Combining Support Vector Machine with Genetic Algorithms to Optimize Investments in Forex Markets with High Leverage

Bernardo Martins Paiva Jubert de Almeida

Thesis to obtain the Master of Science Degree in

Electrical and Computer Engineering

Supervisor: Prof. Rui Fuentecilla Maia Ferreira Neves

Prof. Nuno Cavaco Gomes Horta

Examination Committee

Chairperson: Prof. Paulo Ferreira Godinho Flores

Supervisor: Prof. Rui Fuentecilla Maia Ferreira Neves

Members of the Committee: Prof. Agostinho Cláudio da Rosa

November de 2016

Abstract

This work proposes a new approach, based on Genetic Algorithms and Support Vector Machine to trade in the forex market. In this work, a new algorithm capable of generating technical rules to make investments with a given amount of leverage depending on the certainty of the prediction is presented. To forecast those predictions, a combination of a Support Vector Machine (SVM) algorithm – to identify and classify the market in three different stages –, and a Dynamic Genetic Algorithm – to optimize trading rules in each type of market, is used. The optimization of the trading rules is based on several technical indicators. Forex data for the EUR/USD currency pair, in a timeframe between the years of 2003 and 2016, is used as training and test data. The proposed architecture for the machine learning system, as well as the implementation and study of the proposed system is described in detail. The work shows promising results during the test period between the 2nd of January of 2015 until the 2nd of March of 2016, where the *Return on Investment* obtained is 83%.

Keywords

Leverage, Genetic Algorithms, Forex, Support Vector Machine

Resumo

Este trabalho propõe uma nova abordagem, baseando-se em Algoritmos Evolutivos, mais precisamente, Algoritmos Genéticos e *Support Vector Machine* para transacionar em mercados Forex. Neste trabalho, o objetivo é desenvolver um algoritmo capaz de prever regras técnicas de modo a realizar investimentos com um dado *ratio* de alavancagem (*leverage),* dependendo do nível de certeza da previsão. Utiliza-se uma combinação de algoritmos -*Support Vector Machine* – para identificar e classificar o mercado em três tipos diferentes-, um módulo de Algoritmos Genéticos Dinâmicos– de forma a otimizar as regras de investimento, baseando-se em diversos indicadores técnicos. O par de moeda utilizado no Mercado *Forex* para os períodos de treino e de teste, será o EUR/USD, com uma janela temporal entre 2003 e 2016. Neste trabalho, serão explicados conceitos fundamentais acerca do assunto em questão, como também, a arquitetura proposta do Sistema, em conjunto com a implementação do sistema proposto e as devidas conclusões. O trabalho realizado demonstra resultados promissores durante o período de teste entre 2 de Janeiro de 2015 até 2 de Março de 2016, onde o Retorno obtido foi de 83%.

Palavras-Chave

Alavancagem, Algoritmos Genéticos, Mercados Forex, Support Vector Machine

Acknowledgements

To my supervisors, Professor Rui Neves e Professor Nuno Horta, I would like to express my gratitude for leading this work with excellence, providing the background necessary to develop this thesis and always supporting me when most needed, encouraging and always pushing me further, to develop new ideas that are of great importance in this work. To both my professors, a big thank you, for helping me and to provide me with the most challenging and satisfying semester that I have ever had in IST.

To Gustavo and Gonçalo, for teaching me *Python* and helping me to develop better programing skills, without your help, my work would be hundreds of times harder.

To my family, many apologies for all the family lunches and dinners that I did not attend and for this past few months that I've been missing. This is my retribution to you, my thesis, the final chapter of these five years. For all the education, dedication and care that you gave me, for all the support and help one could ask, all I can say is thank you, without you, none of this would have been possible.

To Teresa, for her patience and dedication, for helping me in times of dissuasion, and for being by my side at all times.

To my friends that worked with me through these five years in MEEC, for showing me what team work is really about, for all the days and nights spent in this institution and for all the stories that I will always remember, thank you for making the IST experience much more fun.

Finally, i would like to express my kind regards to all the reviewers and investigators that work so hard to publish and to present such important contributions to science.

Table of Contents

List of Figures

List of Tables

List of Acronyms

AMSO	Adaptive Multi-Swarm Optimiser
ARBF-PSO	Adaptive Radial Basis Functions with Particle Swarm Optimization
ARIMA	Autoregressive Integrated Moving Average
AUD/USD	Currency pair – Australian Dollar/United States Dollar
B&H	Buy and Hold
BIC	Bayesian Information Criterion
CPSO	Clustering Particle Swarm Optimization
cSVR	Correlation-Aided Support Vector Regression
CV	Cross-Validation
DB	Database
DGA	Dynamic Genetic Algorithm
DOP	Dynamic Optimization Problem
DPGA	Dual Population Genetic Algorithm
DynDE	Dynamic Differential Evolution
EA	Evolutionary Algorithm
EC	Evolutionary Computation
EIGA	Elitism –based Immigrants Genetic Algorithm
EMA	Exponential Moving Average
EUR/USD	Currency pair - Euro /United States Dollar
FNN	Fuzzy Neural Network
FX	Foreign Exchange
GA	Genetic Algorithm
GA-SVM	Hybrid Genetic Algorithm with Support Vector Machine
GA-SVR	Hybrid Genetic Algorithm with Support Vector Regression
GDBG	Generalized DOP Benchmark Generator
Gen	Generation
GPB/USD	Currency pair – Great Britain Pound/United States Dollar
gRatio	Tracking ratio for the global optimum
HMM	Hidden Markov Models
HONN	Higher Order Neural Network
IMGA	Island Model Genetic Algorithm
ISGA	Immune System-based Genetic Algorithm
JPY/USD	Currency pair – Japanese Yen/United States Dollar
MACD	Moving Average Convergence Divergence
MEGA	Memory Enhanced Genetic Algorithm

MIGA	Memory-based Immigrants Genetic Algorithm
MIT	Massachusetts Institute of Technology
MKSVR	Multiple Kernel Support Vector Regression
ML	Machine Learning
MLP	Multi-Layer Perceptron
MPBIL	Memory enhanced Population Based Incremental Learning
MPBILi	Memory enhanced Population Based Incremental Learning with random immigrants
MR	Mutation Rate
MSGA	Memory-Search Genetic Algorithm
NASDAQ	American Index
NAV	Net Assets Value
NIKKEI	Japanese Index
NN	Neural Network
NZD/AUD	Currency pair – New Zealand Dollar/Australian Dollar
NZD/EUR	Currency pair – New Zealand Dollar/EURO
NZD/GBP	Currency pair – New Zealand Dollar/Great Britain Pound
NZD/JPY	Currency pair – New Zealand Dollar/Japanese Yen
NZD/USD	Currency pair – New Zealand Dollar/United States Dollar
PBIL	Population Based Incremental Learning
PDGA	Primal Dual Genetic Algorithm
Pop	Population
Pop-PSC	Population-Proposed Stopping Criteria
PSC	Proposed Stopping Criteria
RBF *v*-SVR	Radial Basis Function Kernel with Fine-tune Support Vector Regression
RG-SVR	Rolling Genetic Algorithm-Support Vector Regression
RNN	Recurrent Neural Network
ROC	Rate of Change
ROI	Return On Investment
RSI	Relative Strength Index
SAMO	Self-Adaptive Multi-Swarm Optimiser
SGA	Static Genetic Algorithm
SKSVR	Single Kernel Support Vector Regression
SMA	Simple Moving Average
SMO	Sequential Minimal Optimization
SPBIL	Standard Population Based Incremental Learning
SPBILi	Standard Population Based Incremental Learning with random immigrants
SVM	Support Vector Machine
SVR	Support Vector Regression
S&H	Sell and Hold

tPercent	Percentage of Peaks being tracked
USD/CAD	Currency pair – United States Dollar/Canadian Dollar
USD/CHF	Currency pair – United States Dollar/Swiss Franc
USD/JPY	Currency pair – United States Dollar/Japanese Yen
XOR DOP	Exclusive OR DOP
WMA	Weighted Moving Average

1. Introduction

This first chapter is an overview of the subject, introducing some historical facts about Evolutionary Computation and an introduction about Genetic Algorithms as well as Support Vector Machine Algorithms. It is also introduced the basic concepts about Forex Markets. Section 1.2 presents the motivation of this work and section 1.3 the objectives that are intended with this thesis, finally the document structure is presented in section 1.4.

1.1 Computational Techniques, An Overview

The study of modern evolution started of with Darwin's Theory of natural selection, and the remarkable power of natural evolution throughout times. First attempts to mimic natural selection and the natural evolutionary process of Nature were made in the 1950's, being unsuccessful, mainly because, it relied too much on the mutation operator, that was very emphasized by biology back in the day. Notable computer scientists like Alan Turing, Norbert Wiener and many others were motivated to introduce intelligence and the ability to self-replicate, with adaptive capabilities to learn and control new environments, onto computers [1]. During this time, scientists were rather interested in biology, psychology and human behaviour, motivating the first attempts to simulate natural evolution. The first field of studies being developed was neural networks, followed by machine learning and lastly, the study of genetic algorithms [1]. *GAs* were first described by John Holland in the mid 1960's, and they were inspired by mimic the simple mechanism of basic biological evolution and molecular genetics [2], and the selection of the fittest individuals from a given population [3]. Holland's book "*Adaptation in Natural and Artificial Systems"* [4] presented the *GA* as an abstraction of biological evolution and gave a theoretical framework for adaptation under the *GA* [1]. Holland had successfully added operators of crossover and mutation and was able to create the first genetic algorithm. Since then, many others have created several applications of Holland's original algorithm and applied it in many different areas of expertize [5].

*GA*s are random heuristic search algorithms that provide a wide range of applications in adaptation and optimization problems [2]. The principle behind this subject is based on a population of individuals (chromosomes) that evolves accordingly to a specific type of selection and genetic operators, such as, selection, crossover and mutation. Each individual is represented by a set of parameters, regarded as genes of a chromosome. In this work, the parameters will be represented through an array that composes the chromosome. The selection will choose the best chromosomes in the population to reproduce, generating, during the process of evolution, new individuals, that are subjected to the mutation operator like any other natural process. The individuals that present a better fitness value have the tendency to produce more and better chromosomes for the next generation than the less fit ones, which means a better solution to the problem and ensures better survival chances of the next generation [3]. The fitness value shows how "good" each chromosome is. This value is given by the "Fitness Function" that is directly correlated with the problem in hand. Another important aspect

on this subject is the diversity introduced on the *Genetic Algorithm*. Diversity is one of the main reasons why every species got to where they are now. Diversity prevents the algorithm from being locked on "Local Maximums" during the evolutionary process, and therefore prohibiting the species to evolve. Further on this work will be explained the methods used to solve the diversity problem.

Traditional genetic algorithms are good for static problems, i.e., problems that do not change through time and remain in the same environment. Real world problems are constantly changing, making species to adapt, and the ones who adapt faster survive, but, what happens when the environment changes? Is it better to restart the algorithm? This problem can be solve using Dynamic Genetic Algorithms that combine a set of tools to predict and better adapt to new environments, giving new solutions without restarting the whole process.

Financial Markets are in a constant change, making *DGA*s and *SVM* algorithms good candidates to predict outcomes and making better investments in the future. For different types of markets, it is mandatory to use different investment strategies in order to succeed. If the market trend goes down, then it does not make any sense to use the same strategy as if the market trend was to go up. It is important to well define each strategy independently before training the GA. In order to do that, this work relies on Support Vector Machine algorithms to identify the market. SVM algorithms can work as classifier algorithms and were created by Vladimir Vapnick in the mid 1960's, in Soviet Union, but the world only knew about Vapnick's brilliant approach in the early 1990's, when Bell Labs discovered Vapnick, and brought him to the United States. This approach became famous when Vapnick solve a hand written recognition problem for Bell Labs, beating approaches like Neural Networks. SVM are very powerful classifiers, that with a specific set of parameters (technical indicators and the currency pair's charts), can distinguish and classify data into different categories.

Nowadays, there are many applications for Evolutionary Algorithms, such as, network management, scheduling, criminal identification, computer chip manufacturing, logistics, and of course, the focus of this work, financial markets [5]. Evolutionary Algorithms, like genetic algorithms, neural networks can also be applied to stock market, portfolio optimization, Foreign Exchange markets for currency trading, database mining and many more. Like stated before, this work's main focus is Foreign Exchange markets using a Genetic Algorithm to explore the possibilities of financial leverage relying on SVM to market identification on Forex. According to the Bank for International Settlements, trading in Forex markets averaged $5.3 trillion per day in April 2013 [6]. One of the great advantages of this kind of markets, compared, for example, with the stock market, is that, they do not require an agreement time, meaning, the investors can trade foreign currencies at current rate almost immediately for major currencies [7].

1.2 Motivation

The Foreign Exchange (*FX*) Market is a global market for currency trading, it is considered the most liquid financial market in the world. This financial problematic is a widely studied subject, and constitutes a rather motivating problem to researchers on the *Machine Learning* field of studies. In this work, more precisely, it is intended to develop an investment strategy, using different "Leverage ratios", applied to Foreign Exchange Markets, or more specifically on FOREX markets. The work is motivated by the study of techniques within the area of Evolutionary Computation, that are capable of learning past events in order to predict future ones, and generate investment rules accordingly, thus, potentiating the quality and the profits of an investment made by a human. Said that, it is intended to study different approaches using *Genetic Algorithms* (*GA*) to optimize investment rules and *Support Vector Machine* (*SVM*) to classify the different types of markets. The FX index is a very oscillating but at the same time is a very interesting market to the study of *GAs*, mainly because, the problem of understanding trends and patterns in the Index can be solved by, a rigorous technical analysis, supported by a genetic algorithm, rather than a fundamental analysis that is based on human decision. To obtain high returns in this type of markets it is essential to apply a good strategy of leverage. Financial leverage it's a very common tool used in financial markets, and its purpose is to increase the return of an investment, being the losses equally great.

1.3 Objectives

With the use of Genetic Algorithms, it is intended to develop an investment strategy based on Artificial Intelligence that allows to obtain, with leverage, high returns. Due to leverage, it is possible to make high investments with a low investment budget. Nonetheless this strategy has a great measure of risk associated, with great potential losses. This thesis is motived by, creating a software that can study the behaviour of EUR/USD currency in *FOREX*, based on a *GA* that can predict its trends and subsequently applying the best strategy of investment with the use of leverage to obtain the highest returns. It is also motivated by the development of a market identifier based on a *SVM*, to split the market in different categories, in order to train the *GA* more accurately. More precisely, the objectives of this thesis are:

- Using Technical Indicators to study future prices of EUR/USD currency;
- Using Price Sequences to study future prices of EUR/USD currency;
- Explore SVM to identify market strategy;
- Introducing new approaches to find the best application of Leverage;
- Explore different approaches in static Genetic Algorithms regarding the FX market;
- Explore new approaches using Dynamic Genetic Algorithms to solve dynamic problems;

The work developed in this thesis, had the following contributions:

- Create a classifier model based on a Support Vector Machine algorithm, that classifies three types of markets to identify the environment used for three different Genetic Algorithms that are trained in each one of the environments;
- Produce Dynamic Genetic Algorithm that introduces adaptability approaches and takes advantage of a memory system to save the populations of each GA, together with the environment information;
- Introduce a new leverage solution based on optimized values from the Genetic Algorithm chromosome, together with the study of the evolution of the price values of Forex market.

1.4 Document Structure

This thesis is composed by five chapters. On Chapter 1 – Introduction-, where was shown a brief historical background and a brief introduction about *GAs*, *SVMs* and the Forex Market. On Chapter 2 – Related Work-, explains some fundamental concepts related to financial markets used, and the methodology already addressed to the present problem, showing some existing solutions in this field of studies. Following, Chapter 3 – Proposed System Architecture -, describes the proposed architecture of the developed system, and the implementation of the theoretical model used on the matter; Chapter 4 – System Validation – describes the several case studies made, showing result analysis and the solutions obtained. Finally, chapter 5 – Conclusion- summarizes the respective work done and the following conclusions about the results obtained and the approaches used, together with several future work propositions.

2. Background and State-of-the-Art

This chapter provides, fundamental concepts about Foreign Exchange Markets, Technical Indicators, and a description of important methodologies addressed to Genetic Algorithms and SVM, focussing on related work on several areas within GAs and some Forex works related to SVM, to better understand the problem and the proposed solution.

2.1 Background

This sub-section describes the fundamental concepts related to financial knowledge and the fundamental concepts of genetic algorithms and support vector machine algorithms.

2.1.1 Financial Concepts

In this sub-chapter it is given some background about Financial Concepts, such as, Forex Index and how it works, Technical Analysis and finally, the main focus of the financial application of this thesis, the concept of leverage and how it can be used.

a) Introduction to Forex

The Forex is a largely well known Foreign Exchange Index, and the approach developed in this work will be applied to this specific market. Forex has seven major currencies that are traded on a daily basis, 24 hours per day, from Monday to Friday. Those currencies are: EUR/USD, GPB/USD, USD/JPY, USD/CHF, USD/CAD, AUD/USD and NZD/USD. It is also possible to trade goods like oil, corn and others. The currencies are called *Cross Currency Pairs*, being the currency from the left called the *base currency* or the *selling currency* and the one from the right the *counter currency* or the *buying currency*, which means, when the trade is on EUR/USD for 1.2, for every 1.2 US Dollars bought, it is automatically sold 1 Euro. Currency quotes can be divided into two parts, the first one is the *ask price* for buying base currency and *bid price* to sell base currency. The currency quotes will be especially important to decide whether you adopt a *Short Position* or a *Long Position* meaning that, if it is profitable to sell your base currency when the market drops (Short Selling), or the other way around, profitable to buy base currency when the market is rising. Another important concept is the *Spread,* not only because this is the cost of each transaction, but this parameter has great weight when it comes to calculate profits and to measure a more realistic prediction of the market behaviour. *Spread* can be calculated as *Bid Price* Minus *Ask Price*. Finally, the profits/losses of currency pairs are calculated through “pips”, that is, smallest amount by which a currency can change (1 pip = 0,0001).

b) Technical Analysis

There are two types of analysis on any type of financial market, i.e., *Fundamental Analysis and Technical Analysis.* In this work the objective is to focus only on the second approach. Fundamental analysis focuses on the economics, the synergy between supply and demand and makes an intensive study about the market factors that cause prices to rise or fall in order to determine its true market price. Technical Analysis is the tool required to analyse future prices of any financial asset and different trends of the market. This is all made with technical indicators or technical indexes that use past variations of assets price to predict future ones [7]. Technical indicators can, some times, be deceiving, meaning that, there is not a better indicator. Gorgulho et al., in "*Applying a GA kernel on optimizing technical analysis rules for stock picking and portfolio composition, 2011"* stated that: "*Indicators should be combined in order to offer us different perspectives*" [8], i.e., a technical indicator cannot be used alone, it has to be a combination of technical indicators that will find the best solution.

It is possible to define different types of technical indicators according to their oscillatory behaviour, e.g., *Trend Following and Momentum Oscillators.* While *Trend Following* indicators are used to understand trends i.e., they work better to understand the present trend in order to learn about the assets behaviour, *Momentum Oscillators* predict sudden changes on the assets behaviour, meaning that *Momentum Oscillators* can keep track with the speed of the price movement variation and are able to measure the strength and speed of the price trend direction[8]. *Trend Following* indicators are of great importance to identify entry or exit points, i.e., these indicators identify if the cycle of a trend has begun (entry point when asset's price is rising), or ended (exit point when asset's price starts to decrease). *Momentum Oscillators* can have major importance to decide the amount of leverage used in the investment [9]. In terms of applying technical indicators to the *GA*, it is most likely to have better performance, by adding more technical indicators, making the confirmation of possible buy or sell more accurate[8]. A wide set of technical indicators can be used, some can be very effective as well as others, can lead to wrong conclusions. The question to be made is one, how is the choice of Technical Indicators made? As Robert W. Colby states in his book, "*Encyclopedia of Technical Market Indicators*", "*We can find specific decision rules that would have maximized profit and minimized risk of significant loss in the past. By testing our ideas in all kinds of market conditions, we can uncover the vitally important information we need about the nature of markets and trading methods."* [9]. Table 1, describes some of the main Technical Indicators used in the market.

