Machine Learning for Factor Investing
Aims and scope:
The field of financial mathematics forms an ever-expanding slice of the financial sector. This series aims to capture new developments and summarize what is known over the whole spectrum of this field. It will include a broad range of textbooks, reference works and handbooks that are meant to appeal to both academics and practitioners. The inclusion of numerical code and concrete real-world examples is highly encouraged.

Series Editors
M.A.H. Dempster  
Centre for Financial Research  
Department of Pure Mathematics and Statistics  
University of Cambridge

Dilip B. Madan  
Robert H. Smith School of Business  
University of Maryland

Rama Cont  
Department of Mathematics  
Imperial College

Metamodeling for Variable Annuities  
Guojun Gan and Emiliano A. Valdez

Modeling Fixed Income Securities and Interest Rate Options  
Robert A. Jarrow

Financial Modelling in Commodity Markets  
Viviana Fanelli

Introductory Mathematical Analysis for Quantitative Finance  
Daniele Ritelli, Giulia Spaletta

Handbook of Financial Risk Management  
Thierry Roncalli

Optional Processes  
Stochastic Calculus and Applications  
Mohamed Abdelghani, Alexander Melnikov

Machine Learning for Factor Investing  
R Version  
Guillaume Coqueret and Tony Guida

Machine Learning for Factor Investing

R Version

Guillaume Coqueret and Tony Guida
To Leslie and Selin.
# Contents

Preface xiii

I Introduction 1

1 Notations and data 3
  1.1 Notations 3
  1.2 Dataset 4

2 Introduction 9
  2.1 Context 9
  2.2 Portfolio construction: the workflow 10
  2.3 Machine learning is no magic wand 11

3 Factor investing and asset pricing anomalies 13
  3.1 Introduction 14
  3.2 Detecting anomalies 15
    3.2.1 Challenges 15
    3.2.2 Simple portfolio sorts 15
    3.2.3 Factors 17
    3.2.4 Fama-Macbeth regressions 22
    3.2.5 Factor competition 25
    3.2.6 Advanced techniques 26
  3.3 Factors or characteristics? 27
  3.4 Hot topics: momentum, timing and ESG 28
    3.4.1 Factor momentum 28
    3.4.2 Factor timing 29
    3.4.3 The green factors 30
  3.5 The links with machine learning 30
    3.5.1 A short list of recent references 31
    3.5.2 Explicit connections with asset pricing models 31
  3.6 Coding exercises 34

4 Data preprocessing 35
  4.1 Know your data 35
  4.2 Missing data 38
  4.3 Outlier detection 40
  4.4 Feature engineering 41
    4.4.1 Feature selection 41
    4.4.2 Scaling the predictors 41
  4.5 Labelling 42
    4.5.1 Simple labels 42
    4.5.2 Categorical labels 43
4.5.3 The triple barrier method ................................................. 44
4.5.4 Filtering the sample ......................................................... 45
4.5.5 Return horizons ............................................................ 46
4.6 Handling persistence .......................................................... 47
4.7 Extensions ................................................................. 47
4.7.1 Transforming features ...................................................... 47
4.7.2 Macro-economic variables ............................................... 48
4.7.3 Active learning ............................................................ 48
4.8 Additional code and results .................................................. 50
4.8.1 Impact of rescaling: graphical representation ....................... 50
4.8.2 Impact of rescaling: toy example ...................................... 52
4.9 Coding exercises ............................................................. 53

II Common supervised algorithms ............................................. 55

5 Penalized regressions and sparse hedging for minimum variance portfo-
lios ................................................................. 57
5.1 Penalized regressions ......................................................... 57
   5.1.1 Simple regressions ..................................................... 57
   5.1.2 Forms of penalizations ................................................. 58
   5.1.3 Illustrations ........................................................... 60
5.2 Sparse hedging for minimum variance portfolios ....................... 62
   5.2.1 Presentation and derivations ......................................... 62
   5.2.2 Example ............................................................. 65
5.3 Predictive regressions ....................................................... 67
   5.3.1 Literature review and principle ..................................... 67
   5.3.2 Code and results ..................................................... 68
5.4 Coding exercise .............................................................. 68

6 Tree-based methods ............................................................. 69
6.1 Simple trees ................................................................. 69
   6.1.1 Principle ............................................................ 69
   6.1.2 Further details on classification .................................... 71
   6.1.3 Pruning criteria ...................................................... 72
   6.1.4 Code and interpretation ............................................. 73
6.2 Random forests .............................................................. 76
   6.2.1 Principle ............................................................ 76
   6.2.2 Code and results ..................................................... 78
6.3 Boosted trees: Adaboost ..................................................... 79
   6.3.1 Methodology ........................................................ 79
   6.3.2 Illustration .......................................................... 82
6.4 Boosted trees: extreme gradient boosting .................................. 82
   6.4.1 Managing loss ....................................................... 83
   6.4.2 Penalization .......................................................... 83
   6.4.3 Aggregation .......................................................... 84
   6.4.4 Tree structure ....................................................... 85
   6.4.5 Extensions ........................................................... 86
   6.4.6 Code and results ..................................................... 86
   6.4.7 Instance weighting ................................................... 88
6.5 Discussion ................................................................. 89
6.6 Coding exercises ............................................................ 90
# Contents

