Open University of Catalonia (UOC) Master's Degree in Data Science

MASTER'S THESIS

Area: Data Analysis & Big Data

Applying Machine Learning to Investment Management

Building a strategy to consistently beat the benchmark

Author: Joan Giné Rabadán

Tutor: David García Agudíez

Professor: Albert Solé Ribalta

Barcelona, January 9, 2024

Credits/Copyright

FINAL PROJECT RECORD

Title of the project:	Applying Machine Learning to Investment Management
Author's name:	Joan Giné Rabadán
Collaborating teacher's name:	David García Agudíez
PRA's name:	Albert Solé Ribalta
Delivery date (mm/yyyy):	01/2024
Degree or program:	Master's Programme in Data Science
Final Project area:	Master's Thesis
Language of the project:	English
Keywords	S&P 500, Exchange Trading Fund, Investments

Abstract

The aim of this work is to build a portfolio that is able to outperform the S&P 500 stock index. This portfolio will be composed by units of the Exchange Trading Funds of the most important sectors of the U.S. economy. The time series of the unit price of the ETFs and the stock index will be obtained with a depth of 10 years. Likewise, it will be attempted to incorporate additional variables that can provide predictive power to the model, such as inflation or gross domestic product. Next, regression machine learning models will be trained in order to determine, for each month, the portfolio that can provide the maximum return. The obtained models will undergo a backtest to determine their predictive capacity. First, a cross-validation will be carried out; if the result is satisfactory, a backtest will be done by means of an out-of-sample sample. Finally, the selected models will be compared with other risk-oriented portfolio management models such as an equal-weighted portfolio or an efficient frontier limit portfolio.

L'objectiu d'aquest treball és construir una cartera que sigui capaç d'obtenir una rendibilitat superior a la de l'índex borsari S&P 500. La cartera estarà composada pels *Exchange Trading Funds* dels sectors més importants de l'economia d'Estats Units. La profunditat històrica de cotitzacions de l'índex de referència i dels ETFs serà de 10 anys. A banda, s'intentaran incorporar altres variables que aportin poder predictiu al model, com per exemple la inflació o el producte interior brut. Seguidament, s'entrenaran models d'aprenentatge automàtic de regressió per tal de, a cada mes, determinar la composició de la cartera que aporti una rendibilitat màxima. Al model obtingut se li durà a terme un exercici de *backtest* per tal de determinar la seva capacitat predictiva. Primerament es durà a terme una validació creuada; si el resultat és satisfactori, es durà a terme un *backtest* mitjançant una mostra *out-of-sample*. Finalment, els models seleccionats seran comparats amb altres estratègies de gestió de carteres orientades a la gestió de riscos, tals com una cartera amb pesos iguals, o en el límit de frontera eficient.

Keywords: S&P 500, Exchange Trading Fund, Investments.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Topic description and reason for the interest and relevance of the topic

The goal of this work is to build a model that manages a portfolio formed by units of the ETFs of the main sectors of the U.S. economy, with the aim of outperforming the S&P500 stock index. Building a strategy that consistently outperforms the benchmark is a goal that has not yet been achieved. What is more, data science has become a tool used to automate decision-making processes through analytics and statistics. Therefore, it is interesting to tackle this financial problem with this cutting-edge methodology.

1.2 Personal Motivation

My academic background has a technical and financial side. Specifically, I hold a Bachelor Degree in Physics, a Master's Degree in Financial Mathematics and I am currently finishing the studies of this Master's Degree in Data Science. The topic of this dissertation brings together the concepts of these studies, i.e., the use of data science knowledge such as machine learning, applied to finance. Although this research is very interesting both academically and practically, it also has a symbolic value since it puts into practice much of the knowledge that I have obtained throughout the course of these years. Besides, with this work I intend to put my academic life to an end. At least, for the time being.

1.3 Hypothesis

The hypothesis that this work seeks to test is that it is possible to build a portfolio that consistently outperforms the benchmark stock index.

1.4 Aims' definition

As mentioned before, the main goal of this work is to build a portfolio that is able to consistently outperform the benchmark. Secondly, it is desired to find additional variables that improve the performance of the statistical model that will predict the return of the assets in the portfolio. In addition, a backtest will be performed to validate the predictive power of the model. Finally, there is a wish to to compare the obtained result with other risk-oriented portfolio management strategies such as an equal-weighted portfolio or an efficient frontier limit portfolio.

1.5 Methodology Description

In this work, some of the methodologies described in Jansen's book (Jansen, 2020) will be used when selecting data and additional variables, training and validating the models. However, it will adapted to the described trading strategy, which is not covered in the mentioned source. That said, the steps that will be followed are listed below:

- Data collection.
- Selection and data collection of additional variables.
- Training of regression and classification models.
- Backtest.
- Selection of the model with the best backtesting results.
- Comparison of the results with those of portfolio management strategies oriented to risk management.

1.6 Competence in Global and Ethical Engagement and Sustainable Development Goals (SDGs)

The UOC is publicly committed to the CCEG and the SDGs, which are included in the Master's Programme, with the following definition: "Act honestly, ethically, sustainably, socially

responsible and respectful of human rights and diversity, both in academic and professional practice, and design solutions to improve these practices."

The CCEG addresses three major dimensions: (I) Sustainability, (II) Ethical Behavior and Social Responsibility (SR), and (III) Diversity (gender, among others) and Human Rights.

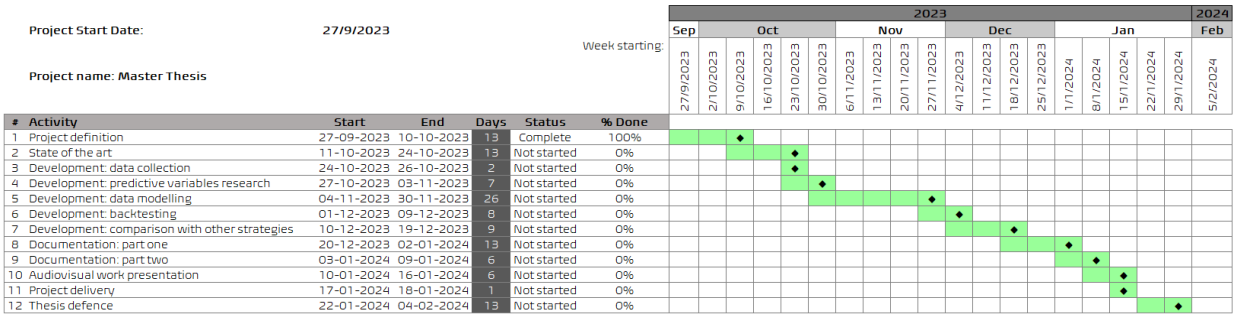
With regards to the first dimension, this work relies on a very limited amount of resources, which are mainly computational ones. They are required to train and validate the models. What is more, if this project was implemented, it could also be done virtually. For instance, it is very sustainable overall, given the very limited amount of resources its development will consume.

Concerning the second dimension, this work uses data that is publicly available. Apart from that, financial data will be used, which does not contain any personal or private information whatsoever. Besides, the documentation of the generated model will be written in full detail.

Finally, regarding the third dimension, it has to be emphasised that this work does not involve the use of any personal or private information. On the contrary, it just uses financial data that is publicly available. As a consequence, this project does not question or disrespect the diversity of gender, sexual orientation or human rights whatsoever.

1.7 Work Schedule

In this Section, a Gantt Diagram is presented in Figure 1.1 in order to illustrate the start and end dates of the distinct parts of the project to be completed throughout the course of the development of this work.



#	Activity	Start	End	Days	Status	% Done
1	Project definition	27-09-2023	10-10-2023	13	Complete	100%
2	State of the art	11-10-2023	24-10-2023	13	Not started	0%
3	Development: data collection	24-10-2023	26-10-2023	2	Not started	0%
4	Development: predictive variables research	27-10-2023	03-11-2023	7	Not started	0%
5	Development: data modelling	04-11-2023	30-11-2023	26	Not started	0%
6	Development: backtesting	01-12-2023	09-12-2023	8	Not started	0%
7	Development: comparison with other strategies	10-12-2023	19-12-2023	9	Not started	0%
8	Documentation: part one	20-12-2023	02-01-2024	13	Not started	0%
9	Documentation: part two	03-01-2024	09-01-2024	6	Not started	0%
10	Audiovisual work presentation	10-01-2024	16-01-2024	6	Not started	0%
11	Project delivery	17-01-2024	18-01-2024	1	Not started	0%
12	Thesis defence	22-01-2024	04-02-2024	13	Not started	0%

Figure 1.1: Gantt diagram that illustrates the time period in which the distinct phases of the project should be completed. The diamond on every line of the calendar indicates the week in which the task shall be completed.

Chapter 2

State of the Art

2.1 A brief introduction to portfolio management

In simple words, the return that a financial asset provides depends on the risk associated with that asset. For instance, since risk factors predict returns, identifying and forecasting their evolution becomes a top priority when it comes to designing an investment strategy.

Modern portfolio theory introduced the distinction between idiosyncratic risk and systematic sources of risk for a given asset. The former can be neutralised by diversification while the latter cannot be eliminated. In 1963, William F. Sharpe formulated the Capital Asset Pricing Model, which identifies a single risk factor that drives all asset returns, the risk factor being the difference between the return of the market portfolio and the return of the Government's treasury bills (Sharpe, 1963). According to that model, the systematic exposure of an asset to the market is measured by beta, β, which is computed as shown below:

$$\beta = \frac{\mathrm{Cov}(r_a, r_b)}{\mathrm{Var}(r_b)} \tag{2.1}$$

In accordance to that, the rate of return of an asset in the CAPM is stated in the following equation:

$$r = R_f + \beta(R_m - R_f) \tag{2.2}$$

Where r accounts for the expected rate of return of the portfolio, R_f is the return of the Government's tresasury bills (or risk-free rate) and R_m is the expected rate of return of the market.

The assumption that the risk of a financial asset depends on how it evolves relative to the

market as a whole, was a major conceptual breakthrough. However, in reality, this model has received multiple criticism with regards to this prediction. Instead, it is claimed that there are additional risk factors that need to be taken into account.

These risk factors were called anomalies since they contradicted the efficient market hypothesis. This hypothesis, formerly known as efficient market theory, states that prices on a financial market reflect all of the available information, which implies that it is impossible to design a portfolio that generates a yield greater than that of the benchmark. Hence, the ongoing discovery of risk factors that are a source of asset returns is a key driver of the appearance of Machine Learning in the investment industry.

2.2 History of algorithmic trading

Since the arrival of the internet in the early 1980's, financial markets have undergone a great modernisation by implementing technologies that have allowed them to execute trades electronically. Along with that, the upcoming of financial instruments pricing models such as the Capital Asset Pricing Model (Sharpe, 1964) or the Black-Scholes-Merton model (Black, Scholes, 1973) also contributed to the aforementioned modernisation, especially within the scope of asset management and trading. In 1997, the order-handling rules established by the U.S. Securities and Exchange Commission introduced competition to stock exchanges through electronic communication networks, which are automated alternative trading systems that match buy-and-sell orders at specified prices. Alternative trading systems allow significant brokerages and individual traders in different geographic locations to trade directly without intermediaries (Jansen, 2020).

Apart from that, direct market access provides traders with great control over trade executions, allowing them to operate in the exchange by means of a broker that is a member of that exchange. Sponsored access removes pre-trade risk controls by brokers, thus establishing the basis for high frequency trading. High frequency trading consists of automated trades executed with extremely low latency in the microsecond range, with the aim of detecting and exploiting inefficiencies in the market micro structure. This new industry has experienced a great growth over the last decade and is estimated to account for 55% of trading volume in US equity markets (Jansen, 2020).

2.3 Algorithmic Trading Performance and Appearance of Machine Learning in the Investment Industry

Since algorithmic trading arrived, many systematic strategies that rely solely on it have emerged and have achieved substantially high yields. Apart from that, it is known that many factors have contributed to applying machine learning to finance: data availability, computing power and advanced statistical methods. As a consequence, fund managers that would implement very different investment strategies, i.e. from hedge funds to passive management, have automated their trading strategies by means of machine learning algorithms. In fact, systematic funds (or algorithm-driven funds) became the largest driver of institutional trading in the US stock market in 2016; what is more, ML algorithms are becoming increasingly predominant (Jansen, 2020).

2.4 Machine Learning Models in Finance: Factor Research

As it has been mentioned, the huge availability of data partly fostered the appearance of Machine Learning in the investment industry. However, it is difficult to select data that can provide actual insights in order to effectively train a Machine Learning model that generates alpha, that is to say, excess returns in relation to a benchmark. There are several types of financial data (de Prado, 2018):

- **Fundamental data**: it consists of information about quality and quantity of equity performance on financial markets. It is commonly used to assess a company's value and for fundamental analysis of the asset.
- **Market data**: it consists of data that comes from the financial markets: volatility, price, dividends, etc.
- **Analytics**: alternative data can be defined as that type of information that comes from financial analysts, credit rating, earnings expectations, etc.

In terms of factor research, it has already been mentioned that apart from the price data, some other predictive variables will be tested in order to improve the performance of the models that are proposed. In fact, some examples were already provided so far: GDP, technical analysis indicators, interest rates, etc. However, fundamental data is very regularised and low frequency. Since it is so accessible to the marketplace, it is rather unlikely that it has much

value left to be exploited. It might, however, be useful in combination with other data types (de Prado, 2018). Hence, the focus will be on market data such as volatility or volume, even though fundamental data will be put to test in any case.

