**Bachelor in Data Science and Engineering - 2021/2022**

**University Carlos III Madrid**

**Data Science Project - Group 96**

# Robo Advisor algorithm design for drawdown-based optimization of investment portfolios.

## Collaborating with **IronIA Fintech**


**Bernardo Bouzas García**
NIA: **100406634**
**David Méndez Encinas**
NIA: **100406667**
**Claudio Sotillos Peceroso**
NIA: **10040940**
**Laura María Torregosa Gómez-Meana**
NIA: **100406691**

# Index

# Introduction

## Introduction to the partner

**IronIA Fintech** has collaborated in the realization of this project: IronIA is a financial asset management platform founded and based in Spain. Most traditional banking networks do not work with open architecture, they only sell their own products, but IronIA is committed to making available to customers more than **18,000 investment funds** in which they can invest their savings.

IronIA stands out with a differential value proposition in a financial sector where restrictions and obstacles make the novice investor's experience a complicated process that sometimes leads to poor investment decisions and loss of capital. By drastically reducing commissions and offering unlimited changes at no cost in its portfolio allocations, IronIA aims to reinvent the way to **invest with freedom**, at a very low cost in the form of a monthly subscription fee.

As it is a large number of funds, they have designed their own ranking to help the client find the best ones. They rate their funds with IronIA points -from 1 to 5, the same as the stars used to evaluate an application-. They have designed a model that makes it possible to discriminate between what is important and what is not in each fund in order to compare them. In this way, the client can see how a certain fund stands out.

IronIA uses **AI models** to rate their funds and then rank them based on their ironIA points. In addition, they use a kind of recommendation system to suggest funds that may be of interest to the clients. In this way, the clients can compare and choose which fund suits them best. This raises the level of the firm's activity from simply offering funds to being able to make **data-driven investment recommendations**, which is a considerable advantage for the small investor who needs additional tools to evaluate the market.

## Introduction to the project

To introduce the approach to our work, we must make clear the central concept on which we build our model: funds, a type of financial product. **Funds** are **collective investment vehicles** managed by a professional fund manager. The fund can invest in a wide range of assets: bonds, equities, derivatives, currencies... They can also invest in any geographical area, as long as they adhere to the estimated risk profile determined for each fund.

The profitability of the investment is given by the performance of the fund. Each investment fund has its own risk exposure, and the investment can be adapted based on the risk that each participant is willing to assume, as well as their interests.

Just as there are shareholders of a company, there are unit holders of an investment fund, who are those investors who become unit holders of the fund in the proportion of the contributions they have made. Each of the unit-holders of a fund may leave the fund, obtaining the reimbursement of the investment, at any time. The units are negotiable securities, not normally traded on any stock market, but sold and repurchased by the management company. Note that IronIA is not the entity that manages the composition of the funds, they offer clients the possibility to invest in them.

Our task is to **explore different methods of portfolio optimization**, that is, to achieve an algorithm that indicates in which funds it is more convenient to invest, and the percentage of capital that should be allocated to that investment, based on the expected return and an exhaustive analysis that we will perform around different risk measures. In this way, we want to improve or complement the currently proposed model, relying on both **classical** portfolio optimization theory and more **modern** and disruptive approaches that we will study.

# Initial thoughts

## Historical background

To design a strategy consistent with the state-of-the-art in the industry, we must understand the historical basis of portfolio optimization, explore the techniques that have evolved since that point, and grasp the economic theory behind them.

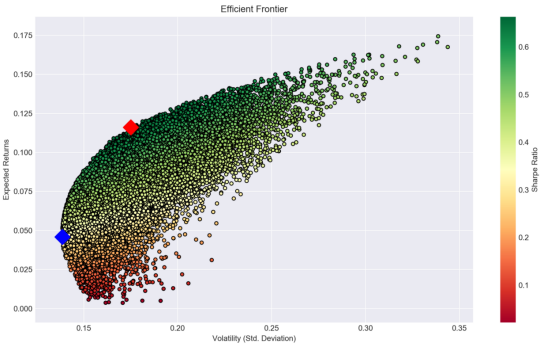
Prior to 1952, portfolio construction was based solely on expected return, and did not take risk as a variable. Since one of the characteristics of the problem at hand involves the investor's risk profile, we must understand the solution to this inconsistency: the **Markowitz Model**.

The theory of portfolio formation consists of three stages:

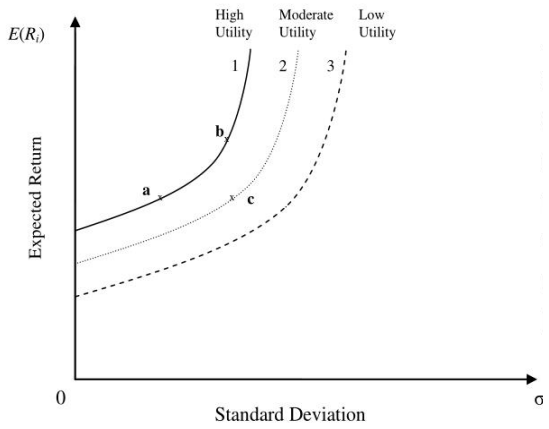
1. Determination of the set of efficient portfolios.
2. Determination of the investor's attitude towards risk.
3. Determining the optimal portfolio.

It also relies on some starting assumptions:

1. The return of a portfolio is given by its mathematical expectation or mean.
2. The risk of a portfolio is measured through volatility (variance or standard deviation).
3. The investor always prefers the portfolio with the lowest risk given an expected return.




An **efficient portfolio** is a portfolio that offers the minimum risk for an expected return value. On the **efficient frontier,** each portfolio minimizes risk for a given return. We should find it by tackling a mathematical optimization problem, which gives us a clue as to which approach we should follow. Ideally, an investor seeks to populate the investment portfolio with assets that offer exceptional returns, but whose combined standard deviation is smaller than the standard deviations of the individual securities. The less correlated the assets are (lower covariance), the lower the standard deviation. If this combination of optimizing the return versus risk calculation is successful, then that portfolio should align along the efficient frontier line. This is what we aim to achieve, building a model starting from this classical paradigm.




