DEPARTMENT OF *BUSINESS AND MANAGEMENT*

Chair of *Asset Pricing*

Characteristic-based portfolio selection using machine learning

Supervisor

Prof. Nicola Borri

Candidate

Emilio Iannone

741981

Co supervisor

Prof. Luigi Laura

Academic Year 2021 – 2022

Contents

INTRODUCTION

Portfolio management is a challenge that has intrigued the world of finance for years, particularly in dealing with perennial issues such as inflation. In this particular period, the latter is reaching unprecedented levels, which is why the search for new wealth management techniques has increased. One of the most interesting frontiers is certainly the use of Machine Learning in finance. It refers to a set of processes that lead to a computer learning how to improve any given performance based on the data it receives. This process can be developed in a supervised or unsupervised manner. In this paper, I will only deal with supervised machine learning. Each machine learning mechanism is characterized by 2 macro phases, a training phase in which the machine understands how to optimize the algorithm and an inference (or testing) phase in which the machine applies what it has learned using the optimized algorithm. In this case study, I will use 75% of the total periods available for training and the remaining 25% for testing the efficiency of the algorithm. The total period used goes from the last quarter of 2013 to the last of 2021. This is due Specifically in this paper I will consider 35 quarters of 19 blue chip companies and, based on 3 particular market characteristics, it will then maximize the perceived utility of the investor through a specific function of preferences over wealth. Going into more detail, the function used to calculate the weights will consist of 4 coefficients, 3 of which will each multiply one of the characteristics mentioned above.

In the first training phase, the purpose of the algorithm will be to find the optimal value of these coefficients in order to maximize the average perceived utility function over the available periods. The values of the characteristics were downloaded from Bloomberg and put into a single excel sheet for computational simplicity.

The entire machine learning algorithm will be developed using the Python programming language. I chose the latter for its intuitiveness both in the input phase and especially in the output phase. All results and graphs in this paper will therefore be generated by the code written in Python, which can be found in the appendix. The objective of the paper is to find optimal coefficients for each characteristic to create an algorithm that manages to choose the optimal percentage of capital to invest in each of the 19 selected companies, thus creating a competitive portfolio that allows significant returns to the investor.

Certainly, this paper does not presume to present an already complete and applicable result. Constraints on the periods that can be selected and the available computing power are certain aspects that do not allow the algorithm to demonstrate its full potential. At the same time, I would like mine to be a starting point, providing a solid basis for possible implementations in such a way as to make it even more performant and easily applicable. I, therefore, preferred not to prioritize profit but stability over the years while trying to keep the algorithm as simple as possible.

1. CHAPTER 1: DATA

1.1 Stock's characteristics

Before going into the details of the features, it is important to dwell on the companies selected for this algorithm. Indeed, I have tried to diversify the sectors in which they operate in order to reduce risk. Another key criterion by which the actions were chosen was the fact that the period considered begins in 2013. In fact, many of the companies that are now considered blue chips at that time were not even born or at least did not have the stability they have now. After the description of each characteristic, I have added a table showing the average results downloaded from Bloomberg for each stock.

The entire selection criterion for my portfolio is based on the financial characteristics of the companies in question, the values of which will influence our purchase choices. In particular, as previously specified, three characteristics will be considered. All three are related to the balance sheet and therefore strongly constrained the timeframe in which I operated. I will in fact reason by quarters as company balance sheets are drawn up every quarter. The values of all three characteristics are standardized cross-sectionally for each time (mean = 0, standard deviation = 1) so that the values are uniform and can be compared with each other when the algorithm calculates the weights. Now I'm going to explain more in detail each characteristic:

1.1.1 Market Cap

Market cap stands for market capitalization, it is the total dollar value of the outstanding shares of a listed company. It is a metric used to relatively tell the size of a company, which can be useful when creating investment plans and strategies. The market cap of a company is obtained by multiplying the total number of shares outstanding by the current market price of a share

Companies are typically grouped into one of the following categories based on their market capital[1]:

[1] https://www.degiro.it/trading-conoscenza/inverstitori-accademia/principianti-corso/market-cap

- Large-cap: Large-cap stocks have a market capitalization of $10 billion or more. Some further specify companies with a market cap of $200 billion or more as mega-cap. Large-capitalization companies are typically established, leaders in their fields. Large-caps generally have the financial resources to better withstand economic downturns and are less volatile, so they tend to be considered less risky than mid-caps and small-caps. They are also more likely to pay dividends to shareholders. On the other hand, some companies have already experienced their peak growth period and, therefore, may see lower short-term returns than mid- and small-caps. All companies considered in this paper belong to this category.
- Mid-cap: Mid-cap companies are those with a market cap between USD 2 billion and USD 10 billion. Companies in this category are generally experiencing or likely to experience growth. They can be considered to have more growth potential than large caps and less risk than small caps.
- Small-cap: Small-cap companies have a market cap between $300 million and $2 billion. Some further define companies in this category as micro-cap (market cap between $50 million and $300 million) and nano-cap (market cap under $50 million). Small-cap stocks tend to have significant growth potential at the expense of higher risk. This is because these companies are generally younger and their business models have not yet stood the test of time.

Knowing the market capitalization of a company can help make an investment strategy and investment decisions. This indicator permits the algorithm to favor companies with a larger presence on the market, and therefore less susceptible to sudden changes in market trends, as they are more stable.

Market capitalization can also help investors to diversify their portfolios. Having a variety of companies' portfolio with different market capitalizations can help spread risk. But the market cap is only one aspect to consider when trying to achieve a diversified portfolio. For example, investing in companies from different sectors or countries, or investing in other financial products such as ETFs and bonds instead of investing exclusively in stocks. In my case, I preferred to build as stable a portfolio as possible, taking only large capitalization companies and investing only in stocks. At the same time, I decided to diversify the sectors and nationalities in which the selected companies operate, to provide further overall stability to my portfolio.

It is important to note that although the market cap can provide a general idea of the level of risk and size of a company, it is not the only indicator. For example, shares may be over- or under-valued by the market, which means that the price investors are willing to pay may be more or less than the real value of the company. For this reason, this characteristic alone is not sufficient as a criterion for stock selection and I decided to use two others more related to the intrinsic values of the company.

Figure 1

1.1.2 Enterprise Value to Book

EV means, in the case of non-debt companies, market capitalization - net liquidity. The value of the company calculated in this way represents the price that someone who wants to acquire the company without debt would have to pay. The value of the company (EV) must be equal to the market value of debt and equity capital. This, therefore, implies that the total value generated by the operating and investment activities of a company has to be divided between the various equity holders (both debt and equity holders). Book value, or book value of equity, defines the net worth of a company or asset according to its financial status. For companies, it consists of the total value of tangible assets minus liabilities. For assets, on the other hand, it is the result of the difference between the price paid and the depreciation, i.e. the depreciation related to market conditions. While the book value reflects the value of a company based on its financial situation (the book), the market value is the price of a company

on the financial markets, i.e. the market cap introduced earlier. Book value is used by traders and investors to compare different companies in order to find undervalued or overvalued shares. The price/book value ratio (or P/B) compares the current market value with the book value. In this paper, I have instead opted to use a value less related to the market but more to the financial performance of the company, such as the Enterprise Value. This is coherent with the nature of the portfolio that I am going to build, in fact, focusing on companies' financial performances makes the investment less affected by possible market irrational behaviors.

Book value shows the value of an asset or company based on hard data and not on opinion or speculation. This is why it is considered a relatively accurate measure of value. It can be useful for learning more about a business or for finding shares at a favorable price.

Taken individually, book value is not a definitive measure of value. This is because it is inefficient in valuing intangible assets, such as intellectual property rights may be. For example, companies that develop software may create products at a relatively low cost and the balance sheet may not reflect the actual value of the assets. In this case, the company's shares may trade at a price much higher than their book value, but would not be overvalued. Therefore, book value should be used as a comparative measure to compare assets and companies with each other. Above all, it is an indicator that becomes efficient when combined with a second indicator as explained above, which can be the share price or, as in this case, the enterprise value.

