based conditional portfolio sorts: The relation between past and future stock returns. *SSRN Working Paper*.

Nair, B. B., Kumar, P. S., Sakthivel, N. R., & Vipin, U. (2017). Clustering stock price time series data to generate stock trading recommendations: An empirical study. *Expert Systems with Applications*, *70*, 20-36.

Patel, J., Shah, S., Thakkar, P., & Kotecha, K. (2015). Predicting stock and stock price index movement using trend deterministic data preparation and machine learning techniques. *Expert Systems with Applications*, *42*(1), 259-268.

Probst, P., Bischl, B., & Boulesteix, A. L. (2018). Tunability: Importance of hyperparameters of machine learning algorithms. *arXiv preprint.*

Probst, P. and Boulesteix, A.-L., (2017). To tune or not to tune the number of trees in random forest? *ArXiv preprint*.

Raffinot, T. (2018). Hierarchical Clustering-Based Asset Allocation. Journal of Portfolio Management, 44 (2) 89-99.

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

Ribeiro M.T., Singh S., Guestrin C. (2016) "Why Should I Trust You?": Explaining the Predictions of Any Classifier, *arXiv preprint*

Schapire, R. E. (1990). The strength of weak learnability. *Machine Learning*, *5*(2), 197-227.

Subrahmanyam, A. (2010). The Cross-Section of Expected Stock Returns: What Have We Learnt from the Past Twenty-Five Years of Research? *European Financial Management*, *16*(1), 27-42.

Van Dijk, M. A. (2011). Is size dead? A review of the size effect in equity returns. *Journal of Banking & Finance*, *35*(12), 3263-3274.