<table>
<thead>
<tr>
<th>Chapter</th>
<th>Title</th>
<th>Pages</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td><strong>Neural networks</strong></td>
<td></td>
</tr>
<tr>
<td>7.1</td>
<td>The original perceptron</td>
<td>91</td>
</tr>
<tr>
<td>7.2</td>
<td>Multilayer perceptron</td>
<td>92</td>
</tr>
<tr>
<td>7.2.1</td>
<td>Introduction and notations</td>
<td>93</td>
</tr>
<tr>
<td>7.2.2</td>
<td>Universal approximation</td>
<td>96</td>
</tr>
<tr>
<td>7.2.3</td>
<td>Learning via back-propagation</td>
<td>97</td>
</tr>
<tr>
<td>7.2.4</td>
<td>Further details on classification</td>
<td>100</td>
</tr>
<tr>
<td>7.3</td>
<td>How deep we should go and other practical issues</td>
<td>101</td>
</tr>
<tr>
<td>7.3.1</td>
<td>Architectural choices</td>
<td>101</td>
</tr>
<tr>
<td>7.3.2</td>
<td>Frequency of weight updates and learning duration</td>
<td>102</td>
</tr>
<tr>
<td>7.3.3</td>
<td>Penalizations and dropout</td>
<td>103</td>
</tr>
<tr>
<td>7.4</td>
<td>Code samples and comments for vanilla MLP</td>
<td>104</td>
</tr>
<tr>
<td>7.4.1</td>
<td>Regression example</td>
<td>104</td>
</tr>
<tr>
<td>7.4.2</td>
<td>Classification example</td>
<td>107</td>
</tr>
<tr>
<td>7.4.3</td>
<td>Custom losses</td>
<td>111</td>
</tr>
<tr>
<td>7.5</td>
<td>Recurrent networks</td>
<td>112</td>
</tr>
<tr>
<td>7.5.1</td>
<td>Presentation</td>
<td>112</td>
</tr>
<tr>
<td>7.5.2</td>
<td>Code and results</td>
<td>114</td>
</tr>
<tr>
<td>7.6</td>
<td>Other common architectures</td>
<td>117</td>
</tr>
<tr>
<td>7.6.1</td>
<td>Generative adversarial networks</td>
<td>117</td>
</tr>
<tr>
<td>7.6.2</td>
<td>Autoencoders</td>
<td>118</td>
</tr>
<tr>
<td>7.6.3</td>
<td>A word on convolutional networks</td>
<td>119</td>
</tr>
<tr>
<td>7.6.4</td>
<td>Advanced architectures</td>
<td>121</td>
</tr>
<tr>
<td>7.7</td>
<td>Coding exercise</td>
<td>121</td>
</tr>
<tr>
<td>8</td>
<td><strong>Support vector machines</strong></td>
<td></td>
</tr>
<tr>
<td>8.1</td>
<td>SVM for classification</td>
<td>123</td>
</tr>
<tr>
<td>8.2</td>
<td>SVM for regression</td>
<td>126</td>
</tr>
<tr>
<td>8.3</td>
<td>Practice</td>
<td>127</td>
</tr>
<tr>
<td>8.4</td>
<td>Coding exercises</td>
<td>128</td>
</tr>
<tr>
<td>9</td>
<td><strong>Bayesian methods</strong></td>
<td></td>
</tr>
<tr>
<td>9.1</td>
<td>The Bayesian framework</td>
<td>129</td>
</tr>
<tr>
<td>9.2</td>
<td>Bayesian sampling</td>
<td>131</td>
</tr>
<tr>
<td>9.2.1</td>
<td>Gibbs sampling</td>
<td>131</td>
</tr>
<tr>
<td>9.2.2</td>
<td>Metropolis-Hastings sampling</td>
<td>131</td>
</tr>
<tr>
<td>9.3</td>
<td>Bayesian linear regression</td>
<td>132</td>
</tr>
<tr>
<td>9.4</td>
<td>Naive Bayes classifier</td>
<td>135</td>
</tr>
<tr>
<td>9.5</td>
<td>Bayesian additive trees</td>
<td>138</td>
</tr>
<tr>
<td>9.5.1</td>
<td>General formulation</td>
<td>138</td>
</tr>
<tr>
<td>9.5.2</td>
<td>Priors</td>
<td>138</td>
</tr>
<tr>
<td>9.5.3</td>
<td>Sampling and predictions</td>
<td>139</td>
</tr>
<tr>
<td>9.5.4</td>
<td>Code</td>
<td>141</td>
</tr>
<tr>
<td>10</td>
<td><strong>Validating and tuning</strong></td>
<td></td>
</tr>
<tr>
<td>10.1</td>
<td>Learning metrics</td>
<td>145</td>
</tr>
<tr>
<td>10.1.1</td>
<td>Regression analysis</td>
<td>145</td>
</tr>
<tr>
<td>10.1.2</td>
<td>Classification analysis</td>
<td>147</td>
</tr>
<tr>
<td>10.2</td>
<td>Validation</td>
<td>151</td>
</tr>
</tbody>
</table>
10.2.1 The variance-bias tradeoff: theory ............................................... 151
10.2.2 The variance-bias tradeoff: illustration ....................................... 154
10.2.3 The risk of overfitting: principle ................................................. 156
10.2.4 The risk of overfitting: some solutions ........................................ 157
10.3 The search for good hyperparameters .............................................. 158
  10.3.1 Methods ...................................................................................... 158
  10.3.2 Example: grid search .................................................................... 160
  10.3.3 Example: Bayesian optimization .................................................... 162
10.4 Short discussion on validation in backtests ......................................... 163