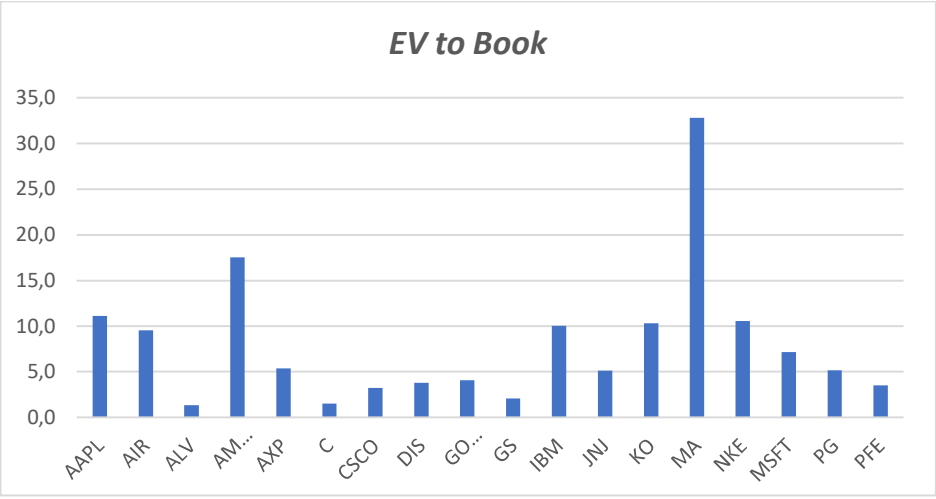

Figure 2

1.1.3 Book Value per Share

It is the ratio of the balance sheet value of equity divided by the number of outstanding ordinary shares. The meaning and strengths of book value have already been explained above. Therefore, it is quite intuitive to understand the usefulness of this indicator. By dividing the book value by the number of shares, one obtains a value that is totally related to the financial characteristics of the company and not to the market in which it is listed. This gives an intrinsic value that can be compared with the price at which shares are bought and sold on the market. With this indicator, we simultaneously understand whether a company is over- or undervalued and what the true value of its shares should be.

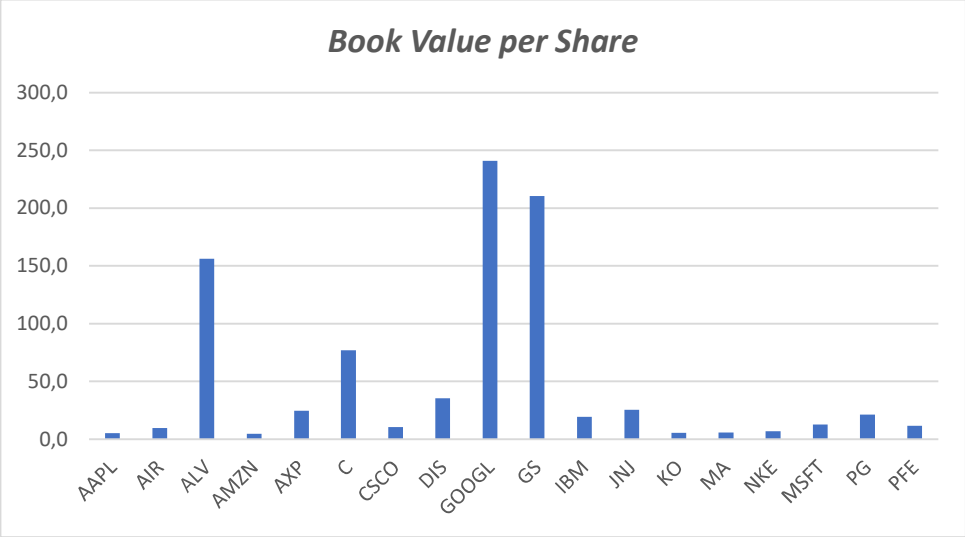

Figure 3

From the graphs can be more clear why I decided to standardize the characteristics cross-sectionally. In fact, since the companies are very different in nature and size, comparison using the characteristics as simple quantities would not have been possible.

1.2 Portfolio weights

The weights play a key role as they determine the percentage of the total capital that is invested in each share for each period. To be able to give them this meaning, however, I had to normalize them once I calculated them using a linear function of the characteristics so that I have homogeneous values and at the same time can be sure not to sell short any stock (short selling constraint). I decided to place a short-selling constraint to make the portfolio more easily realizable in reality. With this constraint, in fact, it avoids problems related to the debt required for short selling of shares and it greatly simplifies the algorithm by allowing it to be calculable even with machines with lower computing power. Although this constraint will probably diminish the yield of the final portfolio, I preferred to place it because I considered it consistent with the philosophy with which I decided to create this portfolio. That is to obtain an instrument that is as simple as possible, easily applicable, and with results that focus more on securing returns than on increasing them. Once the weights have been calculated, I multiply them by the returns of each stock[2] in such a way as to calculate the entire return of the portfolio. Going into more detail, the weights of each action for each period will be calculated using a linear function, consisting of 4 coefficients, 3 of which will multiply each of the 3 characteristics. The fourth is simply added up as it represents the risk-free investments that are not taken into account in the stock selection.

The function in question is as follows:

$$w_{i,t} = e^{\beta_0 + \beta_1 Market\ Cap_{i,t} + \beta_2 EV\ to\ Book_{i,t} + \beta_3 Book\ per\ sha}{}_{i,t}$$

Where:

- $w_{i,t}$ denotes the weight of the stock i at time t
- β_n are the coefficients mentioned before (n=0,1,2,3)

[2] Downloaded from yahoo finance with the functions “from pandas_datareader import data as pdr” and “pdr.get_data_yahoo”

For each period after the computing of the weight of each stock, they are all normalized to obtain a percentage. Using percentages for each weight simplifies the calculations and makes the amount of capital to invest in each share more intuitive once the total capital to be invested in the portfolio has been decided. In addition, as explained above, normalization allows only positive results to be obtained, so that the problem of the short selling constraint can be solved easily and immediately. In the training phase of the algorithm, I will set starting values for each beta which will then be replaced by the results obtained by the optimization algorithm.

1.3 Stock returns

The stock returns used in this paper are percentages of the price changes of the stock itself. Through this indicator, I was also able to calculate the return on a portfolio.

The return can be expressed nominally as the change in the dollar value of an investment over time, or it can also be expressed as a percentage resulting from the ratio of profit to investment. Returns can also be presented as net returns i.e., after fees, taxes, and inflation, or as gross returns that do not take into account anything other than the price change.

Returns on periodic intervals of different durations can only be compared if they have been converted into intervals of the same duration. It is customary to compare returns obtained in one-year intervals. The process of converting shorter or longer return intervals into annual returns is called annualization. In the case study, daily returns are taken into account and then converted into quarterly returns in a way that is consistent with the timeframe of the stock characteristics used.

The returns belong to the profitability indices, which make it possible to understand and compare the earnings associated with different investments.

Profitability ratios perform this comparison by dividing selected or total assets or equity by net income. The result is a percentage return per dollar invested that can be used to assess the soundness of the investment by comparing it to benchmarks such as return ratios of similar investments, companies, sectors, or markets. In this paper, the final return of the portfolio will be compared to that of other standard portfolios that will be used as benchmarks.

Since for the construction and management of this portfolio, it is assumed that no money is taken out during the entire period under consideration, I have decided to consider returns including dividends. This means that the returns used in the algorithm are purposely inclusive of dividends, it is therefore considered that all returns of whatever nature generated by the portfolio are reinvested in the following period, until the investment maturity.

