Master Thesis

Revisiting Equity Strategies with Financial Machine Learning

Luca Schneider

Department of Management, Technology and Economics
Chair of Entrepreneurial Risks

Supervisor:
Prof. Dr. Didier Sornette

September 2019
Acknowledgements

First of all, I would like to express my deep gratitude to my thesis supervisor, Professor Sornette, for this research opportunity, his patience, availability and his suggestions that contributed to my reflection.

I also thank Samuel Fux and Loher Gerson from the IT-Services of ETH Zurich for providing the computational resources for this thesis.

I would also like to extend my thanks to my previous employer, Banque Paris Bertrand SA, and its Systemic Asset Management team for providing the raw data and the expertise in quantitative portfolio management.

I wish to acknowledge the patience and continuing support from my partner, Jasmin Rose Lewis, and my relatives who have made the achievement of this thesis and the Master’s degree possible.

Finally, a big thank you to my student colleagues from ETH, Timothée Barattin, Martin Butterscotch, Emmanuel Profumo and Cédric Travelletti, for their extensive advice in code development and statistics, and their support throughout the writing of this thesis.
Abstract

With the recent trend around Artificial Intelligence and Machine Learning, these techniques have been widely used by academics and practitioners to forecast financial markets. With the most modern concepts of financial machine learning and data science, this paper attempt to derive a sustainable and interpretable Equity strategy with common Equity strategies signals as input. This paper also reviews numerous challenges and issues encountered in financial machine learning application as early modelling have been compromised by data leakage despite cautious procedures. After remedying it, the strategies proposed remain profitable and still beats the Russell 1000 Index, including transaction costs.
# Contents

1 Introduction ...................................................................................................................... 1

2 Thesis Outline ................................................................................................................... 3

3 Essential Concepts in Financial Machine Learning ..................................................... 8

4 Feature Engineering ...................................................................................................... 11

4.1 Context ....................................................................................................................... 11

4.2 Feature Cleaning ........................................................................................................ 15

4.3 Exploratory Data Analysis ........................................................................................ 16

4.4 Initial Feature Selection ............................................................................................. 21

5 Predictive Modelling ...................................................................................................... 25

5.1 Context ....................................................................................................................... 25

5.2 Primary Model ........................................................................................................... 34

5.3 Feature Selection ....................................................................................................... 41

5.4 Data Leakage ............................................................................................................. 44

5.5 Pipeline Performances ............................................................................................... 47

6 Portfolio Allocation and Portfolio Analysis .................................................................. 48

6.1 Context ....................................................................................................................... 48

6.2 Results ........................................................................................................................ 50

7 Conclusion and Further Research .............................................................................. 55

References ............................................................................................................................... 58

Appendix A: Python and Dataset Files .............................................................................. 70

Appendix B: Downloading Bloomberg data with Python ................................................ 71

Appendix C: List of Bloomberg Fields downloaded .......................................................... 73

Appendix D: Dataset Cleaning: a Detailed Description .................................................... 76

Appendix E: List of Features ............................................................................................ 79

Appendix F: Equity Strategies Computation ...................................................................... 87

Appendix G: Portfolio Statistics Computation .................................................................. 112

Appendix H: AutoML and Deep Learning ........................................................................ 115
List of Figures

Figure 2.1 Graphical illustration of the trading framework ............................................................ 4
Figure 2.2 Graphical illustration of the machine learning pipeline .............................................. 5
Figure 4.1 Binary representation of missing values projected in the first two dimensions .......... 13
Figure 4.2 Visualization of missing data patterns over the years .............................................. 14
Figure 4.3 Visualizations of the absolute and relative differences of SIDs over the years after feature cleaning ................................................................. 16
Figure 4.4 Variability summarized across the first 20 components of the FAMD ....................... 17
Figure 4.5 Summary analysis of the first and second components of the FAMD ....................... 19
Figure 4.6 Summary analysis of the third and fourth components of the FAMD ....................... 20
Figure 4.7 Visualization of the quantitative predictors’ correlation matrix ................................ 22
Figure 4.8 Visualization of the uncertainty coefficient matrix .................................................. 24
Figure 5.1: Graphical illustration of the meta-labelling approach ................................................ 25
Figure 5.2: Graphical illustration of the data splitting and the training methodology used ......... 30
Figure 5.3: Graphical illustration of the Repeated Holdout validation ........................................ 31
Figure 5.4: Comparison of training methodologies: Means and standard deviations of metrics computed monthly ................................................................. 36
Figure 5.5: Comparison of training methodologies ....................................................................... 37
Figure 5.6: Comparison of training with missing values: Means and standard deviations of metrics computed monthly ................................................................. 38
Figure 5.7: Comparison of training with missing values ............................................................... 38
Figure 5.8: Comparison of training with rolling pre-processed data: Means and standard deviations of metrics computed monthly ................................................................. 39
Figure 5.9: Comparison of training with rolling pre-processed data: ....................................... 40
Figure 5.10: Predictor selection percentage for selected feature subsets .................................... 41
Figure 5.11: The cross-validated results for RFE and SFS using GBT ........................................ 42
Figure 5.12: Comparison of training with different feature subsets: Means and standard deviations of metrics computed monthly ................................................................. 43
Figure 5.13: Graphical illustration of data leakage ..................................................................... 45
Figure 5.14: Comparison of pipeline performances on the validation set: Means and standard deviations of metrics computed monthly ................................................................. 47
Figure 6.1 Empirical Sharpe Ratio distribution for strategies ..................................................... 50
Figure 6.2 Top panel: Out-Sample cumulative returns for strategies and benchmarks ............. 52
Figure 6.3 Top panel: Out-Sample ratio of the strategies cumulative returns over the Russell 1000 Index cumulative returns ................................................................. 53
List of Tables

Table 4.1 Association between qualitative variables. ............................................................. 23
Table 6.1 Strategies performances for out-of-sample data. ....................................................54
Abbreviations

AI  Artificial Intelligence
ANN  Artificial Neural Network
API  Application Programming Interface
AUC  Area Under the receiver operating characteristic Curve
AutoML  Automated Machine Learning
B/P  Book-to-Price
BDH  Bloomberg Data History
BDP  Bloomberg Data Point
BDS  Bloomberg Data Set
BICS  Bloomberg Industry Classification System
BLPAPI  Bloomberg API
BQL  Bloomberg Query Language
CA  Correspondence analysis
CAGR  Compound Annual Growth Rate
CPCV  Combinatorial Purged Cross-Validation
CUSIP  Committee on Uniform Security Identification Procedures
CV  Cross-Validation
DART  Dropouts meet Multiple Additive Regression Trees
DL  Deep Learning
DT  Decision Tree
EDA  Exploratory Data Analysis
EMH  Efficient-Market Hypothesis
EPS  Earnings Per Share
FAMD  Factor Analysis of Mixed Data
FPR  False Positive Rate
GA  Genetic Algorithm
GBT  Gradient Tree Boosting
GPGPU  General-Purpose GPU programming
GPU  Graphical Processor Unit
HDF5  Hierarchical Data Format
<table>
<thead>
<tr>
<th>Acronym</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>IID</td>
<td>Independent and identically distributed random variables</td>
</tr>
<tr>
<td>IPO</td>
<td>Initial Public Offering</td>
</tr>
<tr>
<td>ISIN</td>
<td>International Securities Identification Number</td>
</tr>
<tr>
<td>JSON</td>
<td>JavaScript Object Notation</td>
</tr>
<tr>
<td>$k$-NN</td>
<td>$k$-Nearest Neighbors</td>
</tr>
<tr>
<td>LASSO</td>
<td>Least Absolute Shrinkage and Selection Operator</td>
</tr>
<tr>
<td>LR</td>
<td>Logistic Regression</td>
</tr>
<tr>
<td>LSTM</td>
<td>Long Short-Term Memory</td>
</tr>
<tr>
<td>MA</td>
<td>Moving Average</td>
</tr>
<tr>
<td>MAE</td>
<td>Mean Absolute Error</td>
</tr>
<tr>
<td>MAR</td>
<td>Missing At Random</td>
</tr>
<tr>
<td>MCA</td>
<td>Multiple Correspondence Analysis</td>
</tr>
<tr>
<td>MCAR</td>
<td>Missing Completely At Random</td>
</tr>
<tr>
<td>MCFD</td>
<td>Mean Correct Forecast Directions</td>
</tr>
<tr>
<td>MedAE</td>
<td>Median Absolute Error</td>
</tr>
<tr>
<td>MFTR</td>
<td>Mean Forecast Trading Returns</td>
</tr>
<tr>
<td>MI</td>
<td>Mutual Information</td>
</tr>
<tr>
<td>ML</td>
<td>Machine Learning</td>
</tr>
<tr>
<td>M&amp;A</td>
<td>Mergers and Acquisitions</td>
</tr>
<tr>
<td>NA</td>
<td>Non-Available</td>
</tr>
<tr>
<td>NB</td>
<td>Naive Bayes</td>
</tr>
<tr>
<td>NLP</td>
<td>Natural Language Processing</td>
</tr>
<tr>
<td>NMAR</td>
<td>Not Missing At Random</td>
</tr>
<tr>
<td>NYSE</td>
<td>New York Stock Exchange</td>
</tr>
<tr>
<td>OLS</td>
<td>Ordinary Least Squares</td>
</tr>
<tr>
<td>PCA</td>
<td>Principal Component Analysis</td>
</tr>
<tr>
<td>PIT</td>
<td>Point-In-Time</td>
</tr>
<tr>
<td>Rep-Holdout</td>
<td>Repeated Holdout validation</td>
</tr>
<tr>
<td>RFE</td>
<td>Recursive Feature Elimination</td>
</tr>
<tr>
<td>RF</td>
<td>Random Forests</td>
</tr>
<tr>
<td>RL</td>
<td>Reinforcement Learning</td>
</tr>
<tr>
<td>RNN</td>
<td>Recurrent Neural Networks</td>
</tr>
<tr>
<td>Acronym</td>
<td>Description</td>
</tr>
<tr>
<td>---------</td>
<td>-------------</td>
</tr>
<tr>
<td>ROC</td>
<td>Receiver Operating characteristic Curve</td>
</tr>
<tr>
<td>SA</td>
<td>Simulated Annealing</td>
</tr>
<tr>
<td>SBS</td>
<td>Sequential Backward Selection</td>
</tr>
<tr>
<td>SDK</td>
<td>Software Development Kit</td>
</tr>
<tr>
<td>SHAP</td>
<td>SHapley Additive exPlanation</td>
</tr>
<tr>
<td>SID</td>
<td>Security Identifier</td>
</tr>
<tr>
<td>SUE</td>
<td>Standardized Unexpected Earnings</td>
</tr>
<tr>
<td>SVM</td>
<td>Support-Vector Machine</td>
</tr>
<tr>
<td>TPE</td>
<td>Tree-structured Parzen Estimator</td>
</tr>
<tr>
<td>TPR</td>
<td>True Positive Rate</td>
</tr>
</tbody>
</table>
Notation
This section provides a concise reference describing the notation used throughout this thesis.

Number & Arrays
\[ a \] A scalar
\[ A \] A scalar constant
\[ {\vec{a}} \] A vector
\[ {\mathbf{A}} \] A matrix
\[ a \] A scalar random variable
\[ {\vec{a}} \] A vector-valued random variable
\[ {\mathbf{A}} \] A matrix-valued random variable

Indexing
\[ a_i \] Element \( i \) of vector \( a \), with indexing starting at 1
\[ a_{-i} \] All elements of vector \( a \) except for element \( i \)
\[ A_{i,j} \] Element \( (i,j) \) of matrix \( A \)
\[ A_{i,:} \] Row \( i \) of matrix \( A \)
\[ A_{:,j} \] Column \( i \) of matrix \( A \)

Sets
\[ \{0,1\} \] The set containing 0 and 1
\[ \{0,1,\ldots,n\} \] The set of all integers between 0 and \( n \)
\[ [a,b] \] The real interval including \( a \) and \( b \)
\[ (a,b] \] The real interval excluding \( a \) but including \( b \)

Calculus
\[ f'(a) \] Derivative of \( f \) at input point \( a \)
\[ \int_{\mathbb{S}} f(x) \, dx \] Definite integral with respect to \( x \) over the set \( \mathbb{S} \)

Probability & Information Theory
\[ P(a | b) \] Short hand for the probability \( P(a = a | b = b) \)
\[ P(a) \] A probability distribution over a discrete variable \( a \)
\[ p(a) \] A probability distribution over the continuous variable \( a \)
\[ H(x) \] Shannon entropy of the random variable \( x \)
Datasets and Machine Learning

- $\mathbf{X}$: A set of training examples
- $\mathbf{x}^{(i)}$: The $i$-th example from dataset
- $y^{(i)}$ or $y^{(i)}$: The target associated with $\mathbf{x}^{(i)}$ for supervised learning
- $\mathbf{X}$: The $m \times n$ matrix with input example $\mathbf{x}^{(i)}$ in row $\mathbf{X}_{i,:}$
- $P(\mathbf{x}, y)$: A data generating distribution
- $\mathcal{F}$: Hypothesis space of function to be learnt, i.e., a model
- $(x', y')$: A testing pair
- $\hat{y}$: Label predicted by a function $f$, i.e., $\hat{y} = f(x')$

Functions

- $f(\mathbf{x}, \theta)$: A function of $\mathbf{x}$ parametrized by $\theta$
- $\ln x$: Natural logarithm of $x$
- $1_{\text{condition}}$: is 1 if the condition is true, 0 otherwise
1 Introduction

Financial markets have always been leading to a scientific and technological arms race among its participants. After the eras of trend-following, of statistical arbitrage, and high frequency-trading, it seems that quantitative trading has been transitioning into a new era recently, with the democratization of Artificial Intelligence and Machine learning brought by digital companies (Sirotuyk, 2018). However, nor machine learning or trying to forecast financial markets with it are new. The academic literature abounds of application of different models and techniques. (Cavalcante, Brasileiro, Souza, Nobrega, & Oliveira, 2016) provides a comprehensive review of techniques used by publication ranging from 2009 to 2015. More recently, (Ryll & Seidens, 2019) perform a meta-analysis on more than 150 articles related financial machine learning published from 1995 to 2018. In addition to rank deep learning models higher than other models, the authors observed the lack of standards shared in financial machine learning as compared to the rest of the machine learning space.

Besides, the exercise of financial machine learning differs sensibly from the other machine learning areas due to the nature of financial markets. Main cited reasons are:

- **Size of datasets in finance**: In computer vision, a single 3-channels picture of 224 pixels format provides already more than 150’528 bits of information, as much as 150 years of OLHCV data for a single stock.

- **Non-Stationarity**: Machine learning theories have been formulated with the assumption of constant data pattern, while financial markets are known for their adaptivity and regime changes (W. Lo, 2017).

- **Noise to signal-ratio**: Machine learning algorithms will always identify a pattern even it is a false signal (Black, 1986).

- **Decision space**: In comparison to the main applications of Reinforcement Learning (RL) (Sato, 2019) in decision space with defined rules, finance involves the choice between numerous assets classes, markets,… without forgetting external uncertainties to be considered such as political or regulatory events.

As such, defining a trading strategy with machine learning is far from trivial due to the challenging nature of the finance area. Recently published, (López de Prado, 2018) laid the ground for the rigorous application of machine learning in real investment scenario cases. As such, this thesis relies on such advanced methods to attempt to derive a sustainable trading strategy from well-known equity anomalies.
The rest of this thesis is organized as follows. Section 2 outlines the thesis workflow and describes computational tools and data used. Section 3 reviews common bias encountered in strategy backtesting. Section 4 describes data exploration and early data pre-processing. Section 5 sets the context of the machine learning models employed for classifying stock returns and deals with issue of data leaking. Section 6 translates predictions into tradable signals before backtesting the final strategies. Finally, concluding remarks are presented in Section 7.
2 Thesis Outline

Context
This thesis focuses on the study of the predictability of stock returns within an interpretable machine learning framework. In addition, it is aimed to be as realistic as possible for an asset management environment. As such, the methodology provided here was designed to avoid common pitfalls encountered in financial machine learning application and strategy backtesting, and ideally result in a sustainable trading strategy. As this thesis deals co-jointly with the fields of quantitative trading, machine learning and statistical learning, the models and methodologies employed here are not sufficiently detailed in order to keep the paper consistent. However, their choices and consequences remain contested, and references are provided for interested readers. Therefore, it is expected that the reader is familiar with machine learning concepts in order to be able to grasp the content of this thesis. Finally, concepts are described progressively along the thesis to match with the paper's workflow, instead of compiling them in an early chapter, as may usually be encountered in other theses.

Predictors
The assumed set of predictors is as given in the equity section of (Kakushadze & Serur, 2018).

Investment Universe
The investment universe is defined as the stock components of the Russell 1000 Index. This choice is motivated by data availability, market efficiency and liquidity. The index measures the performance of the large-cap segment of the U.S. equity universe. It is a subset of the Russell 3000 Index and includes approximately 1000 of the largest securities, representing approximately 92% of the U.S. market capitalisation\(^1\). The main index is updated annually, usually on the last Friday of June. Furthermore, the eligible quarterly Initial Public Offering (IPO) may be included along the year. For more information about the full methodology of the index construction, please refer to the methodology available online\(^2\). Finally, being heavily scrutinised by financial professionals and academic studies, it is expected that an American mid-/large-cap stock subset is an accurate measure to test the sustainability of any trading scheme.

---

\(^1\) FRED: [https://fred.stlouisfed.org/series/RU1000TR](https://fred.stlouisfed.org/series/RU1000TR)

**Trading Frequency**

As equity strategies from (Kakushadze & Serur, 2018) range from intra-day to quarterly, a monthly set up is chosen as a fair trade-off to encompass technical analysis-based strategies, more short-term oriented trading, and fundamental analysis-based ones which are more long-term oriented. This choice is beneficial due to the inclusion of less noisy data, less turnover at the expense of less available data to train the algorithm, and lowered Sharpe ratio in comparison to a daily strategy (E. P. Chan, 2017). As such, monthly predictions are computed after the close of the last trading of month and trades are entered on the close of the following trading day until and kept for one month. This gap prevents look-ahead bias, ensures all data is available and allows for a realistic timeframe needed to run computations. This set up can be visualised in Figure 2.1:

![Graphical illustration of the trading framework.](image)

**Pipeline Outline**

This thesis is guided around three core steps:

1. **Feature generation**: The dataset is generated, cleaned, and predictors computed.
2. **Predictive modelling**: A learning algorithm is trained and optimised for defined machine learning metrics to generate predictions.
3. **Portfolio allocation**: predictions are processed to allocate a strategy optimised for defined backtesting metrics.

A chart with all the details of the pipeline is presented in Figure 2.2.
Cross Validation
Predictive Modelling
Portfolio Allocation

Feature Preparation

Predictors
Outputs Labelling

Index Filtering
Missing Data

Feature Extraction

Redundancy

Raw Data

Dataset Cleaning

Feature Generation

Feature Cleaning

Exploratory Data Analysis

Initial Feature Selection

Processed Data

Bloomberg

Other

Feature Preparation

Predictors
Outputs Labelling

Index Filtering
Missing Data

Feature Extraction

Redundancy

Raw Data

Dataset Cleaning

Feature Generation

Feature Cleaning

Exploratory Data Analysis

Initial Feature Selection

Processed Data

Bloomberg

Other

Predictive Modelling

Processed Data

Sampling

Training Dataset

Test Dataset

Training Window Selection

Pre-Processing

Feature Selection

Hyperparameter Optimization

Post-Processing

Classification Metrics

Final Classification Model

Predictions

Cross Validation

Portfolio Allocation

Predictions

Sampling

Training Dataset

Test Dataset

Strategy Allocation

Trading Parameters Optimization

Post-Processing

Final Strategy

Automation

Evaluation

Portfolio Metrics
Trading Costs

Cross Validation

Figure 2.2 Graphical illustration of the machine learning pipeline.
Software

Most of the computations are conducted in Python; a free and open-source interpreted, inter-
active, object-oriented programming language (Van Rossum & De Boer, 1991). Due to its ac-
tive community around providing a vast selection of libraries and support, Python is currently
among one of the fastest-growing programming languages, especially in the space of data sci-
ence and artificial intelligence. This thesis makes intensive use of its packages for scientific
computing amongst others:

- IPython (Pérez & Granger, 2007), Jupyter (Kluyver et al., 2016), Dask (Dask
  Development Team, 2016) and Joblib (Joblib developers, 2011) for development;
- NumPy (Van Der Walt, Colbert, & Varoquaux, 2011) and Pandas (McKinney, 2010)
  for scientific computing and data processing;
- Matplotlib (Hunter, 2007), Seaborn (Waskom et al., 2018) and Yellowbrick (Bengfort
  et al., 2018) for data visualisation;
- SciPy (Jones, Oliphant, Peterson, & Others, 2001), Statsmodels (Seabold & Perktold,
  2010), Scikit- Rebate (Urbanowicz, Olson, Schmitt, Meeker, & Moore, 2018), Dython
  (Zychlinski, 2018) and Scikit-posthocs (Terpilowski, 2019) for statistical models and
  statistical tests;
- Scikit-learn (Pedregosa et al., 2011), MLxtend (Raschka, 2018), Pomegranate
  (Schreiber, 2018) and LightGBM (Ke et al., 2017) for machine learning;
- Keras (Chollet & Others, 2015) and Tensorflow (Abadi et al., 2015) for deep learning;
- Hyperopt (Bergstra, Yamins, & Cox, 2013), Hyperas (Pumperla, 2016) and Optuna
  (Akiba, Sano, Yanase, Ohta, & Koyama, 2019) for hyperparameter optimisation;
- Lime (Ribeiro, Singh, & Guestrin, 2016), ELI5 (TeamHG-Memex, 2016) and SHAP
  (Lundberg et al., 2019; Lundberg, Erion, & Lee, 2018; Lundberg & Lee, 2017) for
  machine learning explainability;
- Auto-sklearn (Feurer et al., 2015) and TPOT (Olson, Bartley, Urbanowicz, & Moore,
  2016) for Automated Machine Learning (AutoML);
- Pyfolio (Quantopian Inc., 2015) and Empyrical (Quantopian Inc., 2017) for portfolio
  analysis.