Moreover, it is worth mentioning that there are other pricing models that take into account more than one single risk factor. For instance, in 1993, French and Farma developed a model which consists of an extension of the Capital Asset Pricing Model (French, Farma, 1993). The main difference is that it considers more than just one risk factor:

$$r = R_f + \beta(R_m - R_f) + b_s \cdot SMB + b_v \cdot HML + \alpha \tag{2.3}$$

Again, r is the expected rate of return of the portfolio, R_f accounts for the risk-free interest rate, R_m is the return of the market portfolio, SMB stands for Small (market capitalisation) Minus Big (market capitalisation) and HML stands for High (book-to-market ratio) Minus Low (book-to-market ratio). The last two factors measure the historic excess returns of small companies compared with big companies, and value stocks compared to growth stocks, respectively. In simple terms, growth companies or stocks are considered to be able to outperform the market over time while value companies are considered to be trading below their worth, thus providing a greater return.

In 2015, the authors extended the model by adding two extra factors: profitability and investment. The profitability factor (RMW) consists of the difference between the returns of firms with Robust and low (Weak) profitability and the investment factor (CMA) is the difference between the returns of firms that invest in a passive way and the return of firms that invest in an active way (Conservative Minus Aggressive).

2.5 Machine Learning Models

As it has been pointed out in previous sections, the aim of this work is to adapt an already designed methodology to a specific problem which has not yet been addressed. Therefore, in this section, an explanation regarding which machine learning models have been used in finance so far and which of them are intended to be used here so as to implement the project, will be provided.

First, it is desired to implement a Random Forest model. In short, a Random Forest is an ensemble model, that is to say, a collection of single decision tree models. They can be used both in classification and regression problems. In the former case, the average of the predictions

made by each decision tree is the output of the ensemble; in the latter, the class selected by most trees becomes the final output. The main advantage of choosing an ensemble model is that their performance is best among all machine learning models. However, cost of training and complexity increase while interpretability decreases.

In the present case, each individual decision tree model is trained with bootstrapping samples, that is to say, by selecting random samples with replacement from the training set. A random sample without replacement of the features in the dataset is also performed. Another advantage of Random Forests is that they have a built-in method to cross-validate the resulting model, which is called out-of-bag (OOB) sampling. Since every single decision tree is trained with a sample of the training set, it can be shown that, on average, one third of the training set is never used. Hence, these observations are used to perform a cross-validation on the ensemble model.

However, it is also desired to implement a model that uses a boosting strategy. Boosting does also consist of an ensemble of individual models that generate a unique output. Nonetheless, the strategy used when sampling the observations to train the models is different from the one bagging uses. In that case, the individual models are trained sequentially, which means that the data used to train them is repeatedly modified to reflect the cumulative learning, thus ensuring that the next base learner compensates for the drawbacks of the current ensemble.

Throughout the course of the last few years, new gradient boosting algorithms have emerged. They use a number of innovations that accelerate training and allow the resulting algorithm to scale to very large datasets. The most relevant algorithms and their sources are the following:

- XGBoost
- LightGBM
- CatBoost

Regarding relevant works that use these algorithms, Basak et al. trained two algorithms, Random Forests and gradient boosting decision trees (XGBoost). These models predicted whether the stock of some companies would increase or decrease with respect to the price prevailing n days earlier. They tested their approach and reported the accuracies for a variety of companies as an improvement over existing predictions (Basak et al., 2019).

Finally, Deep Learning is a relatively new discipline that has achieved a number of breakthroughs in many domains, ranging from robotics to image processing, among other applications. In particular, the Deep Learning statistical models that are more in use in finance are Neural Networks.

As early as 2016, Krauss et al. developed a deep neural network model that was able to obtain positive returns by means of performing statistical arbitrage (Krauss et al., 2016). In addition, they also trained a gradient-boosted tree and a random forest. Statistical arbitrage involves taking a long position and a short position on securities that are correlated. Therefore, these strategies are able to generate returns as long as there are inefficiencies in the market.

2.6 Sector Rotation Strategy

As it has been mentioned before, in this work there is a wish to build a portfolio made of ETFs that represent the main sectors of the US economy. An ETF is a financial instrument that strongly resembles a regular investment fund. However, the ETF price fluctuates during trading hours just as a regular stock would do; instead, the price of an investment fund remains constant throughout the day. Apart from that, an ETF can be bought and sold during trading hours whereas in traditional investment funds, that can only be done once a day. Finally, ETFs offer low expense ratios and broker commissions.

The composition of the portfolio will vary depending on the predicted return of the ETFs by the model. In 2021, Karatas et al. developed a two-step methodology for ranking sectors using a wide range of common and sector-specific macroeconomic indicators (Karatas et al., 2021). The first step of the methodology predicts future prices of the sector indices and the prediction is made by a neural network model; the second step ranks the sectors according to their predicted rate of return. The top 4 indices will form the portfolio. The strategy was tested over short-term, mid-term and long-term. The results they obtained were able to beat the benchmark consistently.

In the present case, the rotation frequency will be monthly, but this work will surely provide a good reference for this project.

Chapter 3

Content and Methods

In this section, the methodology that has been used to develop the described project will be explained. It has to be mentioned that, due to time constraints, it has not been possible to implement all the models that were proposed in Section 2; instead, the LightGBM Boosting Model and a Random Forest Model, both in their regression versions, have been implemented and successfully validated.

With regards to Random Forest models, as it was mentioned in Section 2.5, they present many advantages, the first one being that they provide a very good accuracy thanks to being an ensemble model, although at the expense of a loss of interpretability. What is more, the trees that conform the random forest are not correlated to one another. Hence, no bias is introduced and as a consequence, overfitting chances decrease.

On the other hand, a boosting model does offer some advantages. Again, the first one of them is that it is an ensemble model. Apart from that, the key difference between random forests and gradient boosting models is that the individual models are trained sequentially. More concretely, the training set is modified every time based on the cumulative errors made by the model so far. In other words, it proceeds sequentially using reweighted versions of the data (Jansen, 2020). In addition, state-of-the-art boosting implementations also adopt the randomisation strategies of random forests. As a result, boosting models tend to have a slightly higher performance than random forest's. Finally, the LightGBM Boosting Model was developed by Microsoft. It is an open-source alternative that offers many advantages over other boosting models, such as faster training speed and higher efficiency, lower memory usage and better accuracy, among others (Microsoft, 2023).

Finally, as it was mentioned before, two models had to be chosen among those proposed in

previous sections due to time constraints. In the present case, the neural network model has been discarded. Even though there is scientific evidence that they perform well (Krauss et. al., 2017), it has been decided to drop their implementation because they are very resource consuming and they have a substantial lack of transparency in their decision-making (also known as black box model).

Each of the implemented models has been studied under two different approaches. Firstly, the target has been defined as the return on a one month horizon. Secondly, the target variable has been redefined as the return on a one month horizon minus that of the benchmark (S&P 500 or, in that case, ETF SPY).

In order to implement the models, the code that is supplied in the book by Stefan Jansen has been adapted. Hence, the methodology used to obtain the data, perform the data cleaning, modelling, validation and backtest of the results will be explained in full detail. The structure of the code is the same in all cases (LightGBM and Random Forest; plain returns and returns adjusted to the benchmark).

3.1 Data Extraction

Firstly, it is important to point out that this script needs an input, which is the historical data of the ETFs under consideration. As Jansen suggests, it has been obtained from the *stooq* website. There, it is possible to obtain a zip file with the desired data. Once that input has been downloaded, the provided code transforms the zip file into an h5 file. An h5 file, or Hierarchical Data Format File, is an open-source file that is able to store large, complex and heterogeneous data. The mentioned output has the following columns:

- **open**: it contains the price of the corresponding asset at the beginning of the trading day.
- **high**: the value of this variable corresponds to the highest price the asset reached during the trading day.
- **low**: the value of this variable corresponds to the lowest price the asset had during the trading day.
- **close**: the value of this variable corresponds to the price the asset had when the trading day ended. That variable will be used in future calculations.

- **volume**: the volume of a share consists of the number of shares of a security traded during a given period of time.

This data is obtained for all the ETFs considered, during the period 2000-2019.

3.2 Feature Engineering

This part of the code addresses, on the one hand, data preprocessing, which consists of selecting those variables of the initial dataset that will be used in the modelling step, dealing with missing values and other data transformations.

The first step is to select the ETFs that will form the desired portfolio:

- **IYR**: Real Estate sector.
- **SPY**: ETF that replicates the Standard and Poors 500 stock index. It will be the benchmark.
- **VOX**: Communication Services sector.
- **XLB**: Materials sector.
- **XLE**: Energy sector.
- **XLF**: Financial Services sector.
- **XLI**: Industrial sector.
- **XLK**: Technology sector.
- **XLP**: Consumer Staples sector.
- **XLU**: Utilities sector.
- **XLV**: Health Care sector.
- **XLY**: Consumer Discretionary sector.

Next, the time period is selected, which will be between the years 2005 and 2023. After that, the resulting datasets (one for each ETF) are modified to have monthly data; for each month, they only contain the row of its last trading day. That is done in order to reduce training time and to be able to experiment with strategies for longer time horizons. There are no missing

values.

The next step requires to compute historical returns over various monthly periods which will be identified by lags. Concretely, historical returns over periods of 1, 3, 6, 9 and 12 months are computed. The code obtains a second dataset whose columns are the distinct computed returns. In addition, this dataframe contains an index that identifies each row by the name of the ETF and the monthly date. The returns are computed as it follows:

$$\text{return}_{\text{lag}} = (1 + r_{\text{lag}})^{1/\text{lag}} \tag{3.1}$$

Where

$$r_{\text{lag}} = \frac{P_t - P_{t-\text{lag}}}{P_{t-\text{lag}}} \tag{3.2}$$

In case that the target variable is defined as the returns minus those of the benchmark, the latter has to be subtracted from the former. Moreover, those values that are outside of the [0.1, 0.99] percentile interval of each return sample are deleted since they are considered to be outliers. Next, those ETFs that have less than 10 years of returns are also discarded; however, there are no such cases.

After that, the 5 Fama-French factors are obtained using the `pandas-datareader` library: **Mkt-RF**, **SMB**, **HML**, **RMW** and **CMA**. However, they are not the plain factors; instead, they represent the monthly returns of such factors. Hence, the monthly series from January 2000 until October 2023 is retrieved. After that, for each ETF and month date, an Ordinary Least Squares regression is trained. It estimates the one month return of the ETF as a function of the Fama-French factors' returns, taking data from two years prior to the current month date. Then, the estimated betas of the regression on the selected ETF and date are assigned respectively. These metrics are a measure of how the ETF's returns will evolve with respect to those of the Fama-French factor.

Subsequently, momentum factors are calculated as the difference between returns over longer periods and the most recent monthly return (1 month), as well as for the difference between 3 and 12 month returns. Apart from that, two additional variables are generated, month and year, which correspond to the month and year of every row, that is stored as an index. Moving on, more variables are created. In that case, they account for historical returns up to the current period. Concretely, the set of variables are return_1m_t-lag, where lag=1,...,6. In other words, the one month returns of the past are projected into the future.

Next, it is desired to compute the holding period returns: at time t, it is necessary to know the return that an asset will provide at time $t + 1$. Hence, the lagged returns are shifted back to obtain such variables (for lags of 1, 2, 3, 6 and 12 months).

The last step is to incorporate macroeconomic variables. The ones that are to be incorporated are the following (again, obtained thanks to the `pandas-datareader` library):

- **recession**: it is a dummy variable that takes the value 1 if a recession in the United States was taking place on a certain date and 0 otherwise.
- **yield curve**: yield curve of the United States expressed as 10-Year Treasury Constant Maturity minus 3-Month Treasury constant maturity.
- **financial conditions**: this variable reflects the United States financial conditions in money markets, debt and equity markets. Positive values indicate financial conditions that are tighter than average, while negative values indicate financial conditions that are looser than average.
- **leverage**: in finance, leverage is a technique which involves borrowing funds to buy an investment, estimating that earnings will surpass the cost of borrowing. This variable consists of debt and equity measures. As in the previous case, positive values indicate financial conditions that are tighter than average while negative values indicate financial conditions that are looser than average.
- **sentiment**: this variable expresses the consumer's confidence towards the future economic situation. An indicator above 100 signals a boost in consumers' confidence towards the future situation of the economy, while a value below that threshold indicates the opposite.
- **unemployment**: it expresses the unemployment rate in the United States.
- **inflation**: it indicates the inflation of the United States economy.

Finally, the obtained data is adjusted to monthly frequency and is joined to the previous dataset. The last modification is the following: some of the variables in the previous list are recalculated as the difference with the value of the previous month. The involved variables are **recession**, **yield curve**, **financial conditions**, **leverage**, **sentiment**, **unemployment rate** and **inflation**. Hence, the final dataset has the following variables:

- Returns: **return_1m**, **return_2m**, **return_3m**, **return_6m**, **return_9m**, **return_12m**.
- Fama-French factors: **Mkt-RF**, **SMB**, **HML**, **RMW**, **CMA**.
- Momentums: **momentum_2**, **momentum_3**, **momentum_6**, **momentum_9**, **momentum_12**, **momentum_3_12**
- Categorical Variables: **year**, **month**, **sector**.
- Lagged returns: **return_1m_t-1**, **return_1m_t-2**, **return_1m_t-3**, **return_1m_t-4**, **return_1m_t-5**, **return_1m_t-6**.
- Target Variables: **target_1m**, **target_2m**, **target_3m**, **target_6m**, **target_12m**.
- Macroeconomic variables: **recession**, **yield curve**, **financial conditions**, **leverage**, **sentiment**, **unemployment rate**, **inflation**.
- Lagged macroeconomic variables: **recession diff**, **yield curve diff**, **financial conditions diff**, **leverage diff**, **sentiment diff**, **unemployment rate diff**, **inflation diff**.