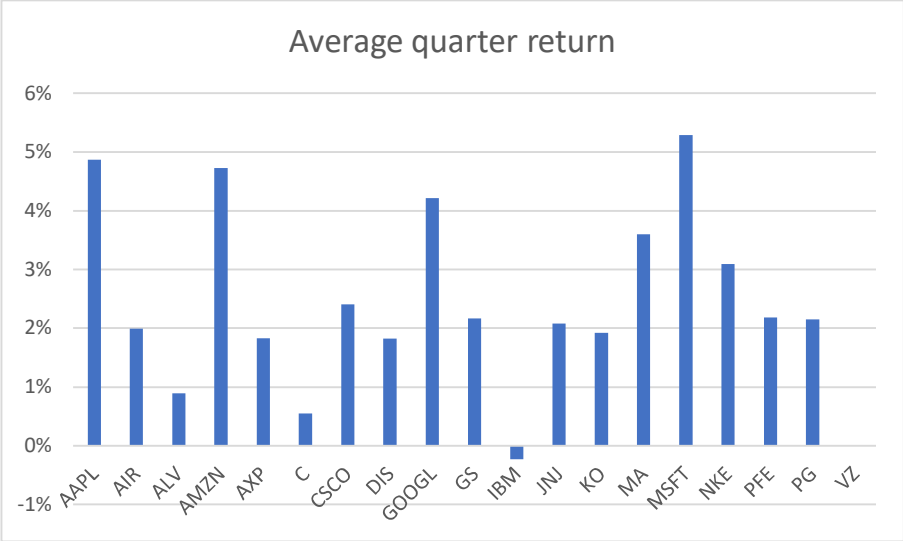

Figure 4

Furthermore, another important calculation is that of market volatility, i.e. its standard deviation. Considering that the investable universe has already been reduced to obtain as stable a section of the market as possible, it is not surprising to see the result shown in the figure below, where the standard deviations of the returns in each analyzed period are represented.

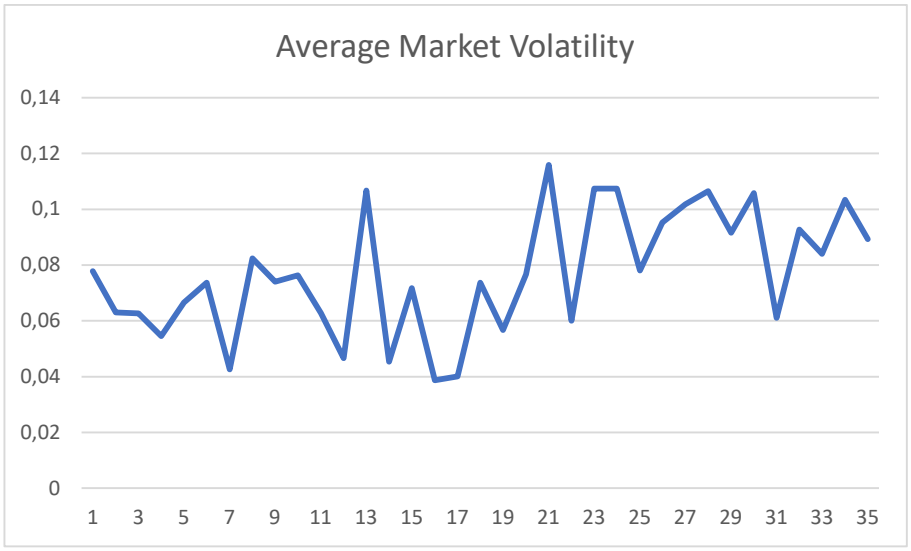

Figure 5

In fact, these are all relatively small values, confirmation of the philosophy with which the shares were chosen. The average standard deviation obtained was 0.0769. Although the average of the standard deviations over the periods is not a useful calculation for understanding new information about the portfolio, it does give a general indication of how volatile the selected market was during the chosen periods.

2. CHAPTER 2: MACHINE LEARNING FOR PORTFOLIO OPTIMIZATION

2.1 What Machine Learning is

Machine learning (ML) is a sub-category of artificial intelligence, which refers to the process by which computers develop pattern recognition, or the ability to continuously learn and make predictions using data and then make changes on their own, without specific programming[3].

Machine learning is incredibly complex and the way it works varies depending on the task and the algorithm used to implement it. However, at its core, a machine learning model is a computer that examines data and identifies patterns, and then uses that information to better complete the assigned task. Any task based on a set of data points or rules can be automated using machine learning, even the most complex ones such as answering customer service calls and examining resumes.

Depending on the situation, machine learning algorithms work using more or less human intervention/reinforcement. The four main models of machine learning are supervised learning, unsupervised learning, semi-supervised learning, and reinforcement learning[4].

- Supervised learning: the computer is given a labeled data set that allows it to learn how to perform a human task. This is the least complex model, as it attempts to replicate human learning, and it is the one used in this paper.
- Unsupervised learning, the computer is given unlabelled data; it extracts previously unknown patterns or information from these. There are several ways in which machine learning algorithms perform these operations, including:
    - Clustering, in which the computer finds similar data points within a dataset and groups them accordingly (creating 'clusters');
    - Density estimation, in which the computer detects information by observing how a dataset is distributed;

[3] www.hpe.com/it/it/what-is/machine-learning, Hewlett Packard Enterprise Development LP
[4] Facchini N. (2020) Machine learning ed investimenti finanziari, Università Ca' Foscari Venezia,

    - Anomaly detection, in which the computer identifies data points within a dataset that are significantly different from the rest of the data;
    - Principal component analysis (PCA), in which the computer analyses a dataset and summarises it in such a way that it can be used to make accurate predictions.
- Semi-supervised learning: the computer is given a partially labeled dataset and performs its task using the labeled data to understand the parameters to interpret the unlabelled data.
- Reinforcement learning: the computer observes its environment and uses this data to identify the ideal behavior that will minimize risk and/or maximize outcomes. This is an iterative approach that requires some sort of reinforcement signal to help the computer identify its best action.

The two main stages in the development of a neural network are training and inference. Training is the initial phase in which the deep learning algorithm is provided with a data set and the task of interpreting which data set it represents. Engineers then provide the neural network with feedback on the accuracy of its interpretation for adaptation. In this case, my feedback was just adjusting the algorithm in case of non-competitive results. Inference occurs when the neural network is deployed and can acquire a dataset it has never seen before and make accurate predictions about what it represents.

In the financial services sector, for example, banks use predictive machine learning models to analyze huge amounts of interconnected measures in order to better identify and meet customer needs. Predictive machine learning models are also able to detect and limit risk exposure. Banks can identify cyber threats, monitor and document fraudulent customer behavior and more accurately predict risks for new products. Key use cases for machine learning in banking include fraud detection and mitigation, personal financial advisory services, credit scoring, and loan analysis.

2.2 Training phase

During the training phase, the designer provides a series of examples for the machine. Each example consists of a series of input values and is accompanied by a label in which the designer indicates the result or a value judgment. The machine processes the data and learns from the examples to identify a predictive function or rule of thumb. Supervised learning is well suited to fully observable environments and in the presence of a human instructor. On the other hand, it is not very effective when analyzing feedback in partially observable environments because the agent cannot identify cause-effect relationships under conditions of certainty. In this case, it is a fully observable environment as it was previously constructed ad-hoc by me, to speed up and exemplify the calculations.

Before the actual learning phase, there is the pre-processing phase in which the information acquired by the machine can be represented in various ways. The main ones are propositional logic, first-order logic, Bayesian networks, neural networks, and decision trees[5]. Knowledge representation is one of the most important technical aspects in the study of machine learning, as it significantly influences the spatial and temporal complexity of the algorithm. Since, as previously explained, one of the criteria for building this portfolio is simplicity and ease of application, I opted for the decision tree model.