The investor's attitude towards risk will depend on their map of **indifference curves**.

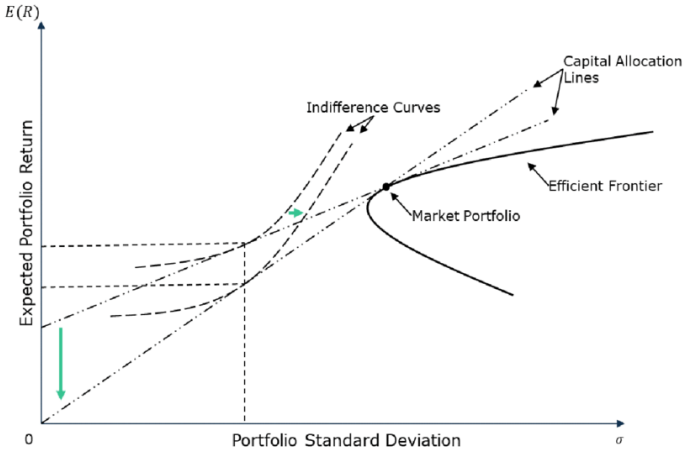
Specifically speaking, a map of indifference curves is a set of contour lines for a utility function. If the utility function changes, then the map changes, unless it is an increasing monotonic transformation of the utility function.

Indifference curve maps must consist of curves whose utility level increases as they are located farther from the origin, strictly convex curves, to reflect the fact that

scarce goods are valued more than abundant ones, continuous curves, to comply with the axiom of completeness, and curves that do not cut each other, to comply with the axiom of transitivity.

Therefore, each investor will have a different risk aversion and for each level of risk they are willing to assume, they will expect a certain return.

An investor's **optimal portfolio** is determined by the tangent point between one of the investor's indifference curves and the efficient frontier. Curves below that point will give less satisfaction and those above that point are not feasible.




This illustrates very well the concept we are trying to develop, and gives us a good understanding of this model, the fundamental basis of modern investment portfolio design theory, we move on to look for indices, or benchmark metrics, that can help us build our system and reliably optimize a set of solutions.

The first one we introduce is the **Sharpe Ratio**, the relationship between the additional return of a mutual fund, measured as the difference between the fund's return and the return of a risk-free asset, and its volatility, measured as its standard deviation. We will take as the "risk-free asset" the return on short-term government bonds in the geographic area that most closely resembles the assets in which the fund invests.

The higher the Sharpe Ratio, the better the fund's performance relative to the amount of risk taken in the investment. If the Sharpe Ratio is negative, it indicates underperformance relative to the risk-free return. Any Sharpe Ratio less than 1 implies that the asset's return is less than the risk we are taking by investing in it. The Sharpe Ratio helps us to analyze whether the risk/return ratio of an investment is appropriate and to compare different funds within the same category, but it should always be considered in conjunction with other relevant metrics that are used as an industry standard.

However, knowing about the existence of metrics such as these has made us delve even deeper into the risk measurement we can perform, and this research has led us to determine certain key metrics that we will explain and formulate in detail in the following pages.

We came to understand that over more than seven decades, the methodology has evolved a lot, as well as the implementation strategies. Therefore, we decided to look for resources that, following the guidelines set by these critical initial models, would facilitate the development of a solution in a testing and benchmarking environment.

# Available data overview

## Working in a cloud environment

To design a suitable methodology, the partner firm for this project, IronIA Fintech, has provided seamless access to a collection of valuable resources. IronIA owns a database hosted on **Google Cloud** with very comprehensive data on tens of thousands of investment funds over the course of lustrums. The information available and its relevance will be detailed in the following paragraphs.

To access this data origin, we have been provided with a service account from which to initiate a connection to the Google endpoint. Therefore, instead of working locally and connecting the project through GitHub, the team has decided to start preliminary testing and pre-processing using **Google Collaboratory**. This allows us to leverage the in-cloud tools offered by Google, particularly the BigQuery API.

**BigQuery** supports a standard SQL dialect that complies with the ANSI:2011 standard, reducing the need to rewrite code. The tool also provides free ODBC and JDBC drivers to ensure that commonly used applications can interact with the platform's powerful engine.

## Prices Data Frame

| fund ID | fund name | currency | date | NAV |
|---|---|---|---|---|

The **Net Asset Value** of a fund is the unit price of each share in the fund at a given point in time. In our case, each date represents the daily **NAV** of each fund available in the dataset.

This value is the result of dividing the fund's net assets by the number of shares in circulation. The fund's NAV represents a "**per-share**" value of the fund, which makes it easier to be used for valuing and transacting in the fund shares. This is a dynamic concept, since it must be calculated on a daily basis according to the market prices of the assets comprising the fund's portfolio.

$$NAV = \frac{Value\ of\ Assets - Value\ of\ Liabilities}{Total\ Shares\ Outstanding}$$

As in the case of IronIA, management fees charged to the clients are implicit for funds. In other words, the percentage corresponding to the fees is deducted from the result of the above formula. This is because these are charged directly and are already deducted from the NAV of the fund. Another nuance to highlight in this section is that the price at which a fund buy or sell transaction materializes may be different from the price on the day the order was actually given. We must bear in mind that the funds offered by IronIA are of a very diverse geographic nature, so it is essential that the market in which each fund is invested is open at the time the NAV is obtained. In this regard, days on which there is no market for assets representing more than 5% of the fund's assets are not considered business days for NAV purposes. In such a case, the NAV that would be available to us is, in effect, the NAV of the next business day.

This is the largest dataset (10.76 GB), as it contains the daily NAV record of each fund. However, it will be very useful for the calculations to be explained shortly.

## Categories Data Frame

| category | subcategory | benchmark | benchmark ID | morningstar ID |
|---|---|---|---|---|

This dataset contains all the fund categories that IronIA considers. On the one hand, we find the Morningstar ID, in terms of the categories they use. Morningstar is a leading financial services provider that assigns ratings to funds based on an analytical estimate of a stock's target price, specifically:

1. Analysis of the company's competitive advantage.
2. Estimation of the target price of the stock.
3. Uncertainty about the estimated target price.
4. Current market price.