Table 1– Example of Technical Indicators

Technical Indicator	Method	Definition and Technical Rule	Type of indicator
RSI – *Relative Strength Index*	$\frac{\lvert R\rvert}{\lvert F\rvert}\times 100$	R- Sum of the absolute value of rising width in the past n days. F – Sum of the absolute value of rising and falling width in the past n days. Aims to buy when currency is sold too much, i.e., the price *i* slow, and to sell when it's bought too much, i.e., the price is high. The value of n used is between 9 and 14 days [7].	Momentum Oscillator
ROC – Rate of Change	$\frac{Xt - Xt - n}{Xt - n}\times 100$	Ratio between the current closing price and the price of the last n time periods. This ratio measures how fast, the price of the asset is changing. When the price is rising to quickly indicates that there is a great possibility of overbought conditions. When the price is falling to rapidly it is possible that the conditions are oversold.	*Momentum Oscillator*
MACD – Moving Average Convergence/ Divergence	$\mathrm{MACDt(s,l)} = \mathrm{EMA\ (s)_t - EMA(l)_t}$ $Triggert(n) = EMA(n)\ of\ MACDt(s,l)$ $Histogram = MACD(s,l) - Triggert(n)$	MACD is the difference between two EMA's where n is the number of periods considered for the trigger signal; s is the number of periods considered for the shorter EMA (12 weeks) and t is the number of periods considered for the longer EMA (26 weeks) [8]. MACD indicates a trend turning point. When the line of the MACD crosses the 0 value to the positive side indicates a "Buying Signal", when the line of the MACD crosses the 0 value to the negative side indicates a "Selling Signal".	Trend/ Momentum Oscillator
MA – *Moving Average*	$\frac{\sum_{i=1}^{n} X(i)}{n}$	Technique to short-term variation of price. Simple mean value of the past n days' prices, where $X(i)$ is the assets price on a specific day. Aims to buy when MA is increasing and/or the MA line crosses price line in a descending way. Aims to sell when MA is decreasing and/or the MA line crosses price line in an ascending way.	Trend Following
EMA – Exponential Moving Average	$\mathrm{EMA_t = EMA_{t-1}(n)}\times \left(1 - \frac{2}{n+1}\right) + \mathrm{X_t}\times\frac{2}{n+1}$	EMA averages asset's price and assigns more weight (exponentially) to the latest data [8]. The technical rule to EMA is the same as the Moving Average.	Trend Following

c) Concept of Leverage

In the first chapter leverage was briefly introduced, now, it is given a more rigorous explanation about the subject. Leverage is used to profit from fluctuations in exchange rates between two different currencies (currency pairs). Leverage is a loan that is provided to the investor by a broker, i.e., companies that are specialized in financial trading lend money to the investors to increase their investment, e.g., take a small investment of 1000 euros, if the leverage applied is 100:1, then the investment made is 100 000 euros. The common amount of leverage provided is 10:1. This can result on two possible outcomes, the first is, if the investment goes well and the profit is for example 3%,

equivalent to 3000 euros, the trader made an investment of 1000 euros and took profits that could only be achieved with leverage. Without leverage, the profits would be 30 euros. The second outcome is more pessimistic, due to the fact that using leverage comes with a greater risk and can turn against investors, i.e., if the investment made was the same 100 000 euros, and looses the same 3%, the trader has to pay to the broker 3,000 euros. To overcome this kind of risks brokers, have “Stop Limits” to avoid this type of actions to occur. Fluctuations in Foreign Exchange Markets do not go over 1% during intraday trading, permitting brokers to provide this much leverage [10]. One of this thesis objectives is to apply different levels of leverage, to compare them and find the one that is most suitable for a specific market strategy. More precisely, each strategy rule has to match a certain level of leverage i.e., the certainty of the strategy chosen has to be proportional to the level of leverage chosen.

2.1.2 Structure of Genetic Algorithms

In this section it is given some background about *Genetic Algorithms*, describing the features of a *GA*, such as, *Selection*, *Mutation*, *Crossover*, and respective chromosome design, followed by the main focus of introducing the importance of new approaches relying on Diversity and Dynamic GAs.

a) Concept of a Genetic Algorithm

In the first chapter was given a brief introduction about the basic concept of a Genetic Algorithm. There are many solutions and different approaches that can be applied to each step of the algorithm.

To describe the chromosome representation, it is necessary to know what kind of problem one has in hand. There are a wide number of problems that can be solved using GAs, and because of that, it is necessary to address the right design of a chromosome to a given problem. Usually, a chromosome (from the biological point of view) is composed by a set of genes that define a genome. From a machine learning point of view, the chromosome can be represented by an array structure, where each element corresponds to a weight, i.e., importance given to the element (in this work, the elements will be technical indicators). A population is a group of chromosomes, and at the beginning, they are randomly generated. This is followed by different chromosome operators, such as *Selection*, *Mutation*, and *Crossover*. *Selection* can be made through several different processes like *Tournament Selection*[7], or *Truncation Selection* [8]. The point being is that, a set of individuals from a given population is chosen to perform *Crossover*, i.e., being selected for the next generation. The “fittest” individuals are the ones selected, using a *Fitness Function*. There many possible ways to make this *fitness function*, depending on the problem in hand. It is very common to use a strategy called *Elitism* that chooses a very small percentage of individuals, according to their fitness value e.g., 1%, of the population to perform crossover automatically to the next generation.

After the selection of individuals, the *Crossover* techniques, like *Two-Cut-Point Crossover* [7], *Single Arithmetic Recombination* or *One-Cut Point Crossover* [8] take place in order to pick the “parents”, i.e., two individuals, to generate “offsprings” (new individuals for the next generation).

Mutation procedures are widely used to maintain diversity. This procedure generates a new random value for each variable selected, of a chromosome, for mutation. The number of mutations can

depend on population size and chromosome size. The probability of occurring a mutation is traditionally very low. On the other hand, diversity may be the key to solve many problems. Strategies like mutation and others, that will be shown further on the next section, may be able to resolve the "Local Maximum" problem. After applying the method for N generations, the GA will converge to an optimum population, that has to be trained, based on the necessary data sets, and then tested to see if it is according to reality. This is an iterative method, so the GA can learn and optimize the population in order to produce the best individuals to reach an optimum solution. Figure 1 shows an example of a structure of a Genetic Algorithm.

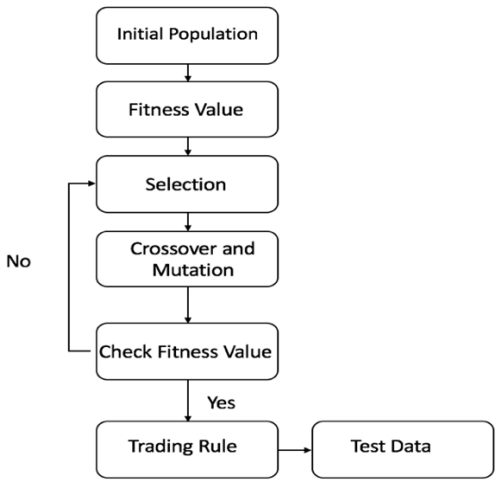

Figure 1 – Structure of a Genetic Algorithm

b) Diversity

Diversity is essential in order to a species to evolve from a generation to the next. One important aspect when initializing a population, is to make sure that the population has both quality and diversity [11]. The lack of diversity can set back a hole species evolution, quoting MIT's Professor Patrick H. Winston, *"…Sometimes species collapse into a state where they don't change for five or six hundred million years, like sharks…, sometimes they only survive if they got a lot of diversity built in their way of life so they can adjust to habitat changes…"*, can give a good example to explain the problem of "Local Maximums". This specific points are created when a given population does not have enough diversity to evolve, i.e., it gets stuck in a point where populations stagnate from generation to generation. The Local Maximums describe the main problem of evolutionary algorithms. They eventually converge to an optimum, consequently loosing their diversity to explore the search space more efficiently [12]. As stated before, Mutation is an operator that contributes to increase diversity, but there are other ways to make diversity stronger and more reluctant to dissipate during the process. A very interesting approach is to, not only choose the quality of a certain individual by is "fitness value", but also, measuring the fitness value combined with how different and individual is from other individuals that already been through the selection method. This way, it is possible to guarantee a certain level of diversity through generations making harder to "Local Maximums" to appear. The idea is to have an individual that can have both good values of fitness rank and good measures of diversity rank combined. Figure 2 exemplifies the idea with a plot "Fitness – Diversity" [11].

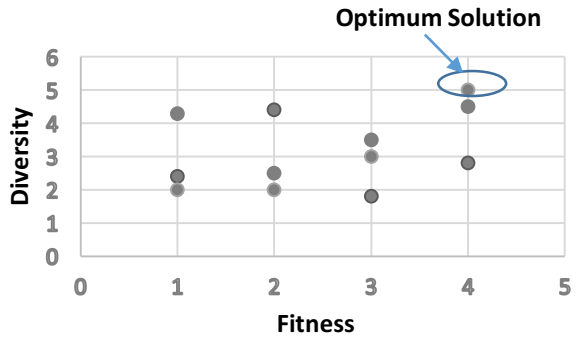

Figure 2 – Combination of fitness and diversity of an optimal individual

The literature defines several ways to improve diversity, to prove the concept, some of them are shown in the following order:

1. Increase Mutation Rate, to raise the number of mutations in a chromosome, the higher the mutation rate, the higher the number of mutations [13];
2. Avoid similarity between chromosomes, especially the ones that are duplicate. Duplicate chromosomes and similar populations may lead to the appearance of undesired "Local Maximums" and the stagnation of a certain species [13];
3. Trade-off between exploitation and exploration, i.e., transferring as much information as possible from the past, even though it is required new information to avoid similarities in the population and consequently early convergences to Local Optimums [12];
4. Multi-population scheme, to generate different populations, evolving separately, although the crossover is made by exchanging individuals between populations to increase diversity and avoid over fitting [14]. There are similar approaches where "immigrants", i.e., randomly generated chromosomes that are introduced in the population to increase diversity.

Hopefully diversity strategies will help to better adapt the optimization model in this thesis, to make better decisions when the environment changes, or, when sudden changes of the market strategy occur. To support this idea, in the next section it is introduced *Dynamic Genetic Algorithm* (*DGA*) to improve the adaptive capabilities of a *GA*.

2.1.3 Dynamic Genetic Algorithm – A Non-Stationary Problem

Traditionally, Genetic Algorithms aimed to solve static problems that usually presented quick and precise solutions. But the real world is filled with more challenging problems that need different and more effective approaches, like dynamic optimization problems (DOP's). This approach is very promising to produce more precise predictions in financial markets, since this markets, although, being proved that they are cyclic in the long term, and history tends to repeat itself, on the short-term, a lot of changes occur, and the algorithm must be ready to adapt to different environments. Whenever a change in the environment occurs and the optimization goal of the problem changes, the optimum to that problem might change as well, then, an adaptation of the old solution is necessary as it can be seen in figure 3. Said that, let's assume that this type of problems are time dependent, i.e., there's a new

variable in the equation that accounts time, like $F = f(\vec{x}, \vec{y}, t)$, where $\vec{x}$ represents decision variables, $\vec{y}$ the system control parameters depending on the problem, and t represents time. This means that optimum solutions may move over time in the search space. This paradigm did not happen quite the same way when the approach was static, i.e., after the algorithm converge it was very difficult to escape from an old optimum [15].

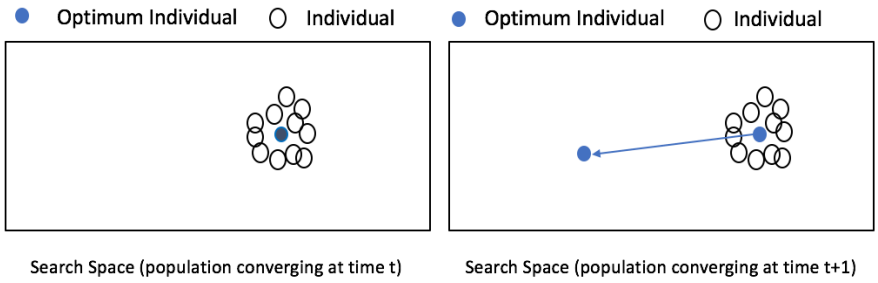

Figure 3 - Adaptation from an old optimum to a new one

The idea behind this approach is to make the algorithm to adapt and to be able to reshape peaks in the fitness landscape. Dynamic Genetic Algorithms (*DGAs*) have to be capable of adapting the solution to a changing environment, reusing the information gained in the past [12]. The premise is, once the environment changes, the fitness of an individual must change also, in order to a species to adapt.

As time goes by, and the environment changes it is important to change, add or remove decision variables, e.g., item weights. It is also important to refer that, not all environments are equivalent. Different environments, require different approaches of optimisation, being necessary to have some selection criteria. According to Shengxiang Yang, from De Montfort University, the classification criteria is based on:

- Predictability: Are changes predictable?
- Cyclicity: Are changes cyclic in the search space?
- Visibility: Are changes detectable?
- Time-linkage: Does the future behaviour of the problem depend on the current solution? (Most common DOP's are non time-linkable, this selection criterion shows that some solutions might influence the environments.)

To assess the predictability of change, the GA must be capable of understanding whether the environment presents patterns or trends that are possible to predict, or changes that are so severe that makes it impossible to predict the direction or the time of the next change. Depending on the problem in hand, predictability might be rather difficult to achieve. Cyclicity is an important measure, alongside predictability, to measure if an optimum point returns to previous locations in the search space, if it does, how many environmental states does the problem has before he encounters the same or similar state? The idea behind this measure, is to decide if past information is useful to find the future optimum. If the previous optimum is not exactly the same as a past optimum, it is important to use the distance between the two optimums in order to determine if the past information is close enough to the encountered one, making this, the decision rule to apply memory strategies. As it was stated before,

problems in Financial Markets can be considered cyclical, the markets have proven that some events occur with similar behaviour from time to time. But are those changes frequent? If so, how is the level of severity of those changes? According to Branke in "*Evolutionary optimization in dynamic environments*", there are two more selection criteria important to address this sort of problems. The first is, *frequency of change,* a measure of how often the environment changes, and if the *GA* is capable of adapting at the same speed. The time response of a *GA* usually relies on hardware and implementation details. Time response is an important factor, mainly because sometimes is unknown how much time does the *GA* has to adapt after the environment changed. The second selection criteria, *Severity of Change,* measures how strongly the environment changes. This measure can be determined by comparing the distance between the old and the new optimum [12]. To adapt to environment changes, one has to detect that a change took place. The *GA* may rely on different tools to detect changes in the environment, e.g., when the population performance gets deteriorated, meaning that probably, the old optimum it is not necessarily the best optimum in the new environment. There are many different solutions that have been studied to solve detection problems, but the objective behind all of them is only one, to monitor the model of the environment and to see if it is still consistent to the real environment. If the response predicted by the *GA* differs too much from the real environment response, this can only lead to the conclusion that the environment has changed.

There is a question that can be made after all this. Why not restart the Genetic Algorithm after a change on the environment? Despite of this being the easy choice, it would only turn the algorithm inefficient, wasting computational resources and too time-consuming. For real-problems, some of the solutions may not differ too much from the solution of the old problem [12]. Depending on the severity of the change, the *GA* must choose which is the most viable solution, whether the solution derives from an old optimum, a reconfiguration of the search space, that can lead the *GA* to search in more promising areas, or even rely on mutation and diversity of the old optimum to find the new one, making the restart of the algorithm always the last resort. DOP's rely on many approaches, such as: Memory, Diversity, Multi-Population or Adaptive. This approaches have been applied by different EA's for DOP's.

- The idea behind memory is to store and reuse useful information. It is possible to store information in two different ways. The first one, *Implicit Memory* uses redundant representations and each individual is represented by a pair of chromosomes. *Implicit Memory* is most used for problems with two state environments, where it has been proved to be very useful [12]. The second one, *Explicit Memory* uses extra space to store information about the population. When a change in the environment occurs, the algorithm uses memory to track a new solution in the search space. It is of major importance to assess when and which individuals are stored in the memory and which ones should be replaced, it is also important to know how many should be stored and how many should be replaced, and lastly which individuals should return to the population. Intuitively, the individuals stored should be above average fitness and this memory stack should be diversified to include as many solutions for the search space as possible. Figure 4 describes one of many possible solutions to implement memory strategies, inspired on Branke's approach using explicit memory based on storing information about the environment and good solutions.

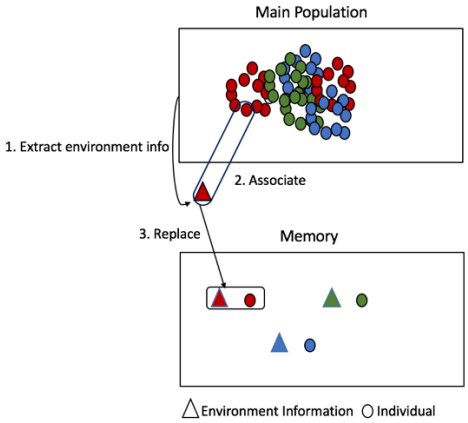

Figure 4 - Explicit memory example

- Diversity, like stated in the previous section, can help to better adapt the population to converge to the new optimum. If the population is diversified, it is most likely to have an individual that better suits the new environment and from there the population can converge to the new solution.
- Multi-population scheme uses different populations or sub-populations to shift the balance of the GA. Different populations explore different search spaces, to find the most promising area, making use of that population to converge to that place. Figure 5 shows a colony search space approach where the main population explores the promising area and the colonies explore the search space.

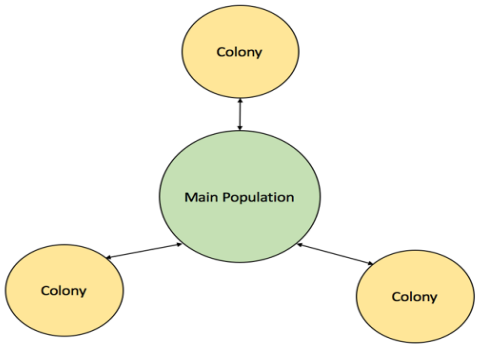

Figure 5 - Colony Search space

- The Adaptive approach aims to adapt operators after a change of environment. It is possible to assign different weights to each parameter depending on the environment, meaning that after a change occurs, it should be given more importance to some parameter that might not be as important before the change. The idea is to adapt operators, temporarily, after a change, e.g.:
    1. *Hyper-mutation*, raises the mutation rate temporarily [16];
    2. *Hyper-selection*, raises selection pressure temporarily [17];
    3. *Hyper-learning*, raises the learning rate for Population-Based Incremental Learning (PBIL) temporarily [18];

DOP's require a large set of performance measures. Table 2 presents some of the main performance measures that were proposed by Nguyen [15] [19].

Table 2 – Main Performance Measures

Performance Measures	Parameters
Optimality-based performance	• Mean best generation • Accuracy and Adaptation • Mean distance to optimum at each generation
Behaviour-based performance	• Reactivity and Stability • Robustness • Diversity Measures

To conclude this section let's divide the *DOP* in three important ideas, that are vital to understand *Dynamic Genetic Algorithms* [12].

1. The *GA* runs the same way as the stationary one, only when the environment changes or the *GA* predicts a change, that actions are taken to increase diversity in order to achieve a new optimum;
2. Early convergence is prohibited otherwise the *GA* has more difficulties to adapt.
3. The *GA* has to have memory to be able to find new optimums recurring to past information, which seems extremely useful for the problem addressed in this thesis.

d) Support Vector Machine and Market Identification

Support Vector Machine algorithms are classifier algorithms that are based on decision boundaries. Let's take an example of defining the difference between a Male and a Female, according to their height and weight respectively, let's assume also that Males are represented with an M and Females with an F. The approach used was introduced by Vladimir Vapnick, and its name is "*Widest Street Approach*" [20]. The idea is to draw a straight line γ between the Male samples and the Female samples, and the objective is to make that line the widest distance between the two samples. To better understand the concept, figure 6 provides a description of the widest street approach.