A decision tree is a system with n input variables and m output variables. The input variables (attributes) are derived from observation of the environment. The output variables, on the other hand, identify the decision/action to be taken. The decision-making process is represented with an inverted logic tree where each node is a conditional function. Each node tests a condition (test) on a particular property of the environment (variable) and has two or more branches downwards in function. The process consists of a sequence of tests. It always starts at the root node, the parent node located higher up in the structure, and proceeds downwards. Depending on the values detected at each node, the flow takes one direction or another and progressively proceeds downwards. The final decision is in the terminal leaf nodes, those furthest down. In this way, after analyzing the various conditions, the agent arrives at the final

[5] Jensen, F. V., & Nielsen, T. D. (2007). *Bayesian networks and decision graphs* (Vol. 2). New York: Springer.

decision. Logic trees have the undisputed advantage of simplicity. They are easy to understand and execute. Compared to neural networks, the decision tree is easily understood by humans. Therefore, humans can check how the machine arrives at the decision and possibly disagree. There are more efficient decision criteria that are more suited to machine logic but less comprehensible to humans. Furthermore, decision trees are easily developed in the form of programming code, because they can be represented in any propositional language.

The machine is then given starting data and, using predefined functions, calculates each possible scenario from the initial data. At the end of the training phase, the algorithm is asked to optimize the average output of all scenarios by modifying the coefficients of the previously fixed weight function. This is therefore a complex and demanding calculation, which is why I preferred to reduce the universe of purchasable stocks to 19 and the characteristics are taken into consideration to 3. Certainly, the algorithm in this paper could be improved by increasing the number of stocks that can be bought and the characteristics taken into consideration, but for each additional stock or characteristic, the possible scenarios increase exponentially. Therefore a computer with significantly more computing power than the average PC would be needed to implement the algorithm.

2.3 Inference phase

Machine learning (ML) inference is the process of running live data points into a machine learning algorithm (or "ML model") to calculate an output such as a single numerical score. This process is also referred to as "operationalizing an ML model" or "putting an ML model into production"[6]. When an ML model is running in production, it is often then described as artificial intelligence (AI) since it is performing functions similar to human thinking and analysis. Machine learning inference basically entails deploying a software application into a production environment, as the ML model is typically just software code (as in my case) that implements a mathematical algorithm. That algorithm makes calculations based on the characteristics of the data.

[6] Kolltveit, A. B., & Li, J. (2022, May). Operationalizing Machine Learning Models-A Systematic Literature Review. In 2022 IEEE/ACM 1st International Workshop on Software Engineering for Responsible Artificial Intelligence (SE4RAI) (pp. 1-8). IEEE.

ML inference is the second phase, in which the model is put into action on live data to produce actionable output. The data processing by the ML model is often referred to as “scoring,” so one can say that the ML model scores the data, and the output is a score.

In machine learning inference, the data sources are typically a system that captures the live data from the mechanism that generates the data. The host system for the machine learning model accepts data from the data sources and inputs the data into the machine learning model. The data destinations are where the host system should deliver the output score from the machine learning model.

The data destinations are where the host system should deliver the output score from the ML model. A destination can be any type of data repository like a database, and from there, downstream applications take further action on the scores. For example, if the ML model calculates a fraud score on purchase data, then the applications associated with the data destinations might send an “approve” or “decline” message back to the purchase site.

When deploying the artificial intelligence model during production, it is necessary to consider how it makes predictions. The two main processes for artificial intelligence models are:

- Batch inference: an asynchronous process that bases predictions on a batch of observations. Predictions are stored as files or in a database for end users or business applications.
- Real-time (or interactive) inference: free the model to make forecasts at any time and trigger an immediate response. This model can be used to analyze data from interactive and streaming applications.

In this paper, I will only use batch inference, so I will only go into detail about the latter. Since batch inference processes do not run continuously, it is advisable to automatically start, stop and size reusable clusters that can handle a range of workloads. Different models require different environments and the solution must be able to deploy a specific environment and remove it at the end of inference so that the calculation is available for the next model.

Although batch inference is an easier way to use and distribute the model in the production environment, it does present some challenges: depending on the frequency with which inference is performed, the data generated may be irrelevant at the time of access.

This is a variant of the cold start problem. Results may not be available for new data. For example, if a new user creates an account and starts shopping with a retail recommendation system, product recommendations will only be available after the next batch inference has been executed. If this is an obstacle for the use case, consider real-time inference.

Distribution in many areas and high availability are not critical issues in a batch inference scenario. It is not necessary to distribute the model at the area level and it may be necessary to distribute the data store with a high availability strategy in many locations. This will generally follow the design and high availability strategy of the application.

3. CHAPTER 3: CASE STUDY APPLICATION

3.1 Introduction

It is now time to go into more detail as to what specifically are the formulas that go into constructing the algorithm analyzed so far.

Starting with the first data implemented in the code, namely the returns. The code downloads from yahoo finance the daily returns of the previously selected shares over a period of 3 months (quarter). In particular:

$$r = \frac{P_c - P_o}{P_0}$$

It selects the opening price (Po) and the closing price (Pc) of each day of the selected period for each company. It then subtracts them and divides the result by Po. I then obtain the monthly percentages of stock returns and add them up to obtain the quarterly percentages.

Turning instead to the data of the characteristics this time will be downloaded directly from an excel file previously made by downloading data from Bloomberg. Once the program reads the data, it is standardized cross-sectionally. That is, for each period the data is transformed so that its average is 0. In particular, the formula used to achieve this is as follows:

$$C_{i,t} = \frac{C_{i,t} - \vartheta_t}{\delta_t}$$

Where:

- $C_{i,t}$ is the characteristic of the stock i at time t
- ϑ_t is the mean of the characteristic of each asset at time t
- δ_t is the standard deviation of the characteristic at time t.

The most important aspect of the parameterization is that the coefficients β are constant across assets and through time. Constant coefficients across assets imply that the portfolio weight in each stock depends only on the stock's characteristics and not on the stock's historic returns. Two stocks that are close to each other in characteristics

associated with expected returns and risk should have similar weights in the portfolio even if their sample returns are very different. The implicit assumption is that the characteristics fully capture all aspects of the joint distribution of returns that are relevant for forming optimal portfolios. Constant coefficients through time mean that the coefficients that maximize the investor's conditional expected utility at a given date are the same for all dates, and therefore also maximize the investor's unconditional expected utility.

A crucial aspect of the algorithm, as mentioned above, is the function that calculates the weights. In detail, the formula is:

$$w_{i,t} = e^{\beta_0 + \beta_1 Market\ Cap_{i,t} + \beta_2 EV\ to\ Book_{i,t} + \beta_3 Book\ per\ share_{i,t}}$$

and the results obtained are normalized, so for each period t, the weight of each share is divided by the weight of all shares in the period. The positive base exponent of the formula[7] ensures that only positive results can be obtained. So that they can be normalized and used as percentages of capital to be invested.

Having obtained the necessary data, I analyzed the objective function in detail. The investor is assumed to have constant relative risk aversion (CRRA) preferences. In particular, the formula in question is as follows:

$$u(r_{p,t+1}) = \frac{(1 + r_{p,t+1})^{1-\gamma}}{1 - \gamma}$$

Where:

- γ denotes the risk aversion of the investor
- $r_{p,t}$ denotes the total return of the portfolio at time t.

Risk aversion is an investor's preference to avoid uncertainty in their financial investments. The phenomenon of risk aversion implies by definition a certain level of risk rejection by those who invest in the financial markets. A person can be risk-averse,

[7] e = 2,71828

risk-neutral or risk-prone in a situation[8]. In this paper it will be used a standard risk aversion of 5, is used in most of the similar optimized portfolios.

The results of the last formula are then used to calculate the mean for each period. This final mean will be the function that I'm going to optimize:

$$\max_{\{w_{i,t}\}_{i=1}^{N_t}} \{ E\big[u\big(r_{p,t+1}\big) \big] \}$$

The result will be the betas that better satisfy the CRRA utility. The advantage of CRRA utility is that it incorporates preferences toward higher-order moments in a parsimonious manner. In addition, the utility function is twice continuously differentiable, which allows us to use more efficient numerical optimization algorithms that make use of the analytic gradient and Hessian of the objective function[9].