For readability and reproducibility purposes, Jupyter notebooks are designed according to
(Rule et al., 2019). Other undirectly relevant utility packages used here are mentioned through-
out this paper with references in footnotes.
As no equivalent Python packages could be found, R (R Development Core Team (R Foundation for Statistical Computing), 2008) and the following packages have been used:

- FactoMineR (Lê, Josse, & Husson, 2008) for multivariate analysis.

Links to the scripts and notebooks used can be found in Appendix A.

**Hardware**

Some of the calculations reported here were performed using the EULER cluster at ETH Zurich and on the Google Cloud Platform, while deep learning models were realised with a Graphical Processor Unit (GPU): an NVIDIA Quadro P5000 with 2560 CUDA cores and 16GB GDDR5. To enable large scale distributed parameters optimizations, a PostgreSQL database (Stonebraker & Rowe, 1986) from ETH Zurich has been used.

**Data**

In this paper, a Bloomberg terminal has been used to retrieve most financial data. While the data provider dominates the market of financial data with 33.2% of market share for institutional investors in 2017, most encountered financial commercial data is not the most suitable for backtesting purposes. Such data might need additional efforts in order to be used appropriately within a strategy development. Bloomberg data can easily be downloaded through their add-in provided for Excel. In order to automate the processing of larger quantities of data, the Bloomberg API (BLPAPI) for Python has been mostly used. More information about BLPAPI can be found in Appendix B. A list of all fields downloaded on Bloomberg can be found in Appendix C. To avoid survivorship-bias (see Chapter 3), security identifiers (SID) for the end-of-month constituents of the Russell 100 Index are obtained from January 1995 (earliest date available for the constituents on Bloomberg) to March 2019 and compiled into a binary matrix, indicating whether the stock is a constituent of the index for the subsequent month or not. Then, after a cleaning of the SIDs list referred in Appendix D, the list is used to download individual security data where stock prices have been adjusted for dividends, corporate actions and stock splits. Finally, after cleaning the equity data, referred to in Appendix D, the dataset is ready for feature generation.

---

3 Financial Times: [https://www.ft.com/content/622855dc-2d31-11e8-9b4b-bc4b9f08f381](https://www.ft.com/content/622855dc-2d31-11e8-9b4b-bc4b9f08f381)

4 Bloomberg API: [https://www.bloomberg.com/professional/support/api-library/](https://www.bloomberg.com/professional/support/api-library/)
3 Essential Concepts in Financial Machine Learning

We review here essential concepts that should be taken into account before any data collecting. High-data quality is critical in a robust strategy development process and could be one of its most time-consuming steps. A common way to validate a trading strategy is to simulate it on past data, namely backtesting the strategy. It is as essential as not trivial to perform it. Naive approaches would lead to bias and flawed result metrics due to bias known among academics and financial professional. Adapted data and limiting bias is a necessary condition for a robust trading strategy. The following bias and errors are compiled from (López de Prado, 2018; Luo et al., 2014). The bias shares different terminologies between the fields of quantitative trading, statistical learning and machine learning but refers to the same underlying issues:

1. **Survivorship bias:** It is usually introduced by considering only the current investment universe across the backtesting. Excluding securities that previously left it (i.e. bankruptcy, acquisition or delisting), would artificially inflate portfolio metrics. (Daniel, Sornette, & Woehrmann, 2009) analyses deeply this issue.

2. **Look-ahead bias:** Also referred as data leakage, (Kaufman, Rosset, Perlich, & Stitelman, 2012), this bias consists of acting upon data that would have not yet been available at the simulated moment. It is ubiquitous that macroeconomic or companies’ fundamental data are lagged and can even be corrected posteriorly. For the same reason data random shuffling or classic cross-validation should be avoided when dealing with non-stationary time series data. Data timestamps of data availability should be monitored carefully in order to limit the likelihood of peeking into the future. It also possible to find Point-In-Time (PIT) data from data providers. In this way, CRSP and Compustat have been the most widely used database in the empirical finance literature.

3. **Storytelling:** Overinterpreting and trying to justify past random patterns.

4. **Data snooping:** Also known as cherry-picking, p-hacking or selection bias, it occurs when a given set of data is used more than once for purposes of inference or model selection. When such data reuse occurs, there is always the possibility that any satisfactory results obtained may be due to chance rather than to any merit inherent in the method yielding the results. It is particularly relevant in finance due to its non-experimental nature. Splitting data into in-sample dataset (training set) and out-of-

---

sample dataset (validation/testing set) and statistical tests, as the Bonferroni correction should be considered to prevent false discovery.

5. **Transaction costs:** Transaction costs could only be known if the actual trade would have been made and are thus complicate to be simulated precisely.

6. **Outliers:** Outliers are observations that deviates so much from others data points, raising the suspicion of being the result of a different generating process. Such extreme outcomes should be considered carefully. They might never happen again and have non-desired effects on a model. Investigating the nature and the severity of outliers and various robust statistical tools can help to mitigate it.

7. **Shorting:** Short selling involves the availability of a lender, a cost associated with it and depends on different other market conditions and restrictions. Some regulators banned it temporarily during the financial crisis of 2007-2008 and other periods.

In addition to these common errors, it is crucial to continually remain critical and look for other flaws that could affect the validity of backtesting results. For instance, the following could also be considered:

- Using algorithms, especially machine learning, or exploiting financial anomalies on data corresponding to before their discovery.

- Daily and lower frequency strategies usually rely on open and closing prices. As these prices are formed through an auction process every day, data providers often use the *consolidated prices* among market centres and might not reflect the trade execution price that one could obtain.

- Original stock prices might display variations despite not reflecting a change in real value for investors. When a dividend is issued, the stock price immediately drops to take into account that the company no longer owns this money. Another case is stock split where a company decides to increase/decrease the number of its outstanding shares. Prices are thus expected to follow proportionally. To prevent these price variations, it is common to find adjusted prices from data providers. The adjustment formulas used should be checked as it is not consistent across data providers. In the case of Bloomberg, backtesting with their adjusted close would not simulate receiving

---

dividend nor returns from re-investing it, underestimating strategy’s long term expected return\textsuperscript{7}.

4 Feature Engineering

4.1 Context

Feature generation
After data collection and cleaning, 92 monthly predictors (57 quantitative and 35 qualitative) have been derived for each SID based on the Equity section of (Kakushadze & Serur, 2018). Depending on the data format required from the libraries used, qualitative data may need to be one-hot encoded, leading to the possibility of different numbers of predictors displayed across figures. The predictors are listed in Appendix E. Backgrounds and computations are presented in Appendix F. As opposed to training algorithms parallelly for each security $i$, all individual stock datasets $\{(X_i, y_i)\}_{i=1,...,d}$ are stacked into a single matrix $(X, y)$ in order to fit one classifier on the investment universe simultaneously. This approach leads to more data for training and consecutively more general conclusion, reduced outliers influence and less overfitting (López de Prado, 2018). The decrease in computational complexity for optimising only one machine learning pipeline can also be added.

Target generation
As explained earlier, the returns to be predicted are computed between the close between each first trading day of months. This one-day gap in comparison to features prevents look-ahead bias in computation. Also, some SIDs might get delisted along the month, and should be taken into account to avoid survivorship bias. Delisted shares were assumed to equal a null share value, as the reason why the stock became delisted was not possible to automate for the quantity of SIDs on Bloomberg. For labelling, the methodology from (Fu et al., 2018) is followed here: stocks risk-adjusted returns are ranked for each timestamp. While the top and bottom quantiles are labelled positively and negatively, the candidates in the middle are discarded to filter out noise. It is proposed here to keep the middle candidates and to assess their values later with the meta-labelling approach in the predictive modelling chapter. Accordingly, target returns are defined as follows.

For each month, the targeted returns $R_i$ of the $n$ stocks constituting the index are firstly ranked with the empirical distribution function $\hat{F}_n(x)$:

$$\hat{F}_n(x) = \frac{1}{n} \sum_{i=1}^{n} \mathbf{1}_{(x_i \leq x)}$$
The associated cumulative probabilities are then labelled according to the quantile $q$:

$$y_i(q) = \begin{cases} 
1 & \text{if } \hat{F}_n(R_i) \geq 1 - q \\
-1 & \text{if } \hat{F}_n(R_i) \leq q \\
0 & \text{otherwise}
\end{cases}$$

While (Fu et al., 2018) labels only risk-adjusted returns, simple returns have also been labelled here, and the association between them tested statistically. For illustration purposes, $q = 1/3$ is set in the following section. Across the entire dataset, approximately 14% of the returns would be labelled differently than the risk-adjusted returns. The Spearman’s rank correlation $r_s$ (Spearman, 1904), Kendall’s rank coefficient $\tau$ (Kendall, 1938), Chi-squared test $\chi^2$ (Pearson, 1900), Cramer’s value $V$ (Harald, 1946) and their respective p-values$^8$ have been computed as:

$$r_s = 0.88, p \approx 0$$
$$\tau = 0.85, p \approx 0$$
$$\chi^2 = 582272, p \approx 0$$
$$V = 0.78, p \approx 0$$

It can be concluded that both types of returns are statistically too associated to be considered separately. Only simple returns are therefore kept.

**Missing data assumptions**

After filtering SIDs according to their index membership for each month, the predictors are first inspected for missing data. The heatmap of Figure 4.2 displays the percentage of missing data for each predictor along time. For a more condensed visualisation of missing patterns across predictors, a Principal Component Analysis (PCA) (Pearson, 1901) can be used. In this setting, the predictors are first converted to a binary matrix where missing data is labelled one, and zero otherwise (Kuhn & Johnson, 2019). The matrix is then transposed for predictors to be in rows before PCA is finally applied. Figure 4.1 illustrates the projections of the predictors on the two first components. A significant part of the missing data comes first from implied volatility-related predictors, for which no data is available before beginning 2005 and secondly from the predictors Standardized Unexpected Earnings and Residual Momentum.

$^8$ p-values are smaller than the system smallest representable positive number and the values 0 have been then returned. See https://stackoverflow.com/questions/20530138/scipy-p-value-returns-0-0
These two have the longest lookback window requirement for computation, namely 24 and 36 months and may most likely be the reason for missingness increase, in addition to the lower availability of fundamental data for early timestamps.

Due to the previous explanation, the present missing data can be categorized as *Not Missing At Random* (NMAR) as opposed to *Missing Completely At Random* (MCAR) or *Missing At Random* (MAR) (Roderick & Donald, 2019). While missing information for categorical variables can be easily encoded as *Non-Available* (NA) (Kuhn & Johnson, 2019), the situation is more challenging for continuous variables. NMAR represents the most complex case of the three, and strategies to handle it are to explore the causes of missing data and perform sensitivity analyses. Due to the temporal aspect of the problem, it is assumed here that observations are not statistically independent leading to a cautious approach for not introducing any bias. Bayesian methods are proposed by (Kofman & Sharpe, 2000; Roderick & Donald, 2019). However, as they are outside of the scope of this thesis, such imputation is not attempted. Instead problematic predictors are momentarily dropped, and value of remaining missing data is assessed later with resistant models such as Decision-Tree (DT) models or Naïve Bayes (NB) (Hand & Yu, 2001).

![Figure 4.1 Binary representation of missing values projected in the first two dimensions.](image-url)
**Figure 4.2** Visualization of missing data patterns over the years.
Outliers handling assumptions

Outliers have been presenting a constant challenge in empirical finance research, mainly when relying primarily on Ordinary Least Squares (OLS). A 30-year review of empirical finance articles was performed by Adams, Hayunga, Mansi, Reeb, & Verardi, (2019), using mainly OLS and erroneous practice of outliers’ mitigation if any: winsorizing, trimming, dropping or univariate outliers’ identification in place of multivariate analysis. Instead, the authors introduce a multivariate identification strategy and an estimator for both cross-sectional and panel regressions. This issue is approached here differently as classification methods have been chosen. Harris (2017) revisits statistical distributions used in the fields of economics and finance, questioning the most use of Frequentist approaches and advocating for a balance with Bayesian methods instead. Thus, this thesis approaches data with distribution-free and robust methods. It is assumed instead that outliers’ origins are not the result of error and might contain valuable information. They should not be corrected nor dropped, assuming Bloomberg has entered its data correctly. In that sense robust algorithms, such as DT or data transformation techniques, such as spatial sign (Serneels, De Nolf, & Van Espen, 2006), should be applied.

4.2 Feature Cleaning

Based on the formulated assumptions, the predictors are cleaned as following:

1. Filtering individuals according to their index membership for each month
2. Encoding missingness for qualitative predictors.
3. Dropping implied volatility-related predictors momentarily. They will be re-assessed in the modelling process.
4. Dropping individuals with remaining missing data. They will be re-assessed in the modelling process.

Figure 4.3 illustrates the effect of the cleaning process. For each month, 12% SIDs are dropped on average, leaving a monthly average of 865 SIDs lefts for training. The main reason for such a drop is related to the window length requirement to compute certain features as mentioned before.
Exploratory Data Analysis

After laying out the dataset and main assumptions, the dataset can be analyzed, and its main characteristics summarised with the help of visual methods, a step namely called Exploratory Data Analysis (EDA). As each of the 92 predictors are evaluated here, the most insightful figures and analyses are presented in this thesis to remain consistent. The interested reader is invited to refer to the Jupyter notebook (see appendix A) displaying the full EDA process where univariate distributions, bivariate distributions and other visual representations are presented.

Feature extraction

To directly visualise 92 dimensions, dimension reduction techniques are applied. As PCA can only be performed on continuous data, a Factor Analysis of Mixed Data (FAMD) (Pages, 2004) is chosen here to simultaneously analyses both qualitative and qualitative data. As an analogy, FAMD can be said to behave as PCA on quantitative variables and as Multiple Correspondence Analysis (MCA) for qualitative variables. MCA designs the extension to more than two categorical variables of a Correspondence Analysis (CA) (Gibrat, 1978). Before performing FAMD with R package FactoMineR, quantitative data needs to be normalised. As many predictors display skewness, a Yeo–Johnson transformation (Yeo & Johnson, 2000) is applied first before standardisation. The data transformation revealed to be non-essential as FAMD
with raw values displayed the same results. Figure 4.4 displays the explained variance across the first 20 components and its cumulative amount. The first four components are here examined in detail, capturing around 45% of the overall variability.

![Explained Variances and Cumulative Variance](image)

**Figure 4.4** Variability summarized across the first 20 components of the FAMD.

Figure 4.5 provides a summary of the analysis for the first and second dimensions, while Figure 4.6 for the third and fourth dimensions. For both figures, part (a) and (b) displays the top 22 variables contributing to the respective dimensions. The red dashed line represents the expected average value if contributions were uniform. Both variables can be represented on plot (c), called the *relationship square*, where the correlation between variables and principal dimensions can be assessed. Quantitative variables can be visualized on the *correlation circle* (d) where the signs of correlations are specified. The categories of qualitative variables are represented in (e). Variable categories with a similar profile are grouped together, while negatively correlated variable categories are positioned on opposite sides of the plot origin. Finally (f) displays individuals projected on the selected plane. In addition, more information is displayed on the plots as follows: variable names have been shortened to preserve space. Variables are coloured according to their contribution to each projection (lowest are in blue and highest are in red) on plots (c), (d) and (e) and only top lowest, and highest contributors have been labelled. Lastly, individuals are coloured according to the return label associated in plot (f). From both figures, it can be observed that neither of the first four components separates the return categories.

In Figure 4.5, the first component seems to have captured a latent variable related to trend-following/momentum (momentum, moving averages, residual momentum, standardized unexpected earnings), while the second one focuses on mean-reversion predictors and related indicators (channels, support and resistance). Volatility and variance have an ambiguous role as
they are negatively correlated to dimension 1, but positively correlated to dimension 2. On the plot of categories, neutral signals are all grouped at the origin. The position of categories seems coherent with the previous interpretation of axis as sell-signals of long channels and of support-resistance are close to buying signals of moving averages and vice-versa.

In Figure 4.6, dimension three have captured all predictors related to volatility and variances. Axis 4 interpretation is more conflictual as no variables seem to be well represented. It can be noticed that short term moving averages contribute the most to it. Again, for categories, buy and sell signals from previously mentioned moving averages are opposed on the 3rd dimension.

In general, the primary information to be collected from these plots is the strong inter- and intra-association of certain continuous and qualitative variables. Redundancy of certainty can easily be suspected, as some variables have been computed with close parameters.
Figure 4.5 Summary analysis of the first and second components of the FAMD.

(a) The 22 predictors contributing the most to the first dimension. (b) The 22 predictors contributing the most to the second dimension. (c) Relationship between quantitative and qualitative predictors. (d) Relationship between quantitative variables and components. (e) Relationship between qualitative variables and components. (f) Projection of individuals on the components.
Figure 4.6 Summary analysis of the third and fourth components of the FAMD.

(a) The 22 predictors contributing the most to the third dimension. (b) The 22 predictors contributing the most to the fourth dimension. (c) Relationship between quantitative and qualitative predictors. (d) Relationship between quantitative variables and components. (e) Relationship between qualitative variables and components. (f) Projection of individuals on the components.
4.4 Initial Feature Selection

**Context**

In general, models face the trade-off between predictive performance and model interpretability: Extensively adding predictors might increase predictive ability, but it would be at the expense of model and computational complexity. In addition, some models may be sensitive to irrelevant predictors, multicollinearity or noise, and it always makes scientific sense to minimise the feature set that provides acceptable results. As such, the goal of feature selection is to reduce the number of predictors as much as possible without compromising predictive performance. Methods of feature selection are usually distinguished by three groups:

1. **Implicit methods**: Some models already incorporate a feature selection in their modelling approaches such as LASSO regression (Tibshirani, 1996) or DT models:

2. **Filter methods**: It designates an initial analysis of predictors before proceeding to modelling. Some algorithms or statistical tests can be used to judge the importance of predictors, such as RelieF (Robnik-Sikonja & Kononenko, 2003) or filtering out correlated predictors.

3. **Wrapper methods**: They are iterative procedures continuously supplying predictor subsets to a learning model. The resulting model performance then guides the selection of the next subset to evaluate. Examples might include Recursive Feature Elimination (RFE) (Guyon, Weston, Barnhill, & Vapnik, 2002) with a greedy approach or non-greedy ones such as Genetic Algorithms (GA) or Simulated Annealing (SA)

Each of these groups presents its advantages and drawbacks. The reader is invited to consult (Kuhn & Johnson, 2019) for an exhaustive description. It is common in early feature engineering steps to remove multicollinear features or irrelevant ones with filter methods. However, for classification tasks, (Guyon & Elisseeff, 2003; Guyon, Gunn, Nikravesh, & Zadah, 2006) provide small but revealing illustrated anti-examples with continuous data. As such, correlation does not always imply redundancy. Two highly correlated features taken together are shown to achieve a perfect class separation while providing poor separation taken independently. Moreover, a variable seeming useless by itself can provide to improve classification when taken with others.
All variables
For exploration purposes, all predictors have been evaluated concerning the target labels with first dependency measurement with the Mutual Information (MI) (Ross, 2014) from Scikit-learn and secondly with the RelieF algorithm from Scikit-Rebate. While MI ranks each predictor separately, RelieF takes into account interaction effects. For raw and scaled data and two or three classes of returns, these two procedures scored **KNN_Prediction** highly discriminative and negligible for the rest. This situation can also be observed with univariate and bivariate representations of the predictor (see Jupyter Notebook from Appendix A)

Quantitative variables
Despite high multicollinearity among quantitative predictors (see Figure 4.7), no filtering is attempted here as discussed above. An RFE will be applied during the predictive modelling process instead.

![Figure 4.7 Visualization of the quantitative predictors’ correlation matrix.](image)

Qualitative variables
Redundancy among certain categorical variables can be assumed from the FAMD results. Instead of Cramer’s V, the uncertainty coefficient (Theil, 1970) is used here as a measure of association between categorical variables. The uncertainty coefficient relies on information entropy and provides to be asymmetric in comparison to Cramer’s V, with possible loss of
information. For two discrete random variables $x$ and $y$, the uncertainty coefficient $U(x \mid y)$ is defined as:

$$U(x \mid y) = \frac{H(x) - H(x \mid y)}{H(x)} = \frac{I(x, y)}{H(x)}$$

Where $H(x)$ is the Shannon entropy of a single distribution, $H(x \mid y)$ is the conditional entropy and $I(x, y)$ is the MI:

$$H(x) = -\sum_x P(x) \ln P(x)$$

$$H(x \mid y) = -\sum_{x,y} P(x, y) \ln P(x \mid y)$$

Where $P(x, y)$ and $P(x \mid y)$ are the joint and conditional distributions. The first expression shows that the uncertainty coefficient is an MI normalised. Uncertainty coefficients between categorical variables are computed with the package Dython and can be visualised in Figure 4.8. The filter threshold is set to 0.8 and variables are removed according to Table 4.1. Exponential moving averages seems not to be statistically different from their simple moving averages counterparts.

**Table 4.1** Association between qualitative variables.

<table>
<thead>
<tr>
<th>Feature removed</th>
<th>High association with</th>
</tr>
</thead>
<tbody>
<tr>
<td>Support Resistance_20D</td>
<td>Support Resistance_10D</td>
</tr>
<tr>
<td>Support Resistance_10D</td>
<td>Channel_10D</td>
</tr>
<tr>
<td>Support Resistance_10D</td>
<td>Channel_20D</td>
</tr>
<tr>
<td>Support Resistance_20D</td>
<td>Support Resistance_20D</td>
</tr>
<tr>
<td>EMA_30D</td>
<td>SMA_30D</td>
</tr>
<tr>
<td>EMA_50D</td>
<td>SMA_50D</td>
</tr>
<tr>
<td>EMA_100D</td>
<td>SMA_100D</td>
</tr>
<tr>
<td>EMA_200D</td>
<td>SMA_200D</td>
</tr>
<tr>
<td>3MA_60_120_200D</td>
<td>3MA_30_120_200D</td>
</tr>
<tr>
<td>3MA_90_120_200D</td>
<td>3MA_90_120_200D</td>
</tr>
</tbody>
</table>


10 Wikipedia: [https://en.wikipedia.org/wiki/Mutual_information](https://en.wikipedia.org/wiki/Mutual_information)
Figure 4.8 Visualization of the uncertainty coefficient matrix.
5 Predictive Modelling

5.1 Context

Forecasting stock returns can be translated into a classification problem where side returns define different classes. A third class for neutral cases can also be considered. Instead, the modelling is divided into two problems as presented in the Meta-Labelling approach of (López de Prado, 2018). The first modelling process focuses on predicting the side of the forecasted returns with targets \{-1,1\}. Neutral cases are dropped in this first step as it is assumed they would be assigned with low probabilities to both classes. Once sides are predicted, a second model re-assess the feature with side predictions to determine the size with targets \{1,0\}, that should be attributed to the bet. As such, the secondary model is expected to filter wrongly classified targets from the first model. This approach also reduces the likelihood of overfitting as models are trained for different targets. Figure 5.1 displays the meta-labelling approach.