11 Ensemble models .................................................................................. 165
  11.1 Linear ensembles .............................................................................. 166
    11.1.1 Principles .................................................................................... 166
    11.1.2 Example ...................................................................................... 167
  11.2 Stacked ensembles ............................................................................ 170
    11.2.1 Two-stage training ........................................................................ 170
    11.2.2 Code and results ........................................................................... 170
  11.3 Extensions ........................................................................................ 172
    11.3.1 Exogenous variables ................................................................. 172
    11.3.2 Shrinking inter-model correlations ............................................. 173
  11.4 Exercise ............................................................................................ 176

12 Portfolio backtesting ............................................................................ 177
  12.1 Setting the protocol ........................................................................... 177
  12.2 Turning signals into portfolio weights .............................................. 179
  12.3 Performance metrics ........................................................................ 181
    12.3.1 Discussion .................................................................................... 181
    12.3.2 Pure performance and risk indicators ........................................... 182
    12.3.3 Factor-based evaluation .............................................................. 183
    12.3.4 Risk-adjusted measures .............................................................. 184
    12.3.5 Transaction costs and turnover .................................................... 184
  12.4 Common errors and issues ................................................................. 185
    12.4.1 Forward looking data .................................................................. 185
    12.4.2 Backtest overfitting ..................................................................... 185
    12.4.3 Simple safeguards ...................................................................... 187
  12.5 Implication of non-stationarity: forecasting is hard ................................ 187
    12.5.1 General comments ...................................................................... 187
    12.5.2 The no free lunch theorem ......................................................... 188
  12.6 First example: a complete backtest .................................................. 189
  12.7 Second example: backtest overfitting .............................................. 193
  12.8 Coding exercises .............................................................................. 196

IV Further important topics ....................................................................... 197

13 Interpretability ..................................................................................... 199
  13.1 Global interpretations ........................................................................ 200
    13.1.1 Simple models as surrogates ...................................................... 200
    13.1.2 Variable importance (tree-based) ............................................... 201
    13.1.3 Variable importance (agnostic) ................................................. 203
    13.1.4 Partial dependence plot ............................................................. 205
  13.2 Local interpretations ......................................................................... 206
<table>
<thead>
<tr>
<th>Section</th>
<th>Content</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>18.8</td>
<td>Chapter 13</td>
<td>281</td>
</tr>
<tr>
<td>18.8.1</td>
<td>EW portfolios with the tidyverse</td>
<td>281</td>
</tr>
<tr>
<td>18.8.2</td>
<td>Advanced weighting function</td>
<td>282</td>
</tr>
<tr>
<td>18.8.3</td>
<td>Functional programming in the backtest</td>
<td>283</td>
</tr>
<tr>
<td>18.9</td>
<td>Chapter 16</td>
<td>285</td>
</tr>
<tr>
<td>18.10</td>
<td>Chapter 17</td>
<td>285</td>
</tr>
</tbody>
</table>

**Bibliography**  

289

**Index**  

319
Preface

This book is intended to cover some advanced modelling techniques applied to equity investment strategies that are built on firm characteristics. The content is threefold. First, we try to simply explain the ideas behind most mainstream machine learning algorithms that are used in equity asset allocation. Second, we mention a wide range of academic references for the readers who wish to push a little further. Finally, we provide hands-on R code samples that show how to apply the concepts and tools on a realistic dataset which we share to encourage reproducibility.

What this book is not about

This book deals with machine learning (ML) tools and their applications in factor investing. Factor investing is a subfield of a large discipline that encompasses asset allocation, quantitative trading and wealth management. Its premise is that differences in the returns of firms can be explained by the characteristics of these firms. Thus, it departs from traditional analyses which rely on price and volume data only, like classical portfolio theory à la Markowitz (1952), or high frequency trading. For a general and broad treatment of Machine Learning in Finance, we refer to Dixon et al. (2020).