3.2 Code Description

The code starts with importing the libraries. More in detail the ones used are "NumPy" to create arranges and matrices, "matplotlib.pyplot" to draw graphs, "scipy" and "random" for minimizing and statistic tools, "pandas" to create dataframes and "datetime" and "dateutil" to select data in the correct timeframes, After having imported the necessary libraries, I begin to define the function I will use to calculate for each period the weights of each action. After that, I define the function that will allow me to download the necessary returns from yahoo finance. In particular, in the latter, I also put a counter to keep track of progress, as this is a computationally time-consuming operation. Having downloaded the necessary data, I move on to reading the excel files in which I will find the values of the selected characteristics. once the matrices have been filled with this data, we move on to standardization. Once I have all the necessary data, I can define the objective function, i.e. the average over the selected periods of the CRRA utility. This will be the function I will then maximize to find the optimal coefficients of the features. In particular, since python does not have a maximization tool, I placed a "-" before the output of the objective function so that I

[8] Cohn, R. A., Lewellen, W. G., Lease, R. C., & Schlarbaum, G. G. (1975). Individual investor risk aversion and investment portfolio composition. The Journal of Finance, 30(2), 605-620.
[9] M. W. Brandt, P. Santa-Clara, R. Valkanov (2009). Parametric Portfolio Policies: Exploiting Characteristics in the Cross-Section of Equity Returns, Oxford University Press on behalf of The Society for Financial Studies

could, instead, minimize it. Once I have used the 'minimize' tool based on the coefficients, I will obtain our result and go on to set the coefficients for calculating the weights in the testing phase. Once the training phase has been completed, it is the moment to move on to what is the final objective of this paper, namely to verify that the algorithm used with Machine Learning is indeed competitive in the market. I am going to redefine the function that calculates the weights but this time with fixed coefficients, the ones I got as an output of the optimization in the training phase. I then calculate the weights for each of the remaining periods.

Once the weights have been obtained, I know exactly for each period the percentage of the portfolio invested in each of the available companies. With this information, I need only multiply each percentage by its respective return to find the performance of my portfolio divided by period and each period divided by assets. To transform this data into an indicator of the generic performance of the portfolio resulting from this machine learning algorithm I will simply add up the percentages[10] for each asset class and then add them together. I then obtain a percentage of 25.44 % total return. However, since this is a return achieved over an unusual period (two and a half years), I have to annualize it in order to obtain an annual result and thus better compare it with the performance of other portfolios.

In detail, the annualization formula is as follows:

$$r_a = (1 + r_t)^{\frac{1}{N}} - 1$$

Where:

- r_a is the annualized return
- r_t is the total return
- N is the number of years

Once used this formula I obtained an 11.51% of annual return.

The last part of the code is dedicated to the statistical analysis of the results and the comparison with classic benchmark portfolios.

[10] They will be percentages as the result between two percentages

To summarise, then, the algorithm starts from the characteristics by calculating the values of the weights using fixed coefficients as a starting point. once the weights for each period have been obtained, they can be multiplied by the returns to obtain how much has been gained period by period. This value is then entered into the utility function in such a way as to calculate how much this gain is perceived by the investor in relation to the risk he took to earn it. Once this latter function has been optimised, we obtain the betas for which the average over the periods of this utility function is maximum. It must therefore be noted that the object of optimisation is the CRRA function and not the simple result of portfolio returns (weights multiplied by returns). Having obtained the optimal betas, I can repeat the process with the new data from the remaining periods (testing), obtaining the new weights and consequently the new portfolio returns. This time I no longer have any reason to use the CRRA function since the risk approach of my investments has already been set with the optimisation. I can then proceed with the statistical analysis to check its viability and strengths.

4. CHAPTER 4: FINAL RESULTS

4.1 Results of the analysis

The final result is not a particularly high return considering that we are reasoning over a period of two years and a half (10 quarters used in the out-of-sample period). At the same time, however, it is in line with the conservative portfolio type I wanted to achieve. The total return is thus limited, but the portfolio enjoys strong stability despite the short time span analyzed. The first statistics I wanted to calculate were those relating to weights. These are in fact key values not only in the success of the portfolio but also in its applicability in a real situation. The results are shown in figure 6:

Statistics	Values
∂^2	5,26%
∂	22,32%
Wmax	30%
Wmin	0,68%
Turnover	3.35%

Figure 6

Where :

- ∂^2 is the mean

- ∂ is the standard deviation

- Wmax and Wmin are the maximum and minimum weights used

Turnover is a crucial indicator because it highlights the trading activity needed to maintain the portfolio competitive. It's calculated with the following formula:

$$T_t = \sum_{i=1}^{N_s} |w_{i,t} - w_{i,t-1}|$$

So, it is simply the sum of all the weight changes during the periods. Such a low percentage obtained in the testing sample means that the portfolio is stable and easy to maintain. This also means that is easily applicable since in a real case there would have been also transaction costs for the trading activity that negatively affect the performance.

I therefore calculated the standard errors of the coefficient of the characteristics, estimated from 1000 bootstrapped samples. Standard error is a mathematical tool used in statistics to measure variability. It enables to arrive at an estimation of what the standard deviation of a given sample is. Standard error is used to estimate the efficiency, accuracy, and consistency of a sample. In other words, it measures how precisely a sampling distribution represents a population. The results obtained are shown in figure 7:

Beta	Boostrapped standard error
β_1	34.15%
β_2	37.76%
β_3	31.93%

Figure 7

When a sample of observations is drawn from a population and the sample mean is calculated, it serves as an estimate of the population mean. Almost certainly the sample mean will vary from the true population mean. The statistician's research will be useful in identifying the extent of the variation. This is where the standard error of the mean comes into play. When drawing different random samples from a population, the standard error of the mean is essentially the standard deviation of the different sample means from the population mean. However, multiple samples are not always available to the statistician. Fortunately, the standard error of the mean can be calculated from a single sample. It is calculated by dividing the standard deviation of the sample observations by the square root of the sample size. In this case, the sample size was particularly small and this is going to be the major feature that will negatively affect the statistics. However, the results are not so discouraging considering the small universe available and the fact that none of them exceed 40%.

Finally, I analyzed what were the actual investment returns. In particular, the following graph in Figure 8 represents the development of the overall portfolio performance period by period. As already mentioned, each period also takes into account the gains of the period before as they are considered reinvested. Of particular note is the fifth period in which a total return of even 68.33% is recorded. As can be seen in the horizontal axis there are only 9 periods, this is due to the fact that I am measuring the inter-period increase, which is why the results obtained from 10 periods become 9.

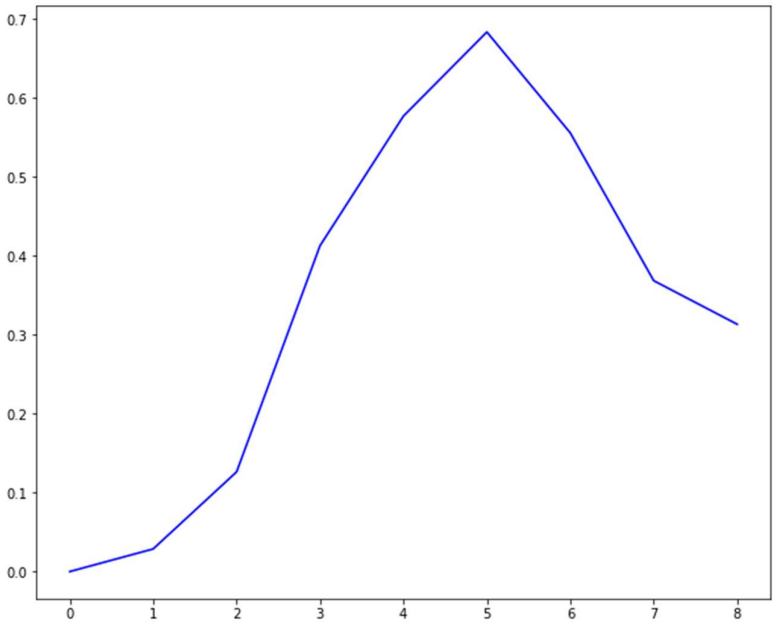

Figure 8

Subsequently, Figure 9 again shows the portfolio performance trend, but this time the periods are considered separately. It is as if every period the same share is reinvested and the profits of the previous period are collected.