Learning algorithms

As missing values and multicollinearity are still present in the dataset, early modellings are performed with DT algorithms for their non-parametric, distribution-free assumptions, and robustness to the presence of outliers and irrelevant attributes (Gama, Medas, & Rodrigues, 2004). DT models can be combined into Ensemble model (Yaliang Li, Gao, Li, & Fan, 2015) in order to obtain a better model than any individuals one. Three ensembling approaches can be distinguished:
- **Bagging**: Learners are created independently by resampling with replacement the dataset. The final forecast is the average of the learners' forecasts.

- **Boosting**: Learners are fit sequentially where subsequent learners address previous models errors. The final forecast is a weighted average of the learners' forecasts.

- **Stacking**: Learners forecasts are combined to train a meta-model to output a final forecast.

Bagging addresses overfitting while boosting addresses underfitting, at the cost of a greater risk of overfitting. López de Prado (2018) suggests the bagging approach for financial machine learning, due to greater concerns of easy overfitting because of the low signal-to-noise ratio.

The two types of approaches are tested here. As such, Random Forest (RF) (Ho, 1995), Gradient Tree Boosting (GBT) (J. H. Friedman, 2001) and Dropouts meet Multiple Additive Regression Trees (DART) (Rashmi & Gilad-Bachrach, 2015) have been selected. GBT popularity is due to its immense success in online data science competitions, and a detailed analysis is provided by (Nielsen, 2016). A less formal description can be found here¹¹,¹² and an intuitive description of RF can be found here¹³. Finally, DART is an extension of GBT (also called Multiple Additive Regression Trees (MART) according to authors) with an implementation of the concept of Dropout (Hinton, Srivastava, Krizhevsky, Sutskever, & Salakhutdinov, 2012), designed to regularise and prevent overfitting for neural networks. Instead of the commonly used XGBoost (Chen & Guestrin, 2016), LightGBM is adopted here for its higher convergence speed in comparison to XGBoost and handling missing values as opposed to Scikit-learn.

**Classification metrics**

Major classification metrics, such as accuracy, sensitivity, specificity and others rely on the implicit assumption of a valid probability cutoff (in general 0.5). As the class separation is expected to be filled with noisy samples, cutoff-free metrics are used initially. As such, the Area Under the receiver operating characteristic curve (AUC) (Green & Swets, 1966; Japkowicz & Shah, 2011) is chosen to indicate the separation capacity of the classifier. For a

---


¹² How to explain gradient boosting?: [https://explained.ai/gradient-boosting/index.html](https://explained.ai/gradient-boosting/index.html)

¹³ Random Forests for Complete Beginners: [https://victorzhou.com/blog/intro-to-random-forests/](https://victorzhou.com/blog/intro-to-random-forests/)
random continuous variable $x$ in a binary classification, AUC is defined as the integral of the Receiver Operating characteristic Curve (ROC)$^{14}$:

$$AUC = \int_{x=0}^{1} TPR(FPR^{-1}(x)) \, dx$$

Where $TPR(T)$ and $FPR(T)$ are the True Positive Rate and the False Positive Rate for a given threshold $T$:

$$TPR(T) = \int_{T}^{\infty} p_1(x) \, dx$$

$$FPR(T) = \int_{T}^{\infty} p_0(x) \, dx$$

Where $p_1(x)$ and $p_0(x)$ are the probability distribution for the positive and negative classes. Additionally, the log loss or cross-entropy is employed for probabilistic confidence across classes. For $N$ samples and $C$ classes as, the log loss $L$ is defined as:

$$L(Y, P) = -\frac{1}{N} \sum_{i=1}^{N} \sum_{j=1}^{C} y_{i,j} \ln p_{i,j}$$

While AUC aims to maximise classes discrimination, the log loss is meant to penalise the divergence between actual and estimated probabilities.

The secondary model is optimized for $F_1$ for two reasons:

1. Relabelling has no impact on log loss.
2. There is more interest in focusing on prediction accuracy for the positive class, as compared in predicting both sides equally before.

$F_1$ score is defined as the harmonic mean of precision and recall

$$F_1 = 2 \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}$$

Where recall and precision are:

$$\text{precision} = \frac{TP}{TP + FP}$$

$$\text{recall} = \frac{TP}{TP + FN}$$

As these metrics are not directly interpretable, benchmark values are computed for performances comparison. They represent the case of a classifier with random guesses for a balanced dataset:

\[
AUC_{\text{random}} = \int_{0}^{1} x \, dx = 0.5
\]

\[
L_{\text{random}, \text{class}} = -\ln(1/2) \approx 0.69
\]

\[
L_{\text{random}, \text{class}} = -\ln(1/3) \approx 1.10
\]

\[
F_{\text{random}} = 0.5
\]

The case for AUC represents the area under a diagonal curve. The full mathematical derivation can be found here\textsuperscript{15}.

**Sample weight**

In addition, Scikit-learn and LightGBM allow users to pass samples weights to learners fitting or to metrics. As higher absolute returns are defined as more important than negligible absolute returns, the sampling weight methodology of (López de Prado, 2018) is followed here, and sample weights are defined as the weighted absolute log returns at each timestamp. Logarithmic returns are chosen here due to their symmetric property.

**Dataset shift**

Machine learning algorithms have been formalised under the presumption of common data generating process between the training and the testing environment. In practice, this assumption often fails to hold, and when the data distribution is subject to change over time, it is subject of *dataset shift*. The academic literature has been using different terminologies to express the same concept (Moreno-Torres, Raeder, Alaiz-Rodríguez, Chawla, & Herrera, 2012), but such a shift may fall in three categories as presented initially by (Kelly, Hand, & Adams, 1999) for a set of predictors \( X \) and the class variable \( y \):

1. **Covariate shift**: the input distribution \( p(X) \) differs while the posterior distribution of the class memberships \( p(y \mid X) \) remains unchanged.

2. **Prior probability shift**: the class prior distribution \( p(y) \) differs while the posterior distribution \( p(X \mid y) \) remains unchanged.

\textsuperscript{15} StackExchange: https://datascience.stackexchange.com/questions/31872/auc-roc-of-a-random-classifier
3. **Concept shift**: the relationship between input and class variables, \( p(y|X) \) and \( p(X|y) \), differ while input and class distribution, \( p(X) \) and \( p(y) \), remains unchanged.

This paper’s problem is naturally exposed to such shifts due to its temporal component. Markets are also known to be adaptive systems (W. Lo, 2017), and signal values are thus expected to keep changing. Developing algorithms to learn under dataset shift is an active area in machine learning research (Gama et al., 2004; Žliobaitė, 2010) and exploring the implementation of such learners is outside of the scope of this thesis. However, the following two adjustments are proposed and tested statistically to make our machine learning pipeline more adaptable:

1. **Sample weighting**: (López de Prado, 2018) time linear decay factors are used to multiply sample weights and to decay older ones. The convention of the author is reused here, and for a decay parameter \( c \in (-1,1] \), the following cases can be observed:
   - \( c = 1 \) means no time decay.
   - \( 0 < c < 1 \) means that weights decay linearly, but are still strictly positive.
   - \( c = 0 \) means that weights converge linearly to zero as they get older.
   - \( -1 < c < 0 \) means that the sample weights of the oldest \( c \) portion are set to zero and are consecutively erased from memory.

2. **Sliding window training**: Predictions are based on a learner periodically re-trained on a defined lookback window. Such methodology is standard in the financial machine learning literature. An example can be found in (Bao, Yue, & Rao, 2017)

These implementations can be seen in Figure 5.2, in the context of the data splitting, described below.

**Data splitting**

The dataset is broken in three sections, as suggested by (Hastie, Robert, & Jerome, 2009):

1. **Testing set**: defined as the last four years of the dataset, namely 16.5% of the entire dataset
2. **Training set**: defined as the first 70% of the remaining data, namely 58.5% of the entire dataset
3. **Validation set**: defined as the last 30% of the remaining data, namely 25% of the entire dataset
The testing set is locked until the entire model calibration process is finished. This prevents underestimation of the true model error and should provide a clean backtest for the last four years at least. Finally, the model is built on the training set, and prediction errors during the modelling process are calculated using the validation set. Figure 5.2 illustrates the splits used.

![Figure 5.2: Graphical illustration of the data splitting and the training methodology used.](image)

Top panel: Growing window illustrates the case where all previous data is considered. Bottom panel: The model is only trained on a specific lookback period.

**Resampling**

The commonly used $k$-fold Cross-Validation (CV) is known not to be suitable for time-dependent data. The in-sample performances are then estimated with Repeated Holdout validation (Rep-Holdout) motivated by the exhaustive comparison of resampling methods for time series from (Cerqueira, Torgo, & Mozetič, 2019). Due to serial correlation and data derived from overlapping points, a monthly purge is applied between the inner-training set and the inner-testing set (López de Prado, 2018). Rep-Holdout and the purging procedures are illustrated in Figure 5.3. If not mentioned otherwise, 30 resamplings are performed to estimate performances.

As observed by (López de Prado, 2018), Scikit-learn cross-validation implementations do not pass sample weights to the scoring function, but only to the estimator class. This issue has still not been solved as of the time of this paper and is common among other major Python machine learning libraries due to their dependency to Scikit-learn. The author’s solution is

---

16 Github: [https://github.com/scikit-learn/scikit-learn/pull/13432](https://github.com/scikit-learn/scikit-learn/pull/13432)
followed, and a function estimating weighted score by repeated holdout validation has also been recorded here.

**Tuning parameters**

Because of the high complexity of LightGBM’s parameters tuning\(^\text{17}\), a grid-/random search (Bergstra & Bengio, 2012) would be computationally too demanding. Instead, Bayesian optimisation frameworks are chosen. Initially, with Hyperopt, hyperparameters search has been finally performed with Optuna. They both rely on a Tree-structured Parzen Estimator approach (TPE) (Bergstra, Bardenet, Bengio, & Kégl, 2011), but Optuna proved to be more performant due to its ability for parallel computing and pruning of unpromising trials. To enable large scale distributed optimisations with Optuna, a PostgreSQL database has been used. Unless mentioned otherwise, 200 trials are performed for each model.

**Statistical tests for model comparison**

The effective practice of statistical tests to compare classifiers has been controversial in the machine learning research space, mainly due to the use of multiple testing without significance tests correction or the use of parametric tests which assumptions cannot be proven to hold (Benavoli, Corani, Demšar, & Zaffalon, 2017; Japkowicz & Shah, 2011). Tests results presented in this thesis follows methodologies from (Demšar, 2006; Japkowicz & Shah, 2011; Salzberg, 1997). As such, monthly performance metrics are drawn from the validation set, meaning the learning algorithms has not been trained nor tuned with them. Then, the non-parametric Wilcoxon signed-ranks test, (Wilcoxon, 1945) and Friedman test (M. Friedman, 1937) are performed to compare two and more classifiers respectively. Finally, whenever the

\[^{17}\] Description of all parameters in LightGBM: [https://lightgbm.readthedocs.io/en/latest/Parameters.html](https://lightgbm.readthedocs.io/en/latest/Parameters.html)
null hypothesis of the Friedman test is rejected ($H_0$: no significant difference between models performances), post hoc tests can be proceeded to identify which classifiers actually differ. For its complementarity with the Friedman test, the Nemenyi test (Nemenyi, 1962) is chosen here as a post hoc test.

**Feature selection**

Despite not affecting the predictive performances of DT algorithms, redundant predictors tend to dilute the feature importance scores (Kuhn & Johnson, 2019). In addition to the previous reasons mentioned, feature selection should also be applied in order to correctly interpret such scores. For this process, RFE is chosen for its complementarity with DT algorithms and its efficient screening of correlated variables (Gregorutti, Michel, & Saint-Pierre, 2017). RFE is a greedy Sequential Backward Selection (SBS) (Ferri, Pudil, Hatef, & Kittler, 1994) of predictors for learners with weight coefficients or importance scores. For tree-based models, RFE primarily works by eliminating features recursively using feature importance, whereas SBS eliminates features based on a user-defined classifier/regression performance metric. These processes need to be paired with external resampling as they could easily overfit.

RFE is performed with RFECV\(^{18}\) from Scikit-learn and SFS with SequentialFeatureSelector\(^{19}\) from Mlxtend. As before, these objects cannot pass sample weights to the scoring function. While sample weights can be passed to the learner for SequentialFeatureSelector, RFECV does not accept any sample weights. Due to their computational costs and the fact these implementations are already optimised for parallel computing, recoding them including a weighted scorer has been considered out of this thesis scope. Instead, both implementations are used and tested on the separate validation set with weighted scorer.

Furthermore, these computational complexities rise another issue concerning selection bias. CV methods are unbiased if all the aspect of classifier training takes place inside the CV (Hastie et al., 2009; Kuhn & Johnson, 2013). Ideally, it means that feature pre-processing, feature selection, classifier type selection and classifier parameter tuning should be performed co-jointly in each CV loop. Due to the size of the dataset and the complexity of this paper machine learning pipeline, such procedures cannot be considered. Instead, we rely on the more pragmatic

---


\(^{19}\) SFS: [http://rasbt.github.io/mlxtend/user_guide/feature_selection/SequentialFeatureSelector/](http://rasbt.github.io/mlxtend/user_guide/feature_selection/SequentialFeatureSelector/)
approach from (Kuhn & Johnson, 2019) and here\textsuperscript{20}. The following assumptions are formulated to mitigate selection bias:

- Precedent feature selections, i.e. filtering methods, have been done in an unsupervised way, meaning not taking into account targets, and should introduce any selection bias (Hastie et al., 2009).

- Learners less sensitive to the choice of tuning parameters might not need to be re-tuned in a nested CV. (Kuhn & Johnson, 2019) used a Random Forest (RF) model for their application of RFE and thus LightGBM’s GBT is considered similarly less sensitive as opposed to Logistic Regression (LR) or Support-Vector Machine (SVM) (Corinna Cortes & Vapnik, 1995).

A selection of promising features subsets and hyperparameters would be tested on the validation set. Finally, to study the impacts of correlated predictors and of the scorer in the selection process, the exercise of (Kuhn & Johnson, 2019) is repeated here. RFE and SBS are performed for AUC and log loss for three initial feature sets defined as:

1. All predictors.
2. Predictors filtered for correlation (threshold set at 0.8).
3. Predictors filtered for multicollinearity with dependency estimation with RF\textsuperscript{21}.

In order to identify dependency among features, each feature can be regressed on the remaining ones with RF. Consecutively for each regression, the corresponding out-of-bag $R^2$ indicates the independent variable dependency, or how it can be easily predicted. Regarding the dependent variables, their importance scores can be interpreted as their predictive powers. Finally, dependent variables with importance scores close to one across several regressions can be removed. The author’s package, \texttt{Rfpimp}\textsuperscript{22}, is used to illustrate this filtering.

\textsuperscript{20} StackExchange: https://stats.stackexchange.com/questions/264533/how-should-feature-selection-and-hyperparameter-optimization-be-ordered-in-the-m
\textsuperscript{21} Dealing with collinear features: https://explained.ai/rf-importance/index.html#corr_collinear
\textsuperscript{22} \texttt{Rfpimp}: https://github.com/parrt/random-forest-importances
5.2 Primary Model

As in the previous chapter, neutral cases, i.e. label 0, are momentarily dropped, leading to a binary classification problem with labels \{-1,1\}. Neutral cases are expected to be later predicted to both classes with low probability, allowing the second model to filter them out. In the next pages, different training configurations are compared. Classifiers are judged based on their capacity to distinguish return sides. For this purpose, performances are measured with non-weighted log loss and AUC on predictions made on the validation set. In addition, each configuration hyperparameters have been retuned for weighted log loss.

Window training

As a first comparison, performances on the validation set are compared for models trained with growing windows and sliding windows. In order to describe training methodologies, the terminology of (Cerqueira et al., 2019) is referred to this study. The term growing window describe including all previous data for training while sliding window includes a fixed size look-back window for training. The following models are considered:

- Growing window training for time decay factor $c \in \{1, 0.75, 0.5, 0, -0.25, -0.5\}$
- Sliding window training for windows of 1, 2, 4 and 8 years.

Each model has been optimized for weighted log loss with internal Rep-Holdout validation. Figure 5.4 displays the statistics metrics computed over months on the validation set. All models seem to have the same level of class discrimination. However, growing windows models seem to have a slight advantage from a probabilistic view. Friedman tests have been then performed for the four metrics, and the null hypothesis is rejected. Nemenyi tests then proceed and respective p-values are displayed in Figure 5.5. Across tests, a significant difference can generally be found between growing window and sliding window models, especially for weighted log loss, the optimised metric. For the rest of the study, sliding window models are then not considered anymore. In addition, to slightly lower performances, they are computationally expensive to optimise as parameters search have to be redone at each training slide.

Missing values

So far, missing values have been dropped. Their value on performances metrics is re-assessed here as LightGBM handles them. In a first time, samples with missing data are added back to the dataset, while the implied volatility features are added in a second step. New models are
compared to the previous growing training models. Figure 5.6 displays the statistics metrics computed over months on the validation set and related p-values from Nemenyi tests are displayed in Figure 5.7. While non-weighted performances slightly decreased, weighted log loss show to be lower and have less variance. P-values seem also to indicates statistical differences between models with and without missing values. As we are more interested in better probabilities for returns with higher weights, missing values are therefore kept.

Rolling preprocessing
When stacking features, (López de Prado, 2018) recommends to standardize them on a rolling training window to ensure some distributional homogeneity in the dataset and numerous equity strategies of (Kakushadze & Serur, 2018) rely on ranking stocks. Ranking is also a way to mitigate outliers. Finally, as we perform cross-sectional analysis, rolling scaling could be suitable. Thus, models trained on monthly pre-processed feature are compared for:

- Ranking.
- RobustScaler(), as outliers are expected.
- PowerTransformer(), as skewness is expected.

Figure 5.8 displays the statistics metrics computed over months on the validation set and related p-values from Nemenyi tests are displayed in Figure 5.9. Monthly means and variances have both improved for weighted and non-weighted metrics, confirmed by the Nemenyi tests. RobustScaler() is kept as it is computationally more efficient than PowerTransformer() and to preserve the data distribution if it had to be discriminating for the secondary model.
Figure 5.4: Comparison of training methodologies: Means and standard deviations of metrics computed monthly
Figure 5.5: Comparison of training methodologies.

Left panel: Monthly metrics time series. Right panel: Visualization of p-values from Nemenyi tests.
Figure 5.6: Comparison of training with missing values: Means and standard deviations of metrics computed monthly.

Base models refers to previous growing window models.

Figure 5.7: Comparison of training with missing values

Visualization of p-values from Nemenyi tests. Base models refers to previous growing window models.
Figure 5.8: Comparison of training with rolling pre-processed data: Means and standard deviations of metrics computed monthly.

Raw models refer to previous models with raw data including all missing values and predictors.
Figure 5.9: Comparison of training with rolling pre-processed data:

Visualization of p-values from Nemenyi tests. Raw models refer to previous models with raw data including all missing values and predictors.
5.3 Feature Selection

The training methodology is defined so far as:

- Growing window
- Keeping missing values
- Rolling pre-processing

We are interested in removing irrelevant features and associated noise from the models. As described earlier, RFE and SFS are performed on all feature and two different feature subsets, previously filtered for correlation and multicollinearity. Results displayed in Figure 5.11 shows that GBT benefits from feature interaction between correlated features. Bests subsets proposed by RFECV and the three subsets that achieved best scores on SFS are selected for further investigation. Figure 5.10 displays selection percentage of predictors for these subsets. Finally, these subsets are used for training and validation performances are reported in Figure 5.12. The subset RFECV LL a11 with 21 predictors is selected at this one the only having a lower mean and lower variance for weighted log loss compared to the base case.

Figure 5.10: Predictor selection percentage for selected feature subsets.
Figure 5.11: The cross-validated results for RFE and SFS using GBT.

Left panel: RFECV selecting subset based on GBT feature importances. Middle and Right panels: SFS selecting subset based on AUC/LogLoss Score. Right panels: GBT trained with sample weight.
**Figure 5.12**: Comparison of training with different feature subsets: Means and standard deviations of metrics computed monthly.

Base models refers to model with RobustScaler() and a time decay factor of 0.5. Value in brackets represents the number of predictors included in the subset.
5.4 Data Leakage

Context
As early backtesting with the previously mentioned models, lead to extraordinary performances for a monthly strategy (Sharp Ratio above 5), the process modelling has been interrupted to investigate the possibility of data leakage. After investigations across the entire pipeline, it has been deduced that the predictions from the $k$-NN were probably the suspects for such leakage. This chapter is meant then to describe and to emphasize the subtlety of data leakage (Kaufman et al., 2012) in financial machine learning application. It was enough to unintentionally leak data at one step to compromise the entire pipeline and invalidate results, despite rigorous procedures along the process:

1. Feature Generation: inclusion of delisted index components; usage of PIT data; rolling prediction for $k$-NN; alignment of sample features and targeted returns with one day gap; Treatments for missing values and outliers not considered;
2. Feature Engineering: Early feature selection without taking targets into account.
3. Predictive Modelling: Dataset splitting; Cross-Validation with one month purging on training set; Results validation with validation set; Locking of the testing set; Rolling feature scaling.