The topics we discuss are related to other themes that will not be covered in the monograph. These themes include:

- Applications of ML in other financial fields, such as fraud detection or credit scoring. We refer to Ngai et al. (2011) and Baesens et al. (2015) for general purpose fraud detection, to Bhattacharyya et al. (2011) for a focus on credit cards and to Ravisankar et al. (2011) and Abbasi et al. (2012) for studies on fraudulent financial reporting. On the topic of credit scoring, Wang et al. (2011) and Brown and Mues (2012) provide overviews of methods and some empirical results. Also, we do not cover ML algorithms for data sampled at higher (daily or intraday) frequencies (microstructure models, limit order book). The chapter from Kearns and Nevmyvaka (2013) and the recent paper by Sirignano and Cont (2019) are good introductions on this topic.

- Use cases of alternative datasets that show how to leverage textual data from social media, satellite imagery, or credit card logs to predict sales, earning reports, and, ultimately, future returns. The literature on this topic is still emerging (see, e.g., Blank et al. (2019), Jha (2019) and Ke et al. (2019)) but will likely blossom in the near future.

- Technical details of machine learning tools. While we do provide some insights on specificities of some approaches (those we believe are important), the purpose of the book is not to serve as reference manual on statistical learning. We refer to Hastie et al.

- Finally, the book does not cover methods of natural language processing (NLP) that can be used to evaluate sentiment which can in turn be translated into investment decisions. This topic has nonetheless been trending lately and we refer to Loughran and McDonald (2016), Cong et al. (2019a), Cong et al. (2019b) and Gentzkow et al. (2019) for recent advances on the matter.

---

The targeted audience

Who should read this book? This book is intended for two types of audiences. First, postgraduate students who wish to pursue their studies in quantitative finance with a view towards investment and asset management. The second target groups are professionals from the money management industry who either seek to pivot towards allocation methods that are based on machine learning or are simply interested in these new tools and want to upgrade their set of competences. To a lesser extent, the book can serve scholars or researchers who need a manual with a broad spectrum of references both on recent asset pricing issues and on machine learning algorithms applied to money management. While the book covers mostly common methods, it also shows how to implement more exotic models, like causal graphs (Chapter 14), Bayesian additive trees (Chapter 9), and hybrid autoencoders (Chapter 7).

The book assumes basic knowledge in algebra (matrix manipulation), analysis (function differentiation, gradients), optimization (first and second order conditions, dual forms), and statistics (distributions, moments, tests, simple estimation method like maximum likelihood). A minimal financial culture is also required: simple notions like stocks, accounting quantities (e.g., book value) will not be defined in this book. Lastly, all examples and illustrations are coded in R. A minimal culture of the language is sufficient to understand the code snippets which rely heavily on the most common functions of the tidyverse (Wickham et al. (2019), www.tidyverse.org), and piping (Bache and Wickham (2014), Mailund (2019)).

---

How this book is structured

The book is divided into four parts.

Part I gathers preparatory material and starts with notations and data presentation (Chapter 1), followed by introductory remarks (Chapter 2). Chapter 3 outlines the economic

---

1For a list of online resources, we recommend the curated page https://github.com/josephmisiti/awesome-machine-learning/blob/master/books.md.
foundations (theoretical and empirical) of factor investing and briefly sums up the dedicated recent literature. Chapter 4 deals with data preparation. It rapidly recalls the basic tips and warns about some major issues.

Part II of the book is dedicated to predictive algorithms in supervised learning. Those are the most common tools that are used to forecast financial quantities (returns, volatilities, Sharpe ratios, etc.). They range from penalized regressions (Chapter 5), to tree methods (Chapter 6), encompassing neural networks (Chapter 7), support vector machines (Chapter 8) and Bayesian approaches (Chapter 9).

The next portion of the book bridges the gap between these tools and their applications in finance. Chapter 10 details how to assess and improve the ML engines defined beforehand. Chapter 11 explains how models can be combined and often why that may not be a good idea. Finally, one of the most important chapters (Chapter 12) reviews the critical steps of portfolio backtesting and mentions the frequent mistakes that are often encountered at this stage.

The end of the book covers a range of advanced topics connected to machine learning more specifically. The first one is interpretability. ML models are often considered to be black boxes and this raises trust issues: how and why should one trust ML-based predictions? Chapter 13 is intended to present methods that help understand what is happening under the hood. Chapter 14 is focused on causality, which is both a much more powerful concept than correlation and also at the heart of many recent discussions in Artificial Intelligence (AI). Most ML tools rely on correlation-like patterns and it is important to underline the benefits of techniques related to causality. Finally, Chapters 15 and 16 are dedicated to non-supervised methods. The latter can be useful, but their financial applications should be wisely and cautiously motivated.

---

**Companion website**

This book is entirely available at [http://www.mlfactor.com](http://www.mlfactor.com). It is important that not only the content of the book be accessible, but also the data and code that are used throughout the chapters. They can be found at [https://github.com/shokru/mlfactor.github.io/tree/master/material](https://github.com/shokru/mlfactor.github.io/tree/master/material). The online version of the book will be updated beyond the publication of the printed version.