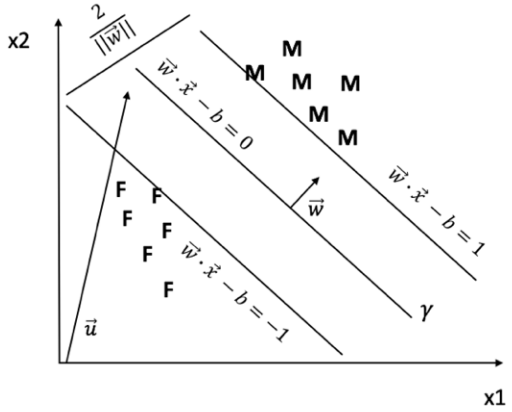

Figure 6 – Widest street approach

Said that, it is necessary to set a decision rule that will use the decision boundary shown on figure 6. First step, let's assume a vector $\vec{w}$ of any length, but has to be perpendicular to γ, then, another unknown vector $\vec{u}$. The next step is to classify if this unknown vector is on the Female side of the street,

or on the Male side of the street. In order to discover it, it is necessary to project vector $\vec{y}$ down on vector $\vec{w}$, to find the distance of vector $\vec{u}$ in the same direction as $\vec{w}$, i.e., let's assume that $\vec{w} \cdot \vec{u} \geq C$, being C a constant that will indicate, the bigger the dot product the bigger the distance will be and eventually the projection will give a distance that will be on the Male side of the street. It is also true to say that if $\vec{w} \cdot \vec{u} + b \geq 0$, $C = -b$ then the sample is a Male sample, making this the decision rule. But there is no knowledge yet about which $\vec{w}$ or b to use, there is still few constrain rules to choose any particular $\vec{w}$ or b. To overcome this issue, Vapnick decided to introduce two constraints, being the first on $\vec{w} \cdot \overrightarrow{x_M} + b \geq 1$ to describe Male samples, and $\vec{w} \cdot \overrightarrow{x_F} + b \leq -1$ to describe Female samples. In order to make this equations "Mathematical Convenient", it will be introduced a new variable $y_i = 1$ for Male samples and $y_i = -1$ for Female samples. Multiplying $y_i(\vec{w} \cdot \overrightarrow{x_M} + b)$ or $y_i(\overrightarrow{w} \cdot \overrightarrow{x_F} + b)$ the result is -1, and where there was two equations, now there is only one due to this "mathematical convenience", making possible to describe this new equation as $y_i(\vec{w} \cdot \overrightarrow{x_i} + b) - 1 \geq 0$ and for the frontier with line γ, $y_i(\vec{w} \cdot \overrightarrow{x_M} + b) - 1 = 0$.

In order to draw the straight line γ it is necessary to calculate the widest distance between the two comparing samples. To do that, first, let's subtract the two vectors and make the dot product with vector $\vec{w}$, that are shown on figure 6, to get the result, $\vec{w} \cdot (\overrightarrow{x_M} - \overrightarrow{x_F}) = 2$, being, $\vec{w} \cdot \overrightarrow{x_M} = 1 - b$ and, $\vec{w} \cdot \overrightarrow{x_F} = 1 + b$ generated from the constrain rules. To get the distance between the samples it is still necessary to divide this result by the length of $\vec{w}$, becoming the final equation:

$$\frac{\vec{w}}{||\vec{w}||} \cdot (\overrightarrow{x_M} - \overrightarrow{x_F}) = \frac{2}{||\vec{w}||} \quad (1)$$

To find the widest distance, it is necessary to maximize the distance function, so it is true to say that maximizing $\frac{2}{||\vec{w}||}$ it is the same as maximizing $\frac{1}{||\vec{w}||}$, because the value 2 is a constant, and is also true to say that, in order to maximize $\frac{1}{||\vec{w}||}$, $||\vec{w}||$ has to be minimized. To make it easier for the next step, it is also true to assume that $\frac{1}{2} \times ||\vec{w}||^2$ has to be minimized (Another "mathematical convenience").

The question to be made at this point is, how is it possible to minimize the magnitude of $\vec{w}$ while honouring the constrains? The answer that Vapnick found, was to use the method devised by Lagrange, revealing how the sample data, figures into the classification formula. In the following steps, it is shown how Lagrange multipliers solve this problem.

Step 1: Find place where L has zero derivatives:

$$\boldsymbol{L} = \frac{\mathbf{1}}{\mathbf{2}} ||\vec{w}||^2 - \sum_{i=1}^{l} \alpha_i \, (y_i(\vec{w} \cdot \vec{x_i} + b) - 1) \quad (2)$$

Step 2: Find the derivatives of L:

$$\frac{\partial L}{\partial w} = w - \sum_{i=1}^{l} \alpha_i \, y_i x_i = 0 \implies w = \sum_{i=1}^{l} \alpha_i \, y_i x_i \quad (3)$$

$$\frac{\partial L}{\partial b} = \sum_{i=1}^{l} \alpha_i \, y_i = 0 \quad (4)$$

Step 3: Substituting (3) and (4) on (2):

$$L = \sum_{i=1}^{l} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{l} \alpha_i \alpha_j y_i y_j (x_i \cdot x_j) \tag{5}$$

As it is possible to see, the optimisation depends only on pairs of samples (x_i, x_j). The new decision rule is now described as (6) for Males and (7) for Females:

$$\sum_{i=1}^{l} \alpha_i \, y_i x_i \cdot \vec{x} + b \; \geq 0 \tag{6}$$

$$\sum_{i=1}^{l} \alpha_i \, y_i x_i \cdot \vec{x} + b \; \leq 0 \tag{7}$$

Concluding this first step of the concept, it is possible to say that classification samples only depend on the dot product of unknown vectors with the sample pairs and the reliance on dot products enables the next approaches in which samples cannot be separated by a straight line, like in figure 6. This takes the problem to the next level. How to classify samples that cannot be separated by a straight line? As MIT's Professor Patrick Henry Winston said, "*When stuck, switch to another perspective*" [20].

To solve this problem, Vapnick used a powerful idea of a *kernel* that had been in his PHD Thesis thirty years before [20]. If the present dimension does not work, get the sample into a high–dimensional space using a transformation $\emptyset(\vec{x})$, where $\vec{x}$ is the sample. In order to maximize the distance between samples it is necessary to make the dot product of $\emptyset(\overrightarrow{x_M}) \cdot \emptyset(\overrightarrow{x_F})$. The idea is the same has explained before, so to classify the samples it will only be necessary to make the dot product of an unknown vector in a high-dimensional space like $\emptyset(\overrightarrow{x_i}) \cdot \emptyset(\vec{u})$, as a function of vectors in the original space. The advantage of this approach is that it is not necessary to know the transformation $\emptyset(\overrightarrow{x_i}) \cdot \emptyset(\vec{u})$, because:

$$\emptyset(\overrightarrow{x_i}) \cdot \emptyset(\vec{u}) = K(x_i, \vec{u}) \tag{8}$$

where, the K function is called a *kernel* and they all that is all that is needed to maximize the distance. To end this section, it is shown some kernels used in machine learning on table 3.

Table 3 – Example of popular Kernels

Kernel	Equation
Polynomial Kernel	$(u \cdot v + 1)^n$, n is the kernel's order
Radial Basis Kernel	$e^{-\frac{\|\|x_i - x_j\|\|}{\sigma}}$

To conclude this section, it is important to say that SVM algorithms are not immune to "over fitting", e.g., if σ is small enough, then the samples will shrink around some points, becoming "over fitted", although there are immune to local maximums, due to the fact that the method works on a convex space.

2.2 State-Of-The-Art

In this section it is introduced some of the existing solutions proposed until now. The section will be divided in works directly related to Forex using several EA techniques, works related to Genetic Algorithms and works related to Support Vector Machine algorithms applied to Forex. Finally, it will be presented a table to resume the existing solutions.

2.2.1 Works on Forex

Works on *Foreign Exchange*, up until now showed some promising results, using different approaches in *Evolutionary Computation*, and many different strategies were used to support the optimization algorithms. In this section, different approaches are divided by EC strategies, starting with *Neural Networks*, *Genetic Algorithms* and a *Decision Tree* algorithm.

Yao and Tan [21], reported empirical evidence that a neural network model is applicable to the prediction of foreign exchange rates, using time series and different technical indicators to capture the underlying "rules" of the movement in currency exchange rates [21]. The forecast of currency rates was made between USD and five major rates, to study issues on the frequency of sampling, choice of network architecture and several measures to evaluate the model's predictive power. Evans et al.[22], focused on predicting intra-day market exchanges, and their research focused on a hybrid solution between *NN* and *GA*. The forecasting model was based on Feed Forward Neural Networks with Back-Propagation architecture, using the GA to optimize the network topology. Their work confirmed with a significance of 95%, that daily Forex currency rates time series are not randomly distributed. Deng and Sakurai [23] applied a traditional genetic algorithm to generate trading rules based on a single technical indicator (RSI), and multiple timeframes. The research concluded that, using more than one timeframe can improve the assessment of the overbought and oversold conditions, and different traders may have different trading time horizons, being important to consider the overall condition for trading a currency pair. Aranha et al. [7] proposed to automatically generate trading rules based on Technical indexes, focusing on calculating the most appropriate trade timing, instead of predicting trading prices. To their research was added additionally buying and selling rule that aimed to increase the profits while managing the risk. Researchers also applied a model to use leverage in an investment, introducing a correlation factor that would indicate the proportion of leverage to use. Mendes et al. [24], used a genetic algorithm based on technical indicators and it is only applied to static problems. An important contribution in this research is based on the use of the "*Stirling ratio*" that takes into account the ratio between the profit and the maximum drawdown. Myszkowski and Bicz [25] approach the subject in hand with two decision trees, that are responsible for taking the decisions of opening long or short positions on EUR/USD currency pair. Trees take into account technical indicators, which are connected by logical operators to identify border values of the technical indicators.

2.2.2 Works on Support Vector Machine Regarding Forex

This section focuses on works related to *Support Vector Machine* only related to the subject in matter that is the FOREX market, due to the fact that, in this thesis the main focus is not the SVM algorithm itself but rather the *Genetic Algorithm* as its main core, being the *SVM* algorithm a tool inserted to increase the performance of the *GA*.

Sermpinis et al. [26], introduce a hybrid Rolling Genetic Algorithm-Support Vector Regression(RG-SVR) model to find optimal parameter solution. RG-SVR method genetically searches over a feature space and combines the optimal feature subsets for each exchange rate. The predictions of each individual are derived from different linear and non-linear models. The focus of this research is the forecast of financial time series because of its non-linear and non-stationary structural nature. Additionally, this work implements a *GA-SVM* and a *GA-SVR* hybrid systems that are evaluated alongside the RG-SVR approach. The results obtained in the trading performance, make a guideline for the work developed in this work [26]. Shioda et al. [27] focused on support vector machine to learn sequences of volatility levels estimated by Hidden Markov Models (HMM), to make predictions about those levels. Many studies in the literature using *HMM* were applied to several applications, e.g., Elliot et al. [28] and Rossi & Gallo [29] applied *HMM* to estimate volatility, showing that two regimes in the market, change according to the economic situation. One regime refers to stable markets where the prices move calmly and the second one refers to markets that move furiously [27]. Yeh et al. [30] implemented a multiple-kernel to overcome the problem of tuning manually the hyper-parameters of the kernel functions. Researcher developed a two-stage multiple-kernel learning algorithm by integrating sequential minimal optimization and the gradient projection method. Results showed that overall system performance is improved by combining different hyper-parameter settings. Shaoning Pang et al. [31] proposed a correlation-aided support vector regression (*cSVR*) for time series application where the *cSVR* was put against the *SVR* showing effectiveness in experiments with 5 currency pairs (NZD/AUD, NZD/EUR, NZD/GBP, NZD/JPY, NZD/USD).

2.2.3 Works on Genetic Algorithms

This section describes different methods in the literature, and will be divided in traditionally GA (stationary problems) and DGA (non-stationary problems). In each step of the algorithm, it is possible to apply different methods and strategies to improve the optimization process.

a) Stationary problems

In the literature it is common to generate populations randomly, and fitness function, for the matter of FX Markets, can rely on different strategies, e.g., Gorgulho et al. [8], uses the technical indicator *Return on Investment* (*ROI*) to evaluate each individual within the population. Aranha et al. [7], uses the profit gained by trading during the period of the training data set. Other interesting method proposed by Aranha & Iba [32], is a multi-objective approach where the goal was to get the best trade off between profit and risk, i.e., the fitness function was composed by two parameters, cumulative return and “Sharpe ratio”, where the first measured the efficiency of the strategy in terms of how much has

the initial investment increased since the beginning and the second indicates the risk associated to an investment made. Cirillo et all. [6], stated in his research that, "… *a proper fitness measure cannot be defined as an error on a per-data point basis, but instead as a measure of performance obtained over the entire dataset*", said that, the fitness value in their research was defined as a ratio between the final Net Assets Value (NAV) and the initial NAV minus 1, to assess the total value of the assets held. Férnandez-Blanco et al. [33] uses a dynamic method, relying on the premise that the fitness landscape may change over time, thus applying different indicators in different time periods.

The selection process can adopt a wide variety of methods. Gorgulho et al. [8][34], used in their research the *Truncation selection method* where, the individuals were sorted according to their fitness values, i.e., the best individuals were selected for reproduction. Another well known method is the *Tournament Selection Method e.g., Hirabayashi et al.* [7] and Yuan [35], used this approach, to select the individuals that are most probable to survive the tournament, i.e., individuals with better fitness values have higher probability to be selected than weaker individuals for crossover. Evans *et al.* [22], and Mendes *et al.* [24], relied on the *Roulette Selection Method*, to find the probability of a certain individual being selected, depending on the individual's fitness compared to the fitness values of the population.

One method that is often applied in the literature is the method of *elitism*, where a certain percentage of the best individuals are directly selected for the next generation without performing any crossover or mutation, usually the percentage below is 10%, this method can be suited in research works made by Gorgulho et al. [8], Hirabayashi et al. [7] or Deng & Sakurai [23].

Many researchers have implemented different strategies to maintain diversity in a population. The first method shown, to maintain diversity is to implement mutation operators with a certain mutation rate, i.e., the probability that each individual has of being mutated, those mutation rates have, often, low percentages e.g., Zhang and Ren [36], Hirabayashi et al. [7] and Gorgulho et al. [34]. Fernández-Blanco et al. [33] used a different approach to maintain diversity that relies on a small set of individuals that are randomly generated every generation, called "Immigrants", choosing this approach to get the trade off between fast computation and diversity. There are, also, several ways to perform crossover, e.g., Gorgulho et al. [37] compared three different methods, *Single Arithmetic Recombination*, *Whole Arithmetic Recombination* and *One-Cut Point*, concluding that the best results were from the *One-Cut Point* solution, in order to generate two distinct offspring. Hirabayashi et al. [7] used a *Two-Cut Point* technique to perform the crossover, instead of choosing one random point in the chromosomes and swap them to generate the offspring, the *Two-Cut Point*, chooses two points at random in each parent individuals, to swap them in the same way as the *One-Cut Point*.

b) Non-Stationary problems

To address non-stationary problems, it is necessary to implement *Dynamic Genetic Algorithms*, thus is important to develop dynamic problem generators to create dynamic test environments [38]. In the last years, researchers studied different approaches on the subject. Yang [39] and Yang & Yao [38] developed a XOR DOP Generator that can create DOPs and detect changes in the environment through the XOR operator, shifting the population of an algorithm to a new location in the fitness landscape [15];

Li &Yang [40] proposed the Generalized DOP Benchmark Generator (GDBG) that can be instantiated into binary space, real space and combinatory space, therefore combining three different benchmark generators. This method results from tuning the system control parameter ϕ, from the current environment with the deviation from the current control parameter $\Delta\phi$, through a XOR operator and throughout time. Contrarily to stationary problems, that only need few performance measures, e.g., convergence speed and success rate of reaching optimality [15], DOPs have over 20 measures that were studied by Nguyen et al. [19], thus dividing it in two types of measures: Optimality-based performance measures and Behaviour-based performance measures. Branke [41] implemented a direct memory approach, to store the good solutions throughout time, and examined in what circumstances memory was useful. Yang and Yao [42] investigated associative memory to store the best solutions as well as environment information to improve the algorithms adaptability to the environment. DOPs require diversity to adapt more efficiently and to better adapt to environmental changes. Park et al. [14] implemented an algorithm with two populations to provide additional diversity to the main population through *crossbreeding.* The results have proven that the distance between the two populations of the algorithm is an important factor that affect performance. Li et al. [43] research, was able to demonstrate that the performance of the algorithm can be significantly affected by maintaining multiple populations on different peaks to locate and track multiple changing peaks simultaneously. Yang [44] developed a Memory-Based Immigrants approach, to maintain the diversity and guide immigrants towards the new environment, using memory scheme to adapt the GA faster to the new search space by reusing past information. This search revealed promising results showing that memory based immigrants improve the performance of *GAs* in dynamic environments. Cobb & Grenfenstette [16] proposed the hyper-mutation method to raise the mutation rate temporarily, to increase diversity between the individuals, in order to improve adaptive performance to the *GA*; Yang & Tinos [17] developed the hyper-selection method that aimed to raise the selection pressure temporarily to improve the quality of individuals; Finally, to complement the adaptive approaches described, Yang & Ritcher [18] based their research on hyper-learning in order to increase the learning rate for a Population-Based Incremental Learning(PIBL) temporarily, proving that the learning rate has great impact on the algorithm performance in order to achieve an optimum solution.

For some DOP's environment changes present predictable patterns that can be useful to improve performance. Bosmam [45] research focuses on predictive techniques using time-linkage, and understanding how past decisions can influence future ones, therefore predicting events in the future from past information; Simões & Costa [46] introduced a predictive model based on linear regressions and a Markov model that memorizes past information of when the changes in the environment occurred, to estimate when the next change will occur.

The works presented in this section were an important contribution for the literature and some of the approaches are going to be taken into account to develop a solution for the problem in hand. The works regarding *GAs* and *SVM* show some promising results, although some of them may seem too optimistic, nonetheless this contribution provides a solid base to prove the potential of the approaches used in this work. Unfortunately, some researches do not provide the necessary data so the user can understand each strategy in a more detailed way. The research made on *SVMs* in the past few years

show a potential path to develop new techniques in financial applications, although the literature does not yet present similar works when it comes to classify the different types of markets, but provides important tools that can help this work's research in the future, e.g., Hybrid system's between *GAs* and *SVMs, ARBF-PSO* approaches [26], or the use of two stage MKSVR [30]. SVM algorithms are a very promising area, that shown good results to solve this financial problematic, using *Support Vector Regression* (SVR) Algorithms mostly. In this thesis, one of the objectives is to use the *SVM* algorithm as a classifier instead of using the SVR approach. Regarding the *GAs*, the literature has proven that this approach provides solid results in this field of studies. The methods presented in the *GA* structure for the different steps, alongside training and test data periods, present great potential to develop the *Optimization Layer*, using a *GA*, e.g., Multi-population method used by Park [14], or the *Selection*, *Crossover, Mutation* used by Gorgulho [8], Aranha et al. [7] or Deng & Sakurai [23]. The *GA* provides a structure where it is possible to introduce technical indicators, as a chromosome constitution, to produce the best trading rule for the investment. Aranha et al. [7] approach on the use of Leverage in Forex Markets is an important starting point that supports some of the approaches that are used in this work. The research done in FX applications using *GAs* or *SVM* do not explore the Leverage tool as much as expected, since this is the main tool to provide higher profit in this sort of market. Finally, it is possible to conclude that GA and SVM strategies are well suited for the problem presented in this work, and these two algorithms make a good choice for the techniques used in this work's research and development. Chapter 3 proposes an architectural system that will be based on some of the approaches seen in this chapter. Some of the GA methods seen in this section are going to be used in this work, and some new approaches are going to be experimented, especially, regarding the SVM to categorize the market. Table 4 resumes the related work referring to Genetic Algorithms and works related to Forex using SVM.