To evaluate the quality of the funds in a comparative way, these funds are grouped by Morningstar **categories**. These categories are referred to in the IDs presented in our database. We also find information about the **benchmark**. By definition, a benchmark is a comparative reference, a tool used to measure something against its comparables.

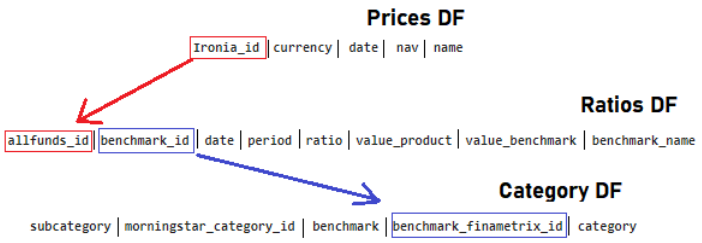
For **actively managed funds**, the benchmark is the reference index with which to compare their performance, but their portfolio does not have to be the same as the composition of their benchmark. In other words, a Spanish fund may have the IBEX 35 as its benchmark, but the fund's positions do not necessarily have to reflect exactly the same composition as the IBEX 35. The objective of the funds in this category is what is commonly known as "beating the benchmark", i.e. to outperform their reference index. In this way, we can test the consistency over time of an actively managed fund by seeing whether it consistently outperforms its benchmark over a moderate period of time or only on an ad hoc basis.

On the other hand, **index funds** and **ETF**s exactly replicate the index they track. This means that the composition of their portfolios does match that of their benchmarks, so that if the index goes up or down, the passively managed fund will register a very similar movement. Understanding this leads us to consider these types of products in our more conservative roboadvisor options, as we have historically found their growth to be more moderate but consistent over the long term.

## Ratios Data Frame

| id | benchmark id | date | period | value product | value benchmark | benchmark name |
|---|---|---|---|---|---|---|

This dataset contains general information of the funds. It consists of 8 columns. The id, the benchmark ID, the date, the period (referring to the year), the . value product, the value benchmark and the benchmark name. This is the second biggest DF (7.05 GB) since it contains the yearly record of each fund for a given ratio. These three datasets are connected as the image shows.



## Overcoming main dataset challenges

As we intend to test several approaches starting from the same initial conditions, it is of vital importance to have a complete and accurate dataset. To obtain it, we have had to face a very common problem in this type of project involving large amounts of data: the presence of **null** values.

### Null values approach

We have null values when the value of a column is unknown or missing. A NULL value is neither an empty string (for character or date and time data types) nor a zero value (for numeric data types). The ANSI SQL-92 specification indicates that a NULL value must be the same for all data types, so that all of them are treated uniformly. It is by these guidelines that we have been guided in designing our null value handling strategy. Our first step is to get all the different ID's from the Ratios dataframe using a standard SQL query.

### COVID19 impact

The second problem we face is the presence of irregularities in the behavior of financial markets in recent years. In this case, we are faced with a phenomenon that interferes abruptly and unpredictably in the global economy, a blackswan, the COVID-19 pandemic.

The impact of this pandemic has left a very particular situation in the markets. A priori, it seems that **fixed-rate products** have ignored the economic consequences of the virus. Last year, the COVID-19 crisis triggered one of the deepest recessions in history, with global growth declining by 3.6% year-on-year. After the initial widespread sell-off in equity markets, global fixed-rate equities (as calculated by the MSCI AC World index) went to offer a 15% return in 2020.

Secondly, investments in countries appear to have been highly influenced by the management of the sanitary situation and vaccine developments. A pattern can be identified in which countries where **economic growth expectations** have improved have shown the best performance in terms of developing a potential vaccine. However, this has not translated into higher **variable rate equity returns** in these markets. Surprisingly the other way around, the relationship between vaccination rates and growth expectations has developed most notably in **foreign exchange** (forex) markets.

In conclusion, the impact of the pandemic has been very diverse, slightly irrational and unpredictable. Therefore, we have decided to design our model with the extra assumption that no black swan like the one we have just witnessed will interfere with its performance. This allows us to perform our analysis with more consistent and robust data. Accordingly, we have looked for all the days that the stock market has been open **from 2016 to 2019** and we have generated a file containing those dates, thus avoiding the influence of 2020 and 2021. In addition, we consider that it can be highly representative, since markets have performed a "V-shaped recovery", returning to a state very similar to what would have been expected if growth had not been affected by the pandemic. To illustrate this, we show the growth of the S&P 500 (SPX), one of the most important stock market indexes in the United States; it is considered one of the most representative indexes of the real market situation. Source: Google Data.

## Cleanness vs representativeness tradeoff

Now that we have the right data collection approach, we begin the extraction process.

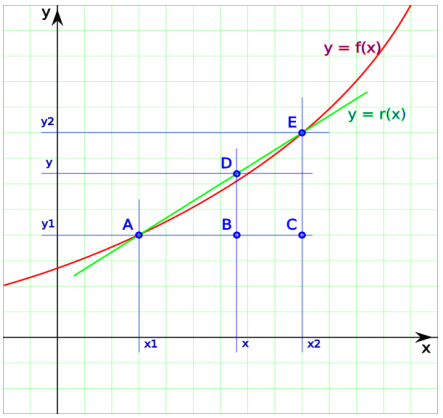
We iterate through each ID and we check that it has at least the length of the dates minus a delta parameter (for the tests is set to 30), otherwise, the ID is discarded. By doing this, we are ensuring that all the IDs have a small value of null values, getting just the funds that have **almost the complete time series**.

By having a substantial number of funds, we have considered the tradeoff between data cleanness and representativeness, since we are removing some options that might be optimal for the model. We have decided that this is a compromise we are willing to make, since there are so many funds, many of them share similar characteristics and we think they could be substituted in many portfolios with little or no change in performance.

## Final linear interpolation

Nevertheless, there are still missing values, so we have to apply a new null processing technique. In this case, we could take the value closest to the NULL under consideration. However, we believe it is more appropriate to **approximate more closely by interpolation**. Interpolation, we expect, will give us a small error with respect to the true function value, but it will always be smaller than taking the nearest value.