---

**Why R?**

The supremacy of Python as the dominant ML programming language is a widespread belief. This is because almost all applications of deep learning (which is as of 2020 one of the most fashionable branches of ML) are coded in Python via Tensorflow or Pytorch. The fact is that R has a lot to offer as well. First of all, let us not forget that one of the most influential textbooks in ML ([Hastie et al. (2009)](https://www.stat.ucla.edu/~hastie/book09/)) is written by statisticians who code in R. Moreover, many statistics-orientated algorithms (e.g., BARTs in Section 9.5) are primarily
coded in R and not always in Python. The R offering in Bayesian packages in general (https://cran.r-project.org/web/views/Bayesian.html) and in Bayesian learning in particular is probably unmatched.

There are currently several ML frameworks available in R.

- **caret**: https://topepo.github.io/caret/index.html, a compilation of more than 200 ML models;

- **tidymodels**: https://github.com/tidymodels, a recent collection of packages for ML workflow (developed by Max Kuhn at RStudio, which is a token of high quality material!);

- **rtemis**: https://rtemis.netlify.com, a general purpose package for ML and visualization;

- **mlr3**: https://mlr3.mlr-org.com/index.html, also a simple framework for ML models;

- **h2o**: https://github.com/h2oai/h2o-3/tree/master/h2o-r, a large set of tools provided by h2o (coded in Java);

- **Open ML**: https://github.com/openml/openml-r, the R version of the OpenML (www.openml.org) community.

Moreover, via the *reticulate* package, it is possible (but not always easy) to benefit from Python tools as well. The most prominent example is the adaptation of the *tensorflow* and *keras* libraries to R. Thus, some very advanced Python material is readily available to R users. This is also true for other resources, like Stanford’s CoreNLP library (in Java) which was adapted to R in the package *coreNLP* (which we will not use in this book).

---

**Coding instructions**

One of the purposes of the book is to propose a large-scale tutorial of ML applications in financial predictions and portfolio selection. Thus, one keyword is REPRODUCIBILITY! In order to duplicate our results (up to possible randomness in some learning algorithms), you will need running versions of R and RStudio on your computer. The best books to learn R are also often freely available online. A short list can be found here https://rstudio.com/resources/books/. The monograph *R for Data Science* is probably the most crucial.

In terms of coding requirements, we rely heavily on the tidyverse, which is a collection of packages (or libraries). The three packages we use most are *dplyr* which implements simple data manipulations (filter, select, arrange), *tidyr* which formats data in a tidy fashion, and *ggplot*, for graphical outputs.
A list of the packages we use can be found in Table 1 below. Packages with a star * need to be installed via bioconductor. Packages with a plus + need to be installed manually.

**TABLE 1**: List of all packages used in the book.

<table>
<thead>
<tr>
<th>Package</th>
<th>Purpose</th>
<th>Chapter(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>BART</td>
<td>Bayesian additive trees</td>
<td>10</td>
</tr>
<tr>
<td>broom</td>
<td>Tidy regression output</td>
<td>5</td>
</tr>
<tr>
<td>CAM +</td>
<td>Causal Additive Models</td>
<td>15</td>
</tr>
<tr>
<td>caTools</td>
<td>AUC curves</td>
<td>11</td>
</tr>
<tr>
<td>CausalImpact</td>
<td>Causal inference with structural time series</td>
<td>15</td>
</tr>
<tr>
<td>cowplot</td>
<td>Stacking plots</td>
<td>4 &amp; 13</td>
</tr>
<tr>
<td>breakDown</td>
<td>Breakdown interpretability</td>
<td>14</td>
</tr>
<tr>
<td>dummies</td>
<td>One-hot encoding</td>
<td>8</td>
</tr>
<tr>
<td>e1071</td>
<td>Support Vector Machines</td>
<td>9</td>
</tr>
<tr>
<td>factoextra</td>
<td>PCA visualization</td>
<td>16</td>
</tr>
<tr>
<td>fastAdaboost</td>
<td>Boosted trees</td>
<td>7</td>
</tr>
<tr>
<td>forecast</td>
<td>Autocorrelation function</td>
<td>4</td>
</tr>
<tr>
<td>FNN</td>
<td>Nearest Neighbors detection</td>
<td>16</td>
</tr>
<tr>
<td>ggpubr</td>
<td>Combining plots</td>
<td>11</td>
</tr>
<tr>
<td>glmnet</td>
<td>Penalized regressions</td>
<td>6</td>
</tr>
<tr>
<td>iml</td>
<td>Interpretability tools</td>
<td>14</td>
</tr>
<tr>
<td>keras</td>
<td>Neural networks</td>
<td>8</td>
</tr>
<tr>
<td>lime</td>
<td>Interpretability</td>
<td>14</td>
</tr>
<tr>
<td>lmtest</td>
<td>Granger causality</td>
<td>15</td>
</tr>
<tr>
<td>lubridate</td>
<td>Handling dates</td>
<td>All (or many)</td>
</tr>
<tr>
<td>naivebayes</td>
<td>Naive Bayes classifier</td>
<td>10</td>
</tr>
<tr>
<td>pcalg</td>
<td>Causal graphs</td>
<td>15</td>
</tr>
<tr>
<td>quadprog</td>
<td>Quadratic programming</td>
<td>12</td>
</tr>
<tr>
<td>quantmod</td>
<td>Data extraction</td>
<td>6</td>
</tr>
<tr>
<td>randomForest</td>
<td>Random forests</td>
<td>7</td>
</tr>
<tr>
<td>rBayesianOptimization</td>
<td>Bayesian hyperparameter tuning</td>
<td>11</td>
</tr>
<tr>
<td>ReinforcementLearning</td>
<td>Reinforcement Learning</td>
<td>17</td>
</tr>
<tr>
<td>Rgraphviz *</td>
<td>Causal graphs</td>
<td>15</td>
</tr>
<tr>
<td>rpart and rpart.plot</td>
<td>Simple decision trees</td>
<td>7</td>
</tr>
<tr>
<td>spBayes</td>
<td>Bayesian linear regression</td>
<td>10</td>
</tr>
<tr>
<td>tidyverse</td>
<td>Environment for data science, data wrangling</td>
<td>All</td>
</tr>
<tr>
<td>xgboost</td>
<td>Boosted trees</td>
<td>7</td>
</tr>
<tr>
<td>xtable</td>
<td>Table formatting</td>
<td>4</td>
</tr>
</tbody>
</table>