3.3 Hyperparameter search and Model Training

In this part of the code, a set of hyperparameter values for both models are tested, thus selecting the ones which predict most accurately. Since this project considers regression models, a metric that indicates the accuracy of the model's predictions has to be chosen. The same metric used in the book by Jansen will be considered, which is the Information Coefficient (IC). It is computed as the Spearman correlation coefficient between the set of predictions of the model and the actual values of the target variable. It is calculated as stated in the following equation:

$$\mathrm{IC} = 1 - \frac{6 \sum_i d_i^2}{n(n^2 - 1)} \tag{3.3}$$

Where n is the number of observations in the sample and d_i is the difference between the predicted value and the actual value.

Firstly, the dataset described in the previous section is selected. Then, three parameters are defined: train length, test length and lookahead. The lookahead is defined as the future time horizon, expressed in months, that will be used to compute the returns; in other words, the target variable. The values 12, 36 and 54 are selected as train lengths; 1 as test length and 1 as lookahead. Hence, 3 combinations of these parameters will be tested.

Apart from that, before training the model, the sector variable, which is the name of each ETF, will be one-hot encoded. Hence, there will be have 12 extra variables (as 12 ETFs were selected) that will take the value 1 in case the data belongs to the i-th ETF and 0 otherwise.

In order to have a first approximation of which set of hyperparameters works best, for each combination of them, a linear regression model will be trained. The target variable will be the one-month target return and the features will be the rest of the dataset (bearing in mind that the sector variable has been one-hot encoded). However, the model will not be trained just once. Instead, it will be trained repeatedly throughout the course of a defined time lapse which will be called validation years. In Figure 3.1 there is a diagram that illustrates the way in which this methodology, the so-called walk-forward procedure, works. In the first place, a time span (the validation years) is selected. In the present case, it will be of 6 years. Thereafter, the model is trained and tested with the data corresponding to the beginning of the time lapse. Subsequently, the model is trained again but the test and train set begin after one test set length, as it can be observed in Figure 3.1. Hence, the model is trained N times, where N is computed as it follows:

$$N_{\text{splits}} = \frac{t_{\text{validation}}}{t_{\text{test}}} \tag{3.4}$$

Since $t_{\text{validation}} = 6 \cdot 12 = 72$ months and $t_{\text{test}} = 1$ month, then $N_{\text{splits}} = 72$.

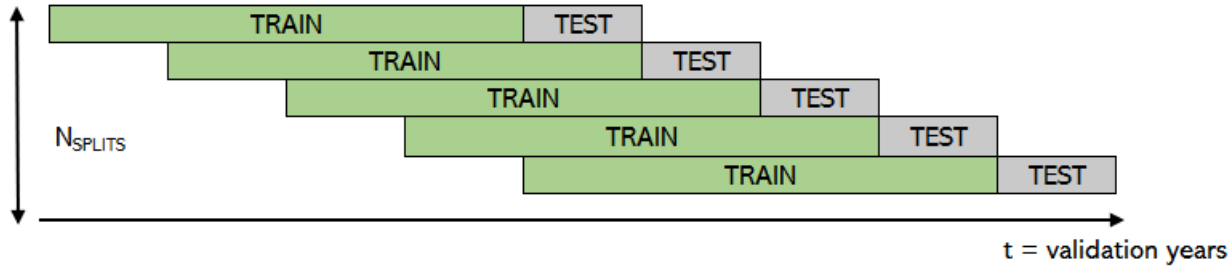


Figure 3.1: This diagram illustrates the walk-forward procedure. The first test set of the model begins at time $t = 0$ until the end of the train set. Next, the second test set begins at $t + \delta$, where δ is the time span of the test set, which in this case is 1 year.

The next step is to select the hyperparameters that will be needed to train the LGBM Boosting Model. They are listed below:

- **Learning rate**: it determines the step size taken at each iteration while the model moves towards a minimum of a loss function. High values would make the minimisation unstable whereas little values would make convergence low.

- **Maximum depth**: it determines a constraint on the structure (the depth) of each tree of the model.
- **Number of leaves**: it is the maximum number of leaves of each node. It will be calculated as $2^{\text{max\_depth}}$.
- **Minimum data in leaf**: it determines the minimum number of registers that have to be present in each row.
- **Feature fraction**: it is the fraction of features to be selected when constructing each tree. It can take values within the $(0, 1]$ interval.
- **Number of boost rounds**. this parameter will be used when predicting the values of the test set, not to train the model.
- **Number of boost iterations**. This hyperparameter will be used to train the model.

In the case of the Random Forest Model:

- **Bagging fraction**: it determines the percentage of the original dataset is used to train each individual model.
- **Feature fraction**: it determines the percentage of the features used to train each individual model.
- **Minimum data in leaf**.
- **Number of boost iterations**.

Hence, in both models, the number of hyperparameter values to be tested is, most likely, very large. For instance, a random, half in size sample (without replacement) of them will be used. Having that said, the next step is to train and test the LGBM Boosting Model and the Random Forest Model. The walk-forward methodology will be used again. Hence, an iteration over the 3 combinations of train length, test length and lookahead is carried out. At each iteration, the hyperparameters values are sampled. Next, the rows of the dataset that contain uninformed values are dropped. It is worth mentioning that in this case, the sector variable will not be one-hot encoded; instead, it will remain as a categorical variable. The next step consists in iterating over the sample of hyperparameters and, for each set, perform the walk-forward cross validation. At each walk-forward iteration, the values of the test set are predicted n times, n being the number of boost rounds values tested. Once the walk-forward loop has finished, the predictions are stored in a dataframe. The next step is the calculation

of the information coefficient, which is computed in two different ways: in the first place, the dataframe that contains the predictions is grouped by date. Then, the mean and the median of the information coefficient of each month is computed (bear in mind that, for each month, there are as much predictions as values for the number of boost rounds). The next step is to select the value of the number of boost rounds that yields the highest IC. Finally, the IC across all the predictions is computed (i.e. across all dates). Finally, the following metrics are stored:

- Hyperparameters set.
- Maximum value of the mean IC.
- Number of boost rounds that yielded the maximum value of the mean IC.
- Maximum value of the median IC.
- Number of boost rounds that yielded the maximum value of the median IC.
- Global IC.

3.4 Validation performance

In this part of the code, the performance of the models that were trained and validated in the last section will be evaluated. In the first place, the data that was obtained in the previous section is retrieved. On the one hand, the dataframe that contains information about the mean and median IC for each hyperparameter combination is also selected. On the other hand, there is a third dataframe involved. That one contains the information coefficient, the hyperparameters, the train and test lengths, the lookahead and the date. In other words, it stores the results of the walk-forward performed at each hyperparameter combination.

The next step is to build a linear regression so that it can be appreciated any dependency between the hyperparameter values and the IC, which again will be the target variable. To this end, the one-hot encoding procedure is used to convert each hyperparameter value into a dummy variable. The resulting variables will be the features. To build the linear regression model, the dataframe that does not contain aggregated data will be used.

Thereafter, the same dataframe is selected, the data is grouped by the hyperparameter values and, for each combination, the mean IC is calculated. That way, it is possible to find out which hyperparameter combination achieves a largest mean IC.

Next, the 10 best performing models and their predictions are selected. In turn, the ETFs' prices throughout the same period are selected.

3.5 Model interpretation

The goal of this section is to interpret the results of the best model. In the first place, the dataframe obtained in Section 3.2 is retrieved, selecting those rows that are between the years 2013 and 2018. The training set is selected as the last N months of the dataframe, N being the train length of the optimal model. It is then trained according to the tested hyperparameters. Thereafter, another metric called feature importance is computed. In short, it is a measure of the relevance each factor of the model has as to making predictions. The LGBM Boosting Model library offers two ways to calculate it:

- **Split Feature Importance**: it measures the number of times a feature is used to split the data across all trees in the model.
- **Gain Feature Importance**: this measure quantifies the improvement in the model's accuracy achieved by using a particular feature for splitting.

The next step is to do a Shapley values analysis. To this end, the `shap` Python library will be used.

The Shapley Values is a methodology that attempts to estimate the marginal accountability of each factor of a model to the final output. Shapley values are a concept borrowed from the cooperative game theory literature and dates back to the 1950s. In their original form, Shapley values were used to fairly attribute a player's contribution to the end result of a game (Gopinath, 2022). In terms of calculation, a Shapley value is computed by performing perturbations to the model's features and observing the resulting changes to the final model prediction. Often, they are computed with respect to a comparison group, which acts as a baseline when it comes to explaining the final output.

According to Gopinath, Shapely values satisfy a number of axioms that are listed below:

- **Completeness or Efficiency**: the sum of the contributions of each feature of the model must add up to the model output minus the average model output of the comparison group.
- **Dummy**: a feature that does not contribute to the model's output must have a contribution of 0.

- **Symmetry**: symmetric features contribute equally to the model output.
- **Monotonicity**: if a feature A that contributes consistently more to the output than another feature B, then feature A will have a higher Shapley value.
- **Linearity**: supposing that the model output was determined based on the sum of two intermediate values, each of which are derived from the input features, then, the contribution attributed to each feature will be equal to the sum of the contributions of the feature towards the intermediate values.

Mathematicaly, a Shapley value or the contribution, Φ, of a feature i is computed as it follows:

$$\Phi_i(v) = \sum_{S \subseteq N \setminus \{i\}} \frac{|S|!(N - |S| - 1)!}{N!} \Big(v\big(S \cup \{i\}\big) - v\big(S\big) \Big) \tag{3.5}$$

Where S is a subset of features, N is the total number of features and v is a function that maps subsets of features into real numbers.

Finally, it is worth mentioning that Shapley values do not necessitate to understand the model; instead, it only needs to use it as a predictor so as to dispose of the feature values and predicted values, thus being able to complete its analysis.

In this part of the script, the `shap` library will be used to perform such an analysis on the best performing model.

3.6 Generating out of sample predictions

In this section, the 10 best performing models are selected and predictions during a 4 year period -which will be called out-of-sample period- are generated.

Firstly, the engineered features data from 2010 until the end of the series is retrieved, as well as the dataframe that contains the mean and median IC for each model that was trained during the first walk forward.

Again, the training period will be the optimal value and the test period will be of 1 month. Hence, the number of splits is determined by the following expression:

$$N_{\text{splits}} = \frac{t_{\text{out of sample}}}{t_{\text{test}}} = 48 \tag{3.6}$$

For each of the 10 best performing models, they are trained during the out-of-sample period by means of the walk forward process. Finally, the resulting data is stored in a dataframe that contains the dates and the test predictions for all of the 10 models.

3.7 Vectorised backtest

The aim of this section is to compare the performance of a portfolio driven by the predictions of the model and the performance of the benchmark. Again, this script will differ on the basis of the target value, which can be defined as the returns of the ETFs or their excess returns with respect to the benchmark. Not to mention that this backtest will be done for the two implemented models, LGBM and Random Forest.

The first step is to collect the dataframe that contains the predictions of the 10 best performing models during the out-of-sample time period, both for the train period and the test period. In fact, the two sets are merged and, if duplicated values of (ETF, Date) occur, then the value that comes from the test set is deleted. Next, the average of the target variable for each month is computed. Hence, there will be just one column. As mentioned, each row is identified by the name of the ETF and the date. Thereafter, by using the `unstack()` method to transform the dataframe so that each row contains the information of each ETF at a fixed date; in the previous version, a row was identified by the date and the ETF name. Now, the ETFs are variables (columns) of the dataframe.

Thereafter, the composition of the portfolio at each month of the aforementioned period is computed. To do so, the first N ETFs that have positive one-month predicted returns are labeled; concurrently, the opposite is done: for each month, the N first ETFs that have negative one-month predicted returns are also labeled.

Next, the predicted one-month returns are assigned to those ETFs marked at each month in the case of positive returns and negative returns, in separate datasets. Thereafter, the two datasets are grouped by date and the average returns are computed in each case. Finally, the estimated return at each date will be the positive returns minus the negative returns; their sign has to be permuted because a short position will be simulated (a selling position) on the assets that have a forecasted negative return. Finally, a graph is simulated, which shows the cumulative returns of the strategy throughout the out-of-sample period, which is 4 years (2019-2023).

Although the IC is the only performance metric so far discussed, the results of the backtest

are of similar importance: at the end of the day, the goal of this Machine Learning project is to simulate a financial strategy that consistently beats the benchmark. For instance, even if a high IC is a good indicador of the predictive power of the model, it has to be able to generate returns that are consistently higher than those of the benchmark.

Chapter 4

Results and Conclusions

In this section, the results obtained after applying the methodology described in the previous section will be explained in detail.

As mentioned before, the methodology has been implemented with two different Machine Learning Models. In addition, every model has been trained and validated with two different target variables. For instance, this chapter will be divided into 2 sections and each of these 2 sections will be divided into two subsections.

4.1 LightGBM Boosting Model

4.1.1 One month excess return

The first result that is presented is the distribution of the IC resulting from the linear regressions explained in Section 3.3. In Figure 4.1, two different distributions of the Information Coefficient can be observed. The first one (on the left) shows the distribution of the cumulative test samples at every walk-forward iteration; the second case (on the right) shows the distribution of the test sample of each walk-forward iteration.

However, it is quite clear that these results are quite poor, which shows that the evolution of stock prices in particular and financial markets in general are not linear, which is why the linear regression model fails.