There is linear interpolation, quadratic interpolation, cubic and nth power interpolation (for data that have the characteristic of being raised to powers of order 2, 3, 4, ..., ..., and up to n), exponential interpolation, etc. In our case, observing linearity in the values we are studying, we choose to apply linear interpolation, which is a particular case of Newton's general interpolation. Moreover, the smaller the interval between the data, the better the approximation. This is due to the fact that, as the interval decreases, a continuous function will be better approximated by a straight line. Our intervals are very short, and the daily data provide a good basis for this interpolation option.




In simplified terms, just for illustration purposes, linear interpolation consists of drawing a line through (x1,y1) and (x2,y2), y = r(x) and calculating the intermediate values along this line instead of the function y = f(x). For this purpose we rely on the similarity of triangles $\widehat{BAD}$ and $\widehat{CAE}$.

$\frac{\overline{AC}}{\overline{AB}} = \frac{\overline{CE}}{\overline{BD}}$ then, $\overline{BD} = \frac{\overline{AB}}{\overline{AC}}\,\overline{CE}$, or what is the same, $(y - y_1) = \frac{(x - x_1)}{(x_2 - x_1)}(y_2 - y_1)$, so $y = \frac{(x - x_1)}{(x_2 - x_1)}(y_2 - y_1) + y_1$

Once all the data is properly filled and cleaned, we split it into **two different data frames**, one with the train set (2016-2018) and the other with the test set (2019). The final result is a train and test set with a total of **12367 different funds**. The time needed to generate the dataset was around **10 hours**, as each ID query takes **3.5 seconds** to complete.

# Initial Portfolio Allocation

Determining the initial structure of the portfolios is one of the instrumental parts of this project. We take our first steps in this direction, because it allows us to familiarize ourselves with basic and complex asset allocation techniques, and it will also produce a framework on which our robo-advisor can iterate and improve as time progresses.

As a general rule, **proper diversification** will provide greater long-term appreciation potential while keeping risk contained. Ideally, the robo-advisor should be based on an optimized pool of funds, a mix of publicly traded assets and debt instruments consistent with the parameterized financial objectives and needs.

The scale and complexity of portfolio optimization over many holdings means that the work generally requires a high computational cost, which must be mitigated by seeking efficiency. Fundamental to this optimization is the construction of the covariance matrix for the rates of return of the assets in the portfolio, and we highlight the two main methods: linear and nonlinear programming.

## Linear vs nonlinear programming algorithms

Linear programming encompasses methods to achieve the best outcome in a mathematical model whose requirements are represented by **linear relationships**. On the other hand, nonlinear programming solves optimization problems where the constraints or the objective functions are nonlinear.

When it comes to finance applications, and more specifically the optimization of portfolios of traded assets, there are several situations that can be approached from both perspectives. However, the state-of-the-art of the industry suggests a linear approach, since it has demonstrated exceptional effectiveness and robustness.  It has been shown that it can, and does, successfully handle portfolio allocation problems effectively with thousands of funds and scenarios, regardless of the risk measure employed.

In addition, there are two main reasons why we prefer to focus on this approach: the computational cost, and therefore the time spent on testing, is much higher in non-linear programming, especially if we are not able to optimize our model to be ultra-efficient. This, moreover, would incur a difficulty that we prefer to avoid in order to dedicate more time to other areas of exploration within the framework of this project. Furthermore, this is a type of optimization problem that we have previously studied in our bachelor's degree in Optimization and Analytics subject, thus it is a familiar realm that we are comfortable working on.

# Risk management optimization

We will be working with a well-known one-parameter family of risk functions defined on portfolio return sample-paths, which is called conditional drawdown-at-risk (CDaR). These risk functions depend on the portfolio drawdown (underwater) curve considered in active portfolio management, and were originally proposed in *PORTFOLIO OPTIMIZATION WITH DRAWDOWN CONSTRAINTS,* a paper researched by Alexei Chekhlov, Stanislav Uryasev and Michael Zabarankin.

The CDaR family of risk functions originates from the conditional value-at-risk (CVaR) measure, as we will comment on later. The Mean-Absolute Deviation and Standard Deviation risk measures are very similar by construction – they both measure average deviation, so their efficient frontiers and transition maps will probably be very close. On the other hand, the Maximum Loss measures the extreme deviation. Thus, and as proposed in the aforementioned paper, we have two classical approaches with which to compare the performance of our specific model.

## Conditional Value-at-Risk (CVaR)

CVaR is a statistical technique used to measure the level of financial risk within an investment portfolio over a specific time frame. It derives from the value-at-risk for a portfolio. VaR allows quantifying the exposure to market risk, i.e. to estimate the loss that could be suffered under normal market conditions within a time horizon, given a confidence level (1- α) -- usually 95% or 99%.

While VaR represents a worst-case loss associated with a probability and a time horizon, CVaR is the expected loss if that worst-case threshold is crossed. CVaR, in other words, quantifies the expected losses that occur beyond the VaR breakpoint.

### VaR calculation (for just 1 fund)

Our VaR is computed using the NAV of the Prices dataframe. First, we compute the Rate of Return by subtracting each daily NAV, the directly previous NAV recorded, and divide by this same value. This generates a daily **Rate of Return vector**.

VaR depends on a risk level. Let's suppose that we want to know the VaR at a risk of 5%. Then, we need to find out the 5% percentile of the Rate of Return vector.

$$Rate\ of\ Return = \frac{NAV_i - NAV_{i-1}}{NAV_{i-1}}; being\ the\ i^{th} day$$

### CVaR calculation (for just 1 fund)

Once we have the VaR, computing the CVaR is simple, as the CVaR is obtained by taking a weighted average of the "extreme" losses in the tail of the distribution of possible returns, beyond the value-at-risk (VaR) breakpoint. In order to do so, we obtain the mean of those Rate of Return values which are smaller or equal than the VaR.