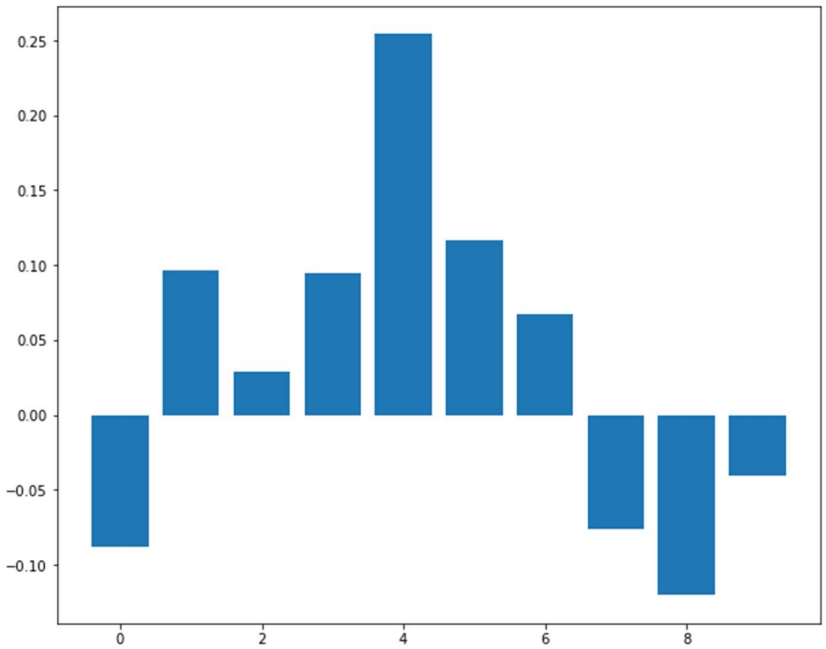

Figure 9

The largest loss recorded is that of the penultimate period in which the portfolio lost 12.03%. I also felt it was important to calculate the standard deviation of portfolio performance. The importance of such a measure is a key factor in the reliability of the algorithm. The results are great since I obtained 1.43% for the overall performance with a maximum of 2.7% counting the single performance of each company.

Another fundamental analysis I can do on the performance of my algorithm is to understand to what extent the results produced are simply caused by market volatility and to what extent they are due to optimal investments. To do that I will use the Sharpe Ratio formula[11]. The Sharpe ratio evaluates the relationship between an investment's return and risk. The idea that excess returns over time may indicate greater volatility and risk rather than investment expertise is expressed mathematically in this way.

To introduce the Sharpe ratio formula, I have to introduce first another important financial measure: the risk-free rate of return. It is the interest rate associated with investing in a risk-free asset whose return is certain, such as US Treasury bonds. The numerator of the Sharpe ratio is the difference over time between realized or predicted returns and the risk-free rate of return. The standard deviation of returns over the same time period, which indicates volatility and risk, serves as the denominator.

$$SR = \frac{R_P - R_f}{\partial}$$

Where

- R_P is the annualized return of the portfolio
- R_f is the risk-free rate
- ∂ is the returns standard deviation

In this case the risk free rate is 3,26%[12] so the SR of this portfolio is 0.972. Sharpe ratios greater than one are typically seen as "excellent," providing excess returns compared to volatility. Investors, however, frequently contrast a portfolio's or fund's Sharpe ratio with those of its competitors or market sector. Therefore, if the majority

[11] Kircher, F., & Rösch, D. (2021). A shrinkage approach for sharpe ratio optimal portfolios with estimation risks. *Journal of Banking & Finance*, *133*, 106281.
[12] 10Y annualized US Treasury Rate

of competitors have Sharpe ratios above 1.2, a portfolio with a Sharpe ratio of 1 might be considered insufficient.

Once all these analyses have been carried out, it is essential to implement one last one, namely a linear regression using Fama and French's three-factor model. The study conducted by Fama and French in 1996[13] shows how market anomalies related to average returns are related to each other and can be explained by the three-factor model developed by Fama and French in 1993.

This model shows that the expected excess return, relative to the risk-free rate of return, of a risky portfolio [E(R_p) - R_f] depends on three factors:

1) The expected excess return of the market portfolio (E[R_{mkt}] – R_f]);
2) The difference between the expected return of a portfolio of small-cap stocks and the expected return of a portfolio of large-cap stocks (SMB, small minus big);
3) The difference between the expected return of a portfolio composed of value securities and the expected return of a portfolio of growth securities (HML, high minus low).

The expected excess return of portfolio p is thus given by:

$$R_p - R_f = \beta_0 + \beta_1 (R_{mkt} - R_f]) + \beta_2 \text{SMB} + \beta_3 \text{HML} + \varepsilon i$$

Fama and French also showed that the slope of the factor HML indices firm difficulty. Indeed, weak firms with continuously low profits have a high and a positive slope of HML (value firms); strong firms with continuously high profits have a low and a negative slope of HML (growth firms). The idea of using HML to explain returns is consistent with Chan and Chen's 1991 study ("Structural and Return Characteristics of Small and Large Firms"), which noted a covariance between returns and firm difficulties, which is not captured by market returns but is offset in average returns.

The three-factor model provides an excellent description of the returns of portfolios sorted by size and how much they are over or undervalued from the market.

[13] "Multifactor Explanations of Asset Pricing Anomalies"

The results achieved by Fama and French in 1996 support the idea that their three-factor model (with the intercept $\beta_0 = 0$)[14] represents a faithful description of the trend in average returns, capturing many of the variations found in the cross-sectional analysis of average stock returns and absorbing the multitude of anomalies that plagued the CAPM.

Given the similarity of the model, I decided to use the three-factor model to compare the results, however, there are more complex and articulated models such as the five-factor model. Obviously, this is a type of regression that makes more sense to do on a number of observations far greater than mine (10), but I nevertheless preferred to include this analysis as it allows some confirmation of the philosophy with which the portfolio was constructed. The results are shown below:

OLS Regression Results

Dep. Variable:	r-RF	**R-squared:**	0.727
Model:	OLS	**Adj. R-squared:**	0.141
Method:	Least Squares	**F-statistic:**	1.491
No. Observations:	10	**Prob (F-statistic):**	0.309
Df Residuals:	6	**Log-Likelihood:**	85.547
Df Model:	3	**AIC:**	-9.109
Covariance Type:	nonrobust	**BIC:**	-7.899

	coef	**std err**	**t**	**P>|t|**
const	0.0071	0.051	18.553	0.000
Mkt-RF	0.0152	0.011	0.965	0.000
SMB	0.0046	0.025	0.990	0.361
HML	-0.0128	0.014	-0.911	0.028

The first result to be analyzed is the R-squared, which is almost close to 1 due to the good diversification of the portfolio. Turning then to Mkt-RF the value is much less than one, this means that the portfolio faces particularly less risk than the average market risk. The result for SMB tells how such a small value indicates that the portfolio prefers to invest in smaller stocks, however it should be noted how such a high P>|t|

[14] Suh, D. (2009). *Stock returns, risk factor loadings, and model predictions: A test of the CAPM and the Fama-French 3-factor model*. West Virginia University.

value makes this statistical measure inconsistent. Turning to HML the negative coefficient is in line with the stock selection criterion of preferring large companies with fixed profits (value stocks). Finally turning to const (β_0) I obtained a positive value, indicating a positive performance with respect to the risk of the Fama-French 3 factor model.

4.2 Benchmark analysis

As comparison portfolios, I chose the one in the Brandt-Santa Clara paper[15] (BSC), an equally weighted (EW,) and a mean-variance portfolio (MV). I have not chosen a large number of benchmarks because it is not so much the results obtained that I would like to dwell on, but the intrinsic characteristics that differentiate them. In particular, the results shown so far certainly give a partially reliable view of the algorithm's capabilities. In fact, one should remember the limited number of periods in both the training and testing phases.