Leakage in financial application occurs when targets from training and testing sets are derived from overlapping points (López de Prado, 2018). The returns derivation illustrates the issue as they are defined from two different price points. Let $y_t$ be the targeted $s$-days returns for two timestamps assigned to the training and testing sets as $t = t_1, t_2$:

\[
y_{t_1} = \frac{P_{t_1+s}}{P_{t_1}} - 1
\]

\[
y_{t_2} = \frac{P_{t_2+s}}{P_{t_2}} - 1
\]

For $t_2 = t_1 + 1$ and then $P_{t_2+s} = P_{t_2+s-1}$, information leakage necessary occurs as prices are shared. Another way more intuitive way to interpret is: By the time one should predict $y_{t_2}$, the estimator could not have been trained with $y_t$ for $t = t_2 - 1, ..., t_2 - s$ as related prices were not
available yet. In order to reduce the likelihood of leakage (López de Prado, 2018) suggests two approaches:

1. **Purging and Embargoing the training set**: Removing data from the training set to ensure non-overlapping data between training and testing sets.

2. **Avoid overfitting**: Impeaching the classifier to profits from leakage even if it occurs. See (López de Prado, 2018) for description of adapted classifiers.

Regarding this paper, the prediction methodologies used for the \(k\)-NN strategy and the main framework are fundamentally different. The \(k\)-NN makes daily predictions of the return over the next 20 days (with one day gap) while the main framework makes monthly predictions only. In that sense, the main framework methodology is bias-free as we purge the last month of the training set. For the \(k\)-NN case, the situation is different as returns are continuously overlapping (see Figure 5.7). As the \(k\)-NN predictions were initially added to the feature set, leakage could not be prevented in the modelling process, even by locking the testing subset.

![Figure 5.13: Graphical illustration of data leakage.](image)

Lenghts of rectangle represent the timerange needed to compute feature and targets. The leakage barrier represents all training returns sharing data with the testing returns.

The situation could have been avoided with:

1. Larger purging or using a different framework for \(k\)-NN predictions.
2. Using one of the classifiers recommended by (López de Prado, 2018).

Due to its nature, data leakage is not trivial to detect and eliminate (Kaufman et al., 2012). The issue seemed obvious from backtests results. However, after reflexions, it could have been spotted earlier as constant flags were raised along the process:
1. **Feature Generation**: High performances for the regression problem (see Appendix F)
2. **Feature Engineering**: High discrimination power of the leaking feature in comparison to the others.
3. **Predictive Modelling**: High performances of learners; Constantly ranked first by feature selections and High feature importances attributed by models.

**Redefined Pipeline**

For ethical and scientific reasons, it has been decided to re-start the modelling processes performed until now. As data leakage was spotted in the latest days of this thesis, the primary model has been designed in a shorter way:

- *k*-NN have been retrained and forecasts re-computed with both a purging period of 21 days
- Models, pre-processing and hyper-tuning selection have been performed within one step with cross-validation.
- Feature selection is skipped due to the nature of DT models
- Analysis from Chapter 4, should remain robust for the rest of the predictors.

The cross-validation performances of the redesigned primary model dropped terribly to $L_{\text{weighted}} = 0.55$. As it is not far from the performances of classifier with random guesses, a second pipeline has been designed parallelly with the primary model optimized for weighted recall. As described in (López de Prado, 2018), this approach aims to classify the most possible number of positive labels at the expense of false-positive increase. False-positive are then expected to be filtered by the secondary model.
5.5 Pipeline Performances

After re-configuring pipelines and optimizing them for their own metrics, the pipeline optimized for log loss selected GBTs for both models while recall selected an RF as primary and a GBT as a secondary model. Performances are compared on the validation set as observable from Figure 5.14. First, secondary models for both models seem to share some performances. Very close hyperparameters have been selected for both of them, interpolating the discrimination capacity of the first two models, as performances obtain very close to the equivalent of random guesses. Due to higher classification performances of secondary models in comparison to the first ones, it raises the issue of accepting a bet that has been predicted for the wrong side.

![Figure 5.14: Comparison of pipeline performances on the validation set: Means and standard deviations of metrics computed monthly.](image)

*Meta* refers to the primary model outputting meta-labels and *Clf* refers to the secondary model. The metric associated determines the pipeline for which the first model has been optimized for.
6 Portfolio Allocation and Portfolio Analysis

6.1 Context

Once ML forecasts have been outputted, it remains to convert them into trading signals and backtest the strategy. For these tasks, the in-sample and out-of-sample time ranges are defined respectively as:

- **In-Sample**: Training and Validation subsets defined in the predictive modelling process, i.e. from the 31st January 1995 to the 31st March 2015.

- **Out-of-Sample**: Testing subset also defined earlier and locked until now, i.e. from the 30th April 2015 to the 29th March 2019.

An Equity Long-Only Investment Strategy is attempted as shorting stocks are not relevant in a monthly setting. Finally, portfolio statistics include trading costs of 0.25% and computations used are presented in Appendix G.

**Signals**

Weights are defined from the combination of sides predicted earlier from the primary model and the size probabilities from the secondary one, with the *Bet Sizing* approach presented by (López de Prado, 2018). From the output of the secondary model, let define \( p(x) \) as the probability that the label \( x \in \{-1,1\} \) takes place. By computing testing the null hypothesis

\[ H_0 : p(x = 1) = \frac{1}{2} \text{ and the test static } z : \]

\[ z = \frac{p(x = 1) - \frac{1}{2}}{\sqrt{p(x = 1)(1 - p(x = 1))}} \sim Z, \text{ with } z \in (-\infty, \infty) \]

The bet size \( m \) is defined as:

\[ m = 2\Phi(z) - 1 \text{ with } m \in (-1,1) \]

Where \( Z \) represents the Standard Normal distribution and \( \Phi \) is the corresponding CDF.

Final signals can be obtained by multiplying the bet size with the predicted side.

**Backtesting through Cross-Validation**

The signals obtained needs to be translated to allocation weights as a long-only strategy assumes:
\[ \sum_{i} w_{ij} = 1 \] with \( w_{ij} \geq 0 \quad \forall t \)

Where \( w_{ij} \) stands for the normalized weight assigned to stock \( i \) at the timestamp \( t \). As we are interested in the bets with the highest probabilities, the \( n \) highest signals are selected and normalized:

\[ w_i = \frac{m_i}{\sum_i m_i} \] where \( i \in I^* \)

Where \( I^* \) denotes the set of indices associated with the \( n \) highest signals. To prevent excess turnover induced by small weight differences at every prediction, the weights are further discretized:

\[ w_i^* = \text{round} \left( \frac{w_i}{d} \right) d \quad \text{where} \quad d \in (0,1] \]

The parameters \( n \) and \( d \) are determined by optimizing the Sharp Ratio strategy due to the pitfall of out-sample performances of strategy optimized for return only as observed by (Wiecki, Campbell, Lent, & Stauth, 2016). To reduce the likelihood of backtest overfitting and obtain the highest true Sharpe ratio possible, we rely on the Combinatorial Purged Cross-Validation (CPCV) (López de Prado, 2018). It recombines groups of observations into training and testing sets where learners can be applied to the forecast the testing sets. Finally \( \varphi \) backtest paths can be recombined from the forecasts and allows to derive an empirical distribution of the Sharpe ratio. Let \( T \) observations be partitioned into \( N \) groups without shuffling of size \( \lfloor T / N \rfloor \); \( k \) is the number of groups assigned to the testing set, the number of possible training/testing splits \( \varphi_{\text{split}} \) is:

\[ \varphi_{\text{split}}[N,k] = \left( \begin{array}{c} N \\ N-k \end{array} \right) = \frac{\prod_{i=0}^{k-1} (N-i)}{k!} \]

The \( k \) testing sets from each combination can be re-grouped into \( \varphi_{\text{path}} \) paths to be backtested:

\[ \varphi_{\text{path}}[N,k] = \frac{k}{N} \left( \begin{array}{c} N \\ N-k \end{array} \right) = \frac{\prod_{i=1}^{k-1} (N-i)}{(k-1)!} \]

From there, the empirical distribution of the strategy’s Sharpe ratio can be derived.
6.2 Results

Empirical Distribution of Strategies’ Sharpe Ratio

The CPCV is applied on the in-sample dataset to determine trading parameters. From the in-sample time range, \( T = 243 \) months can be observed. To let the \( k \) groups timerange matches the out-of-sample time range (~4 years), monthly samples are assigned to \( N = 15 \) groups with size of 16 months and \( k = 3 \). Finally, we obtain the number of splits and backtest paths to be computed:

\[
\varphi_{\text{split}} = 455 \\
\varphi_{\text{path}} = 91
\]

The following trading parameters have been obtained by optimising the Sharpe Ratio from the CPCV:

\[
\begin{align*}
    n_{\text{logloss}} &= 26 \\
    d_{\text{logloss}} &= \frac{1}{1066} \\
    n_{\text{recall}} &= 101 \\
    d_{\text{recall}} &= \frac{1}{6363}
\end{align*}
\]

Empirical Sharpe Ratio distribution can be plotted as follow:

![Empirical Sharpe Ratio distribution for strategies](image)

**Figure 6.1** Empirical Sharpe Ratio distribution for strategies
Backtest Statistics

Strategies are backtested with selected parameters. Cumulative returns for strategies and benchmarks are reported in Figure 6.2. Cumulative returns Ratios of the strategies over the Russell 1000 Index can be visualized in Figure 6.3. Finally, Strategy performances are reported in Table 6.1. Despite similar Sharpe Ratio, the LogLoss pipeline managed to remain more profitable than the benchmark including trading costs while trading costs killed the performances of the Recall pipeline. By a higher differential between average returns from hits and misses, the log loss pipeline manages to recover quicker from drawdowns.

From the trading parameters obtained below, we retrieve the effects from the metrics. As the log loss pipeline seems more confident on its bets, he selects only 26 stocks per time, leading to a higher turnover. The recall pipeline has however kept on increasing false positives and must pick a higher number of stocks per time to reach profitability, leading to lower turnover. Finally as empirical distribution and strategy performances do not correspond, we can question if such performances are not purely due by luck.
Figure 6.2 Top panel: Out-Sample cumulative returns for strategies and benchmarks.

Bottom panel: Out-Sample log cumulative returns for strategies and benchmarks.
Figure 6.3 Top panel: Out-Sample ratio of the strategies cumulative returns over the Russell 1000 Index cumulative returns.

Bottom panel: Out-Sample ratio of the strategies cumulative returns over the Russell 1000 Index cumulative returns displayed in log values.
### Table 6.1 Strategies performances for out-of-sample data.

<table>
<thead>
<tr>
<th>Category</th>
<th>Measure</th>
<th>Pipeline LogLoss</th>
<th>Pipeline Recall</th>
<th>Russell 1000</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>w/o TC</td>
<td>with TC</td>
<td>w/o TC</td>
<td>with TC</td>
</tr>
<tr>
<td><strong>General</strong></td>
<td>Time range</td>
<td></td>
<td>from the 30th April 2015 to the 29th March 2019</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Turnover</td>
<td>16</td>
<td>13</td>
<td>N/A</td>
</tr>
<tr>
<td><strong>Performance</strong></td>
<td>Cumulative return</td>
<td>193.3%</td>
<td>135.8%</td>
<td>113.8%</td>
</tr>
<tr>
<td></td>
<td>CAGR</td>
<td>40.8%</td>
<td>35.3%</td>
<td>33.1%</td>
</tr>
<tr>
<td></td>
<td>Alpha</td>
<td>7.4%</td>
<td>3.2%</td>
<td>2.1%</td>
</tr>
<tr>
<td></td>
<td>Beta</td>
<td>1.35</td>
<td>1.35</td>
<td>1.20</td>
</tr>
<tr>
<td></td>
<td>Stability</td>
<td>0.92</td>
<td>0.89</td>
<td>0.9</td>
</tr>
<tr>
<td></td>
<td>Hit Ratio</td>
<td>53.8%</td>
<td>54.3%</td>
<td>N/A</td>
</tr>
<tr>
<td></td>
<td>Average return from hits</td>
<td>0.34%</td>
<td>0.08%</td>
<td>N/A</td>
</tr>
<tr>
<td></td>
<td>Average return from misses</td>
<td>-0.29%</td>
<td>-0.07%</td>
<td>N/A</td>
</tr>
<tr>
<td><strong>Risk</strong></td>
<td>Volatility</td>
<td>21.1%</td>
<td>21.1%</td>
<td>16.5%</td>
</tr>
<tr>
<td></td>
<td>Skewness</td>
<td>0.20</td>
<td>0.19</td>
<td>-0.29</td>
</tr>
<tr>
<td></td>
<td>Kurtosis</td>
<td>0.75</td>
<td>0.73</td>
<td>0.62</td>
</tr>
<tr>
<td></td>
<td>Maximum drawdown</td>
<td>-21.8%</td>
<td>-24%</td>
<td>-18%</td>
</tr>
<tr>
<td><strong>Efficiency</strong></td>
<td>Sharpe ratio</td>
<td>0.91</td>
<td>0.72</td>
<td>0.77</td>
</tr>
<tr>
<td></td>
<td>Information ratio</td>
<td>0.8</td>
<td>0.5</td>
<td>0.56</td>
</tr>
<tr>
<td></td>
<td>Sortino ratio</td>
<td>1.63</td>
<td>1.23</td>
<td>1.23</td>
</tr>
<tr>
<td></td>
<td>Calmar ratio</td>
<td>0.84</td>
<td>0.58</td>
<td>0.65</td>
</tr>
<tr>
<td></td>
<td>Tail ratio</td>
<td>1.24</td>
<td>1.14</td>
<td>1.10</td>
</tr>
</tbody>
</table>
Chapter 7: Conclusion and Further Research

Because of data leaking has been spotted in the last weeks of this thesis, most findings of this thesis remain invalid. However, practical use of financial machine learning concepts and related issues have been underlined here.

Section 3 reviewed essential concepts before starting to design any trading strategy.

In Section 4, Assumptions about missing data and outliers have been defined, and a FAMD has been applied to both quantitative a qualitative variables to extract information from 92 dimensions. The interpretation of FAMD plots suggested many features were redundant. Correlated quantitative variables have also not been filtered. As shown further by RFECV and SFS, some feature interaction can occur between correlated feature, increasing model performance at the same time. Finally the uncertainty coefficient was used to remove 11 qualitative variables. As an example, exponential moving averages were found too associated with the respective length simple moving averages and were dropped, therefore.

Section 5 presented the modelling process and the meta-labelling approach. Non-parametric statistical tests were also used to compare classifier performances. However, after feature selection to determine a more efficient feature subset, data leakage has been suspected due to early out-of-sample backtest performances. The issue has been reviewed and fixed. New machine learning pipelines have been created to deliver more reasonable backtests.

The predictions were converted, in Chapter 6 to allocation weights with statistical tests. Besides, strategies have been optimized with the Combinatorial Purged Cross-Validation allowing to backtest the entire training time range without overfitting. Finally two monthly equity strategies based on Ensemble of Decision-Tree algorithms were proposed, and their performances analyzed. While trading costs annihilates profitability for the first one, the second one remains profitable and above the selected benchmark. These results were found similar to those of (Fischer & Krauss, 2018; Krauss, Do, & Huck, 2017) for whom, the performances of machine-learning based strategy were continually dropping over time. It can be assumed that financial markets agents have been using machine learning techniques long enough for simple models to capture any profits nowadays. Also, the inputs were derived from already well documented and might no longer offer valuable trading signals or as (López de Prado, 2018) suggests, traditional financial data have been so exploited that new alphas should be looked in alternative data.

Finally, two additional chapters would have been added, if data leakage presence not occurred:
• Machine Learning Interpretation with SHAP (Lundberg et al., 2018) as feature importances interpretation from decision trees algorithms cannot be considered consistent.

• AutoML and Deep Learning. Models have been initially developed before data leakage was noticed. Therefore, it did not seem to be a priority to pursue development.

Further research are suggested with the following five approaches.

1. **Features**: Increasing them, as a limited amount of fundamental data were considered here.

2. **Learners**: The field of AI/ML is currently one of the newest research areas, and it can be hard to keep pace with the constant flow of new learners developed. If the present topic should still be considered as a classification problem, one can consider more advanced deep learning models such as attention-based neural networks (H. Li, Shen, & Zhu, 2018; Youru Li, Zhu, Kong, Han, & Zhao, 2019; Vinayavekhin et al., 2018) or probabilistic deep learning (H. Wang & Yeung, 2016). Framing it as a stock ranking problem, LambdaRank or other ranking algorithms could be used (Burges, 2010; Song, Liu, & Yang, 2017; L. Wang & Rasheed, 2018). Finally, as trading strategies involve optimizing trading decisions, a deep-/reinforcement learning framework could be a natural framework (Sato, 2019).

3. **Asset classes**: The present methodology could be easily extended to other asset classes as Kakushadze & Serur (2018) also define strategies for options, fixed income, indexes, volatility, foreign exchange, commodities, futures and others. If different class portfolios were to be designed, a machine learning asset allocation such as the Hierarchical Risk Parity allocation (HRP) (López de Prado, 2016) could be considered.

4. **Strategy framework**: Compared to the proposed framework by (López de Prado, 2018), this paper works with the monthly fixed-time horizon. The returns are also labelled according to their cross-section percentile. In that sense, positive returns could be labelled negative for the case of a bullish market and vice-versa. If dynamic thresholds were to be defined from a risk management realistic point of view, the triple-barrier method (López de Prado, 2018) could have been used. Future researchers are also invited to explore different trading frequency, considering overlapping bets or long/short implementation. The strategy could also implement a layer of complexity such as stop-loss limits, market regime detection (Chakravorty & Awasthi, 2018; Cook & Smalter Hall, 2017; John M., 2016) or different allocation systems (Black &
Finally, the content of (Cesa-Bianchi & Lugosi, 2006) was recommended, however, not considered in this paper due to time constraints.

5. **Portfolio metrics:** Due to time constraints, advanced backtests statistics (López de Prado, 2018) such as Probabilistic Sharpe Ratio and Deflated Sharpe Ratio, or a strategy risk review (López de Prado, 2018) have not been performed here.

Finally, using R is also a consideration suggested here. While python is better suited for a production environment, the lack of flexibility and features from its ML packages at this time raised along the thesis might question its use for research purposes. For instance, Kuhn & Johnson (2019) rely on the Caret R package (Kuhn, 2008) that includes feature selection with genetic algorithms.
References


Pearson, K. (1900). X. On the criterion that a given system of deviations from the probable in the case of a correlated system of variables is such that it can be reasonably supposed to have arisen from random sampling. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science, 50*(302), 157–175. https://doi.org/10.1080/14786440009463897


Appendix A:  Python and Dataset Files

Due to Bloomberg licensing, datasets are only available upon request, while the Python codes and Jupyter Notebooks can be found in the GitHub repository: https://github.com/lschneidpro

Besides, the following links are provided for Python development on the ETH clusters:

**Python on the ETH clusters:**
https://scicomp.ethz.ch/wiki/Python

**Configuring submission command to the batch systems of ETH clusters:**
https://scicomp.ethz.ch/lsf_submission_line_advisor/

**Managing Python environments with Miniconda on the ETH clusters:**
https://www.tomstesco.com/euler-hpc-cluster/
http://kevinkle.in/jekyll/update/2019/02/28/leonhard.html

**Jupyter Notebooks on ETH clusters:**
https://gitlab.ethz.ch/sfux/Jupyter-on-Euler-or-Leonhard-Open
Appendix B: Downloading Bloomberg data with Python

The Bloomberg Application Programming Interface (API), namely BLPAPI\(^{23}\), does not provide a native interface in Python, but in C++. Therefore, for a successful installation of the Python module, the following would be needed:

- A local installation of the Bloomberg C++ SDK;
- A C/C++ compiler to build the binary part of the Python module;
- Configuring the system environment variables to locate the C++ SDK.

More information can be found on this GitHub repository\(^{24}\) or in the \textit{WAPI} section of a Bloomberg Terminal. For easier processing of the data, it might also be wise to install another module. Indeed, the API responses are in JSON (JavaScript Object Notation). Different Python wrappers for BLPAPI are available. Tia\(^{25}\) is used here in order to directly have the requested data in Pandas object. Depending on the need, the request to the API can include parameters for type of price adjustments\(^{26}\). Data can be now requested while a Bloomberg terminal is running in the background with the help of their provided functions. They are similar to Bloomberg’s Excel functions are summarised in the table below.

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BDP('security','field')</td>
<td>Bloomberg Data Point (BDP) returns data to a single cell. This formula contains ONLY one security and ONLY one field</td>
</tr>
<tr>
<td>BDH('security','field(s)','start date', 'end date', 'opt arg 1', 'opt arg 2')</td>
<td>Bloomberg Data History (BDH) returns the historical data for selected securities and timeframe.</td>
</tr>
<tr>
<td>BDS('security','field','opt arg 1','opt arg 2')</td>
<td>Bloomberg Data Set (BDS) returns multi-cell descriptive data.</td>
</tr>
</tbody>
</table>

\(^{23}\) Bloomberg API: [https://www.bloomberg.com/professional/support/api-library/](https://www.bloomberg.com/professional/support/api-library/)

\(^{24}\) Bloomberg Python API: [https://github.com/msitt/blpapi-python](https://github.com/msitt/blpapi-python)

\(^{25}\) Tia: [https://github.com/bpsmith/tia](https://github.com/bpsmith/tia)

To complete this list, Bloomberg recently introduced a new syntax, Bloomberg Query Language (BQL) with more advanced functionalities. In addition to retrieving both current and historical data, as BDH and BDP, it is capable of performing complex tasks before retrieval (screening, interval calculations, time series calculations and supports the declaration of custom fields). At the time of writing, BQL is only available through Excel and have been only used to retrieve Point-In-Time (PIT) fundamental data.