Of all of these packages (or collections thereof), the tidyverse and lubridate are compulsory in almost all sections of the book. To install a new package in R, just type `install.packages("name_of_the_package")`

in the console. Sometimes, because of function name conflicts (especially with the select() function), we use the syntax `package::function()` to make sure the function call is from the

---


3By copy-pasting the content of the package in the library folder. To get the address of the folder, execute the command `.libPaths()` in the R console.
right source. The exact version of the packages used to compile the book is listed in the “renv.lock” file available on the book’s GitHub web page https://github.com/shokru/mlfactor.github.io. One minor comment is the following: while the functions `gather()` and `spread()` from the `dplyr` package have been superseded by `pivot_longer()` and `pivot_wider()`, we still use them because of their much more compact syntax.

As much as we could, we created short code chunks and commented each line whenever we felt it was useful. Comments are displayed at the end of a row and preceded with a single hashtag #.

The book is constructed as a very big notebook, thus results are often presented below code chunks. They can be graphs or tables. Sometimes, they are simple numbers and are preceded with two hashtags ##. The example below illustrates this formatting.

```
1+2  # Example

## [1] 3
```

The book can be viewed as a very big tutorial. Therefore, most of the chunks depend on previously defined variables. When replicating parts of the code (via online code), please make sure that the environment includes all relevant variables. One best practice is to always start by running all code chunks from Chapter 1. For the exercises, we often resort to variables created in the corresponding chapters.

Acknowledgments

The core of the book was prepared for a series of lectures given by one of the authors to students of master’s degrees in finance at EMLYON Business School and at the Imperial College Business School in the Spring of 2019. We are grateful to those students who asked fruitful questions and thereby contributed to improve the content of the book.

We are grateful to Bertrand Tavin and Gautier Marti for their thorough screening of the book. We also thank Eric André, Aurélie Brossard, Alban Cousin, Frédérique Girod, Philippe Huber, Jean-Michel Maeso, Javier Nogales and for friendly reviews; Christophe Dervieux for his help with bookdown; Mislav Sagovac and Vu Tran for their early feedback; John Kimmel for making this happen and Jonathan Regenstein for his availability, no matter the topic. Lastly, we are grateful for the anonymous reviews collected by John.

Future developments

Machine learning and factor investing are two immense research domains and the overlap between the two is also quite substantial and developing at a fast pace. The content of this book will always constitute a solid background, but it is naturally destined to obsolescence. Moreover, by construction, some subtopics and many references will have escaped our scrutiny. Our intent is to progressively improve the content of the book and update it with
the latest ongoing research. We will be grateful to any comment that helps correct or update
the monograph. Thank you for sending your feedback directly (via pull requests) on the
book’s website which is hosted at https://github.com/shokru/mlfactor.github.io.
Part I

Introduction
1

Notations and data

1.1 Notations

This section aims at providing the formal mathematical conventions that will be used throughout the book.

Bold notations indicate vectors and matrices. We use capital letters for matrices and lower case letters for vectors. \( \mathbf{v}' \) and \( \mathbf{M}' \) denote the transposes of \( \mathbf{v} \) and \( \mathbf{M} \). \( \mathbf{M} = [m]_{i,j} \), where \( i \) is the row index and \( j \) the column index.

We will work with two notations in parallel. The first one is the pure machine learning notation in which the labels (also called output, dependent variables or predicted variables) \( y = y_i \) are approximated by functions of features \( \mathbf{X}_i = (x_{i,1}, \ldots, x_{i,K}) \). The dimension of the feature matrix \( \mathbf{X} \) is \( I \times K \); there are \( I \) instances, records, or observations and each one of them has \( K \) attributes, features, inputs, or predictors which will serve as independent and explanatory variables (all these terms will be used interchangeably). Sometimes, to ease notations, we will write \( \mathbf{x}_i \) for one instance (one row) of \( \mathbf{X} \) or \( x_k \) for one (feature) column vector of \( \mathbf{X} \).