The next step is to explain the hyperparameter values that have been tested. They are listed below:

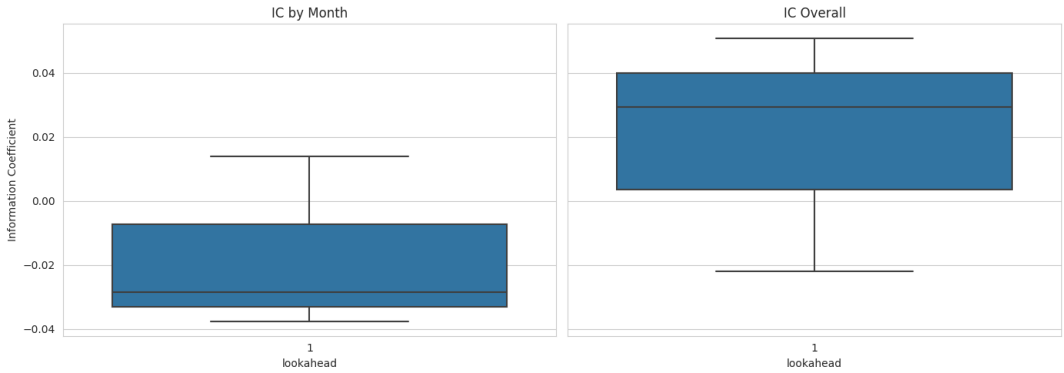


Figure 4.1: Distribution of the monthly IC and overall IC of the linear regressions estimated in Section 3.3.

- **Learning rate**: 0.01, 0.1, 0.3
- **Maximum depth**: 2, 3, 5, 7
- **Number of leaves**: 4, 8, 32, 128
- **Feature fraction**: 0.3, 0.6, 0.95
- **Minimum data in leaf**: 11, 22, 45
- **Boost rounds**: 50, 100

After testing each of them (although not every possible combination of them) by performing the walk-forward cross validation, it is interesting to see if each hyperparameter value has a correlation with the IC. For instance, an ordinary least squares regression has been trained. The independent variables have been each hyperparameter value, that is to say, the hyperparameter data has been one-hot encoded again. The dependent variable has been the IC. In this sense, in Figure 4.2 it can be observed the dependency between the values of the estimated coefficients and the IC. Firstly, it has to be mentioned that the number of tested values for each hyperparameter is not as large as to extract well-grounded conclusions. Besides, the error bars are very wide, which makes it even more difficult. In conclusion, even if some trends can be observed, they are not reliable.

Given that it was not possible to observe any solid dependency between the IC and the hyperparameter values, some other analysis have been explored. In this regard, in Figure 4.3 there are three different boxplots. Each of them illustrate the distribution of the overall IC depending on the train length. In general, it is clear that a 12 months train length yields lower IC values than 36 and 54 months. Nonetheless, it is difficult to tell which of the last two works

best: although they have similar median values, the 54 months distribution can achieve higher IC values even though the resulting distribution is wider than that of the 36 months.

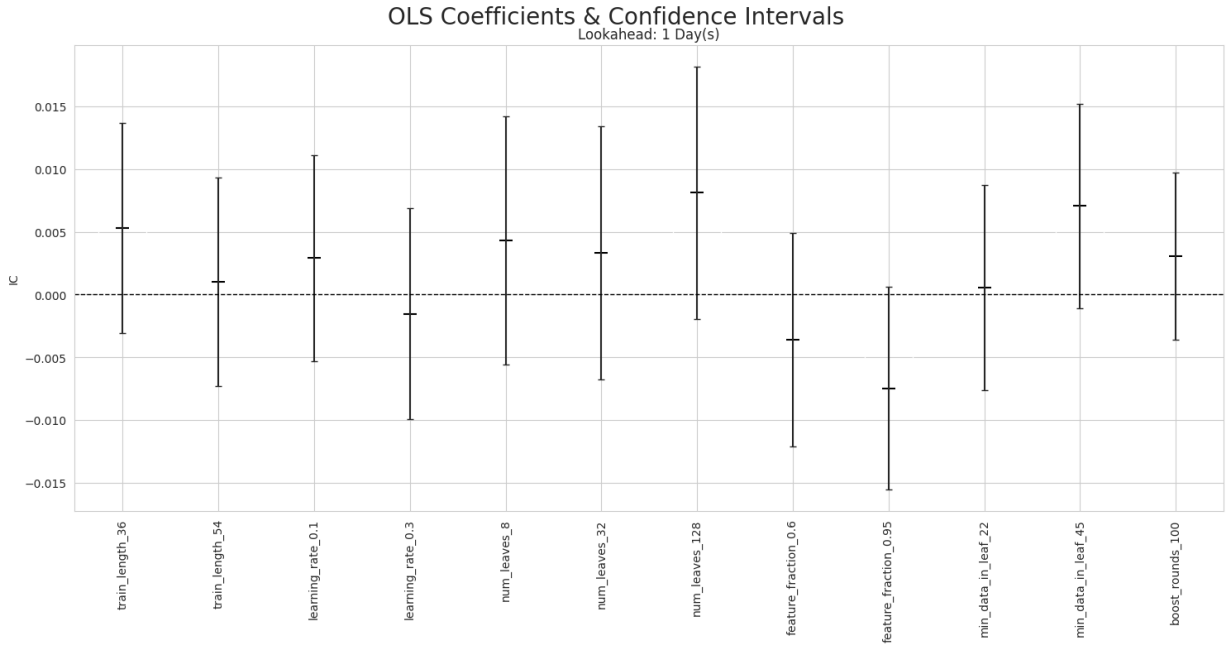


Figure 4.2: This graph shows the value of each regression coefficient estimated for each hyperparameter factor for trainings with 1 month of lookahead. The model in question is the LightGBM Boosting Model and the target variable is the excess return. Error bars are also presented.

In addition, in Figure 4.4, the overall IC distribution by number of boost rounds and train length can be observed. Again, the distributions resulting from a 54 months training length are wider. Moreover, even though the median value of the distributions with 100 boosting rounds is higher, there is not a clear conclusion about which is the optimal value of train length and boosting rounds.

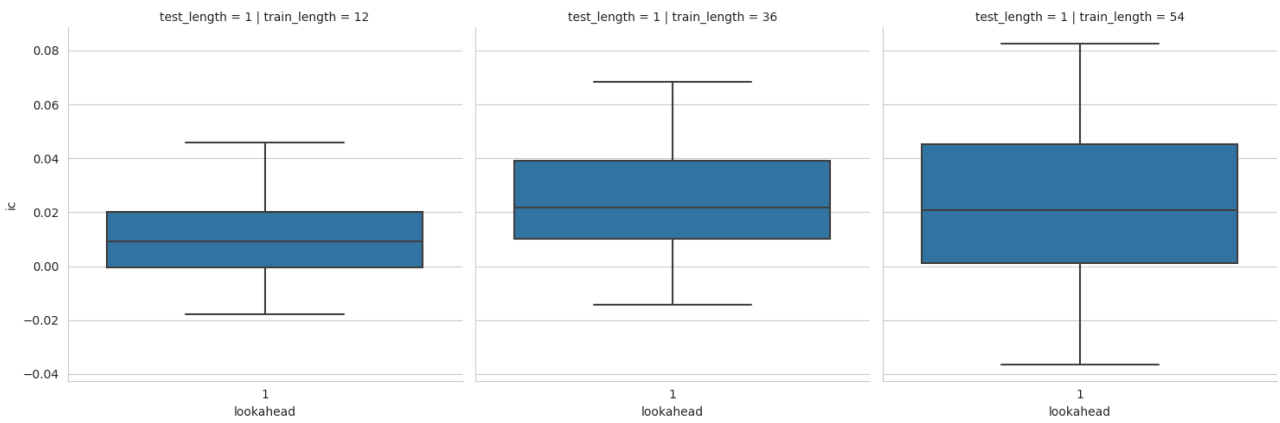


Figure 4.3: The Figure above depicts the distribution of the overall IC by train length. The model in question is the LightGBM Boosting Model and the target variable is the excess return. Although a big difference can be observed when comparing the lowest value (12) and the others (36, 54), the difference between the last two is much subtler.

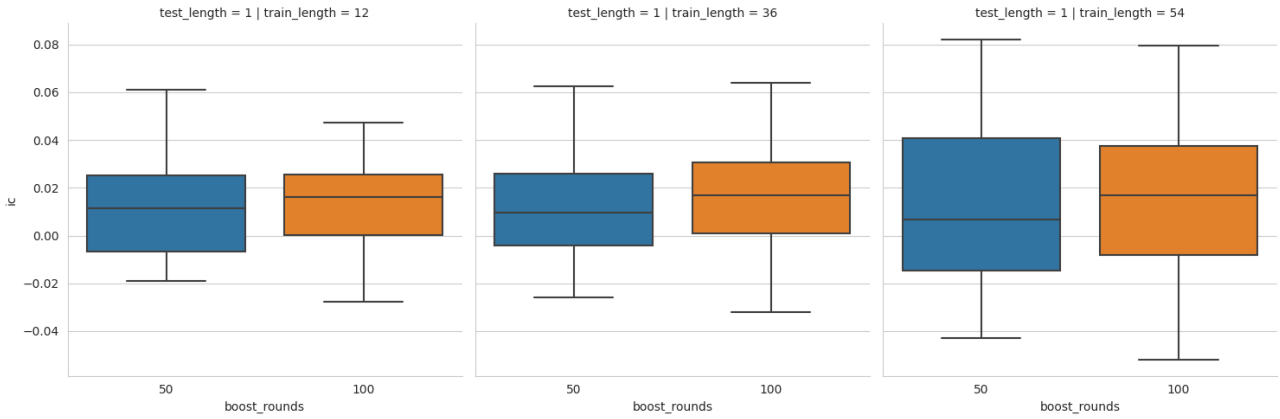


Figure 4.4: The Figure above shows the distribution of the overall IC by number of boost rounds and train length. The model in question is the LightGBM Boosting Model and the target variable is the excess return.

Finally, the optimal hyperparameter combination is presented in Table 4.1. The resulting model has an IC value of 8.26%.

Next, a graph with a three-months rolling average of the overall IC during the years 2013-2018 is presented in Figure 4.5. The series is quite volatile and it undergoes highs and lows throughout the corresponding years.

Train length	54
Test length	1
Learning rate	0.3
Number of leaves	8
Feature fraction	0.3
Minimum data in leaf	45
Boosting rounds	50

Table 4.1: Best hyperparameter set of values for the LGBM Boosting Model with excess return as target variable.

From now on, the best performing model will be analysed in more detail. In Figure 4.6, the normalised feature importance of the best performing model can be observed, both in the Gain and Split methods. Even though the two methods do not coincide in ranking the variables importance, they do coincide in the fact that the ones that contribute most in predicting the output are the lagged returns and the plain returns. Concretely, the one month and six months lagged returns and the 6, 9 and 12 months returns. Besides, the HML and CMA Fama-French factors are of relevance too. In addition, none of the macroeconomic variables that were incorporated (recession, yield curve, and so on) appear in neither of the two Top 10's.

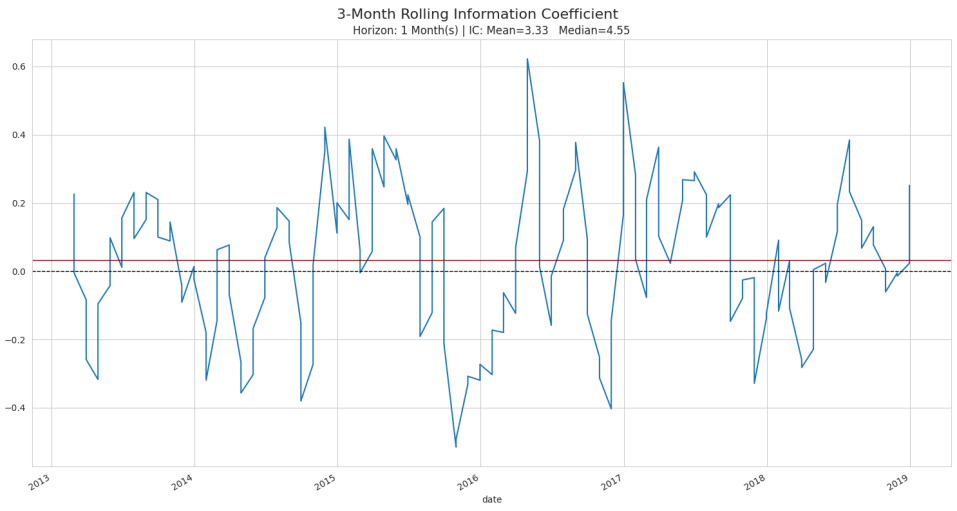


Figure 4.5: Mean three-months rolling average during the 2013-2018 period. The model in question is the LightGBM Boosting Model and the target variable is the excess return.