2.3 Chapter Conclusions

This chapter provided the financial background necessary to better understand the work developed in this thesis, as well as the basic concepts of a *SVM* and a *Genetic Algorithm* that are very important to one, understand the objective and the motivation of applying *Machine Learning* and Evolutionary Computation to a financial application. Moreover, the literature shows important contribution to the matter and also, provides solid works to develop a careful study that is very important starting point to develop a system that is capable of investing in Forex markets, using *EC* and *ML*.

Table 4 - State of the Art Resumed

Work	Date	Heuristic	Approach	Evaluation Function	Result Evaluation	Financial Application	Period	Best Return
[8]	2011	GA	Artificial Immune System	ROI	Comparison with B&H and Random	Stock Market	2003-2009	62.95%
[7]	2009	GA	Immigration Method	Profit Gained	Comparison NN, no leverage, B&H	JPY/USD, AUD/JPY, EUR/JPY	2005-2008	38%
[32]	2007	GA	Seeding Legacy Size	Sharpe Ratio	Comparison to Index	NASDAQ, NIKKEI	2000-2006	57%
[33]	2008	GA	-	Dynamic Method	Comparison to Random and B&H	Stock Market	-	50%
[39]	2003	DGA	Adaptive Selection, Dual GA	Royal Road Function, One Max Problem	Comparison Simple GA, Dual GA and Primal Dual GA	-	-	-
[38]	2005	DGA	PBIL, Dual PBIL	Deceptive Function	SGA, PIBL, Dual BPIL	-	-	-
[42]	2008	DGA	Multi-population; Memory scheme to PIBL; Associative memory;	Dynamic fitness function	ISGA, SPBIL, SPBILi, MPBIL, MPBILi	-	-	-
[14]	2008	DGA	Multi-population scheme with reserve populations	Dynamic fitness function	DPGA, IMGA, PDGA, SGA	-	-	-
[43]	2015	DGA	Moving Peaks Benchmark, Multi-Population Methods	Average Score, tPercent, gRatio	AMSO, SAMO, CPSO, DynDE	-	-	-
[44]	2008	DGA	Elitism, Random immigrants, Memory-based immigrants	Sum of the profits of the selected items	SGA, MEGA, MSGA, MIGA, EIGA	-	-	-
[22]	2008	NN and GA	Elitism, Termination Criteria of 15 generations unchanged	Mean absolute error, Sharpe Ratio, Annualized Return	Monte Carlo Simulation, comparison between actual data and predicted data	GBP/USD, EUR/GBP, EUR/USD	2010-2012	72.5%
[23]	2013	GA	Elitism Multiple time-frame	RSI indicator	B&H, S&H, SVM, GA-simple time-frame, GA multiple time-frame	EUR/USD	2011	1387 pips
[26]	2015	RG-SVR	Roulette Wheel Selection, RBF *v*-SVR	Annualized Return	GA-SVM, GA-SVR, ARBF-PSO, MLP, HONN, RNN, PSN	EUR/USD, EUR/GBP, EUR/JPY	1999-2012	-
[27]	2011	SVM	HMM, Non-linear kernels, Metrics: Accuracy, Precision, Recall	-	BIC of level 2 HMM, level 3 HMM	USD/JPY	2001-2009	-
[30]	2010	SVR	Multiple kernel, Gradient projection method, SMO	Armijo Rule	ARIMA, SKSVR, MKSVR, FNN	Taiwan Stock Index	2002-2005	-

3. Proposed Architecture

The proposed solution intends to develop an algorithm to trade FX currency pairs, using an *SVM* to categorize the type of market and a *Genetic Algorithm* to optimize the investment solution in order to trade with several levels of leverage. This chapter describes the system's architecture, more precisely the different layers of the system application and the way they were implemented. It will be also described the set of technologies implemented, according to the research made in the previous chapter.

3.1 Architecture Description

The system architecture has to be implemented into three separated layers, in order to make them independent, so that the user does not have to be concerned about the different layers at the same time. The system is divided into the *User Interface Layer*, *Forex Data Layer* (FX data Layer), *Optimization Layer*. *FX Data Layer* is divided into *Price Processing Module* and *Technical Rules Module,* and the *Optimization Layer* is divided into *SVM Module* and *GA Module*. Figure 7 illustrates the overall system architecture and the different layers.

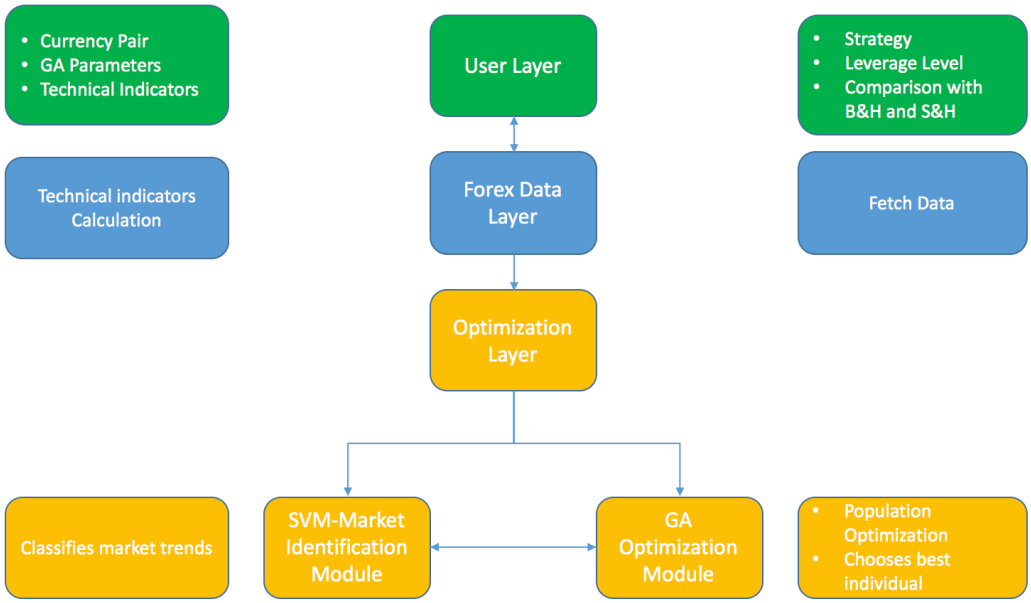

Figure 7 – Overall Architecture

Figure 8 shows the overall description of the proposed system and it is important to explain the general idea behind this approach. The basic concept is to use an *SVM* classifier (*SVM Module*) to classify three types of markets (class "1" for Uptrend markets, class "0" for *Sideways* markets and class "-1" for *Downtrend* markets), in order to train three different *Genetic Algorithms*, for each type of market. The idea behind the proposed system is to use the *SVM* as a classifier (section 3.2.3 a) and also use this environment information to complete the memory approach used on the *Dynamic GA* (section 3.2.3 b), with three different *GAs*, where each one of them invests in the FX market, depending on the type of environment selected, e.g., if the *SVM* classifies the market as an *Uptrend* (class type "1"), then, the

module uses this information to use in the *GA* that was train for *Uptrend* markets. Another important aspect that is important to refer is that the *SVM Classifier* was experimented for two types of *features* used, one with *Technical Indicators* and the other with *Forex Price Sequences*, that are explained in detail in section 3.2.3 a. These two approaches are put against each other, in order to study the consequences of using price sequences instead of technical indicators, that store information about the price sequences. It is important to explain that the technical indicators were calculated in a separate module so they can be used by the *SVM* and *GA Module*. The calculations of the technical indicators are explained in section 3.2.2 b. As for the three GAs, that are exactly coded the same way (only difference is that each GA is trained for different types of markets), the fitness function used is supported by a voting system that uses technical indicators, the decision rule is further explained in section 3.2.2 b and the voting system is explained in section 3.2.3 b.2.

Joining a SVM classifier with a *Dynamic Genetic Algorithm*, making a hybrid system to predict the FX market is one of this work's objectives, and adaptability approaches were introduced in the algorithm in order to improve its dynamical capabilities. Those approaches are shown in section 3.2.3 c.

Architecture layers have to be independent from each other, to facilitate the implementation of different methods during the process. The algorithm will be put against three classic theories, *Random Walk* and *B&H* and *S&H*. The first theory is based on the though of buying and selling assets randomly, and the second is a traditional strategy in the financial market that states that on the long run the asset's price will return profit with "long" trades. The last approach is the same as the second, being the only difference that instead of adopting long positions, the trader adopts a short position. Finally, the proposed system is also put against a *Static Genetic Algorithm* that presents the usual Genetic Operators that are usually seen in the literature, but this algorithm does not present any dynamic characteristic, being the objective, to compare the results between a Static and a Dynamic approach.

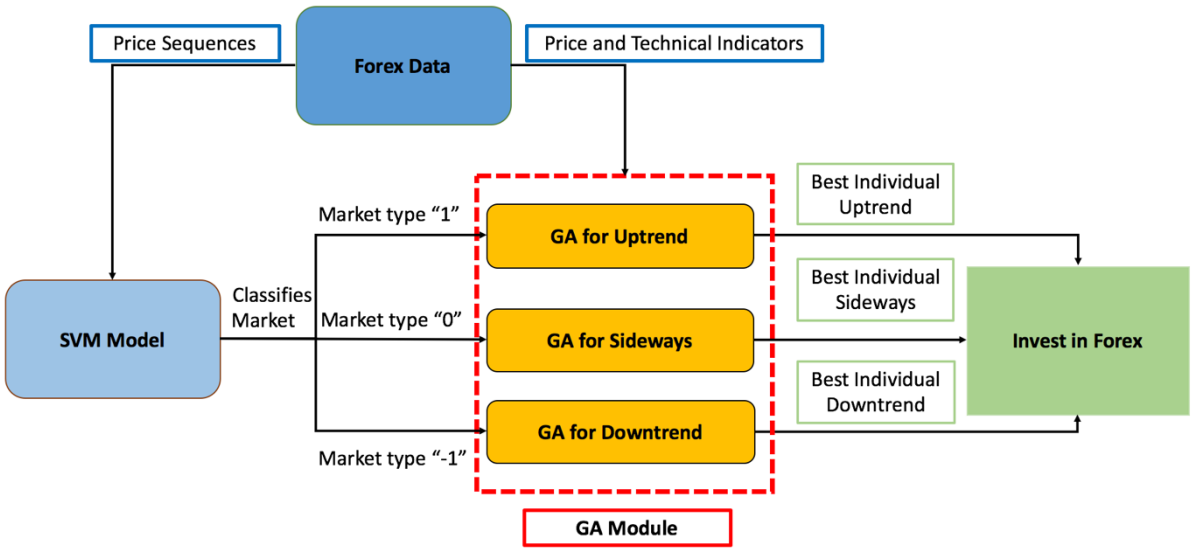

Figure 8 – Schematic for the Proposed System

3.1.1 User Interface Layer

This layer is the upper layer and provides the program interface for the user. This interface has input data, where the user submits a file configuration form, deciding the different parameters for the proposed system, then the interface returns the output data where it is included the *ROI* obtained, as well as, the leverage levels used during the simulation. It is also possible to choose the training and test period for the simulation, making it easier for the user to run the software without needing to use a file configuration that is much harder to interact. In section 3.2.1 it is shown the *User Interface* implementation, as well as, a proposition of implementation, regarding a much more evolved and complete version of the interface developed in this work. Figure 9 shows the proposed features that integrate the user interface.

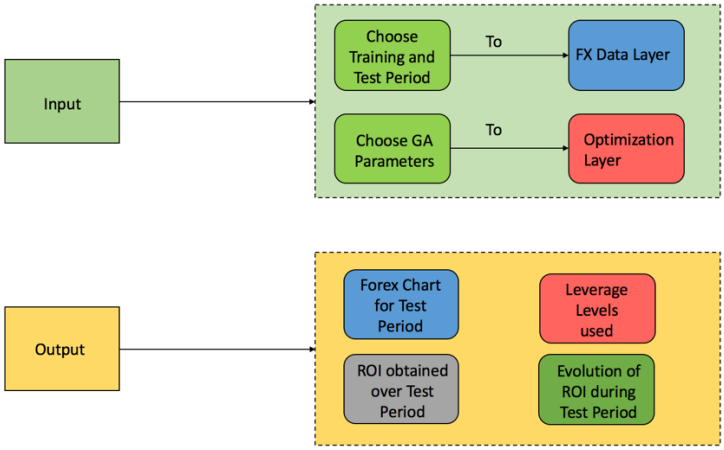

Figure 9 – User interface Schematic

3.1.2 Forex Data Layer

This layer is divided into two modules, the first one is the *Price Processing Module* and the second one is the *Technical Rules Module*. The first module takes the prices of a currency pair, that in this case will be the EUR/USD, calculates the technical indicators used in this work, and saves everything in a database (DB), that was built in order to have fast access time and robustness. The Forex data can come in many types of formats, but more specifically in this work, the data retrieved is based on hourly prices. Like stated above, this layer is also responsible to calculate each technical indicator, to create the input to insert in the *GA*. The retrieving of data is a very important step and has to be made very carefully so the data retrieved does not become damaged. Since the *GA* needs past information to make predictions about the future, it is only natural to conclude that the more information it retrieves the better. Finally, the system has to be robust and since the index is updated on a daily basis, it has to be possible to adapt and correct the data if it ever becomes corrupted. The second module defines the trading rules, based on the technical indicators. This module is very important to define a good trading rule to enter or exiting the market. Also this module has to be prepared to acquaintance changes and has to be as optimized as possible, so it does not work out as a strangling point in the evaluation function of the *GA*.

3.1.3 Optimization Layer

As it was said before, this layer is decomposed into two different modules. The first one, is the *SVM Module*, its objective is to identify different market strategies, i.e., the *SVM* algorithm classifies the currency pair historical data in three groups, *Bullish Market* (Uptrends), *Bearish Market* (Downtrends), and *Sideways* (No Trend). This way, the *GA* trains its populations to specific types of markets, depending on the trend. This feature is expected to return a better performance to the *GA* and increase its success rates.

The *SVM* is trained in a given time period in order to learn to distinguish the different types of markets. This module also works as a "memory" (as in a *Dynamic Genetic Algorithm*), storing the information of the market (environment) and the information about the chromosome associated with it. Figure 10 shows a FX chart divided by types of markets.

Figure 10 - FX chart divided by types of markets

The *GA* module is responsible for the optimization section, where several approaches, seen in the literature, are tested for the different steps of the algorithm. This module is responsible to pick the best individuals of the population, in order to achieve the optimum solution. This work relies on the elitist method, where the fittest individuals of the population will automatically be picked for the next generation, and the worst individuals will be put aside. The chromosome is represented by a float array vector, and each chromosome will identify a single investment strategy. As for the chromosome representation, it will be composed by the six technical indicator's parameters and the respective weights for each one of them and a gene specific to the leverage level that it is used. In the end, the GA, according to the best population generated, will return a technical rule, Buying Rule, Selling Rule and the "Do Nothing" Rule, being the last one relative to the sideways market. Figure 11 provides the schematic, describing the GA module.

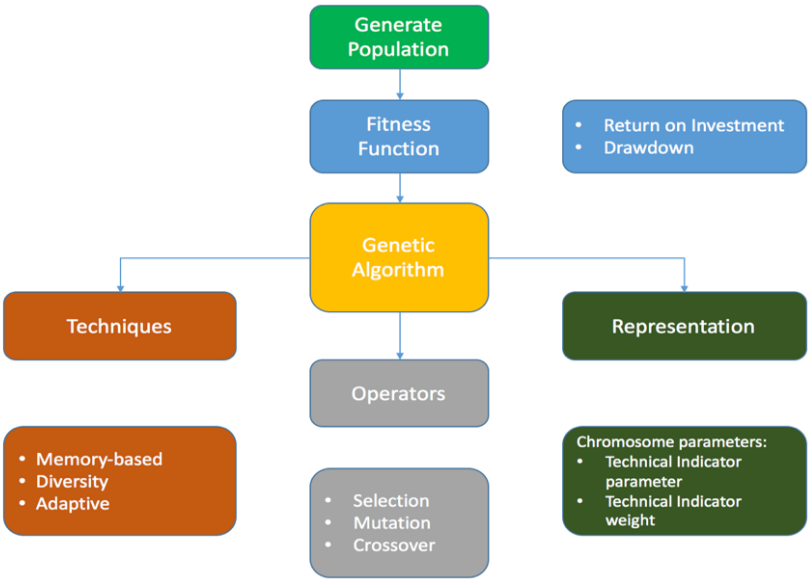

Figure 11 – Schematic for Genetic Algorithm Module

3.2 Architecture implementation

This section will describe how the software was implemented throughout the different layers.

3.2.1 User Interface Layer Implementation

This interface was developed using a python *Graphical User Interface* (GUI) toolkit [47], on a Python IDE called Spyder [48]. The user inserts in the program the following inputs (or uses the default values):

- Start and End date for Training and Test Periods;
- Size of Population;
- Number of Generations for the Proposed Stopping Criteria;
- Cross Over probability;
- Tournament Size for the GA operator of Selection;
- Mutation Rate;
- Button to Run the Simulation of the Proposed System

And the outputs are the following:

- The EUR/USD chart price by the hour during the Test Period chosen;
- The type of market the currency is;
- The leverage levels used during the Test Period;
- The ROI obtained over the Test Period;
- The evolution of ROI during the Test Period;

This interface is important so the user can see what the software is doing and also to make the ultimate decision over the software, if for any reason a special event in the market is happening, e.g., if the central banks publish an important news that might change the market suddenly then the user has the chance to stay out of market or invest accordingly to is opinion on the news. Figure 12 shows the interface developed for this work.

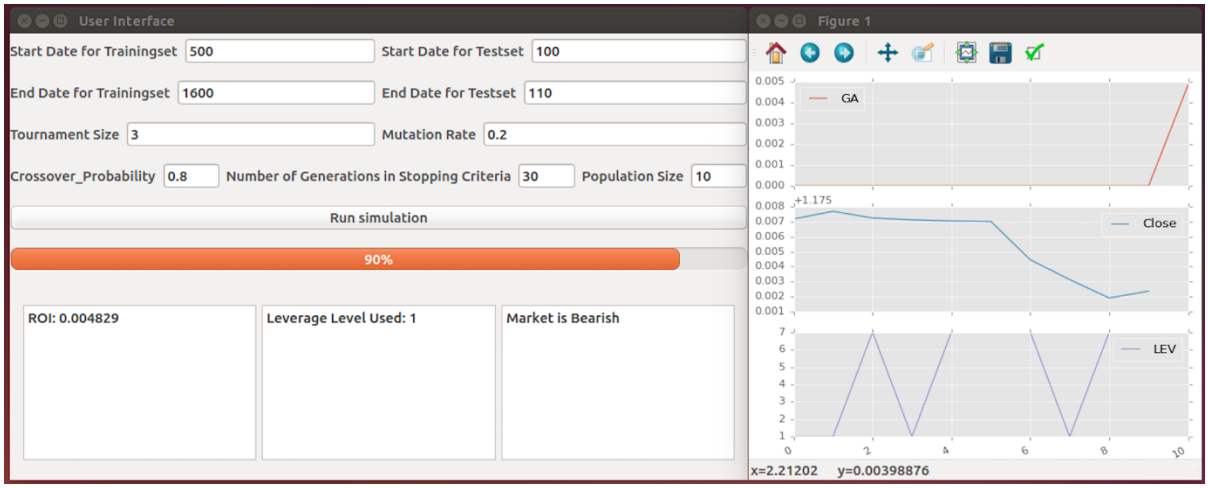

Figure 12 – User Interface developed

As it is possible to see this user interface is quite simple, but, to develop a more sophisticated one would take excessing time that was important to develop the proposed system. Nonetheless, it is important to address some important features that were not developed in the user interface but, are also interesting to take into consideration. Regarding the inputs, the user interface should also be able to:

- Choose which Currency Pair the user wants to invest in;
- Choose which technical indicators the user wants to insert in the chromosome;
- Choose if the investment is made manually or the decision rule is made by the proposed system;
- Choose a currency pair chart and the technical indicators, in order to the user analyse the present prices and following trends;

As for the outputs, the interface should return:

- The FX chart completed with the Technical Indicators in real-time to help the user analysis;
- SVM's classification analysis in real-time;
- A real-time voting system so the user understands what is happening during the time the proposed system is running;

3.2.2 Forex Data Layer Implementation

a) Price Processing Module

The simulations use EUR/USD data since the year 2003 until 2016. To provision the work's database, the software relied on a python library called *Pandas* [49], in order to improve the robustness of data and to reduce the access time when a request to the DB was made. *Pandas* library was design to be faster and more organized as a DB when compared to simple array structures or python types such as "lists" or "matrixes".