To identify a security, Bloomberg uses its own internal Security Identifier (SID) called *Ticker*, but other identifier types can also be passed to the functions, such as CUSIP or ISIN. Few examples to demonstrate how to request Bloomberg’s data are provided below:

**Retrieving the constituents of the Russel 1000 Index on the 31st January 2000:**

```python
1. from tia.bbg import LocalTerminal
2. resp = LocalTerminal.get_reference_data(sids=['RIY Index'],
3.     flds=['INDX_MWEIGHT_HIST'],
4.     END_DATE OVERRIDE='2000/01/31')
5. res = resp.as_frame()
```

**Retrieving the daily closing prices of the Apple stock from the 31st January 2000 to today:**

```python
1. from tia.bbg import LocalTerminal
2. resp = LocalTerminal.get_historical(sids=['AAPL US Equity'],
3.     flds=['PX LAST'],
4.     start='2000/01/31')
5. res = resp.as_frame()
```

**Retrieving the PIT Price/Book ratio of the Apple stock for a range of date with BQL**

```python
1. BQL("AAPL US Equity"; "PX_TO_BOOK_RATIO"; "AS_OF_DATE=RANGE(1990-01-01,1995-01-31)")
```

---

## Appendix C: List of Bloomberg Fields downloaded

### Table C.1 Bloomberg Fields.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>INDX_MWEIGHT_HIST</td>
<td>The list of all the equity members of the index. Historical members can be retrieved by utilising this field in conjunction with the date override field End Date Override.</td>
</tr>
<tr>
<td>NAME</td>
<td>The name of the company</td>
</tr>
<tr>
<td>EQY_INIT_PO_TYP</td>
<td>Initial Public Offer Type. The type of shares offered are common, class A, class B</td>
</tr>
<tr>
<td>EQY_INIT_PO_DT</td>
<td>Initial Public Offer Date, the date the company goes public.</td>
</tr>
<tr>
<td>LAST_UPDATE_DT</td>
<td>Date of Last Update. The date is only available if a trade has occurred in the past six weeks. If the most recent trade occurred before this, blank would be returned.</td>
</tr>
<tr>
<td>LAST_UPDATE_DATE_EOD</td>
<td>End of day value for Date of Last Update</td>
</tr>
<tr>
<td>MARKET_STATUS</td>
<td>Trading status of an equity. Some possible values are: Active, Delisted, Acquired, Unlisted, Suspended, Halted, Private Company, Expired or Postponed</td>
</tr>
<tr>
<td>ID_ISIN</td>
<td>The International Securities Identification Number (ISIN) consists of a two-letter country code, followed by the nine-character alphanumerical national security identifier, and a check digit.</td>
</tr>
<tr>
<td>ID_CUSIP</td>
<td>Security identification number for the U.S. and Canada. The Committee on Uniform Security Identification Procedures (CUSIP) number consists of nine alphanumneric characters. The first six characters identify the issuer, the following two identify the issue, and the final character is a check digit.</td>
</tr>
<tr>
<td>FUNDAMENTALS_TICKER</td>
<td>The ticker to access fundamental equity data for a company</td>
</tr>
<tr>
<td>BICS_LEVEL_1_SEC-</td>
<td>The Bloomberg Industry Classification System</td>
</tr>
<tr>
<td>TOR_NAME</td>
<td>(BICS) level 1 name</td>
</tr>
<tr>
<td>BICS_LEVEL_2_IN-</td>
<td>The Bloomberg Industry Classification System</td>
</tr>
<tr>
<td>DUSTRY_GROUP_NAME</td>
<td>(BICS) level 2 name</td>
</tr>
<tr>
<td>BICS_LEVEL_3_IN-</td>
<td>The Bloomberg Industry Classification System</td>
</tr>
<tr>
<td>DUSTRY_NAME</td>
<td>(BICS) level 3 name</td>
</tr>
<tr>
<td>BICS_LEVEL_4_SUB_IN-</td>
<td>The Bloomberg Industry Classification System</td>
</tr>
<tr>
<td>DUSTRY_NAME</td>
<td>(BICS) level 4 name</td>
</tr>
<tr>
<td>BICS_LEVEL_5_SEG-</td>
<td>The Bloomberg Industry Classification System</td>
</tr>
<tr>
<td>MENT_NAME</td>
<td>(BICS) level 5 name</td>
</tr>
<tr>
<td>PX_OPEN</td>
<td>Price at which the security first traded on trading day</td>
</tr>
<tr>
<td>PX_HIGH</td>
<td>Highest price the security reached during the trading day</td>
</tr>
<tr>
<td>PX_LOW</td>
<td>Lowest price the security reached during the trading day</td>
</tr>
<tr>
<td>PX_LAST</td>
<td>Last price for the security on trading day</td>
</tr>
<tr>
<td>PX_VOLUME</td>
<td>Total number of shares traded on a security on the trading day</td>
</tr>
<tr>
<td>EQY_WEIGHTED_AVG_PX</td>
<td>Volume Weighted Average Price: Trading benchmark calculated by dividing the total value traded (sum of price times trade size) by the total volume (sum of trade sizes), taking into account every qualifying transaction.</td>
</tr>
<tr>
<td>CUR_MKT_CAP</td>
<td>Total current market value of all of a company's outstanding shares stated in the pricing currency.</td>
</tr>
<tr>
<td>HIST_CALL_IMP_VOL</td>
<td>At the money implied volatility of the 1st listed expiry that is at least 20 business days out from today, calculated from a weighted average of the volatilities of the closest out-of-the-money call option.</td>
</tr>
<tr>
<td>CALL_IMP_VOL_10D</td>
<td>Ten days at the money call implied volatility based on the Listed Implied Volatility Engine (LIVE) calculator.</td>
</tr>
<tr>
<td>Symbol</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------</td>
<td>-------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>CALL_IMP_VOL_30D</td>
<td>Thirty days at the money call implied volatility based on the Listed Implied Volatility Engine (LIVE) calculator.</td>
</tr>
<tr>
<td>CALL_IMP_VOL_60D</td>
<td>Sixty days at the money call implied volatility based on the Listed Implied Volatility Engine (LIVE) calculator.</td>
</tr>
<tr>
<td>3MO_CALL_IMP_VOL</td>
<td>Three months at the money call implied volatility based on the Listed Implied Volatility Engine (LIVE) calculator.</td>
</tr>
<tr>
<td>6MO_CALL_IMP_VOL</td>
<td>Six months at the money call implied volatility based on the Listed Implied Volatility Engine (LIVE) calculator.</td>
</tr>
<tr>
<td>12MO_CALL_IMP_VOL</td>
<td>Twelve months at the money call implied volatility based on the Listed Implied Volatility Engine (LIVE) calculator.</td>
</tr>
<tr>
<td>IS_DIL_EPS_BEF_XO</td>
<td>Diluted EPS Before Extraordinary Items: It excludes the effects of discontinued operations, accounting standard changes, and natural disasters. Uses weighted average shares figured as if all of the company's potentially dilutive securities had been changed into shares of common stock.</td>
</tr>
<tr>
<td>PX_TO_BOOK_RATIO</td>
<td>Ratio of the stock price to the book value per share.</td>
</tr>
</tbody>
</table>
Appendix D: Dataset Cleaning: a Detailed Description

High-data quality is critical in a robust strategy development process and obtaining it could be the most time-consuming step. This survey indicates that collecting and cleaning data can be accounted for respectively 19% and 60% of data-scientists work\textsuperscript{28}.

Downloading and Cleaning Historical Index Members

1. End-of-month index members are retrieved from 31\textsuperscript{st} January 1995 (earliest availability on Bloomberg) and set in a binary matrix indicating whether the stock is a constituent of the index.

2. Due to the override date function of Bloomberg, SIDs appears how they were for the given dates, i.e. PIT. As Bloomberg standardised equity data convention\textsuperscript{29} over time, some old SIDs appear to be duplicate, referencing the same underlying security. For instance, \textit{AAPL US} and \textit{APPL UW} refer both to Apple’s stocks but are distinct in our initial binary matrix. All SIDs are renamed with \textit{US Equity} ending, and Booleans data from duplicate SIDs are merged such that continuous historical membership of stocks is preserved.

3. Few SIDs refer to different class of shares\textsuperscript{30} of the same company. Depending on the company, their prices can be highly correlated as they are exposed to the same underlying factors. The aim of this master thesis is to select companies shares in a cross-sectional manner. For this reason, the most common share classes, referred by the field \texttt{FUNDAMENTAL\_TICKER} are kept.

4. Fundamental data is not always available on Bloomberg, especially for older SIDs. As earlier fundamental data might not always be available, Bloomberg tends to refer theses data to the company that acquired the selected company if an acquisition happened. These cases are disregarded with the help of the field \texttt{FUNDAMENTAL\_TICKER}.

\textsuperscript{29} StackExchange: https://quant.stackexchange.com/questions/17157/difference-between-the-two-bloomberg-codes
\textsuperscript{30} Investopedia: https://www.investopedia.com/terms/c/class.asp
Figure D.1 illustrates the effect of the cleaning process. For each month, maximum 1.5% of the original SIDs are removed, and 2993 final SIDs are numbered across the selected timeframe.

![Figure D.1](image)

**Figure D.1** Visualizations of the absolute and relative differences of SIDs over the years after SIDs cleaning.

### Downloading and Cleaning Equity Data

Equity data can be finally downloaded from the cleaned SIDs as follows:

1. Fundamental and Market data are downloaded from Bloomberg. Stock prices are adjusted for dividends, corporate actions and stock splits.
2. Monthly Fama-French 3-/5-Factors are downloaded from Kenneth R. French’s website through the pandas-datareader library.
3. As trading days do not follow the business calendar of pandas due to American holidays and additional market closing days, non-trading days are removed with the NYSE stock exchange calendar from the library pandas-market-calendars.

---

31 Kennet R. French’s website: [http://mba.tuck.dartmouth.edu/pages/faculty/ken.french/data_library.html](http://mba.tuck.dartmouth.edu/pages/faculty/ken.french/data_library.html)
32 Pandas datareader: [https://pandas-datareader.readthedocs.io](https://pandas-datareader.readthedocs.io)
4. If data is missing between earlier and future non-missing data, missing values are filled forward.
5. Fama-French factors and implied volatilities are converted from percentage to decimal to remain consistent with computed returns.
6. Rows full of missing values are dropped.
7. All Pandas objects are converted to type float32 and finally exported as HDF5 for storage efficiency purposes.

The missing data after cleaning are shown in Figure D.2 and spot the lack of availability of implied volatility before 2005.

![Figure D.2 Visualization of missing data patterns over years.]

---

35 Pytables: [https://www.pytables.org/](https://www.pytables.org/)
Appendix E: List of Features

The features are summarised in Table E.1. The computation of each of them and a description of related strategies can be found in Appendix F.
<table>
<thead>
<tr>
<th>Index</th>
<th>Variable Name</th>
<th>Description</th>
<th>Type</th>
<th>Strategy</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>R_1MO</td>
<td>Monthly return</td>
<td>Quantitative</td>
<td>Price-Momentum</td>
</tr>
<tr>
<td>2</td>
<td>R_Cumulative_3MO</td>
<td>Cumulative return with a 1-month skip period and a 3-month formation period</td>
<td>Quantitative</td>
<td>Price-Momentum</td>
</tr>
<tr>
<td>3</td>
<td>R_Mean_3MO</td>
<td>Mean monthly return computed over a 3-month formation period</td>
<td>Quantitative</td>
<td>Price-Momentum</td>
</tr>
<tr>
<td>4</td>
<td>R_Risk_Adjusted_3MO</td>
<td>Risk-adjusted mean monthly return over a 3-month formation period</td>
<td>Quantitative</td>
<td>Price-Momentum</td>
</tr>
<tr>
<td>5</td>
<td>R_Cumulative_6MO</td>
<td>Cumulative return with a 1-month skip period and a 6-month formation period</td>
<td>Quantitative</td>
<td>Price-Momentum</td>
</tr>
<tr>
<td>6</td>
<td>R_Mean_6MO</td>
<td>Mean monthly return computed over a 12-month formation period</td>
<td>Quantitative</td>
<td>Price-Momentum</td>
</tr>
<tr>
<td>7</td>
<td>R_Risk_Adjusted_6MO</td>
<td>Risk-adjusted mean monthly return over a 6-month formation period</td>
<td>Quantitative</td>
<td>Price-Momentum</td>
</tr>
<tr>
<td>8</td>
<td>R_Cumulative_12MO</td>
<td>Cumulative return with a 1-month skip period and a 12-month formation period</td>
<td>Quantitative</td>
<td>Price-Momentum</td>
</tr>
<tr>
<td>9</td>
<td>R_Mean_12MO</td>
<td>Mean monthly return computed over a 12-month formation period</td>
<td>Quantitative</td>
<td>Price-Momentum</td>
</tr>
<tr>
<td>10</td>
<td>R_Risk_Adjusted_12MO</td>
<td>Risk-adjusted mean monthly return over a 12-month formation period</td>
<td>Quantitative</td>
<td>Price-Momentum</td>
</tr>
<tr>
<td>11</td>
<td>Standardized_Unexpected_Earnings</td>
<td>Standardized unexpected earnings</td>
<td>Quantitative</td>
<td>Earnings-Momentum</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>Month_EPS</td>
<td>Labels for month EPS announcement</td>
<td>Qualitative</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>BP_Ratio</td>
<td>Book-to-Price (B/P) ratio</td>
<td>Quantitative</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>VOL_21D</td>
<td>Daily volatility over a 21-day observation period</td>
<td>Quantitative</td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>VOL_126D</td>
<td>Daily volatility over a 126-day observation period</td>
<td>Quantitative</td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>VOL_252D</td>
<td>Daily volatility over a 252-day observation period</td>
<td>Quantitative</td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>VOL_M_21D</td>
<td>Monthly volatility over a 21-day observation period</td>
<td>Quantitative</td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>VOL_M_126D</td>
<td>Monthly volatility over a 126-day observation period</td>
<td>Quantitative</td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>VOL_M_252D</td>
<td>Monthly volatility over a 252-day observation period</td>
<td>Quantitative</td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>VAR_21D</td>
<td>Daily variance over a 21-day observation period</td>
<td>Quantitative</td>
<td></td>
</tr>
<tr>
<td>21</td>
<td>VAR_126D</td>
<td>Daily variance over a 126-day observation period</td>
<td>Quantitative</td>
<td></td>
</tr>
<tr>
<td>22</td>
<td>VAR_252D</td>
<td>Daily variance over a 252-day observation period</td>
<td>Quantitative</td>
<td></td>
</tr>
<tr>
<td>23</td>
<td>VAR_M_21D</td>
<td>Monthly variance over a 21-day observation period</td>
<td>Quantitative</td>
<td></td>
</tr>
<tr>
<td>24</td>
<td>VAR_M_126D</td>
<td>Monthly variance over a 126-day observation period</td>
<td>Quantitative</td>
<td></td>
</tr>
<tr>
<td>25</td>
<td>VAR_M_252D</td>
<td>Monthly variance over a 252-day observation period</td>
<td>Quantitative</td>
<td></td>
</tr>
<tr>
<td>26</td>
<td>CVOL_Diff_10D</td>
<td>First-difference of call implied volatilities for a maturity of 10 days</td>
<td>Quantitative</td>
<td></td>
</tr>
</tbody>
</table>

- Monthly EPS announcement
- Book-to-Price (B/P) ratio
- Daily volatility over a 21-day observation period
- Daily volatility over a 126-day observation period
- Daily volatility over a 252-day observation period
- Monthly volatility over a 21-day observation period
- Monthly volatility over a 126-day observation period
- Monthly volatility over a 252-day observation period
- Daily variance over a 21-day observation period
- Daily variance over a 126-day observation period
- Daily variance over a 252-day observation period
- Monthly variance over a 21-day observation period
- Monthly variance over a 126-day observation period
- Monthly variance over a 252-day observation period
- First-difference of call implied volatilities for a maturity of 10 days
<table>
<thead>
<tr>
<th>No.</th>
<th>Description</th>
<th>Type</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>27</td>
<td>CVOL_Diff_30D</td>
<td>First-difference of call implied volatilities for a maturity of 30 days</td>
<td>Quantitative Implied Volatility</td>
</tr>
<tr>
<td>28</td>
<td>CVOL_Diff_60D</td>
<td>First-difference of call implied volatilities for a maturity of 60 days</td>
<td>Quantitative Implied Volatility</td>
</tr>
<tr>
<td>29</td>
<td>CVOL_Diff_6MO</td>
<td>First-difference of call implied volatilities for a maturity of 6 months</td>
<td>Quantitative Implied Volatility</td>
</tr>
<tr>
<td>30</td>
<td>CVOL_Pct_10D</td>
<td>Percent change of call implied volatilities for a maturity of 10 days</td>
<td>Quantitative Implied Volatility</td>
</tr>
<tr>
<td>31</td>
<td>CVOL_Pct_30D</td>
<td>Percent change of call implied volatilities for a maturity of 30 days</td>
<td>Quantitative Implied Volatility</td>
</tr>
<tr>
<td>32</td>
<td>CVOL_Pct_60D</td>
<td>Percent change of call implied volatilities for a maturity of 60 days</td>
<td>Quantitative Implied Volatility</td>
</tr>
<tr>
<td>33</td>
<td>CVOL_Pct_6MO</td>
<td>Percent change of call implied volatilities for a maturity of 6 months</td>
<td>Quantitative Implied Volatility</td>
</tr>
<tr>
<td>34</td>
<td>CVOL_Spread_10D</td>
<td>Realised-implied volatility spread for a maturity of 10 days</td>
<td>Quantitative Implied Volatility</td>
</tr>
<tr>
<td>35</td>
<td>CVOL_Spread_30D</td>
<td>Realised-implied volatility spread for a maturity of 30 days</td>
<td>Quantitative Implied Volatility</td>
</tr>
<tr>
<td>36</td>
<td>CVOL_Spread_60D</td>
<td>Realised-implied volatility spread for a maturity of 60 days</td>
<td>Quantitative Implied Volatility</td>
</tr>
<tr>
<td>37</td>
<td>CVOL_Spread_6MO</td>
<td>Realised-implied volatility spread for a maturity of 6 months</td>
<td>Quantitative Implied Volatility</td>
</tr>
<tr>
<td>38</td>
<td>CVOL_Cross_10D</td>
<td>Cross-sectional innovation for Quantitative Implied Volatility a maturity of 10 days</td>
<td></td>
</tr>
<tr>
<td>39</td>
<td>CVOL_Cross_30D</td>
<td>Cross-sectional innovation for Quantitative Implied Volatility a maturity of 30 days</td>
<td></td>
</tr>
<tr>
<td>40</td>
<td>CVOL_Cross_60D</td>
<td>Cross-sectional innovation for Quantitative Implied Volatility a maturity of 60 days</td>
<td></td>
</tr>
<tr>
<td>41</td>
<td>CVOL_Cross_6MO</td>
<td>Cross-sectional innovation for Quantitative Implied Volatility a maturity of 6 months</td>
<td></td>
</tr>
<tr>
<td>42</td>
<td>Residual_Momentum_FF3</td>
<td>Risk-adjusted residual returns Quantitative Residual Momentum for 3 Fama-French Factors</td>
<td></td>
</tr>
<tr>
<td>43</td>
<td>Residual_Momentum_FF5</td>
<td>Risk-adjusted residual returns Quantitative Residual Momentum for 5 Fama-French Factors</td>
<td></td>
</tr>
<tr>
<td>44</td>
<td>De-meaned_10D_R1000</td>
<td>10-day demeaned returns Quantitative Mean-Reversion-Single cluster from the Russell 100 Index</td>
<td></td>
</tr>
<tr>
<td>45</td>
<td>De-meaned_10D_SP500</td>
<td>10-day demeaned returns Quantitative Mean-Reversion-Single cluster from the S&amp;P500</td>
<td></td>
</tr>
<tr>
<td>46</td>
<td>De-meaned_21D_R1000</td>
<td>21-day demeaned returns Quantitative Mean-Reversion-Single cluster from the Russell 100 Index</td>
<td></td>
</tr>
<tr>
<td>47</td>
<td>De-meaned_21D_SP500</td>
<td>21-day demeaned returns Quantitative Mean-Reversion-Single cluster from the S&amp;P500</td>
<td></td>
</tr>
<tr>
<td>48</td>
<td>De-meaned_10D_BICS1</td>
<td>10-day demeaned returns Quantitative Mean-Reversion-Multiple Clusters from BICS level 1</td>
<td></td>
</tr>
<tr>
<td>49</td>
<td>De-meaned_10D_BICS2</td>
<td>10-day demeaned returns Quantitative Mean-Reversion-Multiple Clusters from BICS level 2</td>
<td></td>
</tr>
<tr>
<td>50</td>
<td>De-meaned_10D_BICS3</td>
<td>10-day demeaned returns Quantitative Mean-Reversion-Multiple Clusters from BICS level 3</td>
<td></td>
</tr>
<tr>
<td>51</td>
<td>De-meaned_10D_BICS4</td>
<td>10-day demeaned returns Quantitative Mean-Reversion-Multiple Clusters from BICS level 4</td>
<td></td>
</tr>
<tr>
<td>52</td>
<td>De-meaned_10D_BICS5</td>
<td>10-day demeaned returns Quantitative Mean-Reversion-Multiple Clusters from BICS level 5</td>
<td></td>
</tr>
<tr>
<td>53</td>
<td>De-meaned_21D_BICS1</td>
<td>21-day demeaned returns Quantitative Mean-Reversion-Multiple Clusters from BICS level 1</td>
<td></td>
</tr>
<tr>
<td>Page</td>
<td>Description</td>
<td></td>
<td></td>
</tr>
<tr>
<td>------</td>
<td>-------------</td>
<td></td>
<td></td>
</tr>
<tr>
<td>54</td>
<td>De-meaned_21D_BICS2 21-day demeaned returns from BICS level 2 Quantitative Mean-Reversion-Multiple Clusters</td>
<td></td>
<td></td>
</tr>
<tr>
<td>55</td>
<td>De-meaned_21D_BICS3 21-day demeaned returns from BICS level 3 Quantitative Mean-Reversion-Multiple Clusters</td>
<td></td>
<td></td>
</tr>
<tr>
<td>56</td>
<td>De-meaned_21D_BICS4 21-day demeaned returns from BICS level 4 Quantitative Mean-Reversion-Multiple Clusters</td>
<td></td>
<td></td>
</tr>
<tr>
<td>57</td>
<td>De-meaned_121D_BICS5 21-day demeaned returns from BICS level 5 Quantitative Mean-Reversion-Multiple Clusters</td>
<td></td>
<td></td>
</tr>
<tr>
<td>58</td>
<td>Signal_SMA_30D Signals for 30-day SMA Qualitative Single Moving Average</td>
<td></td>
<td></td>
</tr>
<tr>
<td>59</td>
<td>Signal_SMA_50D Signals for 50-day SMA Qualitative Single Moving Average</td>
<td></td>
<td></td>
</tr>
<tr>
<td>60</td>
<td>Signal_SMA_100D Signals for 100-day SMA Qualitative Single Moving Average</td>
<td></td>
<td></td>
</tr>
<tr>
<td>61</td>
<td>Signal_SMA_200D Signals for 200-day SMA Qualitative Single Moving Average</td>
<td></td>
<td></td>
</tr>
<tr>
<td>62</td>
<td>Signal_EMA_30D Signals for 30-day EMA Qualitative Single Moving Average</td>
<td></td>
<td></td>
</tr>
<tr>
<td>63</td>
<td>Signal_EMA_50D Signals for 50-day EMA Qualitative Single Moving Average</td>
<td></td>
<td></td>
</tr>
<tr>
<td>64</td>
<td>Signal_EMA_100D Signals for 100-day EMA Qualitative Single Moving Average</td>
<td></td>
<td></td>
</tr>
<tr>
<td>65</td>
<td>Signal_EMA_200D Signals for 200-day EMA Qualitative Single Moving Average</td>
<td></td>
<td></td>
</tr>
<tr>
<td>66</td>
<td>Signal_2MA_10_30D Signals for 10/30 day SMAs Qualitative Two Moving Averages</td>
<td></td>
<td></td>
</tr>
<tr>
<td>67</td>
<td>Signal_2MA_10_50D Signals for 10/50 day SMAs Qualitative Two Moving Averages</td>
<td></td>
<td></td>
</tr>
<tr>
<td>68</td>
<td>Signal_2MA_30_50D Signals for 30/50 day SMAs Qualitative Two Moving Averages</td>
<td></td>
<td></td>
</tr>
<tr>
<td>69</td>
<td>Signal_2MA_30_100D Signals for 30/100 day SMAs Qualitative Two Moving Averages</td>
<td></td>
<td></td>
</tr>
<tr>
<td>No.</td>
<td>Signal Name</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>-----</td>
<td>---------------------------------</td>
<td>--------------------------------------</td>
<td></td>
</tr>
<tr>
<td>70</td>
<td>Signal_2MA_30_200D</td>
<td>Signals for 30/200 day SMAs</td>
<td></td>
</tr>
<tr>
<td>71</td>
<td>Signal_2MA_50_100D</td>
<td>Signals for 50/100 day SMAs</td>
<td></td>
</tr>
<tr>
<td>72</td>
<td>Signal_2MA_50_200D</td>
<td>Signals for 50/200 day SMAs</td>
<td></td>
</tr>
<tr>
<td>73</td>
<td>Signal_2MA_100_200D</td>
<td>Signals for 100/200 day SMAs</td>
<td></td>
</tr>
<tr>
<td>74</td>
<td>Signal_3MA_5_10_20D</td>
<td>Signals for 5/10/20 day SMAs</td>
<td></td>
</tr>
<tr>
<td>75</td>
<td>Signal_3MA_5_20_60D</td>
<td>Signals for 5/20/60 day SMAs</td>
<td></td>
</tr>
<tr>
<td>76</td>
<td>Signal_3MA_10_30_50D</td>
<td>Signals for 10/30/50 day SMAs</td>
<td></td>
</tr>
<tr>
<td>77</td>
<td>Signal_3MA_30_60_120D</td>
<td>Signals for 30/60/120 day SMAs</td>
<td></td>
</tr>
<tr>
<td>78</td>
<td>Signal_3MA_30_120_200D</td>
<td>Signals for 30/120/200 day SMAs</td>
<td></td>
</tr>
<tr>
<td>79</td>
<td>Signal_3MA_60_90_120D</td>
<td>Signals for 60/90/120 day SMAs</td>
<td></td>
</tr>
<tr>
<td>80</td>
<td>Signal_3MA_60_120_200D</td>
<td>Signals for 60/120/200 day SMAs</td>
<td></td>
</tr>
<tr>
<td>81</td>
<td>Signal_3MA_90_120_200D</td>
<td>Signals for 90/120/200 day SMAs</td>
<td></td>
</tr>
<tr>
<td>82</td>
<td>Signal_Support Resistance_10D</td>
<td>Signals for 10-day Support Resistance Indicator</td>
<td></td>
</tr>
<tr>
<td>Page</td>
<td>Signal</td>
<td>Starts With</td>
<td>Type</td>
</tr>
<tr>
<td>------</td>
<td>--------</td>
<td>-------------</td>
<td>------</td>
</tr>
<tr>
<td>83</td>
<td>Support Resistance 20D</td>
<td>Signals for 20-day Support Resistance Indicator</td>
<td>Qualitative</td>
</tr>
<tr>
<td>84</td>
<td>Support Resistance 30D</td>
<td>Signals for 30-day Support Resistance Indicator</td>
<td>Qualitative</td>
</tr>
<tr>
<td>85</td>
<td>Support Resistance 60D</td>
<td>Signals for 60-day Support Resistance Indicator</td>
<td>Qualitative</td>
</tr>
<tr>
<td>86</td>
<td>Support Resistance 90D</td>
<td>Signals for 90-day Support Resistance Indicator</td>
<td>Qualitative</td>
</tr>
<tr>
<td>87</td>
<td>Channel 10D</td>
<td>Signals for 10-day Channel Indicator</td>
<td>Qualitative</td>
</tr>
<tr>
<td>88</td>
<td>Channel 20D</td>
<td>Signals for 20-day Channel Indicator</td>
<td>Qualitative</td>
</tr>
<tr>
<td>89</td>
<td>Channel 30D</td>
<td>Signals for 30-day Channel Indicator</td>
<td>Qualitative</td>
</tr>
<tr>
<td>90</td>
<td>Channel 60D</td>
<td>Signals for 60-day Channel Indicator</td>
<td>Qualitative</td>
</tr>
<tr>
<td>91</td>
<td>Channel 90D</td>
<td>Signals for 90-day Channel Indicator</td>
<td>Qualitative</td>
</tr>
<tr>
<td>92</td>
<td>KNN_Prediction</td>
<td>Forecasts of monthly returns with k-NN</td>
<td>Quantitative</td>
</tr>
</tbody>
</table>
Appendix F: Equity Strategies Computation