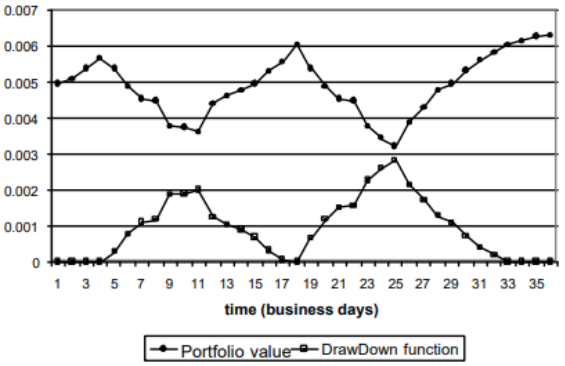
The fundamental difference between VaR and CVaR as risk measures is that VaR is the "optimistic" low bound of the losses in the tail, while CVaR gives the value of the expected losses in the tail. In risk management, we prefer to be conservative rather than optimistic.

## Conditional Drawdown-at-Risk (CDaR)

CDaR is relatively similar to CVaR, since both metrics measure the average value beyond a certain level in the distribution. Thus, similarly, CDaR is the average of all drawdowns, or cumulative losses, in excess of a certain threshold. That threshold is referred to as **drawdown-at-risk**.

Drawdown is a risk measure used to evaluate how long it typically takes an investment to stabilize its net asset value (NAV) from a temporary decline. In other words, measures the **current backward movement in the yield curve** with respect to the previous peak in the curve.

Drawdowns are inevitable in any portfolio: any liquid investment in open markets will, by the simple variation over time of its price, experience drawdowns. Drawdown-based indicators are difficult to implement in discretionary methodologies, as it is often complicated to have objective historical data to analyze drawdown, but in automated computing situations , such as ours, they can be very useful if the market tends to be more volatile.




### DaR calculation (for just 1 fund)

First, we compute the daily maximum cumulative NAV. This is a vector, in which each date it contains, has associated the maximum NAV until that day. Then, we subtract to each daily NAV its respective daily max cumulative NAV, and divide the result by this same value. Finally, we just have to find the Risk% percentile and that will be the fund DaR.

This approach reflects quite well the preferences of investors. For instance, an investor may consider it unacceptable to lose more than 10% of his investment. Another investor may excuse short-term DrawDowns in his account, but he will definitely worry in case his capital suffers a long-lasting DrawDown.

$$Drawdown = \frac{|NAV - Max\ Cumulative\ NAV|}{Max\ Cumulative\ NAV}$$

### CDaR calculation (for just 1 fund)

Once we have the DaR, computing the CDaR is straightforward. We obtain the mean of those DrawDown values which are smaller or equal than the DaR for a specific risk level.

For instance, 0.95-CDaR can be thought of as an average of 5% of the highest drawdowns.

## Mean-Absolute Deviation (MAD)

Generally speaking, the Mean-Absolute Deviation is the average distance between each data point and the mean. That is, it allows us to calculate how much the values of a set of data vary from their mean. A low value for MAD is an indicator that the data values are concentrated close together, while a high value reflects that the values are more widely scattered.

In finance it is used to measure the portfolio's volatility, and it is computed with the portfolio´s rate of return. In our case we will use the Rate of Return to compute it.

$$MAD = E[|\, r_p(\mathbf{x}) - E[r_p(\mathbf{x})]|]$$

It has been proven that portfolios on the MAD efficient frontier correspond to efficient portfolios in terms of the *second-order stochastic dominance*; i.e, for two bets A and B, bet A has *second-order stochastic dominance* over gamble B if the former is more predictable. In relation to portfolio optimization, investing in a portfolio which has second-order stochastic dominance implies making a “safer” investment.

## Maximum Loss (ML)

Maximum Loss is a method introduced for identifying the worst case in a given scenario space, called **Trust Region**. It is one of the simplest and most classic measures of risk, but at the same time it is also intuitive. However, we include it for comparative purposes, since the aforementioned Value-At-Risk indicator arose as a result of the need to improve the risk management of financial institutions on the part of regulatory bodies.

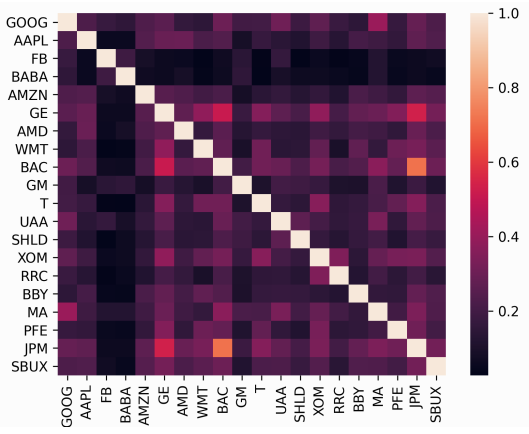
## Market-neutrality (MN):

It is known that the market itself is a risk factor. If the instruments of the portfolio are positively correlated with the market, the portfolio will follow closely the market movements.

In order to avoid a scenario in which a portfolio's assets suffer large declines in the event of a market reversal in trend, portfolios can be designed in a *market-neutral way*. This means that being moderately uncorrelated to the general market behaviour is a factor to be considered when selecting assets.

A fund's **beta parameter** measures the variability of the fund's performance compared to the variability of the benchmark's performance. In other words, it measures whether the fund is more or less volatile than its reference index. In this way, through beta, we can get an idea of the market exposure that each asset is assuming. We have already discussed this in detail previously, when we studied the composition of the dataset provided by IronIA. In order to be market-neutral, ideally,a portfolio should have a zero beta, or at least, close to zero.

Market-neutrality will be used as a constraint in the portfolio optimization problem. However, we will later raise a particular difficulty that we have encountered due to a discrepancy in the way we define the concept of "market neutrality" and how some Python libraries approach it.




The idea is to obtain a result with as **uncorrelated** as possible funds that should show a similar covariance matrix as the one shown here as example. Note that this covariance plot is made using individual stocks. Instead, we must study the covariance matrix of funds, which may be different, as they may contain similar stocks that perform correlatedly.

## Inherited assumptions

To design our model following a rational and well-founded process, we have considered all the assumptions derived from **classical portfolio theory,** including:

- We expect investors to be rational and all have access to the same information.
- All are risk-averse and share the goal of maximizing returns.
- We expect no investor to be able to influence the market.
- All of them base all their decisions in the market on expected returns according to some measures of risk.