The technical indicators were calculated using the *FX* data, in order to be used in the optimization layer. The Technical Indicators were calculated through a *Python library* called *ta-lib* [50], making it possible to only calculate them once, in order to make the software faster and less heavy on computation. The technical indicators used in this work and its parameters are shown in table 5:

Table 5 – Technical Indicators used and respective parameter intervals

Technical Indicators (TI)	Intervals for TI parameters
RSI (Momentum Oscillator)	Period : [5, 30]
ROC (Momentum Oscillator)	Period : [15,30]
SMA (Trend Following)	Period : [10, 100]
EMA (Trend Following)	Period : [10, 100]
WMA (Trend Following)	Period : [10, 100]
MACD (Trend Following)	Fast Period: [10, 20] Slow Period: [20,35] Signal Period: [5,10]

b) Technical Rules Module

One of the main difficulties is to choose the right strategy to invest, whether is the right mix of technical indicators, its thresholds or even the parameters that are chosen to calculate the technical indicators, small variations in one of these decisions can change the rule to buy or sell in instants. The first assumption to make is quite simple, there is no better indicator, only a good combination of them will return a good decision rule. The first technical rule applied was a voting system where a combined set of technical indicators voted, depending on specific rules that are explained in table 6.

Table 6 – Decision Rules for the voting system proposed

Technical Indicator	Trading Rule	Decision
RSI	• RSI<30 • RSI>70	• Buy Rule • Sell Rule
ROC	• ROC>0 • ROC<0	• Buy Rule • Sell Rule
SMA	• SMA<price • SMA>price	• Buy Rule • Sell Rule
EMA	• EMA<price • EMA>price	• Buy Rule • Sell Rule
WMA	• WMA<price • WMA>price	• Buy Rule • Sell Rule
MACD	• MACD>0 • MACD<0	• Buy Rule • Sell Rule

3.2.3 Optimization Layer Implementation

a) SVM Module Implementation

This module is composed by several components, and the first step is to describe the features used in the *SVM* algorithm. The project relied on *scikit learn* [51] framework in order to assure reliability in terms of computation and stability, being that the focus of this work is not the implementation of an SVM itself. Although the *SVM* classifier is the same, there were two possible approaches to choose the type of features used. The first one was the use of the technical indicators as features and the second one was the use of price sequences, in order to classify the type of market. In the first case the features used were the six technical indicators calculated in the FX data layer. In the second case, sequences of 100 prices were used and each one of the prices worked has a feature of the *SVM*. Table 7 shows an example of the two types of features used in each scenario.

Table 7 – Types of features used

Feature type	Features example
Technical indicators	Features=[“RSI”, “ROC”, “EMA”, “SMA”, “WMA”, “MACD”]
Sequence of Prices	Features=[Price1, …., Price100]

The reason to implement this two approaches is that an *SVM* classifier is very effective in high dimensional spaces, it is memory efficient because it uses a subset of training points, i.e. the support vectors make it memory efficient and it is quite versatile, relying on different kernels. The samples applied to this problem are in such large number that abled the construction of a model with 100 features, so the problem of having more features than samples does not apply in this case. Figure 13 shows how *Pandas* library handles the data for the features with an example of a price sequence of 10 values.

Price1	Price2	Price3	Price4	Price5	Price6	Price7	Price8	Price9	Price10
1.36	1.36	1.36	1.36	1.36	1.36	1.36	1.36	1.36	1.36
1.11	1.11	1.11	1.11	1.11	1.11	1.11	1.11	1.11	1.12
1.36	1.36	1.36	1.36	1.36	1.36	1.36	1.36	1.36	1.36
1.33	1.33	1.33	1.33	1.33	1.33	1.33	1.33	1.33	1.33
1.26	1.26	1.27	1.27	1.27	1.26	1.26	1.26	1.26	1.26
1.38	1.38	1.38	1.38	1.38	1.38	1.38	1.38	1.38	1.38
1.15	1.15	1.15	1.15	1.15	1.15	1.15	1.15	1.15	1.15
1.09	1.09	1.09	1.09	1.09	1.09	1.1	1.09	1.09	1.09
1.24	1.24	1.24	1.24	1.24	1.24	1.24	1.24	1.23	1.23
1.13	1.13	1.13	1.13	1.13	1.13	1.13	1.13	1.13	1.13
1.12	1.12	1.12	1.12	1.12	1.12	1.12	1.12	1.12	1.12
1.25	1.25	1.25	1.25	1.25	1.25	1.25	1.25	1.25	1.25
1.28	1.28	1.28	1.28	1.28	1.28	1.27	1.27	1.28	1.28
1.14	1.14	1.14	1.14	1.14	1.14	1.14	1.14	1.14	1.14
1.28	1.28	1.28	1.28	1.28	1.27	1.27	1.28	1.28	1.28
1.11	1.11	1.11	1.11	1.11	1.12	1.11	1.12	1.11	1.11

Figure 13 – Price Sequence example using Pandas library

The algorithm implementation follows the theory explained in chapter 2, section 2.1.4, where the *SVM* takes an *X* array of size *n_samples* and *n_features* like: *X*= [*n_samples, n_features*] of training samples and a *Y* array of class labels of *size n_samples* like: *Y*=[*n_samples*]. In this work's scenario the labels correspond to the market types, where a *Bullish Market* represents a "1", a *sideways* market represents a "0" and a *bearish market* represents a "-1". The classification of the market is then calculated using all the classifications of the predictions, i.e., if the classifier predicted 100 price sequences in a sliding window, the result would be 100 samples that were classified as "0", "1" or "-1".

The calculations are made with a *Weighted Moving Average (WMA),* that analyse the average through a certain period, deciding which type of market is selected, based on the result of the *WMA*. If the result of the WMA is bigger than 0.5, the market is *Bullish*, if the *WMA* is lower than -0.5, the market is *Bearish*, otherwise is *Sideways*. Moreover, the *WMA* computes an average where the last prices have more importance, i.e., weight that influence the moving average.

The *X* and *Y* are then divided into *X_train* and *Y_train* for the training set and *X_test* and *Y_test* for the test set. First, let's focus on the training set, where the *SVM* "learns" from the data in order to classify the market in the test set. To optimize the performance of the *SVM* training model it is necessary to tune its *hyper-parameters*. A *Grid Search* algorithm was implemented to choose parameters for the *SVM* classifier, where an exhaustive search of combinations was made through a specified subset of the *hyper-parameter* space. The *SVM* takes three *hyper-parameters*, where *C parameter* is the parameter for the cost function, which controls the influence of each individual support vector and involves trading error penalty for stability, i.e. *C parameter* controls the cost of misclassification on the training data, and it can influence the variance of the solution (large *C* value makes the cost of misclassification high, *hard margin* and low *C* value makes the cost of misclassification low, *soft margin*). *Gamma* is the *RBF kernel* parameter that handles non-linear classification, i.e., it influences the trade-off between the size of the street (explanation in section 2.1.4) and the margin for misclassification of a label. The *hyper-parameters* values used in this work are shown in table 8.

Table 8 – Hyper-parameters used for the SVM model

Hyper-Parameters	Values
C	[1,10, 100, 1000]
Gamma	[0.0001, 0.001, 0.01, 0.1, 1]
Kernel	• Rbf • Polynomial, n=2,3,4 • Linear

The training solution is then saved to a *pickle* file (saves data that is separated by commas and its objective is to serialize data in python programming) [52] in order to make the classification of the market faster, and to avoid an overkill solution where the SVM model is trained every time the software is run. After the training set is complete the algorithm applies the cross-validation method of *K-Fold* to assure that the trained solution is not biased. The method divides the DB in *K* parts and trains for some of them and then tests the others with the trained model. Figure 14 shows a schematic representation of a 5-Fold strategy example.

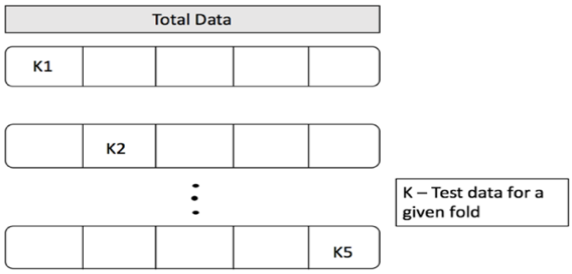

Figure 14 – K-fold strategy

The test set takes sequences of 100 prices, classifies them and then sends the answer to the *GA module*, so that the *GA* can know which environment it is on. Figure 15 describes a pseudo-code that shows how the *SVM model* was trained, and then used in the *GA* later on. The pseudo code shows the several steps that were taken to create the classifier, and shows the two approaches that were tried as features in this work. Both approaches were tested with the same logic and code.

```
features =["Price1", "Price2", ... , "Price100"] #Approach of features type 1
features=["RSI", "SMA", "WMA", "EMA", "MACD", "ROC"] #Approach of features type 2

Database:
    DB= Prices_from_EUR_USD #Sequence of Prices from EUR/USD
    DB =Technical_indicators#Calculated in the Technical Rule Module


X= array(DB[features]) #array with the features chosen from the selected DB
Y=DB["labels"] # Labels from a DB that is classified by the user to train the SVM Model
for DB:

    split X and Y in train_set and test_set

    Parameters=[["rbf", "poly", "linear"], [0, 0.01, 0.001, 0.0001], [1, 10, 100, 1000]] #hyper parameters [kernel, gamma, C]

    find_best_parameters:
        do grid_search(Parameters)

    classifier=train_DB(X, Y) #function where the model learns and creates a classifying model for the SVM

    apply 5-Fold(X) #Cross Validation Method
```

Figure 15 – Pseudo-code explaining the training of the SVM model

Admitting that only the correct evaluations are important as a metric of quality seems to be incomplete. For the *SVM*'s case studies, it is necessary to evaluate the quality of a classification using the metrics, such as *precision, recall and accuracy*. Before describing the measures, it is necessary to explain what are the *True Positives, False Positives* and the *True* Negatives, *False* Negatives. While *True Positives* are the number of items correctly labeled as belonging to a given class, *False Positives* label a number of items as a given class when in fact the item does not belong to that class. *True Negatives,* are the items that are truly rejected when labeled to a given class, e.g., an item that is class "1" gets rejected when the hypothesis is to label it as class "0". *False Negatives*, are items that were not labeled as belonging to a given class but should have been.

Precision is a measure for result relevancy, i.e., the fraction of selected items that are relevant and by its definition is the fraction between *True Positives* and the sum of *True Positives* and *False Positives*. Recall is also a measure of relevancy, but more specifically the fraction of how many truly relevant results are returned, i.e., the measure of how many relevant items are selected as a *True Positive*, and is defined as the number of *True Positives* over the number of *True Positives* plus the number of *False Negatives.*

Accuracy is the measure between the number of items that were labeled correctly (True Values) and the number of total items. Table 9 exhibits the equations for each metric [53].

Table 9 – Definition of the metric used in this thesis

Metrics	Definition
Precision	$\frac{True\ Positives}{True\ Positives + False\ Positives}$
Recall	$\frac{True\ Positives}{True\ Positives + False\ Negatives}$
Accuracy	$\frac{True\ values}{Sample\ Space}$

b) GA Module Implementation

In this module, the training set, takes advantage of three types of chromosomes. Although they are structurally the same, they were trained depending on the type of market the *SVM Module* classified.

To implement this section and to make possible to implement several strategies shown in the literature and add new features, this work's *GA* was developed based on the *DEAP framework* [54], specified for evolutionary algorithms. The reason to use this framework is to get the best performance and computation power from the algorithm.

Before describing the *GA Module* architecture, it is important to show which approaches were used to develop the *GA*, both from the literature and some new approaches tested. Table 10 shows the approaches made in this work.

Table 10 – Approaches used in the GA module

Method	GA approach	Strategy
Selection	Stationary	• Tournament
Mutation	Stationary	• Mutation rate • Mutation Probability
Crossover	Stationary	• Two-Cut-Point
Diversity	Non-Stationary	• Immigrant method
Memory	Non-Stationary	• Associative Memory using SVM
Adaptive	Non-Stationary	• Hyper-selection • Hyper-mutation

b.1) Chromosome Representation

The chromosome structure is an array of floats where each gene of the chromosome is the parameter for each technical indicator and its weight accordingly. Each gene is then transformed to an integer for each parameter of a technical indicator, in order to calculate the results of the given TI. This method provides a modular chromosome structure and faster than if the chromosome was a combination between integer and float array. The values to fill each chromosome, i.e. to create each individual are set with a random function with interval constraints depending on each technical indicator. The weights are a random function that returns a float between zero and one. The chromosome's objective is to optimize two sub-strategies. While, the first one optimizes the importance of each technical indicator, meaning that the *GA* searches for the optimal solution in terms of finding the optimal weights for each TI, the second sub-strategy optimizes the parameters of each technical indicator, i.e.,

parameters need to be dynamic in order to adapt to each sequence of prices, e.g. a RSI with a 14 period price sequence can tell the trader to sell the currency, when in fact it should have bought it if the RSI period was 25. The truth of the rule derives from each price sequence, and thus, being important to optimize the parameters also. The structure of the chromosome is shown in figure 16, red section indicates the weights for the chosen TIs, and the green section indicates the parameters for the second sub-strategy that is optimizing each TI. Finally, the leverage level of the red section optimizes the amount of leverage that is used.

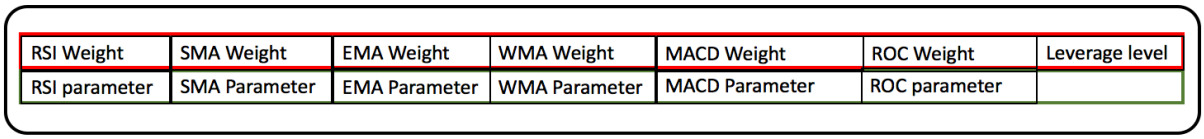

Figure 16 – Chromosome Representation

b.2) Evaluation Function

The fitness function relies on the *Return on Investment* (*ROI*), that is calculated as shown in eq. 9:

$$\frac{Returns - Investment}{Investment} \tag{9}$$

The Genetic Algorithm function is single objective and tracks to maximize the *Return On Investment* approach. To calculate *ROI*, the fitness function relied on the *Technical Rule Module* to make the decisions according to the TI. The trading rule is decided on a voting system, of whom ever won by majority would perform the trade, i.e., if the TIs voted for "Buying" Rule with a *K* % majority over the "Selling" Rule, then the *GA* would perform a "Buying" rule instead of a "Selling" rule, and vice-versa. If a majority was not reached, then the *GA* entered the "Do nothing" rule and waited until the next entering point. The following equations (10, 11, 12 and 13) show how the voting takes place, being *K* the parameter that defines the majority thresh hold.

$$Voter\ Buy = \sum_{i=0}^{n=6} w_i \tag{10}$$

$$Voter\ Sell = \sum_{i=0}^{n=6} w_i \tag{11}$$

$$Voter\ Buy > K * Voter\ Sell \tag{12}$$

$$Voter\ Sell > K * Voter\ Buy \tag{13}$$

After the voting system takes part, the investment module inside the evaluation function opens a "Long" or "Short" position depending on the voting. If there is no "Opened" position, then the voting system decides to enter the market if the voting rules a buying or selling position. If the voting ruled a sideways then it does not enter the market. When there is an "opened" position, then the position maintains opened until the voting system rules otherwise, i.e., if the position is long, then it stays long until the voting system rules a sideways or a short position. Figure 17 describes a pseudo-code to demonstrate the investment logic.

```
For a Trading Period

    If Position Not Open
        check trading rule
        if trading rule is Buy or Sell
            Open Position
    else

        if Trading rule maintains Opinion
            keep Open Position
        else

            Close Position
```

Figure 17 – Pseudo-code demonstrating the investment logic

b.3) GA Operators

b.3.1) Selection

After defining the chromosome representation, it is necessary to define the genetic operators used in the algorithm. The selection method chosen was the *Tournament Selection* approach to define which individuals were to be selected for the next generation. The tournament selection method involves running several tournaments between individuals among the population, and the winner of each tournament is selected to perform crossover. The bigger the tournament size the less chances a weaker individual has to perform crossover. Initially the tournament size was set to 3 and selection probability was 0.5. Only when the hyper-selection was activated, the tournament size was set to 5 to increase the selection pressure in a population. *Hyper-selection* was chosen to be applied in this work, because some periods of the FX market can be more difficult than others because of an ill-defined market tendency, thus, making the *Return On Investment* harder to increase its value. Said that, the selection pressure is increased temporarily to decrease the probability of weaker individuals to perform *Crossover*. This can lead to an over fitting, so it is important to keep in mind that the *Hyper-Selection* should only be temporarily. Figure 18 shows a pseudo code of a tournament selection algorithm.

```
For individuals in population :      #Individuals from Population

    group individuals in tournaments of size k #tournament size can change

    for groups made:

        choose individuals with best fitness within the group

    select best individuals from each group
```

Figure 18 – Pseudo-code explaining the tournament selection algorithm.

b.3.2) Crossover

The method that was most suitable to solve this problem was the *Two-Cut-Point* crossover, where two chromosomes exchange two genes between themselves. This algorithm was chosen over the *One-Cut-Point* because the *Two-Cut-Point*, slowed the early convergence, thus avoiding faster over fitting in the final solution. This *crossover* method consists on picking two random points of the parent chromosomes, then, the genes between the two points, and the ones outside the two points are swapped, originating *offsprings*. A proper example of the algorithm is shown in figure 19.

Figure 19 – Schema explaining the two-cut-point method

b.3.3) Mutation

Individuals may go through mutation in the end of each generation and in this work the mutation that one individual performs is derived from a Gaussian distribution, although its limits are imposed by the gene mutated, and must be higher than zero to be considered valid. The mean and standard deviation values are linked to the parameter of each gene, being the mean the parameter and standard deviation 10 % of the parameter value. Mutation is a random process that depends on the mutation rate and on the probability of each individual to get a mutation. Also in these studies, one introduced *Hyper-Mutation*, and according to changes in the environment the mutation parameter will go up temporarily in order to increase the diversity in a given population. Figure 20 demonstrates the pseudo-code for the mutation operator.

```
if random probability < mutation probability:  #probability of a chromosome being put out for mutation
    for gene in chromosome:
        if random probability < gene probability:
            mut_gene=mutate gene with gaussian function

        add mut_gene and gene to return the mutated gene
```

Figure 20 – Pseudo-code explaining the mutation operator

c) Support Vector Machine and Dynamic Genetic Algorithm Together

Like stated above, this is one of the key issues of this work, where it is intended to implement a memory system, relying on a *SVM* classifier to identify which type of market and then store the information along side an individual that was specifically trained for that situation. The goal is to train three different populations for the three types of markets, and then use them during the investment sessions, taking advantage of knowing the type of market that they are in, i.e. knowing the environment can increase the performance of each population, and this memory approach proved to be efficient, thus, it is not needed to have a new population every time the environment changes. Figure 21 provides a schema to get a better understanding of this approach.