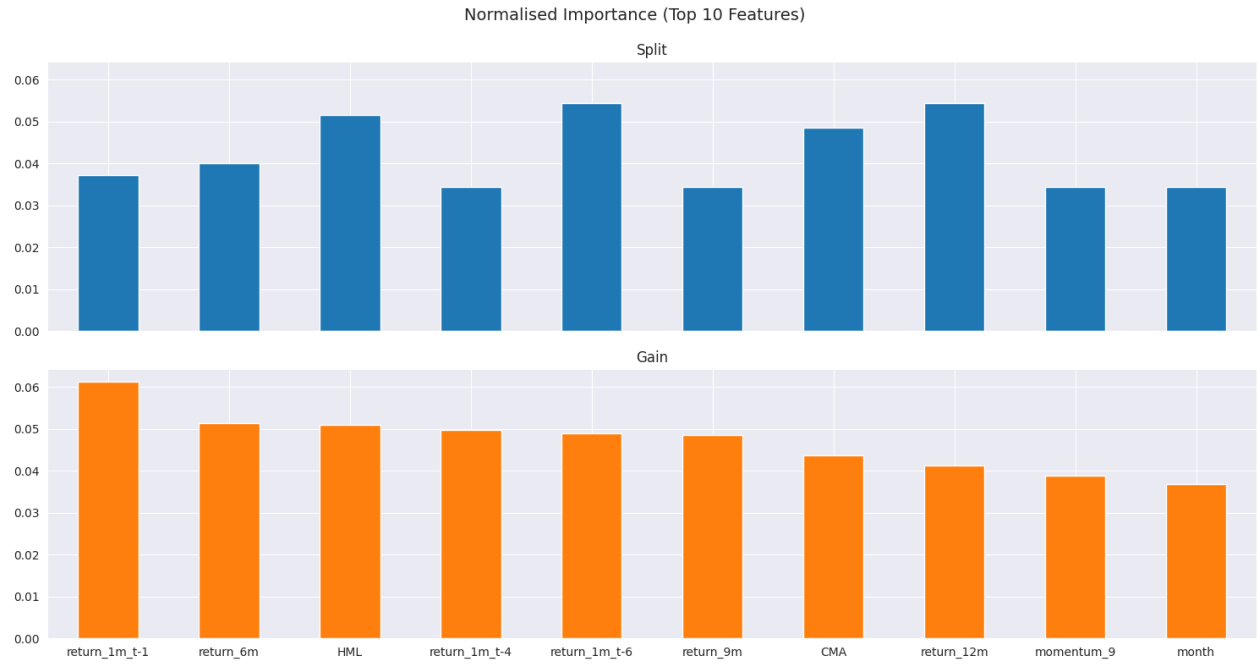


Figure 4.6: The graph above shows the normalised feature importance of the best performing model, both in Gain and split. The model in question is the LightGBM Boosting Model and the target variable is the excess return.

Next, the Shapley values analysis will be explained in detail. As it was mentioned before, the `shap` Python library will be used. Again, in Figure 4.7 it can be observed the Shapley distribution of the 20 most important factors -according to the Shapley analysis- on a 600 sample of the training set. The heat map indicates the actual value of the factor. Again, the Conservative Minus Aggressive Fama-French factor, along with the HML, appear to be among the most important factors. What is more, the CMA is the first one. In fact, according to Figure 4.7, low values of the CMA positively affect the output whereas high values affect it otherwise. In other words, the greater the returns the Conservatives funds achieve, the smaller returns the portfolio predicts.

Also, it is worth mentioning that the month variable is the second most important one. In that case, month is a categorical variable that has been mapped in order for it to take integer values (1 for January, 2 for February, and so on). There seem to be some months, though, that tend to influence the output in one way or another, especially those at the end of the year, which influence the output positively. In addition, it has to be noted that the lagged returns appear again on the top. Interestingly enough, low values of the return_1m_t-1 factor contribute positively to the output. Moreover, the 9 months momentum (the 9 month return minus the one month return) is the third most important variable. High values of it positively predict the

output while low values negatively do so. Finally, with regards to the 6 and 9 month returns variables, high values tend to positively predict the output while low values do the opposite.

Finally, the results of the backtest are presented in Figure 4.8. The left graph shows the cumulative returns throughout the out-of-sample test period, that is to say, 2019-2023. The results are very impressive: the returns accumulated during that period are slightly below 20%. Again, the predicted variable is the one-month excess return with respect to the benchmark. It has to be noted that the model has been trained with monthly frequency. For instance, the same methodology should be applied with daily data, which has not been done in this case due to time and computational resources constraints. Although in that case, the target variable should be redefined too, since using daily data to predict monthly returns may introduce autocorrelation to the model.

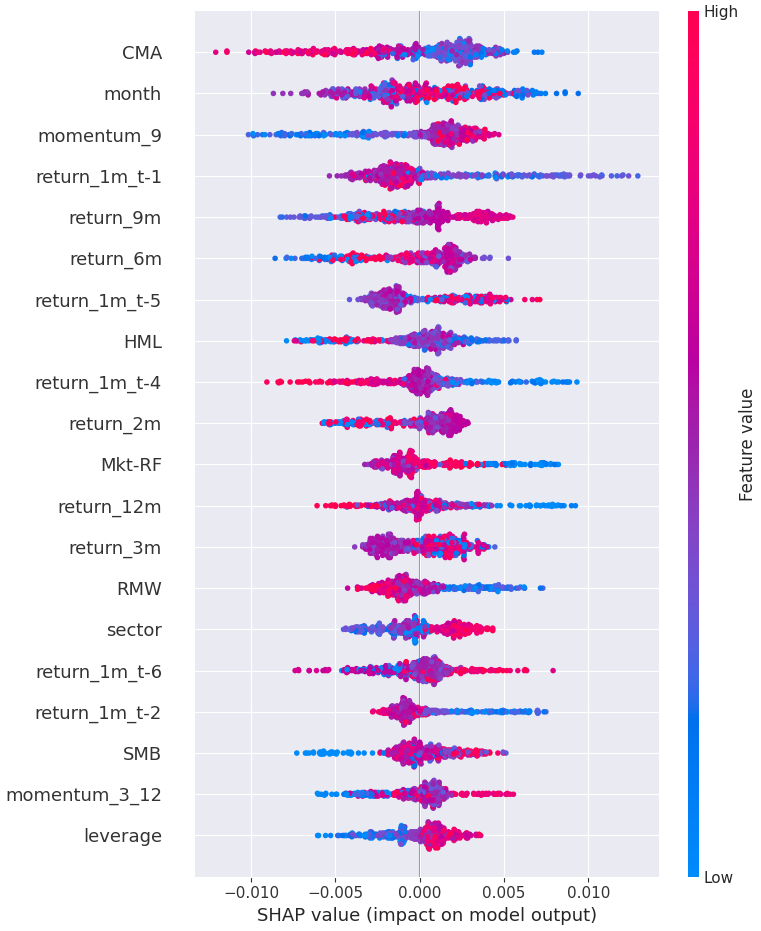


Figure 4.7: The present Figure depicts the Shapley value distribution of the Top 20 contributing factors (according to the Shapley analysis) on a 600 sample of the training set. Additionally, a heat map can be appreciated, which shows the actual value of the factor. The model in question is the LightGBM Boosting Model and the target variable is the excess return.

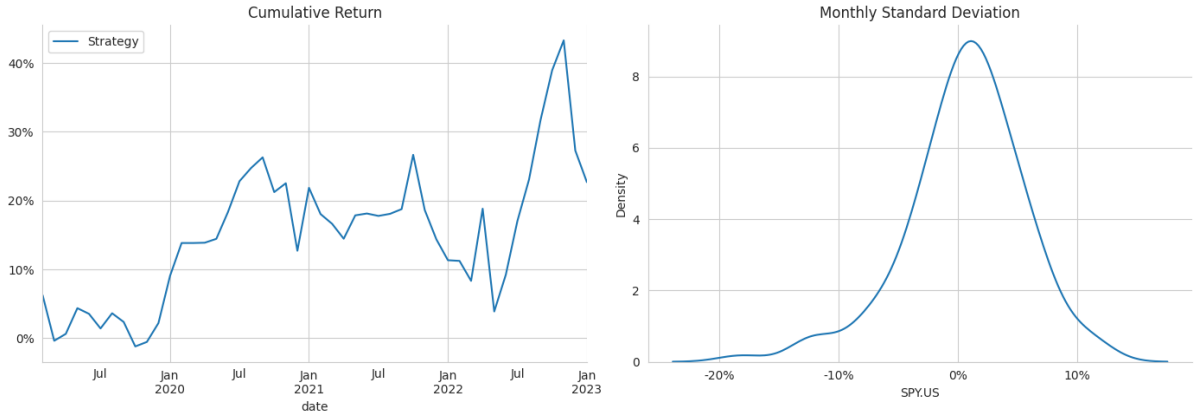


Figure 4.8: The left Figure shows the cumulative returns during the validation period; the right Figure depicts the distribution of the monthly standard deviation of the returns of the implemented strategy. The model in question is the LightGBM Boosting Model and the target variable is the excess return.

4.1.2 One month portfolio return

In this section, the results that were obtained by means of defining the one month return of the portfolio as the target variable and using the LGBM Boosting Model are explained. In order not to include too many graphs that could prolong the section excessively, those analysis that share clear similarities with those of the previous subsection will be commented on, but the corresponding graphs may be omitted.

Analogously to Section 4.1.1, the same hyperparameter values were put to the test. The next step is, again, to compute the dependency between the values of the tested hyperparameters and the IC. However, the same problem observed in Section 4.1.1 occurs here. It is not possible to extract reliable conclusions.

Moving on, the distribution of the overall IC by train length, as well as by train length and boosting rounds, was obtained. The results are very similar and so are the conclusions. In Table 4.2, the best set of hyperparameters is presented. In comparison with Table 4.1, the optimal learning rate in this case is 0.01 instead of 0.3, the feature fraction is 0.95 instead of 0.3, the number of leaves is 32 instead of 8 and the boosting rounds are 100 instead of 50. The mean IC obtained by the model is of 12.05%.

Train length	54
Test length	1
Learning rate	0.01
Number of leaves	32
Feature fraction	0.95
Minimum data in leaf	45
Boosting rounds	100

Table 4.2: Best hyperparameter set of values for the LGBM Boosting Model with portfolio returns as the target variable.

With regards to the graph that shows the 3 months rolling average IC throughout the walk-forward cross validation process, the results are very similar to those of Section 4.1.1.

The next step is to show the feature importance of the resulting best performing model. In Figure 4.9, the Top 10 most importance features, both in Split and Gain, can be observed. Firstly, it is very remarkable that the majority of the factors are those macroeconomic variables that were incorporated into the dataset in Section 3.2. Particularly, the most important variable both in Gain and Split is financial_conditions_diff. Apart from that, the only two variables that are connected to the original data are the 3 month return and the month variable. The most important variables that appeared in the previous section, which are the lagged returns, plain returns, momentum and Fama-French factors, do not appear at all.

Moving on, the results of the Shapley analysis are presented in Figure 4.10. Again, the most important variable is financial_conditions_diff. High values of that variable affect the output negatively and vice versa. Secondly, the differentiated unemployment rate is the 2nd most important variable. High values of it affect the output positively while low values slightly affect it negatively. Then, the third variable is the 9 months momentum, whose positive values affect the output positively and vice versa. The following is the month variable, even though there is no clear correlation between the feature values and the Shapley value. The other variables have a decreasing effect on the output. Again, most of them are macroeconomic variables, although some momentum, lagged returns and plain returns variables are on the bottom of the list.

Finally, in Figure 4.11 the results of the backtest are presented. Since the target variable is the return of the portfolio, a comparison can be established between the cumulative returns of the benchmark and those of the strategy. The obtained model does not obtain a higher accumulated return than the benchmark. In conclusion, having defined the target value as

the returns of the strategy leads to unsuccessful results, in contrast to those obtained in the previous subsection.

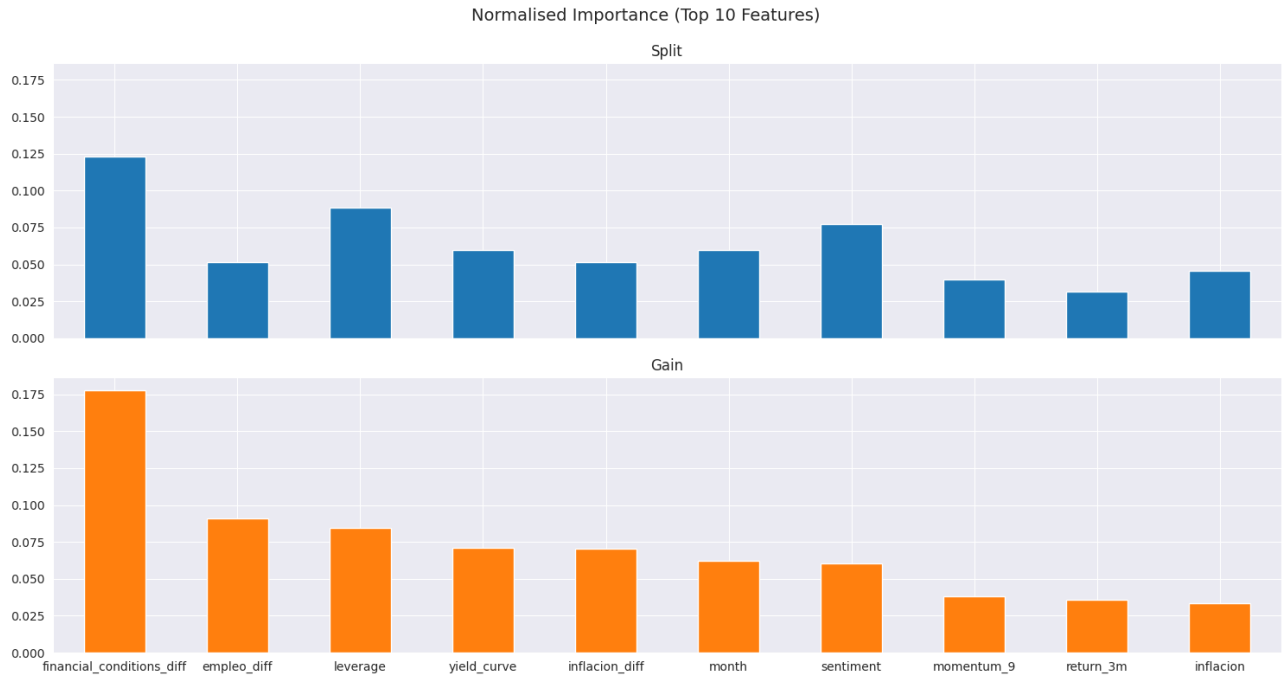


Figure 4.9: The graph above depicts the normalised feature importance of the best performing model, both in Gain and Split. The model in question is the LightGBM Boosting Model and the target variable is the portfolio's return.