The following strategies were derived from the Equity section of (Kakushadze & Serur, 2018). For each strategy, an overview and a few variants are proposed based on academic papers to expand the predictors base. The mathematical notation of the authors is used here.

Table F.1 List of Equity Strategies.

<table>
<thead>
<tr>
<th>Index</th>
<th>Strategy</th>
<th>Variants computed</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Price-Momentum</td>
<td>10</td>
</tr>
<tr>
<td>2</td>
<td>Earnings-Momentum</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>Value</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>Low-Volatility Anomaly</td>
<td>12</td>
</tr>
<tr>
<td>5</td>
<td>Implied Volatility</td>
<td>16</td>
</tr>
<tr>
<td>6</td>
<td>Multifactor Portfolio</td>
<td>0</td>
</tr>
<tr>
<td>7</td>
<td>Residual Momentum</td>
<td>2</td>
</tr>
<tr>
<td>8</td>
<td>Pairs Trading</td>
<td>0</td>
</tr>
<tr>
<td>9</td>
<td>Mean-Reversion-Single Cluster</td>
<td>4</td>
</tr>
<tr>
<td>10</td>
<td>Mean-Reversion-Multiple Clusters</td>
<td>10</td>
</tr>
<tr>
<td>11</td>
<td>Mean-Reversion-Weighted Regression</td>
<td>0</td>
</tr>
<tr>
<td>12</td>
<td>Single Moving Average</td>
<td>8</td>
</tr>
<tr>
<td>13</td>
<td>Two Moving Average</td>
<td>8</td>
</tr>
<tr>
<td>14</td>
<td>Three Moving averages</td>
<td>8</td>
</tr>
<tr>
<td>15</td>
<td>Support and Resistance</td>
<td>4</td>
</tr>
<tr>
<td>16</td>
<td>Channel</td>
<td>4</td>
</tr>
<tr>
<td>17</td>
<td>Event-Driven-M&amp;A</td>
<td>0</td>
</tr>
<tr>
<td>18</td>
<td>Machine-Learning-Single-Stock KNN</td>
<td>1</td>
</tr>
<tr>
<td>19</td>
<td>Statistical Arbitrage-Optimization</td>
<td>0</td>
</tr>
<tr>
<td>20</td>
<td>Market-Timing</td>
<td>0</td>
</tr>
<tr>
<td>21</td>
<td>Alpha Combos</td>
<td>0</td>
</tr>
</tbody>
</table>
Strategy 1: Price-Momentum

The momentum describes the inertia effect in stock returns, where past performing stocks would continue to perform well and vice-versa. The rationale for such an effect points to behavioural biases behaviour (initial underreaction and delayed overreaction over information dissemination), persistent supply, demand imbalances, a positive feedback loop between risk assets and the economy, or market microstructure (Jansen, 2018). The strategy then amounts to buying the best past performer stocks and selling the worst past performers. The most recent month is usually skipped due to a mean-reversion effect empirically observed in monthly-returns (Kakushadze & Serur, 2018). Let $t$ denote time measured in units of 1 month, with $t = 0$ corresponding to the most recent time; $P_i(t)$ is the time series of adjusted closing prices for stock $i$; $R_i(t)$ is the monthly return:

$$R_i(t) = \frac{P_i(t)}{P_i(t+1)} - 1$$

$R_i^{cum}$ is the cumulative return computed over the $T$-month formation period (usually $T = 12$) skipping the most recent $S$-month skip period (usually $S = 1$):

$$R_i^{cum} = \frac{P_i(S)}{P_i(S+T)} - 1$$

$R_i^{mean}$ is the mean monthly return computed over the formation period:

$$R_i^{mean} = \frac{1}{T} \sum_{t=S}^{S+T-1} R_i(t)$$

$\sigma_i$ is the monthly volatility calculated over the formation period:

$$\sigma_i^2 = \frac{1}{T-1} \sum_{t=S}^{S+T-1} (R_i(t) - R_i^{mean})^2$$

And $R_i^{risk.adj}$ is the risk-adjusted mean monthly return over the formation period:

$$R_i^{risk.adj} = \frac{R_i^{mean}}{\sigma_i}$$

The predictors are subsequently computed for, $T \in \{1,3,6,12\}$ as shown in Table F-2.
Table F.2 Price-Momentum variants.

<table>
<thead>
<tr>
<th>Variable name</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>R_1MO</td>
<td>$R_i(1)$</td>
<td>Monthly return</td>
</tr>
<tr>
<td>R_Cumulative_3MO</td>
<td>$R_i^{cum}$ for $T = 3$</td>
<td>Cumulative return with a 1-month skip period and a 3-months formation period</td>
</tr>
<tr>
<td>R_Mean_3MO</td>
<td>$R_i^{mean}$ for $T = 3$</td>
<td>Mean monthly return computed over 3-months formation period</td>
</tr>
<tr>
<td>R_Risk_Adjusted_3MO</td>
<td>$R_i^{risk,adj}$ for $T = 3$</td>
<td>Risk-adjusted mean monthly return over 3-months formation period</td>
</tr>
<tr>
<td>R_Cumulative_6MO</td>
<td>$R_i^{cum}$ for $T = 6$</td>
<td>Cumulative return with a 1-month skip period and a 6-months formation period</td>
</tr>
<tr>
<td>R_Mean_6MO</td>
<td>$R_i^{mean}$ for $T = 6$</td>
<td>Mean monthly return computed over 12-months formation period</td>
</tr>
<tr>
<td>R_Risk_Adjusted_6MO</td>
<td>$R_i^{risk,adj}$ for $T = 6$</td>
<td>Risk-adjusted mean monthly return over 6-months formation period</td>
</tr>
<tr>
<td>R_Cumulative_12MO</td>
<td>$R_i^{cum}$ for $T = 12$</td>
<td>Cumulative return with a 1-month skip period and a 12-months formation period</td>
</tr>
<tr>
<td>R_Mean_12MO</td>
<td>$R_i^{mean}$ for $T = 12$</td>
<td>Mean monthly return computed over 12-months formation period</td>
</tr>
<tr>
<td>R_Risk_Adjusted_12MO</td>
<td>$R_i^{risk,adj}$ for $T = 12$</td>
<td>Risk-adjusted mean monthly return over 12-months formation period</td>
</tr>
</tbody>
</table>

Strategy 2: Earnings-Momentum

Momentum strategies can also be transposed to Earnings Per Share (EPS). Due to the seasonal aspect of business, seasonally different quarterly earnings show a correlation from one quarter to the next (Foster, 1977). As such, whenever a quarterly EPS is up compared to the quarter of the prior year, the next quarter would generate higher EPS expectation from analysts. Seasonal random walks with a trend have been then proposed to model unexpected EPS (Bernard & Thomas, 1989; Foster, Olsen, & Shevlin, 1984)). To capture deviation from this serial correlation and associated future returns, the Standardized Unexpected Earnings (SUE) definition of (L. K. C. Chan, Jegadeesh, & Lakonishok, 1996) is used here. As before, the strategy consists of buying a top percentile and short a bottom percentile. Let $t$ denote time measured in the units
of 1 quarter, with \( t = 0 \) corresponding to the most recent time; \( E_i \) is the most recently announced quarterly EPS of the stock \( i \); \( E_i' \) is the EPS announced four quarters ago; \( \sigma_i \) is the standard deviation of the unexpected earnings \( E_i - E_i' \) over the last eight quarters:

\[
SUE_i = \frac{E_i - E_i'}{\sigma_i}
\]

In addition, as the paper framework is monthly, and returns might be diminishing along holding time (Kakushadze & Serur, 2018), months are labelled according to the previous EPS announcement month:

\[
Label_i \triangleq \begin{cases} 
0 & \text{if }E_i \text{ announced the same month} \\
1 & \text{if }E_i \text{ announced one month ago} \\
2 & \text{if }E_i \text{ announced two months ago}
\end{cases}
\]

The predictors are subsequently computed, as shown in Table F.3:

<table>
<thead>
<tr>
<th>Variable name</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Standardized_Unexpected_Earnings</td>
<td>( SUE_i )</td>
<td>Standardised unexpected earnings</td>
</tr>
<tr>
<td>Month_EPS</td>
<td>( Label_i )</td>
<td>Labels for EPS announcement month</td>
</tr>
</tbody>
</table>

Bloomberg’s field Diluted EPS Before Extraordinary Items\(^{36}\) is used here instead of basic EPS as a precautionary approach. Indeed, diluted EPS take into account securities that can be turned into share while extraordinary items might not have desired impacts.

**Strategy 3: Value**

Value strategies rely on selecting stocks based on accounting-based fundamental metrics and estimation of asset’s fair values. Stocks with low prices relative to their fundamental value, i.e. cheap stocks called *value stocks*, tend to deliver higher returns than the market portfolio (Jansen, 2018). (Rosenberg, Reid, & Lanstein, 1985) shows that a portfolio consisting of buying stocks with high ratio of book value of common equity per share to the market price, i.e.

Book-to-Price ratio (B/P ratio), and selling their highly valuated counterparts, called growth stocks, deliver abnormal returns and goes against market efficiency. Such a phenomenon can be explained by increased premiums, compensating for higher risk and low-performance expectations from analysts (Piotroski, 2006). As before, the strategy consists of buying a top percentile group of stock and short a bottom one. Let denote time measured in units of 1 month, with 0 corresponding to the most recent time; the stock is labelled by : 

\[
B/P\text{ ratio}_i = \frac{\text{Book value per share}_i}{P_i}
\]

The predictor BP_Ratio is subsequently computed, as shown in Table F.4:

<table>
<thead>
<tr>
<th>Variable name</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BP_Ratio</td>
<td>B/P ratio_i</td>
<td>Book-to-Price (B/P) ratio</td>
</tr>
</tbody>
</table>

### Strategy 4: Low Volatility Anomaly

Simple anomalies can challenge the Efficient-Market Hypothesis (EMH) (Fama, 1970). Low-risk stocks exhibit significantly higher risk-adjusted returns than the market portfolio, while high-risk stocks underperform it in the U.S. and across the world (Ang, Hodrick, Xing, & Zhang, 2006, 2009). (Baker, Bradley, & Wurgler, 2011) attributes this anomaly to behavioural finance biases such as fund managers with incentive towards risky assets. As before, the strategy consists of buying a top percentile and short a bottom percentile. Let denote time measured in units of 1 day, with 0 corresponding to the most recent time; is the time series of adjusted closing prices for a stock ; is the cumulative daily return computed over the day formation period:

\[
R_{i\text{cum}}(t) = \frac{P(t)}{P(t + T)} - 1
\]

is the mean return computed over day observation period:

\[
R_{i\text{mean}} = \frac{1}{W} \sum_{t=0}^{W-1} R_{i\text{cum}}(t)
\]

\(\sigma_i\) is the monthly volatility calculated over the observation period:
The predictors are subsequently computed for $T \in \{1, 21\}$ and $W \in \{21, 126, 252\}$ as shown in Table F-5:

**Table F.5** Low volatility variants.

<table>
<thead>
<tr>
<th>Feature name</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>VOL_21D</td>
<td>$\sigma_i$ for $T = 1, W = 21$</td>
<td>Daily volatility over 21-days observation period</td>
</tr>
<tr>
<td>VOL_126D</td>
<td>$\sigma_i$ for $T = 1, W = 126$</td>
<td>Daily volatility over 126-days observation period</td>
</tr>
<tr>
<td>VOL_252D</td>
<td>$\sigma_i$ for $T = 1, W = 252$</td>
<td>Daily volatility over 252-days observation period</td>
</tr>
<tr>
<td>VOL_M_21D</td>
<td>$\sigma_i$ for $T = 21, W = 21$</td>
<td>Monthly volatility over 21-days observation period</td>
</tr>
<tr>
<td>VOL_M_126D</td>
<td>$\sigma_i$ for $T = 21, W = 126$</td>
<td>Monthly volatility over 126-days observation period</td>
</tr>
<tr>
<td>VOL_M_252D</td>
<td>$\sigma_i$ for $T = 21, W = 252$</td>
<td>Monthly volatility over 252-days observation period</td>
</tr>
<tr>
<td>VAR_21D</td>
<td>$\sigma_i^2$ for $T = 1, W = 21$</td>
<td>Daily variance over 21-days observation period</td>
</tr>
<tr>
<td>VAR_126D</td>
<td>$\sigma_i^2$ for $T = 1, W = 126$</td>
<td>Daily variance over 126-days observation period</td>
</tr>
<tr>
<td>VAR_252D</td>
<td>$\sigma_i^2$ for $T = 1, W = 252$</td>
<td>Daily variance over 252-days observation period</td>
</tr>
<tr>
<td>VAR_M_21D</td>
<td>$\sigma_i^2$ for $T = 21, W = 21$</td>
<td>Monthly variance over 21-days observation period</td>
</tr>
<tr>
<td>VAR_M_126D</td>
<td>$\sigma_i^2$ for $T = 21, W = 126$</td>
<td>Monthly variance over 126-days observation period</td>
</tr>
<tr>
<td>VAR_M_252D</td>
<td>$\sigma_i^2$ for $T = 21, W = 252$</td>
<td>Monthly variance over 252-days observation period</td>
</tr>
</tbody>
</table>

The realised variance is also included, as (Kakushadze & Serur, 2018) often mentions allocating stock proportionally to their volatility, and their variance as well.
Strategy 5: Implied Volatility

The option prices can contain predictive information about stock returns. Consequently, (An, Ang, Bali, & Cakici, 2014) observed that stocks with more significant increases in implied volatilities for call options over the previous month have higher future returns on average, while stocks with implied volatilities for put options increases have lower future returns. As before, the strategy consists of buying a top percentile and short a bottom percentile. Other measures of innovation volatility from (An et al., 2014) have were added to the set of predictors. Let $t$ denote time measured in the units of month, with $t = 0$ corresponding to the most recent time; $CVOL_i(t)$ is the time series of call implied volatilities for a stock $i$ for maturity $M$; $\Delta CVOL_{i,M}(t)$ is the first-difference of call implied volatilities:

$$\Delta CVOL_{i,M}(t) = CVOL_{i,M}(t) - CVOL_{i,M}(t+1)$$

$\%\Delta CVOL_{i,M}(t)$ is the percent change of call implied volatilities:

$$\%\Delta CVOL_{i,M}(t) = \frac{\Delta CVOL_{i,M}(t) - \Delta CVOL_{i,M}(t+1)}{\Delta CVOL_{i,M}(t+1)}$$

$\sigma_{i,M}$ is the daily volatility computed over $M$-day window; $RVOL-IVOL_{i,M}$ is the realised-implied volatility spread:

$$RVOL-IVOL_{i,M} = \sigma_{i,M} - CVOL_{i,M}$$

$CVOL_{i,M}^{cs/shock}$ is the cross-sectional innovation:

$$CVOL_{i,M}^{cs/shock} = \epsilon_j$$

Where the residual $\epsilon_j$ is estimated from the cross-sectional regression:

$$CVOL_{i,M}(t) = \alpha(t) + \beta(t)CVOL_{i,M}(t+1) + \epsilon_j(t)$$

The predictors are subsequently computed daily and monthly for maturities of 10, 30 and 60 days and of 6 months, as shown in Table F.6:

<table>
<thead>
<tr>
<th>Variable name</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CVOL_Diff_10D</td>
<td>$\Delta CVOL_{i,10D}$</td>
<td>First-difference of call implied volatilities for a maturity of 10 days</td>
</tr>
<tr>
<td>Variable</td>
<td>Formula</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------</td>
<td>----------------------------------------------</td>
<td>-----------------------------------------------</td>
</tr>
<tr>
<td>CVOL_Diff_30D</td>
<td>$\Delta CVOL_{i,30D}$</td>
<td>First-difference of call implied volatilities for a maturity of 30 days</td>
</tr>
<tr>
<td>CVOL_Diff_60D</td>
<td>$\Delta CVOL_{i,60D}$</td>
<td>First-difference of call implied volatilities for a maturity of 60 days</td>
</tr>
<tr>
<td>CVOL_Diff_6MO</td>
<td>$\Delta CVOL_{i,6MO}$</td>
<td>First-difference of call implied volatilities for a maturity of 6 months</td>
</tr>
<tr>
<td>CVOL_Pct_10D</td>
<td>$%\Delta CVOL_{i,30D}$</td>
<td>Percent change of call implied volatilities for a maturity of 10 days</td>
</tr>
<tr>
<td>CVOL_Pct_30D</td>
<td>$%\Delta CVOL_{i,30D}$</td>
<td>Percent change of call implied volatilities for a maturity of 30 days</td>
</tr>
<tr>
<td>CVOL_Pct_60D</td>
<td>$%\Delta CVOL_{i,60D}$</td>
<td>Percent change of call implied volatilities for a maturity of 60 days</td>
</tr>
<tr>
<td>CVOL_Pct_6MO</td>
<td>$%\Delta CVOL_{i,6MO}$</td>
<td>Percent change of call implied volatilities for a maturity of 6 months</td>
</tr>
<tr>
<td>CVOL_Spread_10D</td>
<td>$RVOL - IVOL_{i,30D}$</td>
<td>Realised-implied volatility spread for a maturity of 10 days</td>
</tr>
<tr>
<td>CVOL_Spread_30D</td>
<td>$RVOL - IVOL_{i,30D}$</td>
<td>Realised-implied volatility spread for a maturity of 30 days</td>
</tr>
<tr>
<td>CVOL_Spread_60D</td>
<td>$RVOL - IVOL_{i,60D}$</td>
<td>Realised-implied volatility spread for a maturity of 60 days</td>
</tr>
<tr>
<td>CVOL_Spread_6MO</td>
<td>$RVOL - IVOL_{i,6MO}$</td>
<td>Realised-implied volatility spread for a maturity of 6 months</td>
</tr>
<tr>
<td>CVOL_Cross_10D</td>
<td>$CVOL_{cs/shock}^{i,30D}$</td>
<td>Cross-sectional innovation for a maturity of 10 days</td>
</tr>
<tr>
<td>CVOL_Cross_30D</td>
<td>$CVOL_{cs/shock}^{i,30D}$</td>
<td>Cross-sectional innovation for a maturity of 30 days</td>
</tr>
<tr>
<td>CVOL_Cross_60D</td>
<td>$CVOL_{cs/shock}^{i,60D}$</td>
<td>Cross-sectional innovation for a maturity of 60 days</td>
</tr>
<tr>
<td>CVOL_Cross_6MO</td>
<td>$CVOL_{cs/shock}^{i,6MO}$</td>
<td>Cross-sectional innovation for a maturity of 6 months</td>
</tr>
</tbody>
</table>

Put implied volatilities, $PVOL$, are disregarded as the difference between $PVOL$ and $CVOL$ downloaded from Bloomberg is null most of the time for the selected timeframe.
Strategy 6: Multifactor Portfolio

This strategy consists of ranking stocks based on multiple factors, such as the previously mentioned ones. However, it is disregarded here as the machine learning framework is expected to rank itself the strategies.