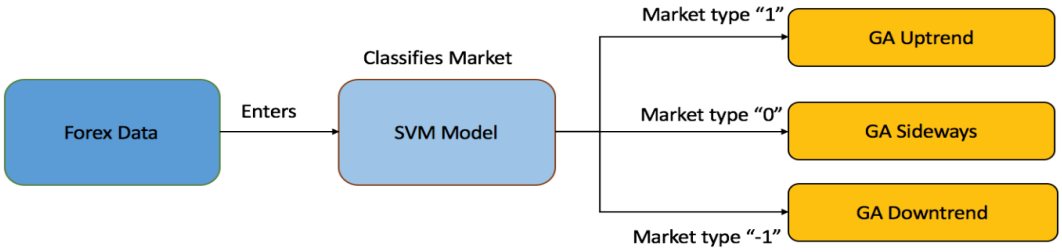

Figure 21 – Schematic that connects the SVM and the GA module together

This solution provides three approaches to solve Dynamic Problems in the *Genetic Algorithm*, which are:

- ***Associative Memory*** – The *SVM* provides the information of the markets(environment) that is saved along side the best individuals of each population, providing a solution where the GA uses this information to adapt to each market (environment) according to the continuous change of FX price sequences.
- ***Hyper-Mutation*** – This method provides a temporarily raise up of the mutation rate, in order to introduce more diversity in the GA and to avoid local maximums. This method makes the GA more adaptable to find new optimal solutions instead of being stuck in a solution that may have been good for a given price sequence, but has gotten worse every time that price sequence is updated.
- ***Hype-Selection*** – This third approach also provides a temporarily condition to the GA but this time is to increase the selection pressure, where the quality of the solution is obligated to increase. This approach is very helpful in the way that it makes possible to increase the quality of the solution without becoming over fitted because it is only a temporary solution.

d) Proposed Stopping Criteria

A proposed stopping criteria was introduced with the objective of increasing the adaptability to new solutions in the search space, rather than having a fixed number of generations. This solution is inspired on [7], where the generations increased limitless as long as the *GA* finds a better solution after each generation. In this work the generations can be limitless as long as the GA finds a solution with better fitness, but the number of generations without finding a better solution is set to 30 generations instead of 10. After a new solution is found, the number of generations is reset to zero and the process is repeated. The reason to choose 30 generations is to explore new solutions that may appear in generations higher than 10, due to the diversity introduced, with the methods shown in section 3.2.3 c. The Stopping Criteria has the following procedure:

- Calculate fitness for best individual of present generation;
- If fitness is higher than the fitness from the previous generation, reset the number of generations to zero
- If fitness from the present generation is not higher than the previous one for 30 generations, then convergence is obtained.

e) Random Immigrants Function

The *Random Immigrants* function is commonly used in the literature, e.g., on works [7] and [44]. For this work, a modified function based on the same principles is introduced. The idea behind this modified function, is that, the *Random Immigrants* are not introduced in same proportion every time, i.e., instead of having a 30% ratio of *Random Immigrants* every time the function is activated, the idea is to evaluate each individual in the population and then eliminate the ones that present themselves below the average fitness. This way, it is possible to only eliminate bad individuals instead of eliminating the 30 % worst. This implementation's objective is to improve the diversity of the populations produced by the three *GAs* without increasing the probability of loosing a good individual. Figure 22 shows the pseudo-code for the modified function.

```
Set Population after Genetic Operators

for individual in population

    individual fitness= calculate fitness
    add to population fitness

average_fitness=Calculate Average fitness

for individual fitness in population_fitness

    if individual_fitness<average_fitness

        Substitute individual for random individual
```

Figure 22 – Pseudo-code for modified random immigrants function

f) Leverage

Leverage is explained in section 2.3 and as it was said before it can bring great returns but also great amounts of risk. To limit the risk exposure, one made the decision to limit the leverage multiplier to a maximum value of 10. This financial tool was applied as the following procedure:

- If the trading rule and the *SVM* classifier are combined, i.e., if the trading rule votes a "Buying Rule" and the *SVM* classifies the market as *Bullish*, or if the trading rule votes a "Selling Rule" and the *SVM* classifies the market as *Bearish*, then leverage is activated; The second step to activate leverage is defined by the *Moving Average* of the "High" values combined with the *Moving Average* of the "Low" values, of each closing price e.g. If the *Moving Average* of the "High" and "Low" values cross the respective "High" and Low" values to an uptrend when the trading rule votes a "Buying Rule", or to a downtrend when the trading rule votes "Selling Rule", then leverage is applied. Figure 23 shows a chart with entry points where leverage is activated.

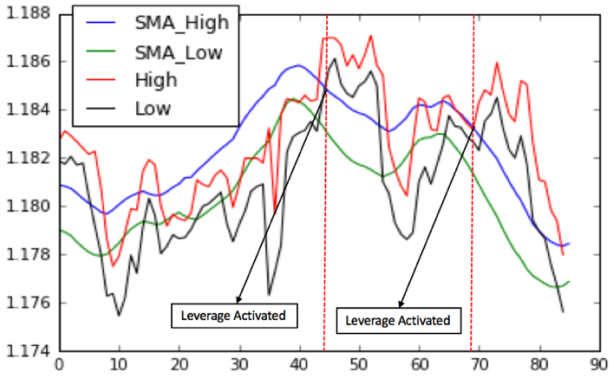

Figure 23– Entry Points where Leverage is activated

- The amount of leverage used, is part of the chromosome structure and when the leveraged is activated the amount of leverage goes between 2 and 10;

3.3 Chapter Conclusions

This section summarizes the work done in this chapter. The overall architecture was described, dividing the different layers in functional modules, in order to make it easier to develop and implement the solutions presented. This chapter also provided the frameworks and python libraries that are used.

4. System Validation

This chapter presents, describes and discusses different strategy evaluation metrics, also it is intended to explore different case studies about the different approaches of the *SVM* and the *GA*, as well as the experiments regarding the investment rules. The algorithm is put against the proposed benchmarks (*B&H*, *S&H* and *Random Walk*) and will be put against a *Static GA* using the most common parameters seen in the literature.

4.1 Evaluation Metrics

To asses the quality of each strategy it is necessary to compare them with the same evaluation metrics. In this work, one used *Return On Investment* (*ROI*) and *Drawdown* as the evaluation metrics to compare investments in the different strategies. Further on this chapter, evaluation metrics for the *SVM* will be taken into account, to assess the quality of the *SVM* model. Those metrics are: *accuracy*, *precision* and *recall*.

4.1.1 Return On Investment

ROI is one of the most used evaluation metrics in the market, and is easily understandable. This metric is the mean profit resulting of an investment strategy, and as described before, can be calculated by equation 14:

$$ROI = \frac{Returns - Initial\ Investement}{Initial\ Investment} \tag{14}$$

Table 11 shows an example of a FX investment, to better understand the evaluation metric from here on out.

Table 11 – Example of a FX investment

Market	Initial Investment	Returns	ROI
EUR/USD	100 000 €	150 000 €	50 %

This metric does not take into account risk measurements, although, this metric provides a faster computation speed to the *GA* making possible to perform different methods in the algorithm, nonetheless, in this work it is important to account for risk because of the leverage. Said that it is introduced a *Stop Loss* of 0.5% loss to account for risk.

4.1.2 Drawdown

Drawdown is a very important metric to measure risk. It can be a very close approximation to calculate the risk associated with an investment since it is very difficult to measure risk with a well defined equation. If an investment wants to take into account the risk, it is always advisable to choose a good trade-off between the ROI and the drawdown. The *Drawdown* can be calculated as the difference of the highest local maximum and the highest local minimum. Figure 24 shows an example of a calculation for *Drawdown*.

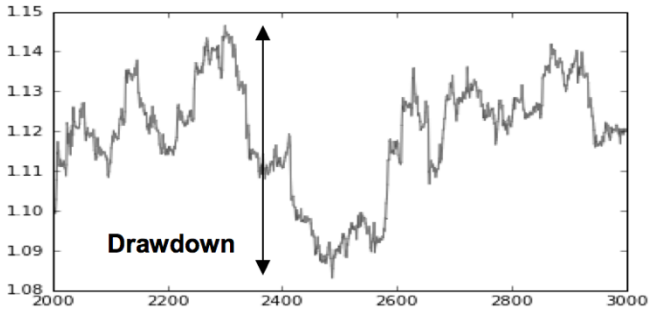

Figure 24 – Drawdown schematic

4.2 Case Studies

This section describes several case studies in which are analysed and compared various changes and configurations of both *Genetic Algorithms* and *Support Vector Machines*, as well as the investment strategies used.

4.2.1 Methodologies

The Case Studies are divided into two groups, being the first group assigned to study the SVM experiences and the second group to describe the experiments made with Genetic Algorithms. For every case study, the price periods used were between 2003 until 2015, also, for both groups the training and test sets were divided equally. For the training set the time periods are from 2003 and 2015, and the test set is the time period of 2016 until the 2nd of March.

Admitting that only the correct evaluations are important as a metric of quality seems to be incomplete. For the *SVM*'s case studies, it is necessary to evaluate the quality of a classification using the metrics, described in section 3.2.3. Reminding that, *precision* and *recall* are calculated as it is described in equation 15 and 16, respectively.

$$Precision = \frac{True\ Positives}{True\ Positives + False\ Positives} \tag{15}$$

$$Recall = \frac{True\ Positives}{True\ Positives + False\ Negatives} \tag{16}$$

An ideal system would present a high precision and high recall, and that is the objective that this works is trying to achieve.

As for the Genetic Algorithm, the evaluation is based on the testing set and then compared to the benchmarks. Several changes are made to the *GA* in order to increase the amount of experiences and depth in the study. It is intended to start with a simple *GA*, with traditional *GA* operators, and then evolve to more dynamic methods. The evaluation of this algorithm is put against the random, B&H, S&H and a *Static GA* approach. The algorithm will be tested in a 50 trial runs and for each case study it is presented the results achieved that include the evaluation metrics used. Finally, the results shown are always referring to the test results of every experiment.

4.2.2 Case Study A – Comparing Features for the *SVM Model*

In this case study, it is made a comparison between the two types of features, that were referred in section 3.2.2, *Price Sequences* or *Technical Indicators*. To compare both approaches, 50 executions were made. The case study also shows the comparison of different kernels and shows the results for the best parameter, in order to describe the evolution of the SVM Model until the final result. Table 12 shows the configuration parameters in which the FX markets were experimented for the development of the *SVM Model*.

Table 12 – Configuration parameters

Parameters	Value
Market	Forex – EUR/USD
Training Period	01/01/03 – 01/01/2015
Real Test Period	02/01/2015 – 02/03/2016
Number of executions	50
Number of Samples	4000

a) Setting The SVM Parameters

The price sequence length was set to 100 because, the objective is to classify a pattern that is useful to classify the market, thus, having to be big enough, so the results may be considerable viable. Before describing the comparison between both features, it is necessary to describe the parameters of the SVM, so the experiment has a baseline, to facilitate the understanding of the experiment's evolution. As it was described in section 3.2.3 a, the parameters for the *SVM* Model were obtained through a *Grid Search Algorithm*, in order to find the best parameter values for each experiment with the two sets of features.

a.1) Parameter Analysis for Price Sequences Approach

This section compares the performance of each kernel and the parameters *C* and *Gamma* that together make the best set of parameters for price sequence features with 100 prices, i.e., 100 features. The objective of this experiment is to study the best parameter for the *SVM* Model to classify a graphical pattern of the 100 prices.

Figure 25 shows the histogram of the score (combined precision and recall) between the three types of kernels used in this experiment. As expected it is possible to conclude, the best solution given by the *Grid Search Algorithm* is the *RBF* kernel, where the best *Gamma* and *C* parameters are described in table 13, because of the great amount of features that this method uses. RBF kernel obtained average scores of 86% against 74% of the *Polynomial* kernel and 69% of the *Linear* kernel. Both *Polynomial* and *Linear* kernels show a higher standard deviation compared with the *RBF* kernel, due to the fact that this two kernels typically do not behave as good as the *RBF* kernel in high dimensions. In the 50 executions, the best parameters for each kernel are described table 13, indicating that the best results for polynomial and linear kernels are over fitted, due to the fact that their hyper-parameters are to high, contrarily to the *RBF* that presents acceptable values for its parameters. High parameters may expose to high false positives and false negatives, that further on, may influence poorly the precision and recall metrics of the *SVM* classification.

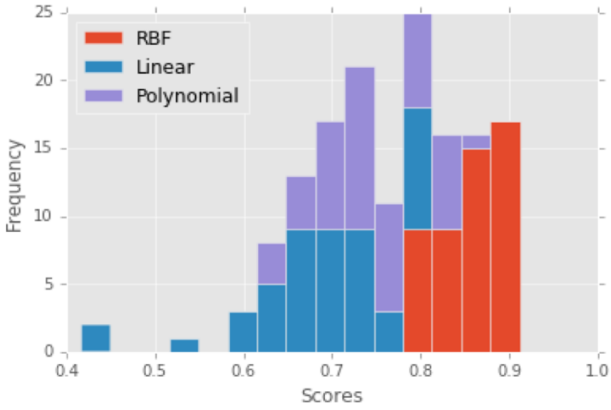

Figure 25 – Histogram for the Kernel scores with Price Sequence approach

Table 13 – Results for the Kernel with Price Sequence approach

Parameters *RBF*	**Gamma**	**C**
Best Score	0.001	10
% of times of best parameter	67	76
Avg. Score (%)	86	
Std. Dev. (%)	3.6	
Parameters *Polynomial*	**Gamma**	**C**
Best Score	0.0001	100
% of times of best parameter	42	57
Avg. Score (%)	74	
Std. Dev. (%)	6.1	
Parameters *Linear*	**Gamma**	**C**
Best Score	Not Applicable	1000
% of times of best parameter	Not Applicable	57
Avg. Score (%)	69	
Std. Dev. (%)	8.2	

a.2) Parameter Analysis for Technical Indicators Approach

The technical indicators used in this experiment were described prior in section 3.2.2 a, being the ones that are used as features in the this *SVM Model* approach. The difference of this approach when compared with the *Price Sequence* approach is that this one does not classify graphical patterns, instead, classifies the correlations between the TIs and the type of market they are in, i.e., instead of classifying 100 prices, the SVM model sets its parameters for the stored information about the 100 prices inside a technical indicator.

Figure 26 shows the histogram for the scores of the parameter *Grid Search Algorithm*, where the best parameter is still the *RBF*, although in this approach the standard deviation for the average scores is higher than expected. Table 14 shows the best hyper-parameters for each of the kernels, and also the average results for the 50 executions that were made. *RBF* kernel presents the best average score with 74% against 69% and 58% of the *Polynomial* and *Linear* kernels respectively. Figure 26 also shows that, this approach presents high standard deviation values, indicating that, may not be a solid solution for every type of *FX* market. Finally, it is important to refer that every approach presents hyper-parameters that are too high, thus, indicating that this model may over fit the *SVM* classifier.

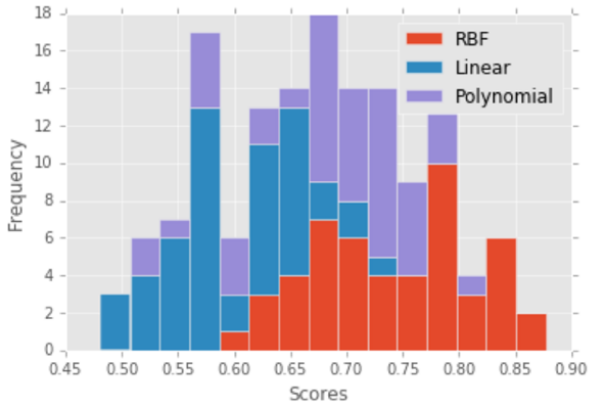

Figure 26 - Histogram for the Kernel scores with Technical Indicator approach

Table 14 - Results for the Kernel scores with Technical Indicator approach

Parameters RBF	**Gamma**	**C**
Best Score	0.00001	100
% of times of best parameter	57	63
Avg. Score (%)	74	
Std. Dev. (%)	7.1	
Parameters Polynomial	**Gamma**	**C**
Best Score	0.00001	1000
% of times of best parameter	45	63
Avg. Score (%)	69	
Std. Dev. (%)	7.4	
Parameters Linear	**Gamma**	**C**
Best Score	Not Applicable	1000
% of times of best parameter	Not Applicable	67
Avg. Score (%)	58	
Std. Dev. (%)	5.7	

a.3) Comparing Both Approaches

To summarize this section, it is important to compare both approaches to understand which one may be the best solution to implement for the rest of the experiment. The *Price Sequence* approach obtained better results, e.g., the average score was 12% higher, the standard deviation was almost 50% lower than the *Technical Indicator* approach, as for the hyper-parameters, *Price Sequence* approach indicated a more reliable solution, since the results were lower than the *Technical Indicator* approach, thus showing higher expectations related to the *SVM* classifier metrics, where the first solution does not show signs of over fitting.

b) Metrics Evaluation for SVM Classifier

The SVM algorithm was trained 50 runs, and the data used to "teach" the model, goes from 2003 until the 1st of January of 2015, using a sliding window approach in order to maximize the solution as much as possible. It was also used 4000 samples to train the *SVM model*, to use more than 4000 samples would require more computational power, and the results would not present a considerable growth. This section follows the logic of the previous section, where a comparison is made between the *Price Sequence* approach and the *Technical Indicator* approach, and concluding which one of the methods is the better one.

b.1) SVM Classifier with Price Sequence Approach

The objective of this approach is to use the 100 price sequence, i.e., 100 features to classify three different types of markets. The model is trained and then Cross-Validation is applied, during the training period. The *Cross-Validation (CV)* method applied was the 5-Fold strategy.

Figure 27, describes a scatter plot for the results obtained by the metrics used in this paper during the test period. Results indicate that label "0", i.e., the label that classifies *Sideways* markets, it is worse than the others. A reason for this to happen, is that this label is more difficult to classify than the *Up* or Down trends, even for the human eye, because it has a higher degree of subjectivity. Nonetheless, it is possible to see that from the metric's results, it is clearly that the *SVM model* can predict successfully the type of markets that are sent to the *GA*, showing promising results for the algorithm proposed in this work. Table 15 describes the average scores of the 50 executions, and it is possible to conclude that the *SVM Model* works as a reliable classifier, making an average score of 82%.

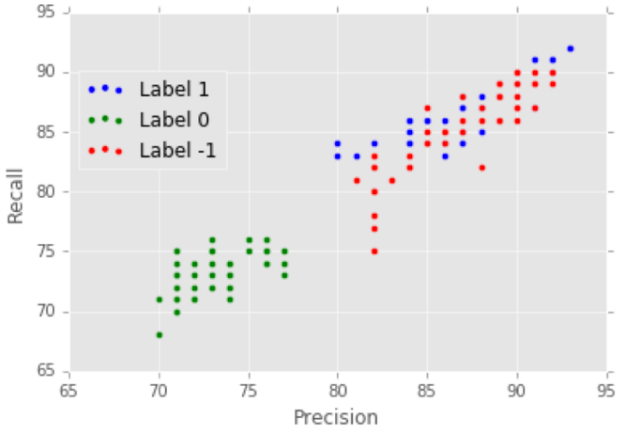

Figure 27 – Average metric results of each label with Price Sequence approach

Table 15 – Average scores for the 50 executions using Price Sequence approach

Parameter	Precision	Recall	Accuracy
Avg. Score for each label (%)	[87.4, 73.2 , 86.8]	[87.1, 72.9, 85.2]	85. 64
Avg. Score (%)	82	82	85. 64
Max. Score (%) for each label	[93, 77, 92]	[92,76,90]	91
Min. Score for each label (%)	[80, 70,81]	[82, 69, 75]	76
Std. Dev. for each label (%)	[3.15, 2.04, 3.31]	[2.59, 1.8, 3.74]	3.934
Avg. CV scores (%)	[84.3, 82.5, 84.3, 84.1, 86.2]		

b.2) SVM Classifier with Technical Indicator Approach

This section describes the approach of a SVM classifier with technical indicators as features. This approach uses the six technical indicators chosen for this work and the objective is to classify the market based on the information that each technical indicator has.

As it was seen in the choice of the best kernel and its hyper-parameters, this approach seems to show less confidence than the first approach. Figure 28 shows that the average metric results have a higher standard deviation, showing that this solution may not be reliable. The reasons for this to happen is that this approach may need more samples than the first one to achieve good results, due to the fact that the correlation between the several TI's may not be an easy problem to solve. The samples used in this approach were also 4000 samples, and the computation came to be heavier than in the first approach, so it may not be reliable to increase the number of samples when the other approach returns better results. Moreover, this approach also shows better results for label "1" and label "-1" when compared with label "0". Table 16 describes the average scores that were obtained in this experiment.