The equally weighted simply distribute the capital evenly among all available companies without changing the percentages during the periods. This is the simplest portfolio, but it gives us an idea of what a more passive and therefore more market-driven approach would look like.

Mean-variance analysis or Modern portfolio theory is a mathematical framework for assembling a portfolio of assets such that the expected return is maximized for a given level of risk. It rewards the idea that owning different kinds of financial assets is less risky than owning only one type. Its key insight is that an asset's risk and return should not be assessed by itself, but by how it contributes to a portfolio's overall risk and return. It uses the variance of asset prices as a proxy for risk[16].

EW and MV were applied to the same data used by my algorithm and with the same universe of stocks. BSC's statistics, on the other hand, are taken directly from their previously cited paper. Figure 10 shows the key results of each.

[15] M. W. Brandt, P. Santa-Clara, R. Valkanov (2009). Parametric Portfolio Policies: Exploiting Characteristics in the Cross-Section of Equity Returns, Oxford University Press on behalf of The Society for Financial Studies

[16] Zhou, X. Y., & Yin, G. (2003). Markowitz's mean-variance portfolio selection with regime switching: A continuous-time model. *SIAM Journal on Control and Optimization*, *42*(4), 1466-1482.

	r_t	r_a	T_t
EW	8,11%	3,16%	0%
MV	14,63%	5,61%	13,28%
BSC	26,20%	5,40%	6,21%

Figure 10

It should be emphasized that the turnover of the EW is zero because the principle is to always allocate the same percentages in each share, and therefore the weights do not change from period to period, resulting in no turnover. Furthermore, it can be seen that the BSC has a higher total return than the other two but is developed over a longer period and therefore loses much of the gap when discounted. Of course, all BSC statistics are only taken in the testing phase (Out of Sample).

It should also be noted that the latter is a much more complex algorithm than mine, there are many more constraints and the university of stocks available is much larger. In its simplicity, however, my portfolio achieves a much higher discounted return (11.51%) than that of the previous paper. It is interesting to see how simplifying an algorithm as complex as BSC's still leads to better results. At the same time, however, it must also be emphasized that it may also simply have been influenced by general market trends or the years under consideration (crises, conflicts, etc.). One would have to do a more in-depth analysis and test my algorithm over a longer period in order to actually have solid results.

For Sharpe ratios calculations, the results are shown in Figure 11. it is important to emphasise, however, that the results of the BSC portfolio were taken directly from the paper and thus calculated with a different risk free[17]

	SR
EW	0,324
MV	0,511
BSC	0,941

Figure 11

Although the risk free used in the other paper is different, this ratio still allows me to compare the two algorithms. At the same time, it also gives more meaning to the previously calculated SR for this portfolio. It is clear how this measure can also be

[17] 6,61 %

relative, since although my result was less than one it is still greater than similar types of portfolios despite only the period of the testing phase being considered.

CONSIDERATIONS AND CONCLUSIONS

Overall, I can be satisfied with the success of the algorithm. At the same time, many improvements could be implemented. Starting with the simplest, and already abundantly mentioned, such as increasing the training and testing periods and allowing the investor to be able to decide between a larger universe of stocks. However, other types of improvements can also be implemented. Realistic transaction costs could be introduced to understand how much of the profit is dissipated in trading costs between periods. Another important improvement could be to allow short selling. It would greatly distort the nature of my portfolio but would certainly lead to higher returns (however at higher risk). Another aspect that could be improved would be to use a fourth characteristic, so as to have a more complete view of the intrinsic properties of each stock. Given the high inflation that has been hitting the markets, particularly the European markets, in recent months, it would also be interesting to understand how it affects portfolio returns. Finally, an improvement that I am particularly interested in could be to implement the possibility for investors to also dedicate part of their investments to risk-free assets, such as government bonds or Eurobonds. The possibility of also being able to invest in these types of financial products would give enormous elasticity to the algorithm, allowing the investor to 'withdraw' from the equity market when the situation becomes riskier by investing in securities with lower returns but significantly safer; thus being able to go positive even during unfavorable market trends. This is, however, a far from simple implementation as the given utility function and the principle by which the weights are calculated would have to be reviewed.

It must be noted, however, that even though my algorithm has strong limitations, I have managed to be consistent with the philosophy with which I constructed it. In its simplicity, I quickly solved complicated problems such as the short selling constraint or excessive turnover. Dealing with a non-linear weight function would not have been easy, especially solving the second problem.

I hope this paper can be a pretext to bring even less related audiences closer to the subject due to its simplicity, demonstrating that machine learning can become the solution to a large number of problems both in everyday life and in the long term.

BIBLIOGRAPHY

- Cohn, R. A., Lewellen, W. G., Lease, R. C., & Schlarbaum, G. G. (1975). Individual investor risk aversion and investment portfolio composition. The Journal of Finance, 30(2), 605-620.
- Facchini N. (2020) Machine learning ed investimenti finanziari, Università Ca' Foscari Venezia,
- Jensen, F. V., & Nielsen, T. D. (2007). *Bayesian networks and decision graphs* (Vol. 2). New York: Springer.
- Kircher, F., & Rösch, D. (2021). A shrinkage approach for sharpe ratio optimal portfolios with estimation risks. *Journal of Banking & Finance*, *133*, 106281.
- Kolltveit, A. B., & Li, J. (2022, May). Operationalizing Machine Learning Models-A Systematic Literature Review. In 2022 IEEE/ACM 1st International Workshop on Software Engineering for Responsible Artificial Intelligence (SE4RAI) (pp. 1-8). IEEE.
- M. W. Brandt, P. Santa-Clara, R. Valkanov (2009). Parametric Portfolio Policies: Exploiting Characteristics in the Cross-Section of Equity Returns, Oxford University Press on behalf of The Society for Financial Studies
- Suh, D. (2009). Stock returns, risk factor loadings, and model predictions: A test of the CAPM and the Fama-French 3-factor model. West Virginia University.
- www.degiro.it/trading-conoscenza/inverstitori-accademia/principianti-corso/market-cap
- www.hpe.com/it/it/what-is/machine-learning, Hewlett Packard Enterprise Development LP
- Zhou, X. Y., & Yin, G. (2003). Markowitz's mean-variance portfolio selection with regime switching: A continuous-time model. *SIAM Journal on Control and Optimization*, *42*(4), 1466-1482.