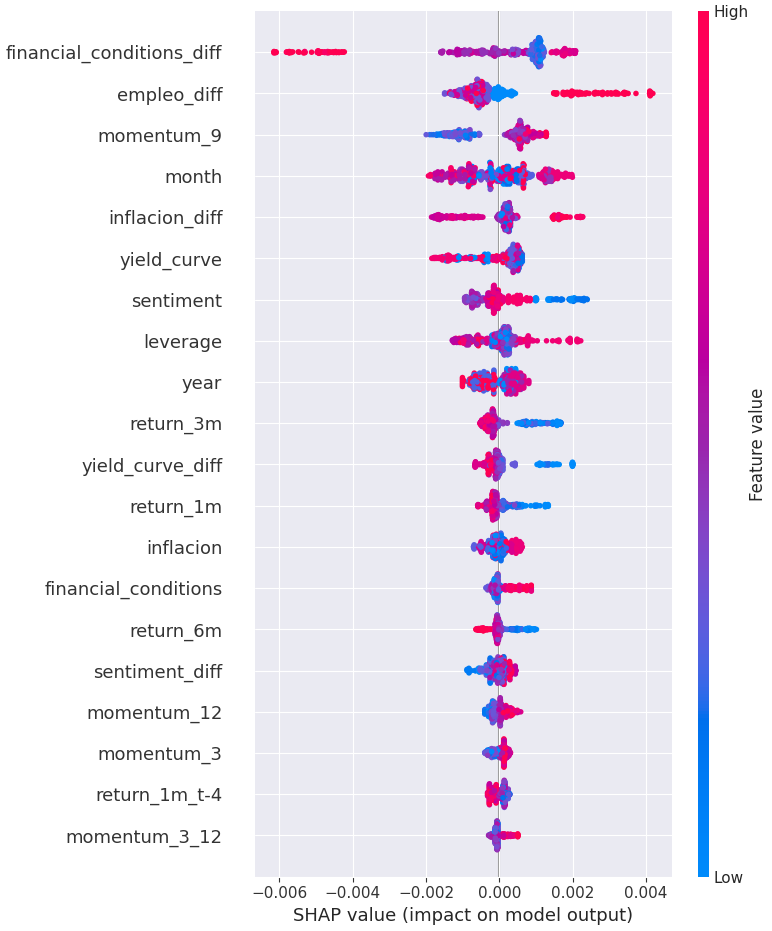


Figure 4.10: The present graph shows the Shapley analysis distribution of the Top 20 contributing factors on a 600 sample of the training set. Additionally, a heat map can be appreciated, which reflects the actual value of the factor. The model in question is the LightGBM Boosting Model and the target variable is the portfolio's return.

Figure 4.11: The left Figure shows the cumulative returns of the strategy and those of the benchmark during the out-of-sample test period; the right Figure depicts the distribution of the monthly standard deviation of the returns of the implemented strategy. The model in question is the LightGBM Boosting Model and the target variable is the portfolio's return.

4.2 Random Forest

4.2.1 One month excess return

Firstly, the hyperparameter set of values that has been tested is presented:

- **Bagging Fraction**: 0.50, 0.75, 0.95
- **Feature Fraction**: 0.75, 0.95
- **Minimum Data in Leaf**: 11, 22, 45
- **Boosting Rounds**: 50, 100

The following step that has been taken has been to compare the distribution of the overall IC and the monthly IC achieved by the two models during the hyperparameters testing. In this regard, in Figure 4.12 the resulting distributions can be compared. The LightGBM Boosting Model distributions are very similar to one another. The Random Forest ones are slightly narrower, even though lots of outliers appear in the Monthly IC distribution.

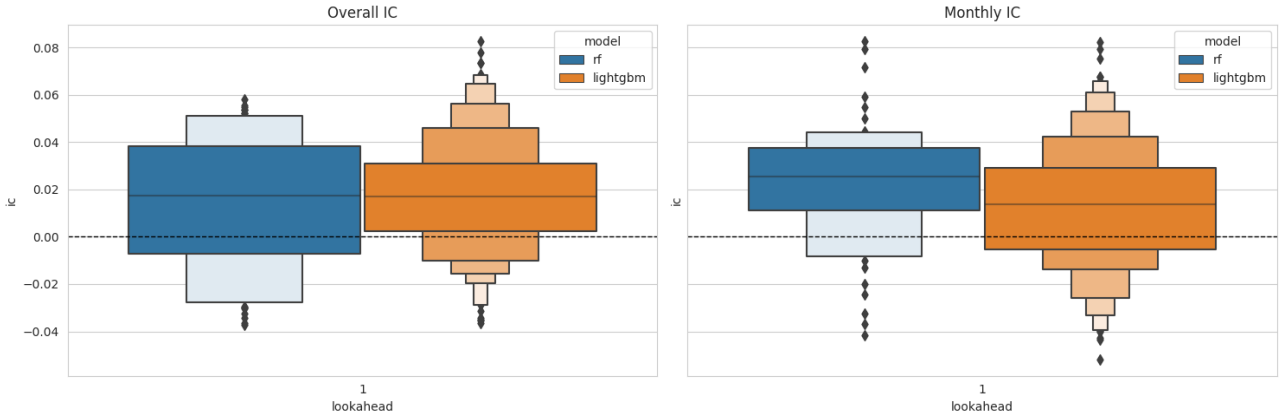


Figure 4.12: These graphs show the distribution of the IC, both in the overall and monthly versions, are presented. The distributions of the IC achieved by the Random Forest models are much narrower than those of the LGBM's. The model in question is the Random Forest Model and the target variable is the excess return.

Again, the next step is to illustrate the dependence between the tested hyperparameters and the IC. However, the same problems that arose in Section 4.1.1 are here too. It is not possible to observe any trend between the IC and the hyperparameters.

Now, the distribution of the IC coefficient by train length is presented in Figure 4.13. In comparison with Figure 4.3, the IC values are generally lower. However, it is clearer that a 36 month training length period achieves higher values, which could not be observed in Figure 4.3.

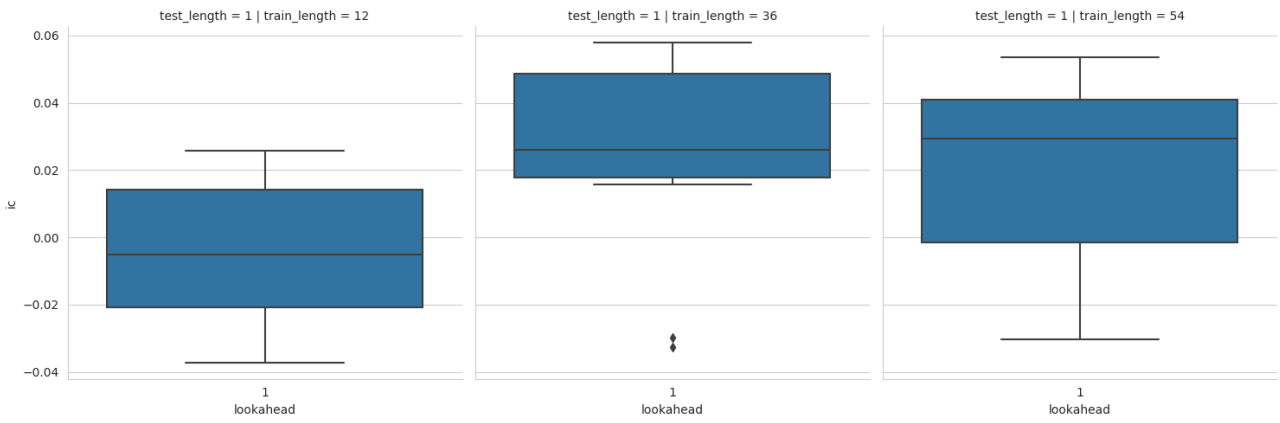


Figure 4.13: Distribution of the overall IC by train length. The model in question is the Random Forest Model and the target variable is the excess return.

Next, in Figure 4.14, the distribution by training length and boost rounds is presented. In general terms, it is not possible to find an optimal value of train length and boost rounds that yields the highest IC. Finally, in Table 4.3, the best performing set of hyperparameters (in terms of IC) is presented. The resulting model has an IC value of 5.80%.

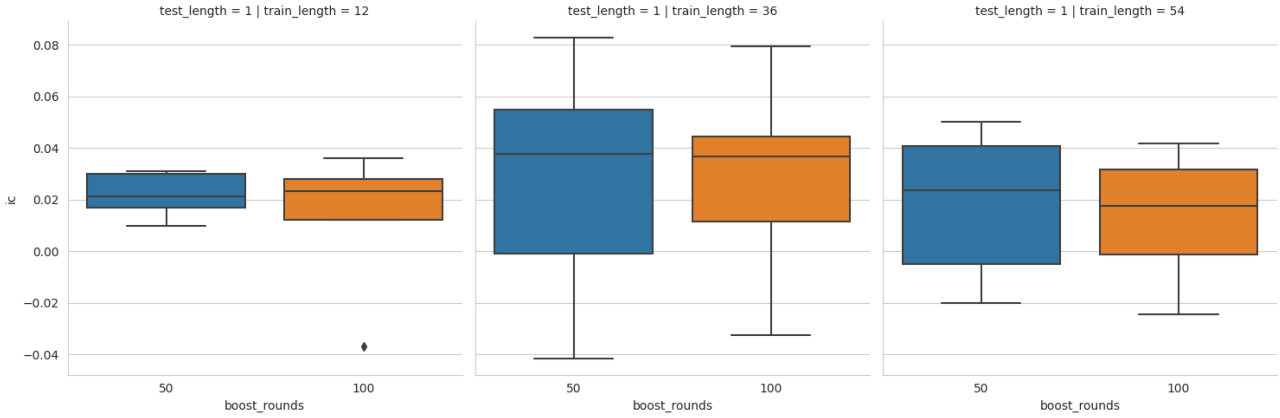


Figure 4.14: The figure above shows the distribution of the overall IC by number of boost rounds and train length. The model in question is the Random Forest Model and the target variable is the excess return.

Train length	36
Test length	1
Bagging fraction	0.5
Feature fraction	0.95
Minimum data in leaf	22
Boosting rounds	50

Table 4.3: Best hyperparameter set of values for the Random Forest where the target variable is the excess return.

At this point, in analogy with what it was presented in Section 4.1.1, the rolling IC graphs that were shown in Figure 4.5 should also be shown for the Random Forest Model. Nonetheless, the obtained results are very similar to those observed in the aforementioned Figure, which entails that the conclusions that were extracted in Section 4.1.1 are valid in the present section too. For instance, the graph will not be included for the sake of simplicity.

From now on, the best performing model will be analysed. Firstly, the feature importance will be discussed, both in the Gain and Split methods. The results can be observed in Figure 4.15. They differ greatly from those observed in Figure 4.6: one single factor, which again is

the CMA factor, accounts for more than 25% of the importance in Gain. The results are quite similar in both methods; however, the CMA has a lower weight in Split. The other variables have a weight of approximately 5% or lower.

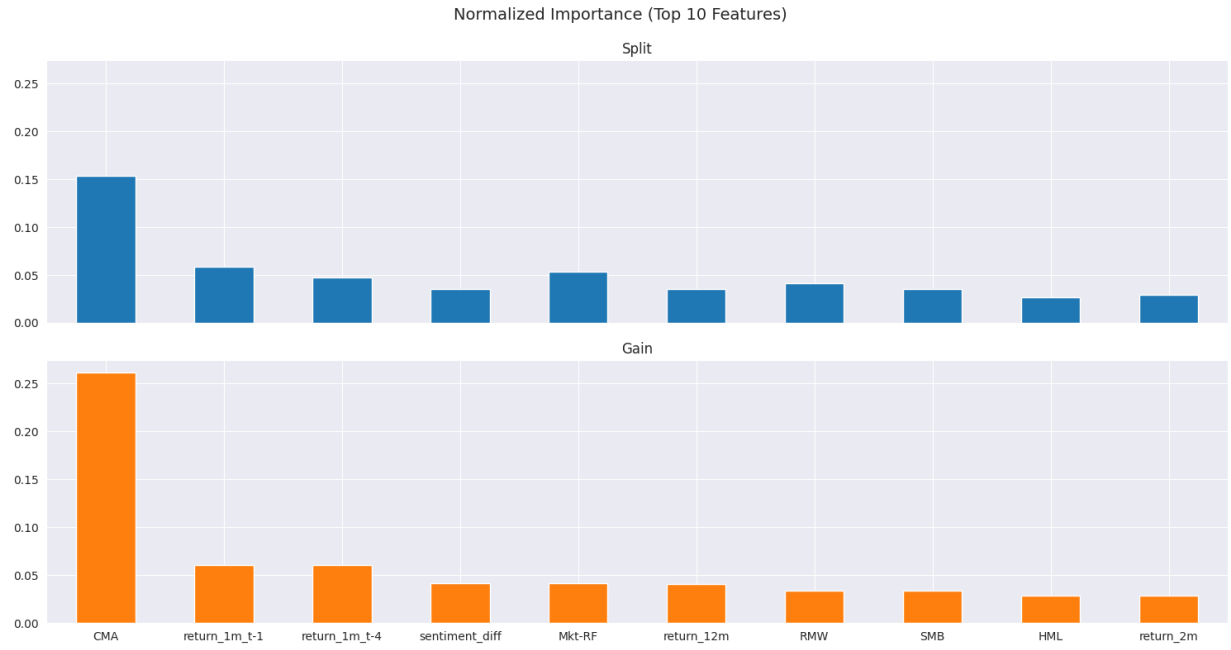


Figure 4.15: The figure above shows the normalised feature importance of the best performing model, both in Gain and Split. The model in question is the Random Forest Model and the target variable is the excess return.