In addition, we consider **modern nuances** such as market players having access to unlimited funds at a risk-free rate, as well as assumptions inherited from the **aforementioned paper**, which we detail in depth as being the most relevant:

### Liquidity

Liquidity considerations are not taken into account. We define an asset in terms of its expected return and risk, but liquidity should be taken for granted, as we are implicitly assuming assets are listed on a global-scale, liquid market. Indeed, this is the case for the funds that IronIA offers.

### Transaction costs

Transaction costs are the trading costs of changing the weights of portfolio elements. Since the optimal portfolio changes over time, there is an incentive to re-optimize frequently. This variable is doubly neglected: because of the assumptions inherited from the model we are trying to replicate and because our partner IronIA Fintech does not charge transaction fees to its clients.

### Investing restrictions

A country's law may prohibit certain investors from owning some assets. Sometimes, it is not practical to hold an asset because the associated tax cost is too high. However, as Spain is not a country with restrictive legislation in terms of the profile of investors allowed access to financial products, and the tax burden of the State is not a point we particularly want to delve into in this project, we will assume that there are no investing restrictions.

### Credit and non-reflected risks

Credit and other risks which directly are not reflected in the historical return data are not taken into account. **Credit Risk** is the possibility of a loss resulting from a borrower's failure to repay a loan or meet contractual obligations.

### Survivorship bias

In finance, survivorship bias is the tendency for failed companies to be excluded from performance studies because they no longer exist. In our model, it is not considered.

## Core approach

It is assumed that we have '***N'*** funds available in order to generate our optimal portfolio. The general idea is to maximize the expected return of the portfolio subject to different operating, trading and risk constraints. Let's go one by one.

First, the **Objective function** which represents the expected return of the portfolio. $x_i$ is the position of asset (fund) $i$ in the portfolio. $r_i$ is the rate of return of asset $i$.

$$\max_x \; E\left[\sum_{i=1}^{n} r_i x_i\right]$$

This objective function is subject to four different constraints. These are:

### Fund limitation

We limit the number of funds which can be used $(i = 1,\ldots, n)$and also, the weight of the fund must be a value in the interval [0,1] $(0 \leq x_i \leq 1)$.

### Budget constraint

It ensures that the sum of the fund weights does not surpass one.

$$\sum_{i=1}^{n} x_i \leq 1,$$

### Risk of financial loss

In this constraint we can use one of the four previously mentioned methodologies (CVaR,CDaR,MAD or MaxLoss). For instance, let´s work with the CVaR methodology. In this case, we will compute the CVaR function over the selected fund´s weights and upbound this resulting value with a risk tolerance level (this is the fraction of the portfolio value that is allowed for risk exposure). $\Phi_{RISK}(x_1,\ldots, x_n) \leq \omega$

### Market neutrality

This constraint also controls the risk of financial losses. In this constraint we force the portfolio to be market-neutral. We bound the portfolio´s correlation with the market, and as a result the portfolio won't follow significant market drops. $\beta_i$ represents market´s beta for the fund $i$ . $k$ is a small number to ensure that the beta sum remains close to zero.

In the paper, they follow two approaches. Using this last constraint or not using it. They anticipate that using this constraint significantly improves the out-of-sample performance of the algorithm.

## Linearization

The problem with these four risk constraints is that they aren't linear (either due to maximum functions or absolute values). Thus we have to linearize them in order to be able to program them in our optimization modeling language (we will use Pyomo) . In the below images you will see their linear to non-linear transformations. We have obtained the linearizations from the paper, except from the **CDaR linearization**, which we have obtained through our own calculations.

### CVaR calculation (for the Portfolio)

$$\zeta+\frac{1}{(1-\alpha)J}\sum_{j=1}^{J}\max\left\{0,-\sum_{i=1}^{n}r_{ij}x_i-\zeta\right\}\le\omega \longrightarrow \begin{cases} \zeta+\frac{1}{1-\alpha}\frac{1}{J}\sum_{j=1}^{J}w_j\le\omega, \\ -\sum_{i=1}^{n}r_{ij}x_i-\zeta\le w_j,\ j=1,\dots,J, \\ \zeta\in\mathbb{R},\quad w_j\ge 0,\quad j=1,\dots,J. \end{cases}$$

### CDaR calculation (for the Portfolio)

$$\eta+\frac{1}{1-\alpha}\frac{1}{J}\sum_{j=1}^{J}\max\left[0,\max_{1\le k\le j}\left\{\sum_{i=1}^{n}\left(\sum_{s=1}^{k}r_{is}\right)x_i\right\}-\sum_{i=1}^{n}\left(\sum_{s=1}^{j}r_{is}\right)x_i-\eta\right]\le\omega, \longrightarrow \begin{cases} \eta+\frac{1}{1-\alpha}\frac{1}{J}\sum_{j=1}^{J}w_j\ \le\ \omega\ ; \\ Z-\sum_{i=1}^{n}\left(\sum_{s=1}^{j}r_{is}\right)x_i-\eta\le\ w_j\quad j=1,\dots,J\ ; \\ \sum_{i=1}^{n}\left(\sum_{s=1}^{K}r_{is}\right)x_i\le\ Z\quad K=1,\dots,j\ ; \\ \eta\in\mathbb{R},\ w_j\ \ge 0\ \ j=1,\dots,J. \end{cases}$$

### MAD calculation (for the Portfolio)

$$\frac{1}{J}\sum_{j=1}^{J}\left|\sum_{i=1}^{n}r_{ij}x_i-\frac{1}{J}\sum_{k=1}^{J}\sum_{i=1}^{n}r_{ik}x_i\right|\le\omega \longrightarrow \begin{cases} \frac{1}{J}\sum_{j=1}^{J}(u_j^{+}+u_j^{-})\le\omega, \\ \sum_{i=1}^{n}r_{ij}x_i-\frac{1}{J}\sum_{j=1}^{J}\sum_{i=1}^{n}r_{ij}x_i=u_j^{+}-u_j^{-},\ j=1,\dots,J, \\ u_j^{\pm}\ge 0,\ j=1,\dots,J. \end{cases}$$