Strategy 7: Residual Momentum

The performance of Momentum strategies is linked to the Fama-French factors. These strategies are then exposed to losses whenever such factors switch signs. To remove momentum dependency on systemic factors, (Blitz, Huij, & Martens, 2011) propose a momentum strategy on the residuals. As before, the strategy consists of buying a top percentile and short a bottom percentile. Let \( t \) denote time measured in units of 1 month, with \( t = 0 \) corresponding to the most recent time; \( R_i(t) \) is the monthly return for a stock \( i \); \( R_{free}(t) \) is the risk-free monthly rate; \( X_{FF3}(t) \) and \( X_{FF5}(t) \) are the 3-/5 Fama-French factors from Kenneth R. French’s website\(^{37}\):

\[
X_{FF3}(t) = \begin{pmatrix}
MKT(t) \\
SMB(t) \\
HML(t)
\end{pmatrix}
X_{FF5}(t) = \begin{pmatrix}
MKT(t) \\
SMB(t) \\
HML(t) \\
RMW(t) \\
CMA(t)
\end{pmatrix}
\]

The serial regression is run over a first \( T_1 \)-month observation period (here \( T_1 = 36 \)) with a \( S \)-month skip period (here \( S = 1 \)) to estimate the regression coefficients \( \alpha_i, \beta_{1,i}, \ldots, \beta_{N,i} \) for \( i = 1, \ldots, I \):

\[
R_i(t) - R_{free}(t) = \alpha_i + \sum_{n=1}^{N} \beta_{n,i} X_n(t) + \epsilon_i(t) \quad \text{for} \quad t = S, S + 1, \ldots, T_1
\]

The residuals \( \epsilon_i(t) \) are estimated over a second \( T_2 \)-month observation period (here \( T_2 = 12 \)) with again a \( S \)-month skip period and by dropping the intercept \( \alpha_i \):

\[ \varepsilon_i(t) = R_i(t) - R_{free}(t) - \sum_{n=1}^{N} \beta_{nj} X_n(t) \quad \text{for } t = S, S+1, \ldots, T_2 \]

The residuals are then used to compute the risk-adjusted residual returns \( \tilde{R}^{\text{risk adj}}_i \).

\[ \varepsilon_i^{\text{mean}} = \frac{1}{T} \sum_{t=S}^{S+T-1} \varepsilon_i(t) \]
\[ \tilde{\sigma}_i^2 = \frac{1}{T-1} \sum_{t=S}^{S+T-1} \left( \varepsilon_i(t) - \varepsilon_i^{\text{mean}} \right)^2 \]
\[ \tilde{R}^{\text{risk adj}}_i = \frac{\varepsilon_i^{\text{mean}}}{\tilde{\sigma}_i} \]

The predictors are subsequently computed for the 3-and 5 Fama-French factors as shown in- Table F.7:

<table>
<thead>
<tr>
<th>Variable name</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Residual_Momentum_FF3</td>
<td>( \tilde{R}^{\text{risk adj}}_i ) for FF3</td>
<td>Risk-adjusted residual returns for the 3 Fama-French Factors</td>
</tr>
<tr>
<td>Residual_Momentum_FF5</td>
<td>( \tilde{R}^{\text{risk adj}}_i ) for FF5</td>
<td>Risk-adjusted residual returns for the 5 Fama-French Factors</td>
</tr>
</tbody>
</table>

**Strategy 8: Pairs Trading**

This strategy attempts to capture mispricing from a pair of historically correlated stocks. However, it is disregarded here as the stock universe keeps changing each month. However, it is extended to clusters, as explained below.

**Strategy 9: Mean-Reversion-Single and Multiple Clusters**

Mean-reversion strategies, or contrarian strategies, assume that prices of correlated stocks follow an underlying trend and prices would eventually revert if stocks are temporarily under- or over-priced with respect to the other related securities. Consequently, the strategy properties are opposite to those of momentum factors (Jansen, 2018) and are associated with market over-reaction (Avellaneda & Lee, 2010). It consists then to buy the underpriced securities and sell the overpriced ones. Let \( t \) denote time measured in units of 1 day, with \( t = 0 \) corresponding to
the most recent time; $P_i(t)$ is the time series of adjusted closing prices for a stock $i$; $R_i(t)$ is the logarithmic return computed over the $T$-day formation:

$$R_i(t) = \ln \left( \frac{P_i(t)}{P_i(t+T)} \right)$$

**Single Cluster**

For a single cluster of highly correlated stocks, the demeaned return $\tilde{R}_i$ is derived by removing the cluster's average return $\bar{R}$:

$$\tilde{R}_i = R_i - \bar{R}$$

The predictors are subsequently computed by subtracting returns from benchmarks for $T \in \{10, 21\}$ as shown in Table F.8:

**Table F.8** Single cluster variants.

<table>
<thead>
<tr>
<th>Variable name</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>De-meaned_10D_R1000</td>
<td>$\tilde{R}_i^{R1000}$ for $T = 10$</td>
<td>10-days demeaned returns from the Russell 100 Index</td>
</tr>
<tr>
<td>De-meaned_10D_SP500</td>
<td>$\tilde{R}_i^{SP500}$ for $T = 10$</td>
<td>10-days demeaned returns from the S&amp;P500</td>
</tr>
<tr>
<td>De-meaned_21D_R1000</td>
<td>$\tilde{R}_i^{R1000}$ for $T = 21$</td>
<td>21-days demeaned returns from the Russell 100 Index</td>
</tr>
<tr>
<td>De-meaned_21D_SP500</td>
<td>$\tilde{R}_i^{SP500}$ for $T = 21$</td>
<td>21-days demeaned returns from the S&amp;P500</td>
</tr>
</tbody>
</table>

**Multiple Clusters**

The previous strategy can be extended to several clusters. For $K$ clusters of historically highly correlated stocks (such as industry group), labelled $A = 1, \ldots, K$, $N$ stocks can be associated through the $N \times K$ binary loading matrix $\Lambda$ where $\Lambda_{i,A} = 1$ for stock $i$ belonging to cluster $A$; otherwise $\Lambda_{i,A} = 0$, such that:

$$N_A = \sum_{i=1}^{N} \Lambda_{i,A} > 0$$

$G$ is the map between stocks and clusters:

$$G : \{1, \ldots, N\} \rightarrow \{1, \ldots, K\}$$
The set $J_A$ comprises the stocks belonging to cluster $A$ such that:

$$J_A = \{G(i) = A\} \subset \{1, \ldots, N\}$$

$\overline{R}_A$ is the mean return for a cluster $A = G(i)$:

$$\overline{R}_A = \frac{1}{N_A} \sum_{j \in J_A} R_j$$

The demeaned return $\tilde{R}_i$ is obtained by subtracting the cluster mean return:

$$\tilde{R}_i = R_i - \overline{R}_{G(i)}$$

The predictors are subsequently computed with clusters defined as the five different levels of the Bloomberg Industry Classification System (BICS)\textsuperscript{38} for $T \in \{10, 21\}$ as shown in Table F.9:

<table>
<thead>
<tr>
<th>Variable name</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Demeaned_10D_BICS1</td>
<td>$\tilde{R}^{BICS1}_i$ for $T = 10$</td>
<td>10-days demeaned returns from BICS level 1</td>
</tr>
<tr>
<td>Demeaned_10D_BICS2</td>
<td>$\tilde{R}^{BICS2}_i$ for $T = 10$</td>
<td>10-days demeaned returns from BICS level 2</td>
</tr>
<tr>
<td>Demeaned_10D_BICS3</td>
<td>$\tilde{R}^{BICS3}_i$ for $T = 10$</td>
<td>10-days demeaned returns from BICS level 3</td>
</tr>
<tr>
<td>Demeaned_10D_BICS4</td>
<td>$\tilde{R}^{BICS4}_i$ for $T = 10$</td>
<td>10-days demeaned returns from BICS level 4</td>
</tr>
<tr>
<td>Demeaned_10D_BICS5</td>
<td>$\tilde{R}^{BICS5}_i$ for $T = 10$</td>
<td>10-days demeaned returns from BICS level 5</td>
</tr>
<tr>
<td>Demeaned_21D_BICS1</td>
<td>$\tilde{R}^{BICS1}_i$ for $T = 21$</td>
<td>21-days demeaned returns from BICS level 1</td>
</tr>
<tr>
<td>Demeaned_21D_BICS2</td>
<td>$\tilde{R}^{BICS2}_i$ for $T = 21$</td>
<td>21-days demeaned returns from BICS level 2</td>
</tr>
<tr>
<td>Demeaned_21D_BICS3</td>
<td>$\tilde{R}^{BICS3}_i$ for $T = 21$</td>
<td>21-days demeaned returns from BICS level 3</td>
</tr>
</tbody>
</table>

\textsuperscript{38} More details in Appendix C
Strategic 10: Mean-Reversion–Weighted Regression

The previous strategy assumed a binary loading matrix and can be extended to any non-binary matrix. However, it is disregarded here as no relevant inputs could be found.

Strategy 11: Single Moving Average

This strategy and the following ones belong to the field of technical analysis which (Murphy, 1999) define as the forecasting of future prices trends, primarily with charts, with three principal sources of information available: price, volume and open interest. The early principles have been compiled initially at the end of 19th century by Charles H. Dow in the Dow theory, and can be summarised with the following three main assumptions:

- The market discounts everything and prices might be affected politically, fundamentally, psychologically, and others.
- The price move in trends and can be identified with trend-following indicators.
- The history repeats itself, and past successful patterns should continue based on human psychology.

Despite its success, the field remains controverted among academics (Brock, Lakonishok, & LeBaron, 1992) and provides to be less effective in more extended frequency trading strategies in opposition to fundamental analysis. As an example, (Hsu & Kuan, 2005) examine the profitability of such strategies in different markets and add proof to its lack of effectiveness in mature markets.

The Moving Averages (MA) are one of the most widely used technical indicators. In the statistics field, it is a type of finite impulse response filters used to analyse a set of data points by

\[ \hat{R}_t^{\text{BICS4}} \text{ for } T = 21 \]

\[ \hat{R}_t^{\text{BICS5}} \text{ for } T = 21 \]

creating a series of averages of different subsets of the full data set. A moving average is commonly used with time-series data to smooth out short-term fluctuations and highlight longer-term trends or cycles. The threshold between short-term and long-term depends on the application, and the parameters of the moving average are set accordingly. Let \( t \) denote time measured in units of 1 day, with \( t = 1 \) corresponding to the most recent time; \( P(t) \) is the time series of adjusted closing prices for a stock; \( SMA(T) \) is the simple moving average of length \( T \)-day:

\[
SMA(T) = \frac{1}{T} \sum_{i=1}^{T} P(t)
\]

\( EMA(T) \) is the exponential moving average from the observation window:

\[
EMA(T) = \frac{1 - \lambda(T)}{1 - \lambda(T)^T} \sum_{i=1}^{T} \lambda(W)^{T-1} P(t)
\]

Where the smoothing factor \( \lambda(T) \) can be derived from \( T \):

\[
\lambda(T) = \frac{2}{T + 1} \quad \text{for } T \geq 1
\]

The strategy is finally based on the stock price crossing the moving average:

\[
\text{Signal} = \begin{cases} 
\text{Establish long/liquidate short position if } P > MA(T) \\
\text{Establish short/liquidate long position if } P < MA(T)
\end{cases}
\]

The predictors are subsequently computed for \( T \in \{30, 50, 100, 200\} \) as shown in Table F.10:

<table>
<thead>
<tr>
<th>Variable name</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Signal_SMA_30D</td>
<td>Signal_{SMA} for ( T = 30 )</td>
<td>Signals for 30-days SMA</td>
</tr>
<tr>
<td>Signal_SMA_50D</td>
<td>Signal_{SMA} for ( T = 50 )</td>
<td>Signals for 50-days SMA</td>
</tr>
<tr>
<td>Signal_SMA_100D</td>
<td>Signal_{SMA} for ( T = 100 )</td>
<td>Signals for 100-days SMA</td>
</tr>
<tr>
<td>Signal_SMA_200D</td>
<td>Signal_{SMA} for ( T = 200 )</td>
<td>Signals for 200-days SMA</td>
</tr>
<tr>
<td>Signal_EMA_30D</td>
<td>Signal_{EMA} for ( T = 30 )</td>
<td>Signals for 30-days EMA</td>
</tr>
<tr>
<td>Signal_EMA_50D</td>
<td>Signal_{EMA} for ( T = 50 )</td>
<td>Signals for 50-days EMA</td>
</tr>
<tr>
<td>Signal_EMA_100D</td>
<td>Signal_{EMA} for ( T = 100 )</td>
<td>Signals for 100-days EMA</td>
</tr>
</tbody>
</table>

Strategy 12: Two Moving Averages

The previous strategy can be extended to two moving averages. For two lengths \( T_1 \) and \( T_2 \) where \( T_1 < T_2 \), the signal is given by:

\[
\text{Signal} = \begin{cases} 
\text{Establish long/liquidate short position if } MA(T_1) > MA(T_2) \\
\text{Establish short/liquidate long position if } MA(T_1) < MA(T_2)
\end{cases}
\]

The predictors are subsequently computed for 30, 50, 100 and 200 days as shown in Table F.11:

<table>
<thead>
<tr>
<th>Variable name</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Signal_2MA_10_30D</td>
<td>Signal for ( T_1 = 10, T_2 = 30 )</td>
<td>Signals for 10/30 days SMAs</td>
</tr>
<tr>
<td>Signal_2MA_10_50D</td>
<td>Signal for ( T_1 = 10, T_2 = 50 )</td>
<td>Signals for 10/50 days SMAs</td>
</tr>
<tr>
<td>Signal_2MA_30_50D</td>
<td>Signal for ( T_1 = 30, T_2 = 50 )</td>
<td>Signals for 30/50 days SMAs</td>
</tr>
<tr>
<td>Signal_2MA_30_100D</td>
<td>Signal for ( T_1 = 30, T_2 = 100 )</td>
<td>Signals for 30/100 days SMAs</td>
</tr>
<tr>
<td>Signal_2MA_30_200D</td>
<td>Signal for ( T_1 = 30, T_2 = 200 )</td>
<td>Signals for 30/200 days SMAs</td>
</tr>
<tr>
<td>Signal_2MA_50_100D</td>
<td>Signal for ( T_1 = 50, T_2 = 100 )</td>
<td>Signals for 50/100 days SMAs</td>
</tr>
<tr>
<td>Signal_2MA_50_200D</td>
<td>Signal for ( T_1 = 50, T_2 = 200 )</td>
<td>Signals for 50/200 days SMAs</td>
</tr>
<tr>
<td>Signal_2MA_100_200D</td>
<td>Signal for ( T_1 = 100, T_2 = 200 )</td>
<td>Signals for 100/200 days SMAs</td>
</tr>
</tbody>
</table>

Strategy 13: Three Moving Averages

The previous strategy can be extended to three moving averages. For three simple moving averages with lengths \( T_1 < T_2 < T_3 \), the signal is given by:
The predictors are subsequently computed, as shown in Table F.12:

**Table F.12** Three moving average variants.

<table>
<thead>
<tr>
<th>Variable name</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Signal_3MA_5_10_20D</td>
<td>Signal for $T_1 = 5$, $T_2 = 10$, $T_3 = 20$</td>
<td>Signals for 5/10/20 days SMAs</td>
</tr>
<tr>
<td>Signal_3MA_5_20_60D</td>
<td>Signal for $T_1 = 5$, $T_2 = 20$, $T_3 = 60$</td>
<td>Signals for 5/20/60 days SMAs</td>
</tr>
<tr>
<td>Signal_3MA_10_30_50D</td>
<td>Signal for $T_1 = 5$, $T_2 = 30$, $T_3 = 50$</td>
<td>Signals for 10/30/50 days SMAs</td>
</tr>
<tr>
<td>Signal_3MA_30_60_120D</td>
<td>Signal for $T_1 = 30$, $T_2 = 60$, $T_3 = 120$</td>
<td>Signals for 30/60/120 days SMAs</td>
</tr>
<tr>
<td>Signal_3MA_30_120_200D</td>
<td>Signal for $T_1 = 30$, $T_2 = 120$, $T_3 = 200$</td>
<td>30/120/200 days SMAs</td>
</tr>
<tr>
<td>Signal_3MA_60_90_120D</td>
<td>Signal for $T_1 = 60$, $T_2 = 90$, $T_3 = 120$</td>
<td>Signals for 60/90/120 days SMAs</td>
</tr>
<tr>
<td>Signal_3MA_60_120_200D</td>
<td>Signal for $T_1 = 60$, $T_2 = 120$, $T_3 = 200$</td>
<td>60/120/200 days SMAs</td>
</tr>
<tr>
<td>Signal_3MA_90_120_200D</td>
<td>Signal for $T_1 = 90$, $T_2 = 120$, $T_3 = 200$</td>
<td>90/120/200 days SMAs</td>
</tr>
</tbody>
</table>

**Strategy 14: Support and Resistance**

Price of security will tend to stop and reverse at a certain level. The strategy here intends to capture such peaks, or *resistance*, and through, or *support*. (Kakushadze & Serur, 2018) provides a strategy for daily frequency trading. It is extended here to a monthly frequency to fit
this paper framework without high expectation. Let $t$ denote time measured in the units of day, with $t = 1$ corresponding to the most recent time; $P(t)$ is the time series of closing prices for a stock; $B_{up}$ and $B_{down}$ are the highest and lowest prices from the $T$-day observation period:

$$B_{up} = \max\left(P(t), P(t+1), \ldots, P(t+T)\right)$$
$$B_{down} = \min\left(P(t), P(t+1), \ldots, P(t+T)\right)$$

The pivot point, or centre, $C$ is:

$$C = \frac{B_{up} + B_{down} + P}{3}$$

The support $S$ and resistance $R$ levels are:

$$R = 2 \times C - B_{down}$$
$$S = 2 \times C - B_{up}$$

The trading signal is defined as follows:

$$\text{Signal} = \begin{cases} 
\text{Establish long position if } P > C \\
\text{Liquidate long position if } P \leq R \\
\text{Establish short position if } P \leq C \\
\text{Liquidate short position if } P \leq S 
\end{cases}$$

The predictors are subsequently computed for $T \in \{10, 20, 30, 60, 90\}$ as shown in Table F.13:

<table>
<thead>
<tr>
<th>Variable name</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Signal_Support_Resistance_10D</td>
<td>Signal for $T = 10$</td>
<td>Signals for 10-days Support Resistance Indicator</td>
</tr>
<tr>
<td>Signal_Support_Resistance_20D</td>
<td>Signal for $T = 20$</td>
<td>Signals for 20-days Support Resistance Indicator</td>
</tr>
<tr>
<td>Signal_Support_Resistance_30D</td>
<td>Signal for $T = 30$</td>
<td>Signals for 30-days Support Resistance Indicator</td>
</tr>
<tr>
<td>Signal_Support_Resistance_60D</td>
<td>Signal for $T = 60$</td>
<td>Signals for 60-days Support Resistance Indicator</td>
</tr>
<tr>
<td>Signal_Support_Resistance_90D</td>
<td>Signal for $T = 90$</td>
<td>Signals for 90-days Support Resistance Indicator</td>
</tr>
</tbody>
</table>
Strategy 15: Channel