The second notation type pertains to finance and will directly relate to the first. We will often work with discrete returns \( r_{t,n} = p_{t,n}/p_{t-1,n} - 1 \) computed from price data. Here \( t \) is the time index and \( n \) the asset index. Unless specified otherwise, the return is always computed over one period, though this period can sometimes be one month or one year. Whenever confusion might occur, we will specify other notations for returns.

In line with our previous conventions, the number of return dates will be \( T \) and the number of assets, \( N \). The features or characteristics of assets will be denoted with \( x_{t,n}^{(k)} \): it is the time-\( t \) value of the \( k \)th attribute of firm or asset \( n \). In stacked notation, \( \mathbf{x}_{t,n} \) will stand for the vector of characteristics of asset \( n \) at time \( t \). Moreover, \( \mathbf{r}_t \) stands for all returns at time \( t \) while \( \mathbf{r}_n \) stands for all returns of asset \( n \). Often, returns will play the role of the dependent variable, or label (in ML terms). For the riskless asset, we will use the notation \( r_{t,f} \).

The link between the two notations will most of the time be the following. One instance (or observation) \( i \) will consist of one couple \( (t,n) \) of one particular date and one particular firm (if the data is perfectly rectangular with no missing field, \( I = T \times N \)). The label will usually be some performance measure of the firm computed over some future period, while the features will consist of the firm attributes at time \( t \). Hence, the purpose of the machine learning engine in factor investing will be to determine the model that maps the time-\( t \) characteristics of firms to their future performance.

In terms of canonical matrices: \( \mathbf{I}_N \) will denote the \( (N \times N) \) identity matrix.

From the probabilistic literature, we employ the expectation operator \( \mathbb{E}[] \) and the conditional expectation \( \mathbb{E}_t[] \), where the corresponding filtration \( \mathcal{F}_t \) corresponds to all information
available at time \( t \). More precisely, \( \mathbb{E}_t[\cdot] = \mathbb{E}[\cdot | \mathcal{F}_t] \). \( \nabla[\cdot] \) will denote the variance operator. Depending on the context, probabilities will be written simply \( P \), but sometimes we will use the heavier notation \( \mathbb{P} \). Probability density functions (pdfs) will be denoted with lowercase letters \( (f) \) and cumulative distribution functions (cdfs) with uppercase letters \( (F) \). We will write equality in distribution as \( X \overset{d}{=} Y \), which is equivalent to \( F_X(z) = F_Y(z) \) for all \( z \) on the support of the variables. For a random process \( X_t \), we say that it is stationary if the law of \( X_t \) is constant through time, i.e., \( X_t \overset{d}{=} X_s \), where \( \overset{d}{=} \) means equality in distribution. Sometimes, asymptotic behaviors will be characterized with the usual Landau notation \( o(\cdot) \) and \( O(\cdot) \). The symbol \( \propto \) refers to proportionality: \( x \propto y \) means that \( x \) is proportional to \( y \). With respect to derivatives, we use the standard notation \( \frac{d}{dx} \) when differentiating with respect to \( x \). We resort to the compact symbol \( \nabla \) when all derivatives are computed (gradient vector).

In equations, the left-hand side and right-hand side can be written more compactly: l.h.s. and r.h.s., respectively.

Finally, we turn to functions. We list a few below:

- \( 1_{[x]} \): the indicator function of the condition \( x \), which is equal to one if \( x \) is true and to zero otherwise.
- \( \phi(\cdot) \) and \( \Phi(\cdot) \) are the standard Gaussian pdf and cdf.
- \( \text{card}(\cdot) = \#(\cdot) \) are two notations for the cardinal function which evaluates the number of elements in a given set (provided as argument of the function).
- \( [\cdot] \) is the integer part function.
- for a real number \( x \), \( [x]^+ \) is the positive part of \( x \), that is \( \max(0, x) \).
- \( \tanh(\cdot) \) is the hyperbolic tangent: \( \tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} \).
- \( \text{ReLU}(\cdot) \) is the rectified linear unit: \( \text{ReLU}(x) = \max(0, x) \).
- \( s(\cdot) \) will be the softmax function: \( s(x)_i = \frac{e^{x_i}}{\sum_{j=1}^{n} e^{x_j}} \), where the subscript \( i \) refers to the \( i^{th} \) element of the vector.