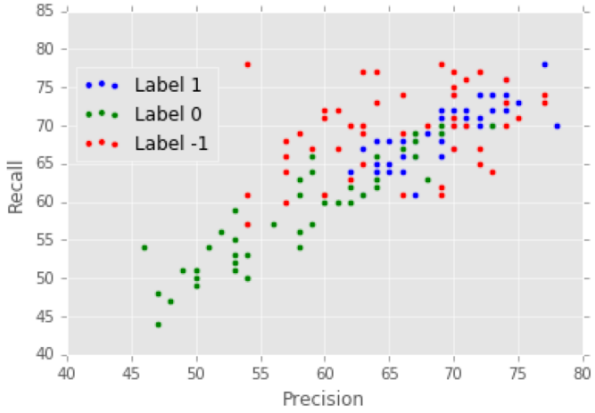

Figure 28 – Average metric results of each label for Technical Indicator approach

Table 16 - Average scores for the 50 executions using Technical Indicator approach

Parameter	Precision	Recall	Accuracy
Avg. Score for each label (%)	[69.5, 57.7 , 65.8]	[68.8, 58.6, 69.56]	65. 34
Avg. Score (%)	64.3	65.65	64. 90
Max. Score (%) for each label	[78, 73, 77]	[78,70,78]	74
Min. Score for each label (%)	[62, 46,54]	[61, 44, 57]	53
Std. Dev. for each label (%)	[4.09, 7.23, 6.61]	[4.16, 7.13, 5.64]	5.63
Avg. CV scores (%)	[66.3, 63.4, 64.2, 66.2, 62.1]		

c) Comparing the two features for the SVM classifier

The first approach clearly outperforms the second one in every aspect, showing better metric results and average scores. Also shows, that the first solution proved itself to be more consistent throughout the executions, being more reliable to implement with the *GA* for the proposed system. Moreover, this solution is faster in computation than the second solution, when compared with the same number of samples. Figure 29 shows a histogram that presents the results for the 50 executions, comparing the scores for average precision and recall of the 3 labels. The values obtained on *Precision* and *Recall* for the *Price Sequence* (PS) approach are positioned in the interval of [75,85], while the values of *Precision* the *Recall* for the Technical Indicator (TI) approach are on the interval of [55,75], showing a disperse and volatile solution thus indicating that, the *Price Sequence* approach is better than the TI approach. The objective is to have both high *Precision* and *Recall* and the only approach that obtains that criteria is the *PS* approach. This criterion has to be accomplished in order to ensure that the model is not over fitted, i.e., having a good training model with good metric results and after in the test period returning a lot of false positives and false negatives.

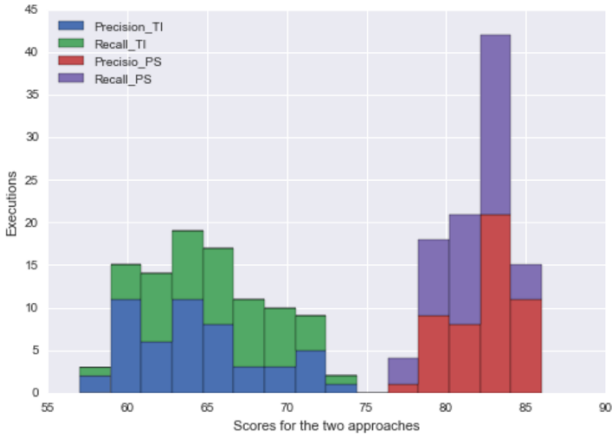

Figure 29 – Histogram showing the results for 50 executions comparing the two approaches

4.2.3 Case Study B – Impact of parameters in the GA of the Proposed Solution

In order to provide the best solutions, it is necessary to evaluate different parameters in matters of size of the population, number of generations or even mutation rates. This case study focuses on different configurations for parameters that are shown in table 17, in order to maximize the performance of the *GA*. The baseline configuration is 100 Generations, 100 individuals in a population and 20% of *Mutation Rate (MR)*. The *Proposed Stopping Criteria* (PSC) was presented in section 3.2.3 d, and is put against all the other configurations also. Note that, in this experiment, the number of executions is also 50.

Table 17– Configuration Parameters

Number of Generations	Size of Population	Mutation Rate(%)	Configuration
50	100	20	1/2 Gen
100	100	20	Base
200	100	20	2xGen
Proposed Stopping Criteria	100	20	PSC
100	50	20	1/2 Pop
100	200	20	2xPop
Proposed Stopping Criteria	50	20	1/2 Pop-PSC
Proposed Stopping Criteria	200	20	2xPop-PSC
Proposed Stopping Criteria	100	10	1/2MR
Proposed Stopping Criteria	100	40	2xMR

a) Parameter Evaluation

The evaluation metric used to perform this experiment is the *Return on Investment* (ROI) and the *Drawdown* and the results obtained are from the period presented in table 18.

Table 18 – Experiment configuration

Parameters	Value
Market	Forex – EUR/USD
Training Period	01/01/03 – 01/01/2015
Test Period	02/01/2015 – 02/03/2016
Number of executions	50

Figure 30 shows the histogram of the configurations for the variation of the number of generations. The results obtained are referred to the ROI during the 50 executions.

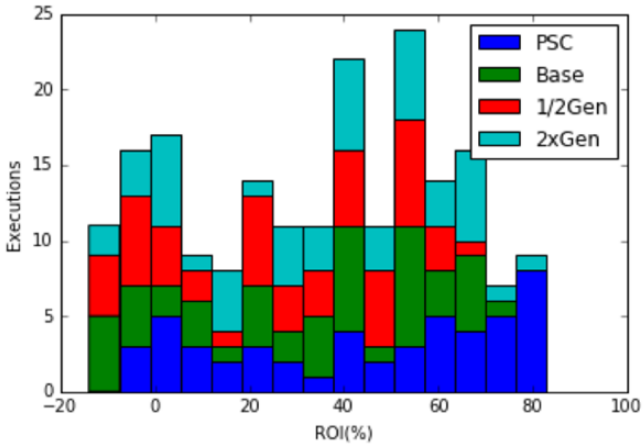

Figure 30 - Histogram of the configurations for the variation of the number of generations

From the results obtained by varying the number of generations, it is possible to conclude that the *Proposed Stopping Criteria* (PSC) has an average ROI higher than the others in the 50 runs. PSC scored 43.8%, against the 32.2%, 27.1% and 34.0% of *Base*, *½Gen* and *2xGen*, respectively. Although the proposed solution returned better results, the runtime of this configuration may take longer than the other two configurations, because the stopping criteria depends on the fitness convergence that the algorithm goes through. A reasonable number of generations for the algorithm to converge is 30 generations. This value was chosen to make a trade-off between runtime and early convergence.

Values higher than 30 could increase run time to 5, to 10 times higher than the other solutions, making it an inviable solution to address the problem in question. *½ Gen* returns worst average results than the other configurations, due to the fact that this configuration may not have enough generations for the population to converge to an optimal solution. On the other side, *2xGen* may be a to "static" configuration, for reasons such as:

- To many generations may be a waste of time and computation, i.e., hypothetically if the optimal solution is found in the first generation and the best fitness does not change during a large number of generations, then it is not needed to keep them running until the end.
- The solution may get over fitted throughout the generations. If the solution does not change, but the algorithm still makes the chromosomes go through the genetic operators over and over until the end of the 200 generations, then it is possible that the population gets over fitted and converges to a local maximum, where the chromosomes converge around a fitness value.

These two possibilities may be avoided with a *Stopping Criteria*, nonetheless, it is very important to define a good convergence criterion, otherwise the algorithm may become very heavy on computation, endless or may loose the optimum solution along the way.

Figure 31 shows the histogram of the results for the configurations where the population varies from *½ Pop* and *2xPop,* comparing also the configuration with the stopping criteria.

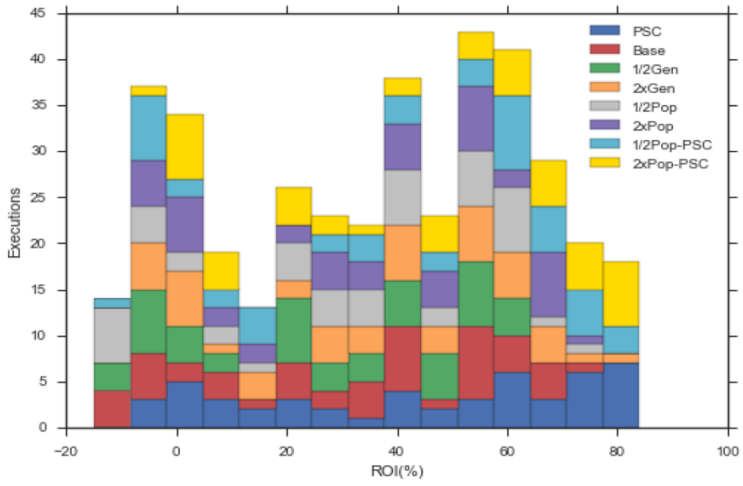

Figure 31 - Histogram of the results for the configurations where the population varies

As expected, the configurations with 2xPop return better average results than the ones with ½ Pop, being that the average score for 2xPop, with 100 generations, is 34.9% and the 2xPop - PSC is 44.6%. Although the 2xPop - PSC configuration has a higher run time than the other it does not affect the performance of the algorithm in a considerable way, so it is still possible to use that configuration. The size of population was chosen in order to achieve a trade off between the best solution and the computation speed of the algorithm. The results are aligned with the theoretical assumptions, because it is only natural that a population that starts with the double of chromosomes has twice the chances of finding a good solution for the problem. Moreover, the histogram shows that the PSC outperformed the baseline configuration of 2xPop also, giving high confidence for the proposed solution. Finally, the histogram shows that the configuration of ½ Pop shows more disperse returns over the 50 executions,

due to the fact that in every run the chromosomes that are created are random, concluding that, bigger populations indicate less deviation in that randomness because a bigger population has a higher probability of having a better solution.

The histogram in figure 32 shows the third experiment with different *Mutation Rates* for each configuration, using the PSC as *Number of Generations*.

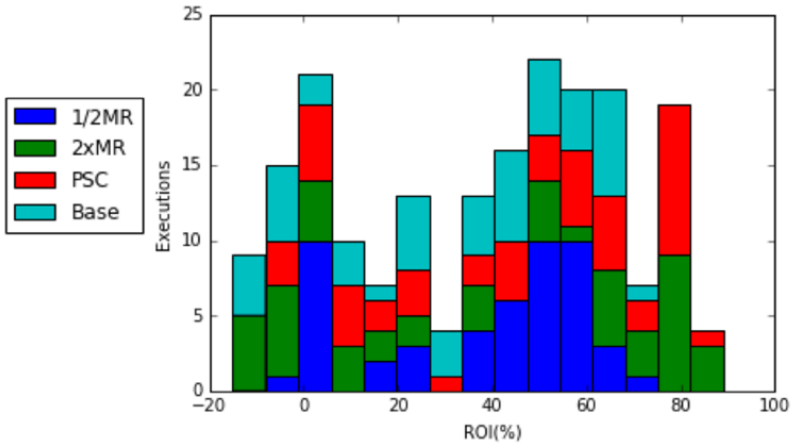

Figure 32 – Histogram for varying mutation rates

Table 19, describes the scores obtained for each configuration, where the 2xPop-PSC returned slightly better average results than the other configurations, with 44.6% of average returns. Although the ROI obtained for this configuration is only slightly better than the PSC, the results were expected due to the fact that the population size increased, thus, providing more individuals that make the probability, of finding an optimal solution, to increase.

A more interesting result is the 2xMR, where the average result is slightly worse than the PSC and 2xPop-PSC, but it returned better *Max. Return*s compared to the PSC and 2xPop-PSC. This may be linked to the fact that the *Mutation Rate* is higher, thus inducing the probability of discovering new solutions in the search space, that could lead to better results, when compared to the PSC or 2xPop-PSC configuration. Although this configuration may lead to better maximum solutions, it is not advisable to use this configuration as a baseline because, as it is possible to see by its *Average Returns*, the configuration is slightly worse when the diversity introduced is too high, i.e., too much diversity may cause the algorithm to loose its best chromosome because it suffered a mutation, nonetheless it is a good sign that it is possible to use *hyper-mutation*, temporarily, to discover better solutions in the search space, i.e., introducing diversity in extreme cases can reveal good solutions as a dynamic approach.

Table 19 – Scores obtain for every configuration

Method	PSC	Base	$\frac{1}{2}$ Gen	2xGen	$\frac{1}{2}$ Pop	2xPop	$\frac{1}{2}$ Pop -PSC	2xPop -PSC	$\frac{1}{2}$ MR	2xMR
Avg. ROI (%)	43.8	32.2	27.1	34.0	29.7	34.9	40.2	44.6	37.2	38.1
Max. ROI (%)	83.1	74.3	64.6	79.2	72.8	75.1	82.9	83.7	69.4	89.3
Min. ROI (%)	-6.1	-14.2	-13.3	-8.2	-15.0	-4.3	-9.3	-3.2	-2.4	-15.9
Std. Dev (%)	30.2	26.7	24.2	25.8	27.1	26.1	29.7	28.4	23.8	36.2
Drawdown (%)	10.3	9.9	10.6	9.6	11.2	9.3	10.2	9.1	7.9	14.3

b) Conclusion of Case Study B

This section concludes Case Study B, where it is possible to conclude that configuration 2xPop-PSC slightly outperforms the PSC. Although this solution takes a little bit more on computation time it is still reasonable to use it as a solution. As it can be seen by table 19, the configuration 2XMR provides promising results to use *hyper-mutation* temporarily, due to the fact that this approach reaches higher *Returns on Investment* but presents worse *Drawdown* and a higher *Standard Deviation*.

4.2.4 Case Study C – Static VS Dynamic Approach

This case study, compares two different approaches when using a *Genetic Algorithm*, the first section is composed by the experiment of a *Static Genetic Algorithm*, followed by the second section that presents the work developed regarding the *Dynamic Genetic Algorithm* approach, and finally, both experiments are compared against each other. It is intended to discuss the results of this experiment in a more conceptual way, i.e., to better understand the differences between the two approaches it is important to discuss and analyse the measures and criterions used, before discussing the "mathematical" results. Table 20 shows the case study configuration for the work developed.

Table 20 – Case Study C configuration

Parameters	Value
Market	Forex – EUR/USD
Training Period	01/01/03 – 01/01/2015
Real Test Period	02/01/2015 – 02/03/2016
Number of executions	50
Number of Generations	Proposed Stopping Criteria
Population Size	200
Evaluation Functions	• Return On Investment (ROI) • Drawdown
Leverage	[2 , 10]

As it was discussed in the literature, there are many differences between static problems and dynamic problems. One of this work's objectives is to implement dynamic GAs in order to improve the performance of the algorithm. More specifically, this case study intends to compare two approaches, comparing two Genetic Algorithms. The baseline of both GA is the same, the objective is to compare two different approaches for the given problematic, i.e., case study C puts a static GA against a dynamic GA. Table 21 describes the configuration of each one of the GA approaches

Table 21 – Configuration of the two GA approaches

Parameters	Static GA	Dynamic GA
Selection	Tournament Selection	Tournament Selection
Mutation	Gaussian Function	Gaussian Function
Crossover	Two-Cut-Point	Two-Cut-Point
Elitism	Yes	Yes
Immigrant Function	No	Yes
Hyper-Mutation	No	Yes
Hyper-Selection	No	Yes
Associative-Memory	No	Yes

a) Genetic Algorithm – Static Approach Configuration

This section describes a *Static Genetic Algorithm* (SGA) approach, in order to be compared further on the work with a Dynamic Approach. This approach aims to develop a traditional Genetic Algorithm that does not contain dynamical constrains, containing only the original features of a GA regarding *GA operators* and *elitism*.

It is also important to say that the Gaussian Function used has some constraints in order to adapt the mutation for the problem in hand. Such constrains are the following:

- If the mutated gene obtained, returns a negative value, the mutated gene is invalid and automatically gets the value of zero, thus avoiding negative values for the weights or invalid values for the technical indicators that complete the structure of the chromosome;
- The mean value of the Gaussian function depends on each gene, so it is possible to apply the mutation for the two sub-strategies (optimization of weights and technical indicator parameters);
- Standard Deviation value takes into account a factor of 10% regarding each type of gene, for the same reason explained for the utilization of the mean value.

Table 22 shows the configuration used for this approach regarding the *GA Operators* and the elitism function.

Table 22 - Configuration used for static approach

Parameters	Method	Parameter Values
Selection	Tournament Selection	Tournament Size: 3
Mutation	Gaussian Function	• Mean Value: Gene Value • Std. Dev.: 10% x Gene Value • Mutation rate: 20%
Crossover	Two-Cut-Point	Cut-Points: Random within the gene
Elitism	Yes	1%

b) Genetic Algorithm – Static Approach Analysis

This section, analyses the performance of the *Static GA* during the 50 executions and for the experiment configuration shown in table 22. As it can be seen by figure 33, where the histogram for the results of the 50 executions is shown, the solution of the *SGA* presents a wide variation on the results, where 90% of the runs obtained a positive *ROI*, but 26% of the runs obtained a *ROI* between 0 and 10%. Although the results are, in the majority, positive, this solution presents it self as being too volatile in its results. This scenario, may be linked with the fact that, as it was explained before, the FX market is very volatile and oscillatory, thus being dynamic. Nonetheless, this solution shows good results, when taking into account that is a very simple strategy that has a wide number of solutions to explore, it can be considered a good starting point to develop a *GA* that is suitable for this problem.

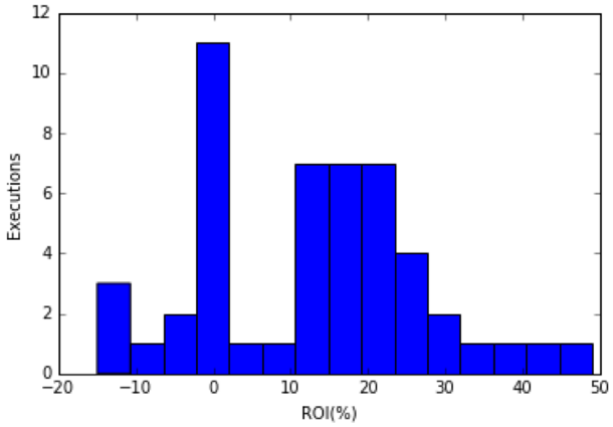

Figure 33 – Histogram for the 50 executions of the SGA

Figure 34, presents the solution for the results of the average *ROI* for the *SGA* and the *Best SGA*. During the test period, it is possible to see that this solution is very volatile when analysing the *ROI*. As for the average returns, the average *ROI* obtained was 12.5% and for the *best SGA*, 43.2%, although the best *SGA* shows that this solution is even more oscillatory than the average solution.