The next step is to describe the results from the Shapley analysis (see Figure 4.16). Although the analysis is different, the resulting conclusions are very similar: there is only one variable that is able to influence the model's prediction, which is the CMA Fama-French factor. High values of that variable affect the output negatively while low values affect it positively. The other variables have very low impact overall.

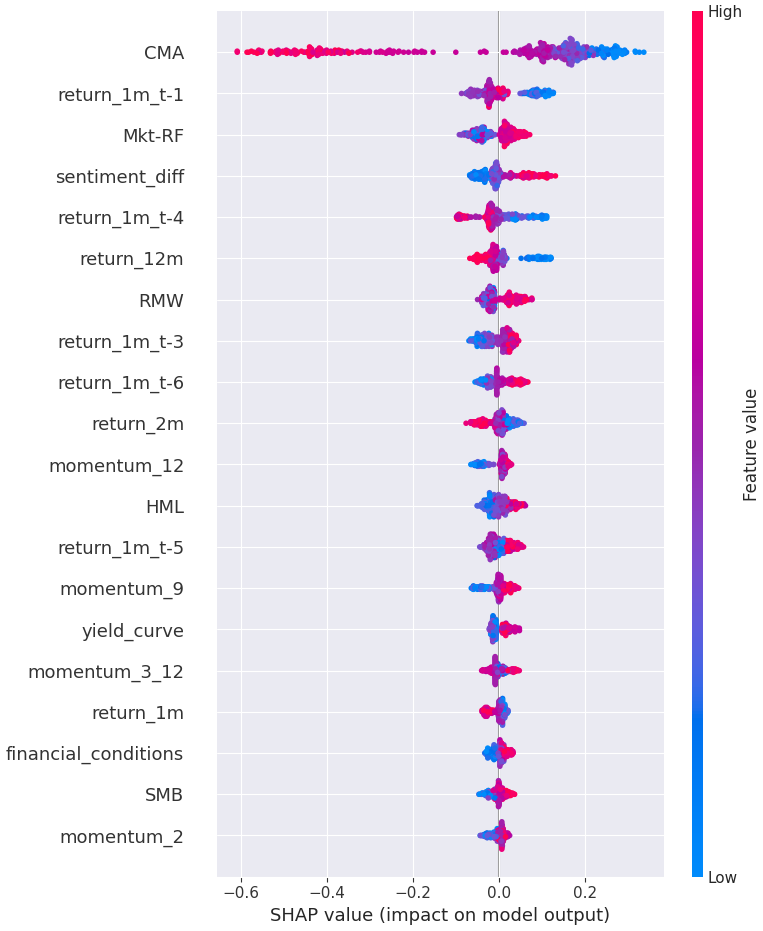


Figure 4.16: The present picture shows the Shapley value distribution of the Top 20 contributing factors on a 600 sample of the training set. Additionally, a heat map can be appreciated, which reflects the actual value of the factor. The model in question is the Random Forest Model and the target variable is the excess return.

Finally, it is left to present the results of the backtest. They are included in Figure 4.17. Even though the results are not as outstandingly positive as those of Section 4.1.1, a 5% excess return is obtained, which is also a very remarkable result.

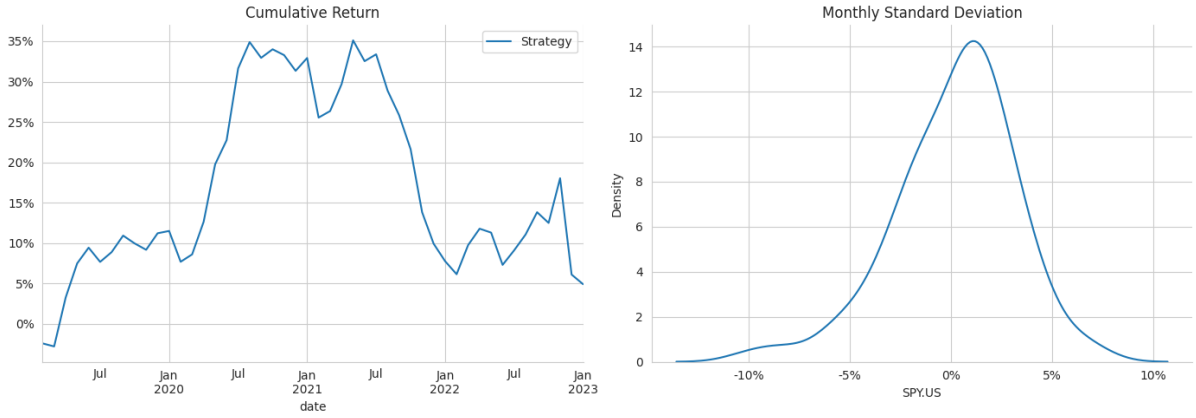


Figure 4.17: The present figure shows the cumulative returns during the out-of-sample test period; the right figure shows the distribution of the monthly standard deviation of the returns of the implemented strategy. The model in question is the Random Forest Model and the target variable is the excess return.

4.2.2 One month portfolio return

In this section, the results obtained after defining again the one month return of the portfolio as the target variable and using the Random Forest model will be explained. As it was mentioned in previous sections, those results that have appeared already will be commented on, but the corresponding graphs will be omitted.

The same set of hyperparameters used in Section 4.1.2 were trained in the present case. The next step, again, was to find any new dependencies between the IC and the hyperparameters. However, it was not possible to draw conclusions.

The succeeding step has been to establish the same comparison between the overall IC and the monthly IC distributions achieved by the two models, that is, those trained in Section 4.1.2 and the ones trained in this section. The results can be observed in Figure 4.18. In that case, the monthly IC distributions are much narrower than the overall's and achieve higher values.

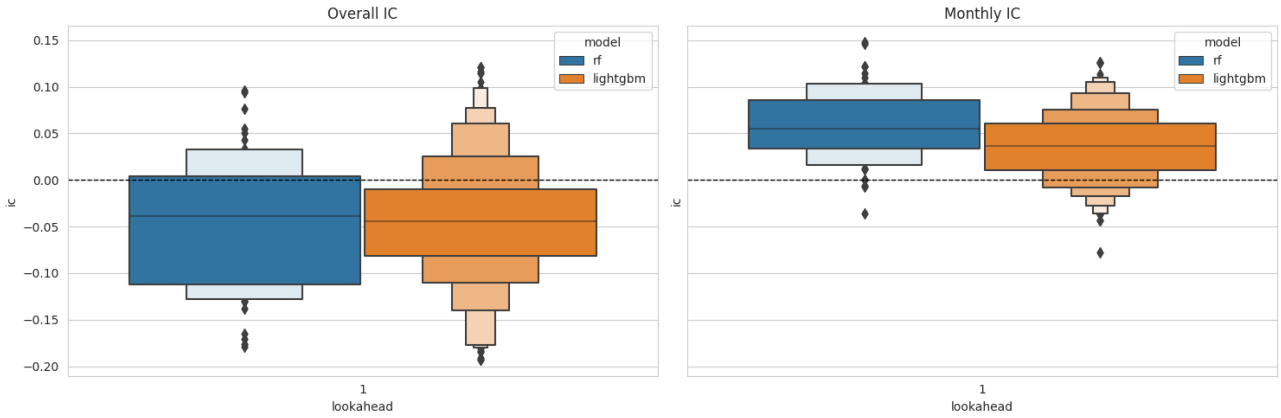


Figure 4.18: These graphs show the distribution of the IC, both in the overall and monthly versions, are presented. The distributions of the IC achieved by the Random Forest models are much narrower than those the LGBM's. The model in question is the Random Forest Model and the target variable is the excess return

Moving on, the distribution of the IC by train length was graphed. The results show some difference compared to the tendencies that were observed so far. These differences lie in the fact that the distributions are much narrower and that the 54 months training length achieves the greatest median IC value.

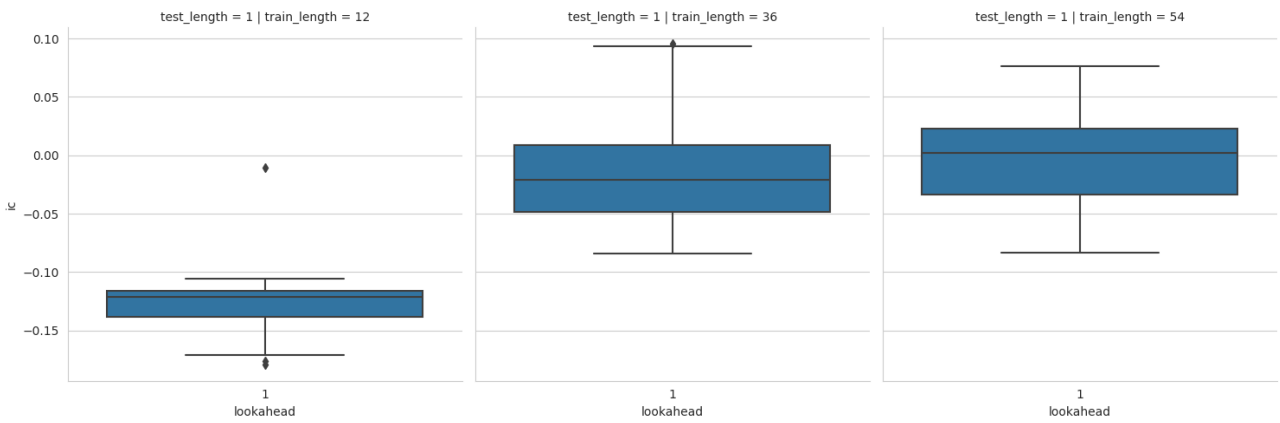


Figure 4.19: Distribution of the overall IC by train length. The model in question is the Random Forest Model and the target variable is the portfolio's return.

Moreover, the distribution of the IC by train length and boost rounds was also graphed in Figure 4.20. Again, there is not a clear correlation between the number of boosting rounds and the IC. However, in this case it does seem that, in general, a 54 months training length achieves slightly higher values.

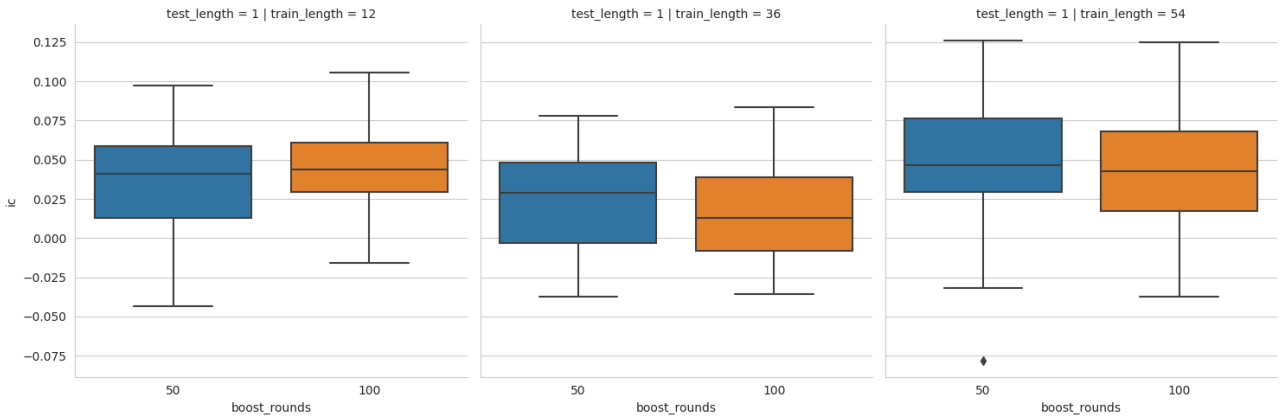


Figure 4.20: This figure shows the distribution of the overall IC by number of boosting rounds and train length. The model in question is the Random Forest Model and the target variable is the portfolio's return.

Subsequently, the best performing set of hyperparameters is presented in Table 4.2.2. The overall IC takes a value of 9.57%. From now on, the results of the best performing model will be presented and discussed. In Figure 4.21, the feature importance of the model graphs, both in gain and split, are presented. The results resemble those of Section 4.1.1. To start

with, the most important variable is financial_conditions_diff again, followed by leverage and the differentiated unemployment. Again, the first variable in Gain accounts for more than a 20% of the variability of the target variable. In addition, it is worth mentioning that the lagged returns, which successfully explain the models of Sections 4.1.1 and 4.2.1, do not appear in the Top 10 once again in exception of the 3 months return. Rather than that, the most important variables are again the macroeconomic factors. Finally, in Split, all variables have very low weights, which suggests that the variables of the model are not able to explain the target variable.

Train length	36
Test length	1
Bagging fraction	0.95
Feature fraction	0.95
Minimum data in leaf	11
Boosting rounds	100

Table 4.4: This table shows the best performing set of hyperparameters for the Random Forest predicting the monthly results of the portfolio.