### MaxLoss calculation (for the Portfolio)

$$\max_{1\le j\le J}\left\{-\sum_{i=1}^{n}r_{ij}x_i\right\}\le\omega \longrightarrow \begin{cases} w\le\omega, \\ -\sum_{i=1}^{n}r_{ij}x_i\le w,\ j=1,\dots,J. \end{cases}$$

# Technical methodology

## PyPortfolioOpt library

This is a python library that implements portfolio optimization methods. With it we have been able to develop the approaches of CVaR and CDaR. It relies on two main design principles: it should be easy to swap out individual components of the optimization process with the user's proprietary improvements and that it is better to be self-explanatory than consistent. They present some advantages over existing implementations:

- Easy to combine with our strategies and models.
- Includes both classical methods (Markowitz 1952 and Black-Litterman), suggested best practices (e.g covariance shrinkage), along with many recent developments and novel features, like L2 regularisation, exponential covariance, hierarchical risk parity.
- Provides native support for pandas dataframes.
- It is quite robust to missing data, and price-series of different length.

### Usability

In order to solve this specific problem is a quite good library in order to start playing with the data we have.

It uses as input data the mean historical return (mean of the returns of each fund along the days), the returns covariance matrix, the risk level that the client is willing to take and the amount of money the client is willing to invest.

Also it includes the option of imposing an L2 regularization term. By doing this, the optimization algorithm spreads more the weight values, creating a portfolio with more funds. The amount of regularization can be modified using the gamma parameter.

The method will give us as output the weights given to each of the funds, the amount of money inverted on each of the funds, the expected annual return (in percentage) and the remaining amount of money that the client will keep after investing (since the algorithm may determine that investing all the budget isn't necessary) .

### Limitations

In addition to that it does not allow us to implement the optimization algorithm with the MAD and MaxLoss constraints, it also limitates us in the Market Neutral constraint. This library enforces the portfolio to be market neutral by making the sum of its fund´s weights equal to zero. In order to achieve this, they modify the weight range of values to [-1,1]. Thus now weights can be negative (a negative weight would mean to perform short selling for that specific fund). Performing short selling can be a riskier activity, and since we want to have control over the risk of our portfolio it isn't a good approach to use the market neutrality provided by *PyPortfolioOpt*.

## Pyomo library

Pyomo is a Python-based software commonly used for formulating, solving, and analyzing optimization models. We have decided to use this software for the formulation and solving of our Linear Problem due to the versatility and easy use it offers. Pyomo has a notation similar to what we would use in the mathematical definition of these problems.
With it, we have been able to program the linear problem from scratch. The most challenging part was to program the 4 risk constraints in an efficient way, since some of the constraints took very long time to be executed because of their linear formulas (i.e. MAD, CDaR).

## Experiment

In this subsection we will compare the results obtained with the two approaches. Also, we want to make clear which is the best risk measure (CVaR, CDaR, MAD or MaxLoss) and check to what extent adding the market neutral constraint improves the portfolios. To carry it out, tests have been performed in different environments: using the **CPLEX** solver, locally, and **ECOS-BB** solver in the preliminary tests carried out with Google Collaboratory.

It is important to note that we have noticed that on many occasions, the result of the optimization yields a very low number of funds, i.e. the vast majority of weights allocation is 0. This is a problem, mainly due to lack of diversification. We decided to choose a diversification strategy for several financial reasons:

1. **Mitigates investment risk**. Spreading capital over several instruments allows offsetting losses on some assets with gains on others.
2. **Allows to set several investment objectives**. In case of further customization of these portfolios, in the future, investing in instruments of various maturities facilitates the achievement of profitability goals.
3. **Allows for currency hedging**. This will potentially improve returns and allow hedging against exchange rate fluctuations.

We can illustrate this situation with the following plot, where we show the variance using only 1 fund, compared to a portfolio of 30 optimized funds. Although it is true that we could obtain more gains at some moments, we prefer the stability of a portfolio to the risk of putting all eggs in the same basket.

To solve this problem, we use a built-in objective function which borrows the idea of **regularization** from Machine Learning. The loss of precision needs to be accounted for by something else to maintain accuracy levels. This trade-off will be borne by the bias portion of the model equation, being the bias the part of the model equation that does not depend on the feature data.

In order to coerce the optimizer to produce more non-negligible weights, *pyportfolioopt* library adds what can be thought of as a "small weights penalty" to all of the objective functions, parameterised by $\gamma$ **(gamma)**. This is the parameter we fine-tuned.

Considering, for example the minimum variance objective for instance, the library performs as follows:

$$\text{minimize } w\ \{wT\Sigma w\} \ \longrightarrow\ \text{minimize } w\ \{wT\Sigma w+\gamma wTw\}$$

After applying regularization, we set our risk measure and our risk value. With those values, we pass our train set but keeping just 500 funds.

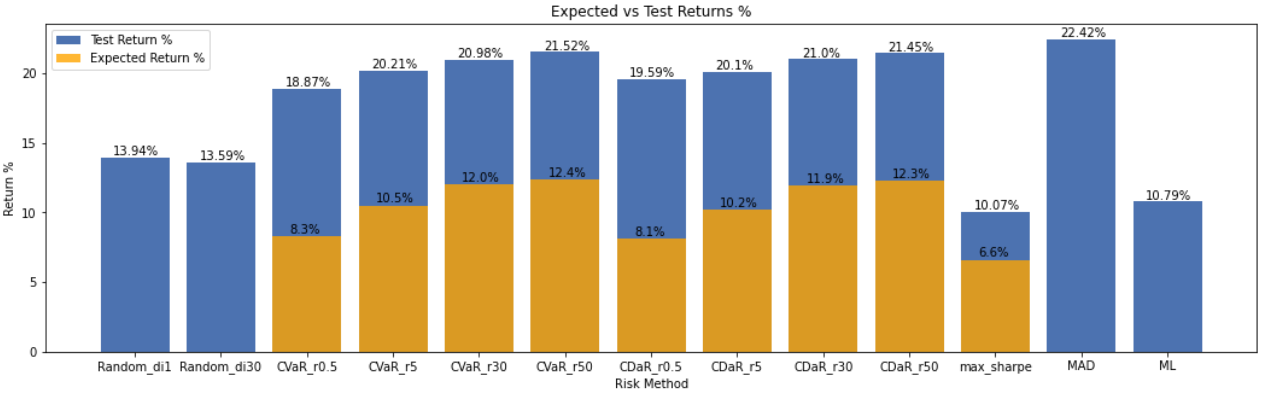
The reason why we need to limit the selection of funds to only 500 funds is because, when the number of funds is increased, due to the L2 regularization, **the problem becomes non-linear**, therefore, not solvable.