The Donchian channel indicator (Donchian, 1960) allows for a visualisation of the volatility of a market. It will be relatively narrow if prices are stable and vice-versa. Let \( t \) denote time measured in the units of day, with \( t = 1 \) corresponding to the most recent time; \( P(t) \) is the time series of closing prices for a stock; \( B_{up} \) and \( B_{down} \) are the ceiling and floor from the \( T \)-day observation period:

\[
B_{up} = \max \{ P(t), P(t+1), \ldots, P(t+T) \}
\]
\[
B_{down} = \min \{ P(t), P(t+1), \ldots, P(t+T) \}
\]

The trading signal is defined as follows:

\[
\text{Signal} = \begin{cases} 
\text{Establish long/liquidate short position if } P = B_{down} \\
\text{Establish short/liquidate long position if } P = B_{up} 
\end{cases}
\]

The predictors are subsequently computed for \( T \in \{10, 20, 30, 60, 90\} \) as shown in Table F.14:

<table>
<thead>
<tr>
<th>Variable name</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Signal_Channel_10D</td>
<td>Signal for ( T = 10 )</td>
<td>Signals for 10-days Channel Indicator</td>
</tr>
<tr>
<td>Signal_Channel_20D</td>
<td>Signal for ( T = 20 )</td>
<td>Signals for 20-days Channel Indicator</td>
</tr>
<tr>
<td>Signal_Channel_30D</td>
<td>Signal for ( T = 30 )</td>
<td>Signals for 30-days Channel Indicator</td>
</tr>
<tr>
<td>Signal_Channel_60D</td>
<td>Signal for ( T = 60 )</td>
<td>Signals for 60-days Channel Indicator</td>
</tr>
<tr>
<td>Signal_Channel_90D</td>
<td>Signal for ( T = 90 )</td>
<td>Signals for 90-days Channel Indicator</td>
</tr>
</tbody>
</table>

Strategy 16: Event-driven-M&A

This strategy attempts to capture excess returns generated via corporate actions such as Mergers and Acquisitions (M&A). However, it is disregarded here as no relevant data could be found.
Strategy 17: Machine Learning–Single-Stock KNN

The authors (Kakushadze & Serur, 2018) present here a strategy to predict future returns based on moving averages of the price and volume of varying lengths with a $k$-nearest neighbour (k-NN) algorithm (Altman, 1992). A less formal description can be found here. As opposed to the authors, a cross-sectional version is implemented here. Let $t$ denote time measured in units of 1 day, with $t=0$ corresponding to the most recent time; $P(t)$ is the time series of closing prices for a stock; $V(t)$ is the time series of the volume; $R(t)$ is the daily return. The target variable $y(t)$ is defined as the cumulative return from the next trading $T$ days. To prevent look-ahead bias and remain consistent with the rest of the thesis framework, a skip-day $S$ is added:

$$y(t) = \frac{P(t-S-T)-1}{P(t)}$$

The predictor variables $X_a(t)$, $a = 1, ..., m$, are the moving averages for volumes, for prices and the volatility for the $T_a$-day observation window:

$$X_1(t) = \frac{1}{T_1} \sum_{s=1}^{T_1} V(t+s)$$

$$X_2(t) = \frac{1}{T_2} \sum_{s=1}^{T_2} P(t+s)$$

$$X_3(t) = \sqrt{\frac{1}{T_3-1} \sum_{s=1}^{T_3} \left( R(t+s) - \frac{1}{T_3} \sum_{s=1}^{T_3} R(t+s) \right)^2}$$

... 

As opposed to training algorithms parallely for each security $i$, as the authors suggest, all individual stock datasets $\{(X_i, y_i)\}_{i=1,...,I}$ are stacked into a single one $(X, y)$ in order to fit one classifier on the investment universe simultaneously. The predictors are further scaled by rolling sample size $T_{train}$ with scaling function $\varphi$:

$$\tilde{X}_{a}(t:t+T_{train}) = \varphi(X_a(t:t+T_{train}))$$

For an out sample subset at the timestamp \( t \), the \( k \) nearest neighbours of \( \tilde{X}_a(t) \) among \( \tilde{X}_a(t'), t' = t+1, ..., t+T_{\text{train}} \) can be found with the Minkowski distance\(^{42}\):

\[
[D(t,t')]^p = \sum_{a=1}^{m} |\tilde{X}_a(t) - \tilde{X}_a(t')|^p
\]

Where the equation is the Manhattan-distance for \( p = 1 \), or the Euclidean distance for \( p = 2 \). Let \( \tilde{X}_a(t'_\alpha(t)) \), \( \alpha = 1, ..., k \), be the \( k \) nearest neighbours of \( \tilde{X}_a(t) \). Finally, the predicted return \( \hat{y}(t) \) is the weighted corresponding realised values:

\[
\hat{y}(t) = \delta(k,y(t'_\alpha(t)))
\]

The exercise was repeated, as presented in Table F.15, and the predictor KNN_Prediction is subsequently computed, after cross-validations, as:

### Table F.15 KNN variants.

<table>
<thead>
<tr>
<th>Variable name</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>KNN_Prediction</td>
<td>( \hat{y}(t) ) for ( S = 1 ), ( T = 20 ), ( m = 9 ), ( T_a \in {21,63,126} ), ( T_{\text{train}} = 100 ), ( \varphi \equiv \psi(\lambda,x) ), ( k = 3 ), ( p = 1 ), ( \delta \equiv 'distance' )</td>
<td>Forecasts of monthly returns with ( k )-NN</td>
</tr>
</tbody>
</table>

Where \( \psi(\lambda,x) \) is the Yeo–Johnson transformation\(^{43}\) (Yeo & Johnson, 2000).

**Application**

The exercise settings are defined as follows: after computing the moving average of prices, of volumes and volatility for windows of 21-, 63- and 126 days (hence \( a = 9 \)), targets returns are set monthly (\( T = 20 \)) with a skip-day (\( s = 1 \)). One of the major drawbacks of \( k \)-NN, is its scalability. For \( D \) predictors and the sample size \( N \), the time complexity for the naïve approach


can reach $O(DN^2)^{44}$. To mitigate this, the returns at each timestamp are continuously predicted based upon the predictors of the last $T_{m_{	ext{ax}}} = 100$ days. The rest of the hyper-parameters are set through RandomizedSearchCV with the parameter’s distributions set at Table F.16. The regression performances are measured with the Median Absolute Error $\text{MedAE}$, the Mean Absolute Error $\text{MAE}$, the Mean Forecast Trading Returns $\text{MFTR}$ (Lee, 2007) and the Mean Correct Forecast Directions $\text{MCFD}$ (Lee, 2007):

$$\text{MedAE}(y, \hat{y}) = \text{median} \left( \left| y_1 - \hat{y}_1 \right|, \ldots, \left| y_n - \hat{y}_n \right| \right)$$

$$\text{MAE}(y, \hat{y}) = \frac{1}{n} \sum_{i=0}^{n-1} \left| y_i - \hat{y}_i \right|$$

$$\text{MFTR}(y, \hat{y}) = \frac{1}{n} \sum_{i=0}^{n-1} \left( y_i - \text{sign}(\hat{y}_i) \right)$$

$$\text{MCFD}(y, \hat{y}) = \frac{1}{n} \sum_{i=0}^{n-1} \mathbf{1}_{(\text{sign}(y_i) - \text{sign}(\hat{y}_i)) > 0}$$

The two first ones are losses to asses predictions range errors and have been chosen as outliers are expected, while the latter ones measure the trading returns and the correctness of side predictions.

**Table F.16 Hyperparameters grid.**

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
<th>Distributions of parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\varphi$</td>
<td>Scaler$^{45}$</td>
<td>$\varphi \in { \text{'MinMaxScaler'}, \text{'RobustScaler'}, \text{'StandardScaler'}, \text{'QuantileTransformer'}, \text{'PowerTransformer'}, \text{'Normalizer'} }$</td>
</tr>
<tr>
<td>$k$</td>
<td>Number of neighbours</td>
<td>$k \in {3,5,11,25,40,100}$</td>
</tr>
<tr>
<td>$\delta$</td>
<td>Weight function$^{46}$</td>
<td>$\delta \in { \text{'uniform'}, \text{'distance'} }$</td>
</tr>
<tr>
<td>$p$</td>
<td>Power parameter for distance metric</td>
<td>$k \in {1,2,3,4}$</td>
</tr>
</tbody>
</table>

The following methodology was used to prevent overfitting:

---


Table F.17 Implementation of the hyperparameters research

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Sample 200 hyper-parameters combination from the parameter grid;</td>
</tr>
<tr>
<td>2</td>
<td>Sample 5 portions of the first half of the dataset;</td>
</tr>
<tr>
<td>3</td>
<td><strong>for each sample</strong> do</td>
</tr>
<tr>
<td>4</td>
<td>Generate 40 splits of training and testing subset with Rep-Hold;</td>
</tr>
<tr>
<td>5</td>
<td><strong>for each split</strong> do</td>
</tr>
<tr>
<td>6</td>
<td><strong>for each combination of hyper-parameters</strong> do</td>
</tr>
<tr>
<td>7</td>
<td>Fit model on the training subset;</td>
</tr>
<tr>
<td>8</td>
<td>Predict the testing subset and report performances;</td>
</tr>
<tr>
<td>9</td>
<td>end</td>
</tr>
<tr>
<td>10</td>
<td>end</td>
</tr>
<tr>
<td>11</td>
<td>end</td>
</tr>
<tr>
<td>12</td>
<td>Average performances and rank combination of hyper-parameters;</td>
</tr>
<tr>
<td>13</td>
<td>Select the best combination;</td>
</tr>
</tbody>
</table>

The average metrics and their standard deviation across resample can be seen for all parameter’s combination on Figure F.1. The lowest MedAE corresponds already to the lowest MAE and highest MFTR and highest MCFD. Consequently, the parameters combination can be easily selected as there is no presence of trade-offs.

![Figure F.1 Cross-Validation performances over hyper-parameters combination](image-url)
The precedent methodology is biased, as hyper-parameters are not determined on an expanding rolling basis. The lookback windows $T_{\text{cov}}$ could also have been determined with cross-validation. However, this approach has been chosen for computational reasons (already 80'000 fits on matrices of size $120'000 \times 9$), and it is assumed it would not lead to significant look-ahead bias as the selected parameters continuously rank amongst the first ones across resamples and forecast remains predicted on a rolling basis. Figure F.2 shows the regression metric across time for the test subsets, i.e. forecasted returns. Despite outliers, 85% of side predictions are on average correct, and half of the predicted returns are wrong by less than 0.02 percent-point. Finally, predictions appear meaningful with and without outliers, as depicted by the scatterplots on Figure F.3. It should be remembered that returns have been predicted across all stocks and for the entire time range and thus, will not be all passed to the final learning algorithm. The individuals would be still selected according to their index memberships for each month, and this step is expected to filter outliers from stocks that have been dropped from the index.

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{figure_f2.png}
\caption{Forecast performances over months}
\end{figure}

Besides, a different approach has been taken as $k$-NN can be sensitive to redundancy. Predictors with Pearson correlation coefficient above 0.9 have been dropped (see Figure F.4), and the same methodology was performed. However, this led to slightly worse performances across months such as $\text{MedAE} = 0.027 \pm 0.13$, in comparison to $0.024 \pm 0.12$ previously.
Strategy 18: Statistical Arbitrage-Optimization

This strategy allocates a portfolio requiring its Sharpe ratio to be maximised. However, it is disregarded here. Calculation of weights requires the inversion of a covariance matrix and would not be robust here for the following reasons:

- The investment universe comprises correlated stocks, and the inversion of its covariance matrix would lead to unstable solutions.
- It requires at least $0.5N(N+1)$ of IID observation in order to derive a non-singular covariance matrix of size $N$. Assuming 800 stocks must be allocated each month (assumed value of the Russell 1000 Index constituents after cleaning), it would require at least 1276 years of daily data (López de Prado, 2018).
Strategy 19: Market-Making

This strategy attempts to capture the bid-ask spread for a given stock (Kakushadze & Serur, 2018). However, it is disregarded here as it does not match with the frequency trading of this paper.

Strategy 20: Alpha Combos

With technological advances, this strategy attempts to mine a big dataset of expected returns, called *alphas* by (Kakushadze & Serur, 2018). On their own, alphas are ephemeral, and any profit would be taken away by trading costs. Therefore, the idea is to combine them into a large *alpha combo strategy*. However, it is disregarded here as:

- As the academic literature is minimal on the formulation of alphas in detail, researching such a large number of alphas is outside of the scope of this thesis.
- While (Kakushadze, 2016) provides formulas for 101 alphas, their average holding period approximately ranges between 0.6 to 6.4 days compared to the monthly framework defined here.
- In addition to the computational complexity, the 101 formulas presented are enigmatic and could compromise the interpretability of the final strategy.
Appendix G: Portfolio Statistics Computation

The following assumptions are defined. Let \( t \) denote time measured in units of 1 month for an observed period \( t = 1,2,...,T \), with \( t = 0 \) corresponding to the initialization of the portfolio; \( N \) is the annualization factor (\( N = 12 \)); the risk-free rate \( R_f \) is assumed to be null; \( R_{jt} \) is the monthly return for a stock labelled \( i \) from an investment universe \( i = 1,2,...,I \); \( R_{b,t} \) is the monthly return of a benchmark (defined here as the Russel 1000 Index); \( R_{P,t} \) is the monthly return of the portfolio defined as the weighted average of the returns from the invested assets:

\[
R_{P,t} = \sum_{i} w_{i,t-1} R_{i,t}
\]

The portfolio return with transaction costs \( c \) (here \( c = 0.25\% \) per trade) is defined as (E. P. Chan, 2009):

\[
R_{P,t}^c = R_{P,t} - c \sum_{i} |w_{i,t} - w_{i,t-1}|
\]

The \( r \)-moment for sample size \( n \) is given by:

\[
m_r = \frac{1}{n} \sum_{i} (x_i - \overline{x})^r
\]

The unbiased cumulant estimates as reported in (Harald, 1946):

\[
k_2 = \frac{n}{n-1} m_2
\]

\[
k_3 = \frac{n^2}{(n-1)(n-2)} m_3
\]

\[
k_4 = \frac{n^2}{(n-1)(n-2)(n-3)} \left\{ (n+1)m_4 - 3(n-1)m_2^2 \right\}
\]

Finally, the metrics from Table G.1 are derived (DeMiguel, Garlappi, Nogales, & Uppal, 2009; Goodwin, 1998; Meucci, 2010; Sharpe, 1964, 1966; Sortino & Price, 1994; Young, 1991). For metrics annualization, (Goodwin, 1998) compares four practices: usage of arithmetic-, geometric-, continuously compunded-mean and frequency-converted data. To remain consistant across metrics computations, the arithmetic mean is used.
Table G.1 Portfolio Statistics

<table>
<thead>
<tr>
<th>Portfolio Statistics</th>
<th>Formula</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>General</strong></td>
<td></td>
</tr>
<tr>
<td>Turnover</td>
<td>Turnover\textsubscript{year} = \frac{N}{T} \sum_{t=1}^{T} \sum_{i=1}^{I}</td>
</tr>
<tr>
<td><strong>Performance</strong></td>
<td></td>
</tr>
<tr>
<td>Cumulative return</td>
<td>CR\textsubscript{t} = \left(\prod_{i=1}^{T} (1 + R_{i})\right) - 1</td>
</tr>
<tr>
<td>CAGR</td>
<td>CAGR = \left(\prod_{i=1}^{T} (1 + R_{i})\right)^{\frac{p}{T}} - 1</td>
</tr>
<tr>
<td>Alpha</td>
<td>\alpha\textsubscript{yearly} = \bar{\alpha} N \quad \text{where } \alpha_{t} = R_{t} - \beta R_{B,t} \quad \text{for } t=1,...,T</td>
</tr>
<tr>
<td>Beta</td>
<td>\beta = \frac{\text{cov}(R,R_{B})}{\text{var}(R_{B})}</td>
</tr>
<tr>
<td>Stability</td>
<td>R^2 \text{ from } \ln(CR)\textsubscript{t} = a + bt \quad \text{for } t=1,...,T</td>
</tr>
<tr>
<td>Hit Ratio</td>
<td>HR = \sum_{t=1}^{T} \sum_{i=1}^{I} \frac{1_{(w_{i,t},R_{i,t}&gt;0)}}{1_{(w_{i,t-1}&gt;0)}}</td>
</tr>
<tr>
<td>Average return from hits</td>
<td>\frac{\sum_{t=1}^{T} \sum_{i=1}^{I} 1_{(w_{i,t},R_{i,t}&gt;0)} w_{i,t-1} R_{i,t}}{\sum_{t=1}^{T} \sum_{i=1}^{I} 1_{</td>
</tr>
<tr>
<td>Average return from misses</td>
<td>\frac{\sum_{t=1}^{T} \sum_{i=1}^{I} 1_{(w_{i,t},R_{i,t}&lt;0)} w_{i,t-1} R_{i,t}}{\sum_{t=1}^{T} \sum_{i=1}^{I} 1_{(w_{i,t-1},R_{i,t}&lt;0)}}</td>
</tr>
<tr>
<td><strong>Risk</strong></td>
<td></td>
</tr>
<tr>
<td>Volatility</td>
<td>\sigma\textsubscript{year} = \frac{k_{1/2}}{2} N^{1/2}</td>
</tr>
<tr>
<td>Skewness</td>
<td>g_{1,\text{year}} = \frac{k_{3}}{k_{2}^{3/2}} \frac{1}{N^{1/2}}</td>
</tr>
<tr>
<td>Kurtosis</td>
<td>g_{2,\text{year}} = \frac{k_{4}}{k_{2}^{2}} \frac{1}{N}</td>
</tr>
<tr>
<td>Maximum drawdown</td>
<td>MD = \max_{t=1,...,T} \max_{t'=1,...,T} (R_{t} - R_{t'})</td>
</tr>
</tbody>
</table>
Efficiency

Sharpe ratio

\[ SR_{year} = \frac{k_1}{k_2^{1/2}} N^{1/2} \]

Information ratio

\[ IR_{year} = \frac{k_1(R_p - R_b)}{k_2^{1/2}(R_p - R_b)} N^{1/2} \]

Sortino ratio

\[ Sortino_{year} = \frac{k_1}{dsr} N^{1/2} \text{ where } dsr = \sqrt{\frac{1}{T-1} \sum_{t} \min(0, R_t)} \]

Calmar ratio

\[ Calmar_{year} = \frac{k_1}{MD} N \]

Tail ratio

\[ Tail\ ratio = \left| \frac{\hat{F}^{-1}(0.95)}{\hat{F}^{-1}(0.05)} \right| \text{ where } \hat{F}^{-1}(x) \text{ is the quantile function} \]
Appendix H: AutoML and Deep Learning

As the process of designing and validating a machine learning pipeline can be time-consuming, the question of whether other faster-implemented tools could be used to compete with the work achieved so far has been raised. As such predictive performances are compared with AutoML and deep learning techniques.

AutoML

As machine learning pipeline performances can be very sensitive to numerous decisions, the field of automated machine learning (AutoML) emerged to ease the workflow of ML practitioners. These systems aim to automate the entire machine learning pipeline in a data-driven way: feature pre-processing, feature selection, model selection and hyper-parameters tuning. These techniques are now mature enough to compete with and even outperform human-machine learning experts (Hutter, Lars, & Joaquin, 2019). As such, Auto-sklearn and TPOT are used here. While Auto-sklearn optimizes pipeline decision with Bayesian optimization, TPOT relies on Genetic Algorithm (GA). In addition to be very computationally demanding, the drawbacks of using such tools is their parameters complexity and their lack of flexibility as they provide to be only wrappers of other machine learning libraries such as Scikit-learn. As such, sample weights can be passed only to the estimator using TPOT, but not for Auto-sklearn. They also both imputes automatically missing values using the median.

Deep learning

As opposed to shallow learning, deep learning depends less on the representation of data they are given and can avoid the time-consuming step of feature hand-designing (Goodfellow, Bengio, & Courville, 2016). For this reason, a deep learning implementation has been considered here. Among the family of Artificial Neural Networks (ANN), the Recurrent Neural Networks (RNN) (Rumelhart, Hinton, & Williams, 1986) are a group that can model the relationship between sequences instead of considering each sample independently and are thus suitable for application in Natural Language Processing (NLP) or Time Series. A significant issue with early RNNs implementations was their vanishing gradient problem. Neural networks work mainly by adjusting their internal weights by backpropagating the gradient computed from the loss function. Vanishing gradient illustrates the case where neural networks cannot converge as gradient drops or explodes. As solution, emerged the sub-group of Gated RNN, such as Long Short-Term Memory (LSTM) (Hochreiter & Schmidhuber, 1997). In addition to share weights
across sequences, Gated RNNs include an internal gates mechanism to accumulate information and forget it when useful. Finally, the choice of using an LSTM has also been motivated by (Ryll & Seidens, 2019) for LSTM overall better performances for stock forecasting in comparison to more classic ANNs. For more information about LSTM architecture, the reader should consider (Goodfellow et al., 2016) or this less formal description\(^\text{47}\). The LSTM is implemented with Keras, a high-level API running on top of TensorFlow. This set up enables fast experimentation in a research environment, instead of relying directly on TensorFlow, more suitable for a production environment. For faster training of the LSTM, a GPU has been used. As deep learning involves mainly matrix operations computation, GPU offers a solution for scalability due to their architectures. For other ends than video processing, computing with GPU is referred as General-Purpose GPU programming (GPGPU). Finally, the content of (Chollet & Others, 2015; Ng, 2019) has been followed for deep learning development good practices.

\(^{47}\text{Understanding LSTM Networks: }\text{http://colah.github.io/posts/2015-08-Understanding-LSTMs/}\)