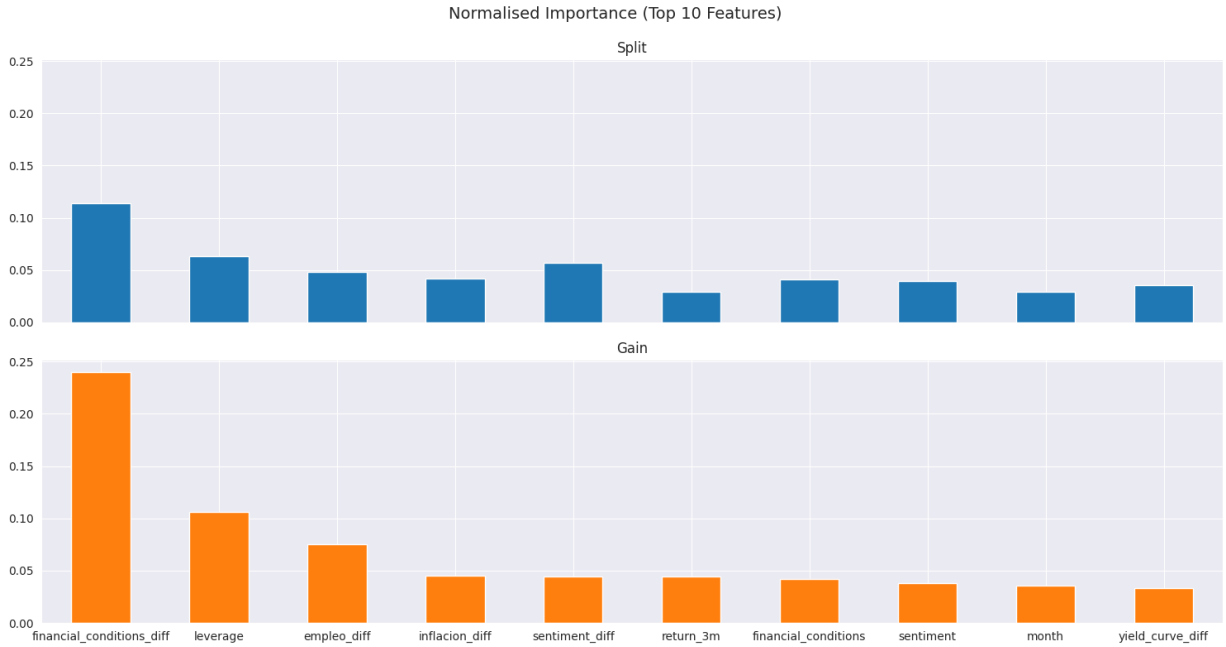


Figure 4.21: The present graph shows the normalised feature importance of the best performing model, both in Gain and Split. The model in question is the Random Forest Model and the target variable is the portfolio's return.

Next, in Figure 4.22, the Shapley analysis is included. Again, it has to be remarked that the return variables play a very limited role, which did not happen in Sections 4.1.1 and 4.2.1. Also, the Top 2 variables contribute significantly to the model's output. Nonetheless, the other 18 variables' contribution is much less significant. The most important variable is financial_conditions_diff again. High values of that variable contribute negatively to the output and vice versa. Apart from that, it is worth mentioning that the differentiated unemployment is the 2nd most important variable, whose high values contribute positively to the output. Low values have a mild negative effect.

Finally, the results of the backtest are included in Figure 4.23. Over the out-of-sample test period, the strategy obtains a net return of approximately -20%, which of course is less than what the benchmark achieves. In conclusion, neither the LGBM Boosting Model nor the Random Forest Model are able to outperform the benchmark by defining the monthly returns as the target variable.

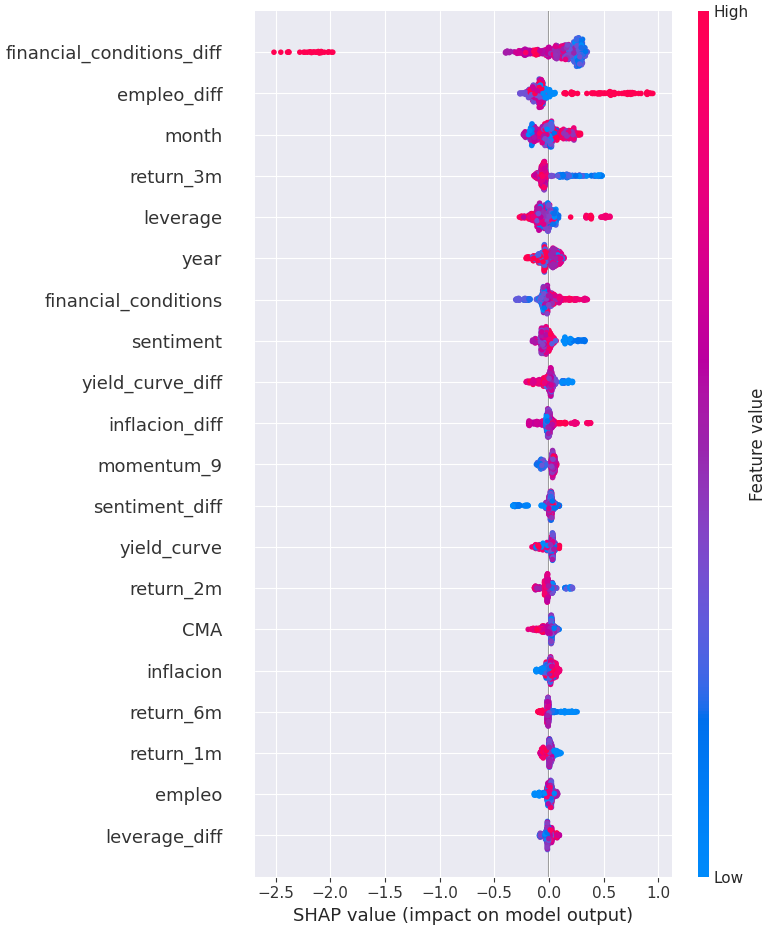


Figure 4.22: The present picture shows the Shapley value distribution of the Top 20 contributing factors on a 600 sample of the training set. Additionally, a heat map can be appreciated, which reflects the actual value of the factor. The model in question is the Random Forest Model and the target variable is the portfolio's return.

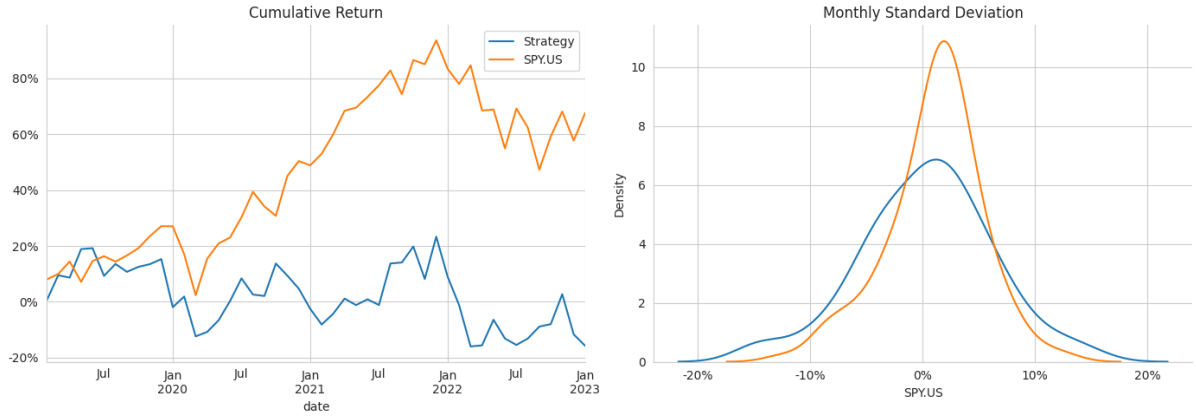


Figure 4.23: The left figure shows the cumulative returns of the strategy and those of the benchmark during the out-of-sample test period; the right Figure depicts the distribution of the monthly standard deviation of the returns of the implemented strategy. The model in question is the Random Forest Model and the target variable is the portfolio's return.

4.3 Conclusions and future investigations

As it was commented in Section 1.5, the aim of this work was to train Machine Learning models so as to build a portfolio that was able to consistently outperform the benchmark, which was defined as the Standard & Poors stock index or the SPY ETF. Additionally, the obtained models would be compared with the results of other risk-oriented portfolio management strategies. Due to time constraints, it has not been possible to complete all these tasks. More concretely, two different models have been trained and validated successfully: the LightGBM Boosting Model and the Random Forest model. The one that has not finally been taken into consideration is the Neural Network model. Moreover, comparing the obtained results with risk-oriented portfolio management strategies has not been possible either. Nonetheless, the two models have been trained considering two different target variables, the first of them being the monthly returns; the second one, the monthly returns minus the benchmark's monthly returns. Hence, four sets of results have been presented.

The performance of the models strongly depend on the defined target variable: those trained to predict the excess return with respect to the benchmark have successfully outperformed the benchmark in the backtest. Conversely, those models trained to predict the future one month returns have underperformed the benchmark, even having overall negative returns in the case of the Random Forest. With regards to the two models that have outperformed the benchmark, the LightGBM Boosting Model has obtained better results than those achieved by the Random

Forest model.

Concerning the feature importance, three different analysis have been done: the Gain method, the Split method and the Shapley analysis. The results strongly depend on the target variable definition. In the case of the excess return, the most contributing variables have generally been the lagged returns, that is to say, the returns with respect to the previous month obtained N months prior to the current date. Moreover, some of the Fama-French factors are of great importance too: HML (High book-to-market ratio Minus Low book-to-market Ratio), Mkt-RF (Market Risk Factor) and CMA (Conservative Minus Aggressive, which is by far the most important variable in the Random Forest model). In addition to that, the variables that account for the returns accumulated during the last N months, where $N = 1, 2, 3, 6, 9, 12$, also make significant contributions. Even though the feature importance ranking varies depending on the used method, the most important variables are the same. The macroeconomic variables that were also added to the initial dataset (e.g. unemployment rate, interest rate curve, and so on) have not been significantly relevant.

On the other hand, in the case in which the estimated variable has been the one month returns, the most important set of variables changes significantly: they are the macroeconomic variables that were not significant in the previous case. Concretely, the first one is the differentiated financial conditions, followed by the unemployment rate and the interest rate curve, to name a few. However, other variables that appeared before do appear again, namely the plain returns and the three Fama-French factors. In addition to that, it has to be mentioned that the relative weight of each variable rapidly decays, which entails that very few variables (compared to the total of them the model has) make the most substantial contributions in order to predict the target variable. The conclusion to what has been discussed is that very few variables are able to predict the monthly returns of the strategy.

In conclusion, by defining the target variable as the excess return it has been possible to confirm the formulated hypothesis. Moreover, the results are very promising. That makes one wonder their validity: consistently outperforming the defined benchmark, which is the Standard & Poors 500, is a problem that has been addressed countless times in quantitative finance. Plus, a permanent solution to it has not been found yet. That does not mean that the present work has not been done correctly or is not valid; rather than that, it should be seen as a research opportunity. Future investigations should try to validate these results. To start with, this work has been done with monthly data due to the computational cost that working with daily data would have had. Hence, it should be proved whether or not these results are so promising in

that case. In addition to that, this experiment could be repeated considering a different time lapse, both with monthly and daily data. Having that said, the obtained results are satisfactory.

Finally, on a personal level, this work has been very rewarding to me for two particular reasons. The first one is because I have always been passionate about quantitative finance and data science since they are disciplines that can be applied in the technological and financial industry. Secondly, because the conclusions that I have reached differ completely from the ones I expected to obtain. I hope that someone retakes this investigation in the future in order to confirm these conclusions.

Chapter 5

Glossary

- **Exchange Trading Fund**: an Exchange Trading Fund is a pooled investment security that holds a number of underlying assets, similarly to an investment fund. They are traded in the stock exchange.
- **Long position**: a long position refers to the purchase of an asset expecting that its price will increase in the future, which is also called a bullish attitude.
- **Short position**: a short position consists of selling an asset with the expectation that its price will decrease in the future, which is also called a bearish attitude.
- **Stock Index**: a stock index market or a stock index is a statistical metric which measures market fluctuations. Usually, it is built by composing a portfolio of stocks listed on the exchange.
- **Standard & Poors 500**: the Standard & Poors 500 is a stock index that tracks the stock performance of the 500 largest US companies listed on US stock exchanges. It is one of the most used stock indices.
- **Risk-free rate**: it is defined as the rate of return on an investment that has no chance of default. Usually, it is associated with the rate of return of a country's treasury bills, the country usually being the US.
- **Financial risk**: financial risk can be defined as the probability of the occurrence of an event that has negative financial consequences for an enterprise or an investor that possesses a financial asset.

Bibliography

[1] Jansen, S. (2020). *Machine Learning for Algorithmic Trading: Predictive models to extract signals from market and Alternative Data for systematic trading strategies with Python.* Packt Publishing.

[2] de Prado, M. (2018). Advances in Financial Machine Learning. (1st edition). Wiley.

[3] Sharpe, W. F. (1964). Capital asset prices: A theory of market equilibrium under conditions of risk. *The journal of finance, 19(3)*, 425-442.

[4] Black, F. & Scholes, M. (1973). The pricing of options and corporate liabilities. *Journal of political economy*, 81, 637.

[5] Fama, E. F.; French, K. R. (1993). Common risk factors in the returns on stocks and bonds. *Journal of Financial Economics*. 33: 3–56.

[6] Karatas, & T., Hirsa, A. (2021). Two-stage sector rotation methodology using machine learning and deep learning techniques. *arXiv preprint arXiv:2108.02838.*

[7] Krauss, C., Do, X.A., & Huck, N. (2017). Deep neural networks, gradient-boosted trees, random forests: Statistical arbitrage on the S&P 500. Eur. J. Oper. Res., 259, 689-702.

[8] Basak, S., Kar, S., Saha, S., Khaidem, L., & Dey, S.R. (2019). Predicting the direction of stock market prices using tree-based classifiers. *The North American Journal of Economics and Finance.*

[9] Gopinath, D. (2022, January 4). The Shapley Value for ML Models - Towards Data Science. *Medium.*
`https://towardsdatascience.com/the-shapley-value-for-ml-models-f1100bff78d1`

[10] Microsoft (2023, January). *GitHub - microsoft/LightGBM: A fast, distributed, high performance gradient boosting (GBT, GBDT, GBRT, GBM or MART) framework based on decision tree algorithms, used for ranking, classification and many other machine learning tasks.* GitHub. `https://github.com/microsoft/LightGBM`