In further expansions of this approach we will be trying different techniques involving hierarchical computation methods and sharding to solve this problem, thus making our model more accurate and scalable.

## Results

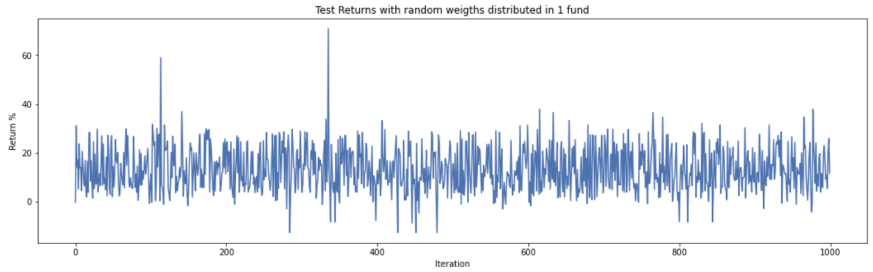
With this initial procedure, we are getting a set of optimal efficient portfolios using the weights obtained in the train (years 2016 to 2018) and evaluating them on the test (year 2019). In the following figure, we compare the different approaches proposed. For a correct analysis, it is necessary to take certain considerations into account:

- **Test Return** is the return that the optimal portfolio, for each risk measure to be optimized, would have offered to an investor in 2019.
- **Expected Return** is the return that the *pyportfolioopt* library model predicts that an optimal portfolio could achieve, for each risk measure to be optimized, also in the year 2019, without knowing the ground-truth.
- The models called **Random** are a comparative benchmark that simply provides an approximation to the average return of random funds selected with random weights. Basically, it measures the expected return for a certain random sample. It serves for comparison purposes.
- Some risk methods do not show any expected return. On the one hand, random models are not provided with one, due to their relative triviality. On the other hand, MAD and ML approaches were solved using pyomo library, which in our current implementation isn't able to provide this comparative additional information.
- The notation '**_r*X***' refers to the *X*% risk tolerance that is associated with that optimal portfolio. For random portfolios, '**_di1**' means that only 1 fund is selected, while '**_di30**' is the random allocation for a 30-fund portfolio.




The first noteworthy finding is that the optimal efficient portfolios obtained through our comparative models **outperform the returns that could be expected from them**. (Test Returns, in blue, over Expected Returns, in yellow). At least, this would have been the case during 2019. This, we believe, is due to the particular choice of risk measures chosen in the scope of this project; we have fortuitously targeted a segment of financial market activity where value and drawdown at-risk have turned out to be a more representative metric than perhaps might be expected.

Within this framework, the family of special interest was the drawdown-related. The CDaR-optimized portfolios outperformed the CVaRs in the **0.5%** and **30%** risk categories, while the opposite happens in the **5%** and **50%** risk categories, leaving us with almost even results. Compared to the rest, our "conditional-at-risk" family portfolios **outperform** those optimized for **2 of the 3 classical comparison metrics** (Sharpe Ratio and ML) and **all randomly selected ones**. They are only bested by those where MAD is used, which is satisfactory enough for us.




```
Mean: 13.375350000000001
Variance: 86.3397482775
Min value: -12.65
Max value: 70.86
95% Quantile: 28.4
75% Quantile: 20.58
25% Quantile: 6.3125
5% Quantile: 0.5835000000000004
```


```
Mean: 13.6053
Variance: 3.88936191
Min value: 7.48
Max value: 19.66
95% Quantile: 16.83
75% Quantile: 14.9925
25% Quantile: 12.21
5% Quantile: 10.4795
```

To conclude this analysis, we delve a little deeper into the experiment performed with the random selection of portfolios with 1 and 30 funds. As the number of iterations is large, we observe similar means, but we know that this is not a real market situation. We can observe especially the volatility derived from a 1-asset portfolio, while the range in which potential gains and losses oscillate is much narrower in a diversified portfolio ( **[7.48% - 19.66%]** vs **[(-12.65) - 70.86]** ) . This confirms our previous assumption that it is safer to have a diversified portfolio, even if we sacrifice a potential triple return.

## Next steps

As this is an intermediate delivery, changes in the actual content and format should be expected, providing relevant details as well as additional perspectives and conclusions along with, of course, references that back up many of the claims and assumptions made in this research. Additionally, we are considering new improvements of the project scope.

### Market scenarios characterization and clustering

Following the conclusions inferred from the results presented, we have focused our attention on the fact that, for this market situation, in 2019, our models outperform what would be expected of them. For this reason, we are particularly interested in studying how the portfolio portfolio would evolve according to the type of market. Therefore, one of our next objectives is to identify and define different market contexts, characterizing these stages through metrics that represent their state.

Ideally, segmentation criteria will be established and a "market status profile" will be designed for different situations, with data collected from reliable sources. Our portfolio optimization models will be applied to these different scenarios, allowing us to be prepared to rebalance portfolios in the event of changes from one market paradigm to another. The way to analyze if these changes occur will be by clustering the current market situation with the ones we have predefined.

### Hierarchical computation

As previously mentioned, we will be trying different techniques involving hierarchical computation methods to refine our current models, so that we can try to optimize with more than 500 portfolios. This will give consistency to our project and help us to speed up obtaining more reliable results.

### Follow risk metrics evolution

Also, it is among our ideas for potential implementations to perform a broader analysis on the risk measures used, monitoring their flow of variation in terms of market impact, to determine whether in other market situations they could invalidate our results or whether, on the contrary, they are resilient to the natural evolution of markets.