APPENDIX

Training

```
>>> Market_cap = np.ones ((19,35))

>>> EV_to_Book = np.ones ((19,35))

>>> Book_Value_per_Share = np.ones((19,35))

>>> beta = np.array((0.01,0.01,0.01,0.01))

>>> Pesi = np.zeros((19,35))

>>> asset_list = ["AAPL", "AIR", "ALV", "AMZN", "AXP", "C","CSCO", "DIS",
"GOOGL", "GS", "IBM", "JNJ", "KO", "MA", "NKE", "MSFT", "VZ", "PG", "PFE"]

>>> norm = np.zeros(35)
```

```
>>> y=5

>>> start_sp = datetime.datetime(2013,10,1)

>>> end_sp = datetime.datetime(2013,12,1)

>>> def calcolapeso(beta, Market_cap, EV_to_Book, Book_Value_per_Share, Pesi,
t):

    w = np.zeros(19)

    norm = 0

    p=np.zeros(19)

    for i in range (19):

       w[i] = np.exp (beta[0] + beta[1]*Market_cap[i,t] +
       beta[2]*EV_to_Book[i,t] + beta[3]*Book_Value_per_Share[i,t])

        norm += w[i]

        Pesi[i,t] = w [i]

    for i in range (19):

        Pesi[i,t]=Pesi[i,t]/norm

        p[i]=Pesi[i,t]

    return (p)


>>> def scaricodati(start_sp, end_sp):

    ret = pd.DataFrame()

    historical_returns = pdr.get_data_yahoo(asset_list, start_sp, end_sp,
interval= "mo")

    r = (historical_returns['Close'] - historical_returns['Open']) /
historical_returns['Open']

    m=r.mean()*3

    ret=ret.append(m, ignore_index=True)

    for i in range(34):

        start_sp=end_sp

        #end = datetime.timedelta(months=3)

        end_sp = start_sp + relativedelta(months=+3)

        historical_returns = pdr.get_data_yahoo(asset_list, start_sp, end_sp,
interval= "mo")

        r = (historical_returns['Close'] - historical_returns['Open']) /
historical_returns['Open']

        m=r.mean()*3
```

```
        ret=ret.append(m, ignore_index=True)
        print(i)
    return ret


>>> Ret=scaricodati(start_sp, end_sp)
>>> for j in range (19):
    M_c = pd.read_excel('DATI.Final.xlsx', sheet_name=j)
    EV_t_b = pd.read_excel('DATI.Final.xlsx', sheet_name=j)
    BV_p_S = pd.read_excel('DATI.Final.xlsx', sheet_name=j)
    for i in range (35):
        Market_cap[j,i] = M_c.iat[2,2+i]
        EV_to_Book[j,i] = EV_t_b.iat[3,2+i]
        Book_Value_per_Share[j,i] = BV_p_S.iat[4,2+i]


>>> def standardization (c):
    for t in range (35):
        ch=np.zeros(19)
        for i in range (19):
            ch[i]=c[i,t]
        for i in range(19):
            c[i,t] =( ch[i] - ch.mean() ) / ch.std()
    b=np.zeros(35)
    for t in range (35):
        a=np.zeros(19)
        for i in range(19):
            a[i]=c[i,t]
        b[t]=a.mean()
    print(b)


>>> standardization (Market_cap)
>>> standardization (EV_to_Book)
>>> standardization (Book_Value_per_Share)
```

```
>>> def Funz_Ob (beta, Market_cap, EV_to_Book, Book_Value_per_Share,
Risk_Aversion, Returns, start, end):

    sol=np.zeros(end)

    for t in range (start, end):

        k=calcolapeso (beta, Market_cap, EV_to_Book, Book_Value_per_Share,
Pesi, t)

        r=Returns.loc[t:t,:]

        sol[t]=+ -(((1+r@k)**(1-Risk_Aversion))/(1-Risk_Aversion))

    out= -sol.mean()

    return (out)

>>> res = minimize(Funz_Ob, beta, args=(Market_cap, EV_to_Book,
Book_Value_per_Share, y, Ret, 0, 25), method='SLSQP', options={'ftol': 1e-09})

>>> print (res)


fun: -0.2593103290186905

     jac: array([0., 0., 0., 0.])

 message: 'Optimization terminated successfully'

    nfev: 130

     nit: 26

    njev: 26

  status: 0

 success: True

       x: array([ 1.00042757e-02, -3.70373349e+01, -4.36879048e+01,
4.32887936e+01])


>>> Beta=res.x

array([1.00042757e-02, -3.70373349e+01,-4.36879048e+01, 4.32887936e+01])
```

Testing

```
>>> Weights =np.zeros((19,10))

>>> start=25

>>> end=35

>>> def Calcolapeso (beta ,  Market_cap, EV_to_Book, Book_Value_per_Share,
Weights, t):
```

```
    w = np.zeros(19)
    norm = 0
    p=np.zeros(19)
    for i in range (19):
       w[i] = np.exp (beta[0] +beta[1]*Market_cap[i,t] +
       beta[2]*EV_to_Book[i,t] + beta[3]*Book_Value_per_Share[i,t])
        norm += w[i]
        Weights[i,t-25] = w [i]
    for i in range (19):
        Weights[i,t-25]=Weights[i,t-25]/norm
        #p[i]=Weights[i,t-25]
    return (Weights[:,t-25])


>>> r=Ret.loc[25:35,:]
>>> b=np.transpose(r)
>>> a=b*Weights
>>> z=np.transpose(a)
>>> z.mean()
>>> a=a+1
>>> a.prod()
>>> portfolio_return = a.prod().prod() - 1
>>> def annualization (returns, t):
       return ((1+returns)**(1/t))-1
>>> annualization (portfolio_return, 2.5)
```

Statistics

```
>>> Weights.mean()
>>> Weights.std()
>>> def weight_analysis (a,b):
    if b == 'max':
        max=0
        for i in range (19):
            for t in range (10):
```

```
                if a[i,t] > max:
                    max=a[i,t]
        print (max)
    if b=='min':
        min=1000
        for i in range (19):
            for t in range (10):
                if a[i,t] < min:
                    min=a[i,t]
        print (min)

>>> weight_analysis(Weights, 'max')
>>> weight_analysis(Weights, 'min')
>>> def turnover (a):
    r=0
    c=np.zeros((19,10))
    for i in range(19):
        for t in range (1,10):
            c[i,t]=abs(a[i,t]-a[i,t-1])
            r=+ c[i,t]
    return (r)
>>> def Bootstrapped_Standard_Errors (characteristic):
    bootstrap_means=np.zeros(19)
    for i in range (19):
        bootstrap=random.sample(characteristic.tolist(), 4)
        bootstrap_means[i]=np.mean(bootstrap)
    sample_std=np.std(bootstrap_means, ddof=1)
    print (sample_std)
>>> gra=np.zeros(9)
>>> gra[0]=gr[0]*gr[1]
>>> for i in range (1,9):
    gra[i]=gra[i-1]*gr[i+1]
```

```
>>> gra=np.zeros(9)
>>> gra[0]=gr[0]*gr[1]
>>> for i in range (1,9):
    gra[i]=gra[i-1]*gr[i+1]

>>> def graphs1(what,t):
    plt.figure(figsize=(10, 8.3))
    y1list = what
    tlist = np.arange(0,t,1)
    plt.plot(tlist, y1list, 'b-')
    plt.show()
graphs1(gra-1, 9)

>>> def graphs2(what, t):
    plt.figure(figsize=(10, 8.3))
    y1list = what
    tlist = np.arange(0,t,1)
    plt.bar(tlist, y1list, 0.8)
    plt.show()

>>> def graphs2(what, t):
    plt.figure(figsize=(10, 8.3))
    y1list = what
    tlist = np.arange(0,t,1)
    plt.bar(tlist, y1list, 0.8)
    plt.show()

>>> graphs2(a.prod() - 1, 10)
```

ABSTRACT

In this paper, I deal with the independent construction of a portfolio of 19 stocks based on some of their market characteristics (Market Cap, EV to Book and Book Value per Share). I use a code I wrote in Python and implement data obtained from Bloomberg in an algorithm I created inspired from the one of Brandt-Santa Clara. The resulting algorithm uses simplified machine learning processes due to the computational complexity of the operation. The main objective of this paper is to present how to independently construct a portfolio model that is as competitive and applicable in real markets as possible. The first part is devoted to the criteria according to which I decided to construct the algorithm and how I processed the data in such a way that it could be optimally used for the purpose. The downloaded data is divided temporally, the first 75% is used for the training phase of the algorithm, in which it calculates the optimal coefficients by which the market characteristics mentioned above influence the decision to invest or not in a stock. The remaining 25% is used to test the applicability of the portfolio and see if the previously calculated coefficients allow competitive returns by also calculating the risk taken.

The second part is more dedicated to Machine Learning in general, always referring, however, to where my algorithm stands within the vast world of ML. Once the results have been obtained, I analyse them to understand the applicability of the portfolio and compare them to other types of portfolios on similar data in order to understand its competitiveness as well. I also decided to print a piece of code in the appendix so as to show how the process has been simplified even in its most operational part, i.e. the code.

As previously explained the algorithm is quite simple in comparison to other similar portfolios created by researchers, the key point of this paper is that it was created entirely by me, with limited IT and even financial skills given the subject matter.

For these reasons, the paper is intended to be a demonstration of how it is feasible to autonomously create a basic portfolio that is also competitive in the long term; implementing a concept as complex at first glance as Machine Learning, which, if properly simplified, can be easily accessible and very useful.