### 1.2 Dataset

Throughout the book, and for the sake of reproducibility, we will illustrate the concepts we present with examples of implementation based on a single financial dataset available at [https://github.com/shokru/mlfactor.github.io/tree/master/material](https://github.com/shokru/mlfactor.github.io/tree/master/material). This dataset comprises information on 1,207 stocks listed in the US (possibly originating from Canada or Mexico). The time range starts in November 1998 and ends in March 2019. For each point in time, 93 characteristics describe the firms in the sample. These attributes cover a wide range of topics:

- **valuation** (earning yields, accounting ratios);
- **profitability** and quality (return on equity);
- **momentum** and technical analysis (past returns, relative strength index);
- **risk** (volatilities);
1.2 Dataset

- estimates (earnings-per-share);
- volume and liquidity (share turnover).

The sample is not perfectly rectangular: there are no missing points, but the number of firms and their attributes is not constant through time. This makes the computations in the backtest more tricky, but also more realistic.

```r
library(tidyverse) # Activate the data science package
library(lubridate) # Activate the date management package
load("data_ml.RData") # Load the data
data_ml <- data_ml %>%
  filter(date > "1999-12-31",
         date < "2019-01-01") %>%
  arrange(stock_id, date) # Order the data

data_ml[1:6, 1:6] # Sample values
```

The data has 99 columns and 268336 rows. The first two columns indicate the stock identifier and the date. The next 93 columns are the features (see Table 17.1 in the Appendix for details). The last four columns are the labels. The points are sampled at the monthly frequency. As is always the case in practice, the number of assets changes with time, as is shown in Figure 1.1.

```
data_ml %>%
group_by(date) %>%
  summarize(nb_assets = stock_id %>%
            as.factor() %>% nlevels()) %>%
  ggplot(aes(x = date, y = nb_assets)) + geom_col() +
  coord_fixed(3)
```

![Graph showing the number of assets through time.](image)

**FIGURE 1.1:** Number of assets through time.
There are four immediate labels in the dataset: R1M_Usd, R3M_Usd, R6M_Usd and R12M_Usd, which correspond to the 1-month, 3-month, 6-month and 12-month future/forward returns of the stocks. The returns are total returns, that is, they incorporate potential dividend payments over the considered periods. This is a better proxy of financial gain compared to price returns only. We refer to the analysis of Hartzmark and Solomon (2019) for a study on the impact of decoupling price returns and dividends. These labels are located in the last 4 columns of the dataset. We provide their descriptive statistics below.

```r
## A tibble: 4 x 5
##  Label mean     sd    min    max
##  <chr>    <dbl>  <dbl>  <dbl>  <dbl>
##1 R12M_Usd 0.137  0.738 -0.991  96.0
##2 R1M_Usd  0.013  0.176 -0.922  30.2
##3 R3M_Usd  0.037  0.328 -0.929  39.4
##4 R6M_Usd  0.072  0.527 -0.980 107.
```

In anticipation for future models, we keep the name of the predictors in memory. In addition, we also keep a much shorter list of predictors.

```r
```

The predictors have been uniformized, that is, for any given feature and time point, the distribution is uniform. Given 1,207 stocks, the graph below cannot display a perfect rectangle.

```r
data_ml %>% filter(date == "2000-02-29") %>% ggplot(aes(x = Div_Yld)) + geom_histogram(bins = 100) + coord_fixed(0.03)
```

**FIGURE 1.2:** Distribution of the dividend yield feature on date 2000-02-29.

The original labels (future returns) are numerical and will be used for regression exercises, that is, when the objective is to predict a scalar real number. Sometimes, the exercises can be different and the purpose may be to forecast categories (also called classes), like “buy”, “hold” or “sell”. In order to be able to perform this type of classification analysis, we create additional labels that are categorical.

```r
data_ml <- data_ml %>% group_by(date) %>% mutate(R1M_Usd_C = R1M_Usd > median(R1M_Usd), R12M_Usd_C = R1M_Usd > median(R12M_Usd))
```
The new labels are binary: they are equal to 1 (true) if the original return is above that of the median return over the considered period and to 0 (false) if not. Hence, at each point in time, half of the sample has a label equal to zero and the other half to one: some stocks overperform and others underperform.

In machine learning, models are estimated on one portion of data (training set) and then tested on another portion of the data (testing set) to assess their quality. We split our sample accordingly.

```r
separation_date <- as.Date("2014-01-15")
training_sample <- filter(data_ml, date < separation_date)
testing_sample <- filter(data_ml, date >= separation_date)
```

We also keep in memory a few key variables, like the list of asset identifiers and a rectangular version of returns. For simplicity, in the computation of the latter, we shrink the investment universe to keep only the stocks for which we have the maximum number of points.

```r
stock_ids <- levels(as.factor(data_ml$stock_id)) # A list of all stock_ids
stock_days <- data_ml %>%
  group_by(stock_id) %>%
  summarize(nb = n()) # Compute the number of data points per stock
stock_ids_short <- stock_ids[which(stock_days$nb == max(stock_days$nb))] # Stocks with full data returns
filter(stock_id %in% stock_ids_short) %>%
  dplyr::select(date, stock_id, R1M_Usd) %>%
  spread(key = stock_id, value = R1M_Usd) # 3. Put in matrix shape
```
Bibliography


Bibliography


Bibliography


Bibliography