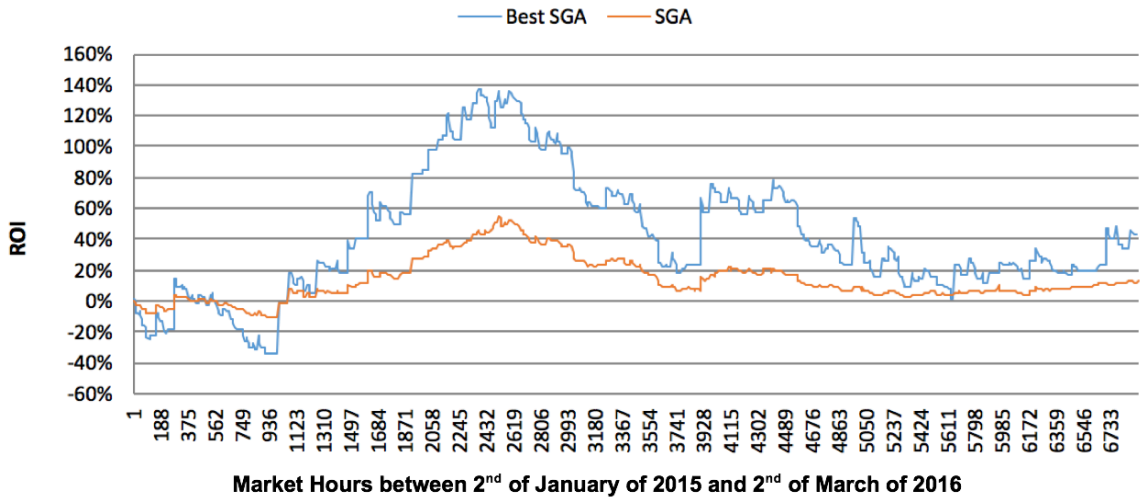

Figure 34 - Results of the average *ROI* for the *SGA* and the *Best SGA*

The *SGA* solution provides a good starting point for the problem proposed for this thesis, but it is not enough, due to the fact that this solution it is not capable of complying with certain characteristics of the FX market, such as volatility, and sudden changes induced by exterior factors like important news or Central Banks that influence the index. Moreover, this solution is not capable of adapting to reality as fast as one wanted, and that can be seen by figure 34, where the *ROI* takes great variations that are not good for the investor.

c) Genetic Algorithm – Dynamic Approach Configuration

The following section describes the work developed for the *Dynamic Genetic Algorithm* approach, where the configuration shows, in table 23, the methods used to perform the *Genetic Operators* and also the approaches introduced to develop a *DGA* approach. Moreover, this section, presents the result analysis of a *DGA* and how the proposed system performs in a FX market. The proposed system intends to explore adaptability, while using *hyper-mutation* and *hyper-selection*, explore memory methods, using an *Associative Memory*, where it relies on the *SVM* classifier to obtain environment information.

Table 23 – Configuration for the DGA approach

Operators and Constraints	Method	Parameter Values
Selection	Tournament Selection	Tournament Size: 3
Mutation	Gaussian Function	• Mean Value: Gene Value • Std. Dev.: 10% x Gene Value • Mutation Rate: 20%
Crossover	Two-Cut-Point	Cut-Points: Random within the gene Crossover Prob.: 80%
Elitism	Yes	1%
Immigrant Function	Proposed by Author	Yes
Hyper-Mutation	Increase Mutation Pressure	Mutation Rate: 40%
Hyper-Selection	Increase Selection Pressure	Tournament Pressure: 5
Associative-Memory	SVM Classifier	Case Study A

d) Genetic Algorithm – Dynamic Approach Analysis

The following section intends to study the *Dynamic Genetic Algorithm* approach. The experiment is made within 50 executions and in the same train and test period of the *Static Genetic Algorithm* approach. Moreover, this section, intends to prove that some dynamic criterions (there are many that can be considered), shown in section 2.1.3, are compliant with this financial problematic, more specifically in this section, the following criterions are addressed and explained in detail further on:

- Predictability;
- Cyclicity;
- Visibility;
- Time-Linkage;
- Stability;
- Diversity Measures;
    - ⇒ Adaptability.

Figure 35 shows a histogram with the results obtained for the test period, using leverage from the chromosome structure. As it is possible to see, the *Avg. Proposed System* shows steady growth through time and does not present considerable volatility during the evolution of the chart. The *Avg. Proposed System* obtained a *ROI* of 43% and the *Best Proposed System* obtained a score of 83%. The histogram shows that the executions may vary more than expected. This may be related with the fact that the population is not big enough, compared with the size of the chromosome, to evolve to an optimum solution that is identical in every execution, nonetheless, the 50 executions show that in 94% of the executions the *Proposed System* obtains positive ROIs. Figure 36 shows that the *Proposed System* behaves with great stability, and as it was said before, it has a steady growth. The best solution within the 50 executions has a little bit more volatility, but it is still possible to see a steady growth, only with bigger variations. This is normal due to the fact that one line takes one of the executions only, and the other is the combination of the 50 executions.

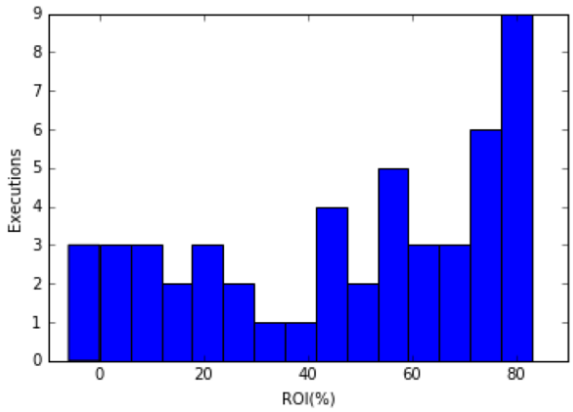

Figure 35 - Histogram with the results obtained for the test period, using leverage

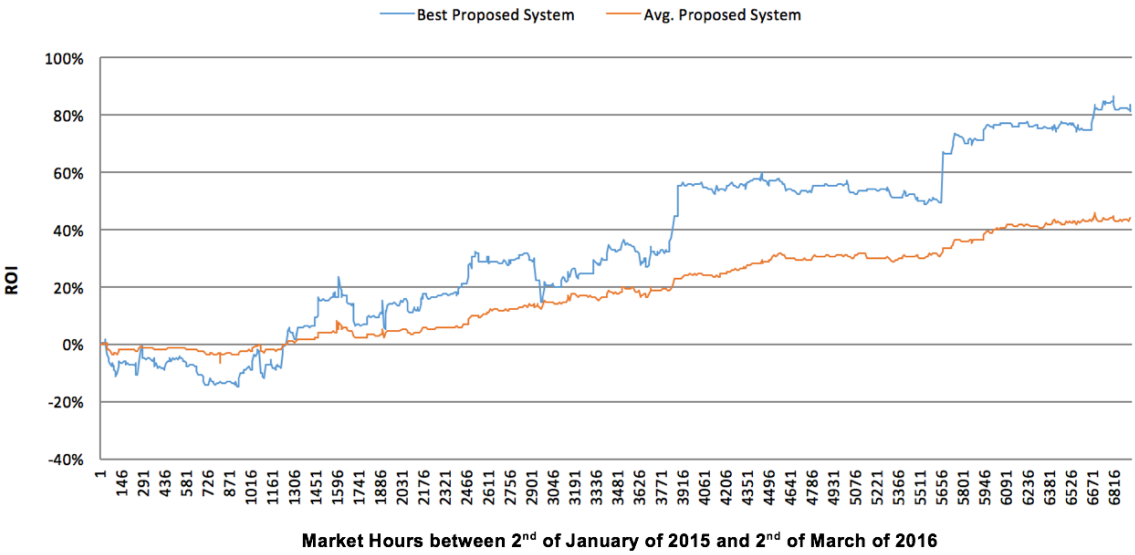

Figure 36 – Results obtained for the Best and Avg. Proposed System with the DGA approach

Before comparing the Static and the Dynamic approach, it is important to evaluate the criterions shown above, in order to evaluate the behaviour and how the proposed system reacts to the FX market.

The following points discuss how the Proposed System reaches those criterions:

- **Predictability** – Although this work does not propose any predictive method like *Hidden Markov Models* or *Naive Bayes* (predictive methods that are usually seen in the literature), the proposed system shows that in the 50 executions, the losses seen are not considerable, even more, the solution presents a steady growth, where, in 57% of times, the Proposed System has profitable positions. By using *Adaptive* approaches (*Hyper-Selection* and *Hyper-Mutation*) and the *Memory* approach relying on the *SVM* classifier, made the *Proposed System*, quite adaptable to new environments, despite of the fact that this system does not contain any "true" predictive method. The *SVM* classifier is used to classify the present hour, based on a time-window of 100 hours, then, the result of the classification is the prediction for the next hour, based on the idea that the type of market remains unchanged in the next hour.
- **Cyclicity** - The *Proposed System* takes advantage of the cyclicity in FX markets, by using the *SVM classifier*, that detects in which type of market the investor is inserted, then, the *Associative Memory* approach used in this work, stores the information (population and environment) about the different markets that are trained by the three *GAs*, in order to use them when the market changes, i.e., the *GA Module* optimizes three different *GAs* depending on the classification of the *SVM* and then stores the information about the population and the environment trained, so that it can be used when the market repeats itself, e.g., let us imagine that the market is *Bullish* (Uptrend), in that case, the *GA* selected is the *Uptrend GA* that has a population that was specifically trained for Uptrends, if the market starts to change and the *SVM* classifies it as a *Sideways*, then the *GA Module* shifts the *GA* for a *Sideways GA*. This characteristic allows the *Optimization Module* to track *Cyclicity* through time, and can shift the populations according to the type of market, if the market is bullish and then shifts to a *Bearish* (downtrend) market and after that returns to a bullish market again, the *GA Module* can adapt and take advantage of an already trained population for that specific environment;
- **Visibility** – There are many ways of detecting changes, whether is in the environment itself or in the performance of the *GA*, the *Proposed System*, as it was referred before, can detect changes in the environment through the *SVM* model, although it is almost impossible to detect sudden changes, that were also referred in previous sections, like important news or press conferences from important politicians or central banks from the world. These sudden changes are impossible to predict when the algorithm only aims to analyse prices and the TIs. So to conclude this idea, the *Proposed System* does not detect those sudden changes, but can detect changes in the environment when its trend changes. As for the *GA* performance, the *Proposed System,* tracks the *ROI* during the investment sessions, and if the performance decreases to losses over 0.5% of the *ROI*, the *GA* closes the position and starts a new evaluation from that point without having to reset the algorithm. This measure proved to be very helpful to control the losses and the *Drawdown* in figure 36, showing a more stable and adaptable solution.
- **Time-Linkage –** During the several experiments and after the careful study made to this problematic, it is possible to assume that this problematic does not present a time-linkage characteristic, i.e., actions made by the selection criterion do not influence the environment.

The FX market is completely independent from an action of a single investor, and the proposed system does not influence the environment whatsoever. The only theoretical way of this solution influencing the market is that if the great majority of traders used it. That option seems far way from reality, so it is automatically put aside. The only real influencers, in the short term, are the central banks, due to the fact that their news influence the great majority of traders, thus making sudden and great variations in the FX market.

- **Stability/Robustness –** The *Proposed System*, as it can be seen in figure 36, shows great stability with a steady growth. It is possible to see that the leverage does not influence the algorithm with great *Drawdowns*, meaning that the implementation of leverage did not destabilized the proposed system. The subject of leverage is discussed in more detailed way in case study E. The introduction of dynamic methods show that the solution obtained, proved to be very stable, as it was intended, the objective is to have a steady growth, rather than huge variations that might have great returns but show great quantities of volatility.

The *Proposed System*, also, proved to be quite robust, because the testing period chosen, has the three types of market addressed in this thesis, and especially the year of 2016, that has been a very difficult year for traders, and yet, the algorithm shows a good performance. The proper comparison with the market are made in case study D.

- **Diversity Measures –** One of this thesis objective is to explore the possibilities of diversity, as it was stated before, the proposed system uses *Adaptability* methods to increase diversity, *Hyper-Mutation* and *Hyper-Selection*, providing new solutions in the search space, i.e., when these measures were activated the *Proposed Stopping Criteria* (explained in section 2.3.2 d) found, for 58% of times, new solutions in the search space, that obtained a better fitness than the previous ones. As it was seen in case study B, higher mutation rate induced higher *ROI*, and also induced higher volatility in the results. In this case study, when introducing hyper-mutation, i.e., when the mutation rate was increased temporarily, the *GA* is able to find solutions with higher *ROI* and at the same time avoid higher volatility because this measure is only temporarily, meaning that, when a new solution is found, the *Stopping Criteria* is activated and the generation number is reset to zero again. *Hyper-Mutation* and *Hyper-Selection* are only activated when the solution does not change for 15 generations, being this method, a good trade-off between diversity, and higher fitness function. Moreover, if the hyper-mutation is used during several generations, it is possible that the best solution may be lost due to a random mutation. Finally, the use of three different *GAs* provided three different populations that were trained for three different environments. Although this method may not be considered a *Multi-Population* method, it provides a reasonable increase in the diversity when compared to a single *GA*.

Finally, the *Dynamic GA* provided a solution that is possible to utilize in real life trading, being that the Proposed System is heavy on computation when running the whole Test Period, that takes about 6 hours, but if one takes into account that this system is ran hourly in real time, then, shows very promising results when referring to computation time and power, i.e., the system takes about 30 to 55 seconds to decide how to invest in the next hour.

e) Genetic Algorithm – Comparing the two Approaches

This section concludes this case study, and compares the two approaches previously analysed. The *Proposed System* with a *DGA* outperformed the *Static GA* solution, both in *ROI*, *Drawdown*, and stability of growth. Figure 37, shows the two approaches together, and it is possible to conclude that the growth obtained by the *DGA* is the one that is intended (with few volatility) for this work, both the *Avg.* and *Best Proposed System* show more stability and better returns than the *SGA*. In addiction to the charts, figure 38 shows a subplot between the FX prices used in the test period and the evolution of *ROI*, where it is shown some points where the *SVM* classifier decides the type of environment is activated. It is important to refer that the points shown are only very few of them, in order to better understand the *SVM* classification. The *SVM* module is used hourly, i.e., every hour the *SVM* classifier sends the environment information to the *GA Module*, and then the *GA* selected, according to the environment, decides how to invest in the FX market.

Finally, the behaviour of both approaches indicates that the *DGA* implemented in the *Proposed System* is capable of adapting to the FX market with more success than the *SGA*, this factor is linked with both *Adaptive* approaches used, and the *Associative Memory* together with the SVM classifier. These methods have shown that it is possible to provide more robustness and less reactivity to the system, when compared with a static approach. Moreover, to quantitate the results in a more detailed way, case study D, provides a comparison with different benchmarks and only then the results are discriminated.

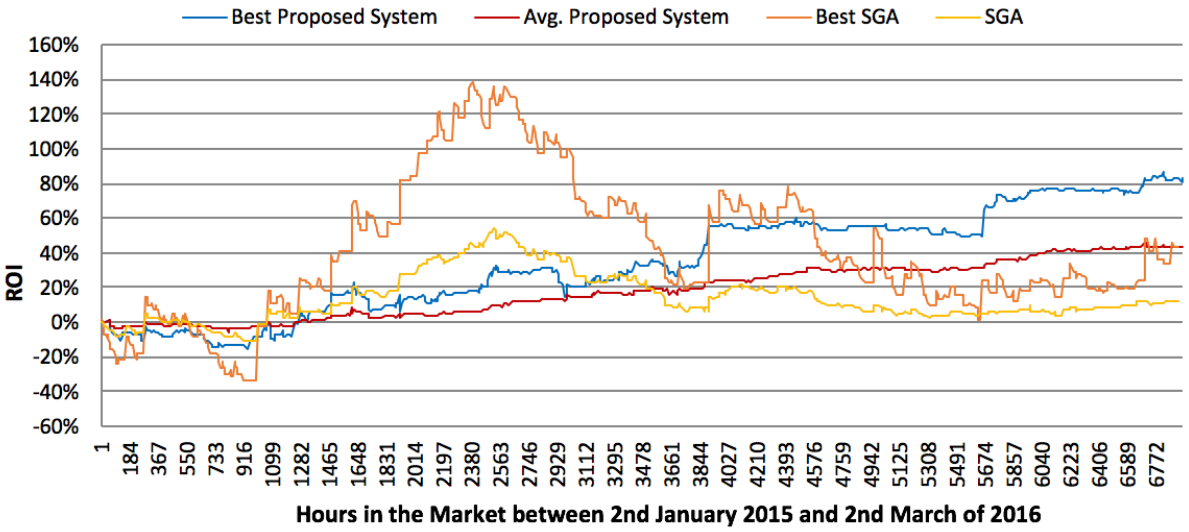

Figure 37 – Results comparing the two approaches

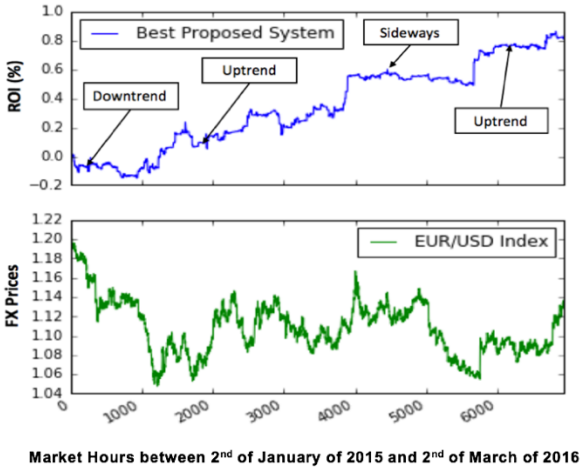

Figure 38 – Evolution of ROI, showing market classifications made by SVM

4.2.5 Case Study D – Comparing Evolutionary Computation with Benchmarks

This case study, focuses on comparing and analysing the results obtained for the proposed system and for the SGA approach against the benchmarks referred in chapter 3, that are, *Random Walk*, *Buy & Hold* and *Sell & Hold*.

In order to include the most possible data for different types of trends in this work training period was set between 1st of January of 2003 and 1st of January of 2015, followed by a test period from 1st January of 2015 until the 2nd of March of 2016. Like stated above, this approach is compared against *Random Walk*, *B&H, S&H*, and a *Static GA* approach.

- ***Buy & Hold / Sell & Hold*** – This two classical approaches take the believe that the market always takes a trend and it is not possible to predict market fluctuations relying on past data, so, whether the market is bullish or bearish, traders takes the position and holds until the opinion is changed.
- ***Random Walk*** – This theory relies on the fact that the market is random, and thus, making random investments as a consequence. This approach decides in totally a random way, if it takes a short or long position and if a position is opened or closed.

Moreover, the *Proposed System* is compared with the work done by Hirabayashi et al. [7], due to the fact that it is based on Forex Market, although this work is based on Japanese Yen (JPY) against US Dollar and EUR, thus making the behaviour of this specific market different from the EUR/USD, it is an important benchmark to compare the behaviour of the returns and in the implementation of leverage. Also, the *Proposed System* is compared with the work done by Sermpinis et al. [26], in terms of *Drawdown*, since there are no charts showing the evolution of ROI. For both works, actual results on *ROI* are not reasonable to compare, because the Testing Period is different, and the type of market is not always the same.

Finally, real life transactions are a bit more difficult to calculate in this work, because each broker decides its own cost transaction. The transaction costs are usually low comparing to other types of markets, thus the results do not diverge considerably when applying an average transaction cost of 0.02% of the transaction made.

a) Parameter setup

The parameters are divided between the two modules, the *SVM* and the *GA*. In this approach it is allowed to perform long and short selling, including leverage, and also, this approach takes advantage of technical analysis only.

As for *SVM* module, the configuration parameters for the *SVM classifier* are shown in table 24. The *SVM* parameters were selected out of 50 experiments set, where these, were the values that presented the best average performance.

Table 24– Best parameter configuration for the SVM Model

SVM Parameters	Values
Model: SVM Classifier	-
Kernel	RBF
C	10
Gamma	0.001
Price Sequence Length	100
Number of executions	50

The results obtained for the parameters, regarding the choice of the kernel used and its *hyper-parameters*, show that the solution does not indicate over fitting in the *SVM model*, because the *C* and *Gamma* obtained are not to "optimized", i.e., in the C interval of [1, 10, 100, 1000], the value obtained is not too high. As for the *Gamma* interval of [0.1, 0.01, 0.001, 0.0001] the value 0.001, although, is a bit closer to the limit, is still a reliable value to use. When comparing the proposed solution against a *Linear* kernel that has *C* parameter equal to 1000, it is possible to conclude that, the parameter setup chosen by the proposed solution, is the most appropriate to design the *SVM* model.

As for the *DGA* configuration, table 25 shows the different methods applied in this process, as well as, the value parameters for the proposed methods. A sliding window approach was used to perform training and test sets. The stopping criteria used in this work is inspired on Hirabayashi's approach [7], where the generations increased limitless as long as the *DGA* finds a better solution after each generation. If there is no better solution at the end of 30 generations then the *DGA* stopped there.

Table 25 – Configuration of the DGA

Parameter	Value
Population Size	100
Mutation Rate	• Hyper-Mutation: 0.4 • Static Mutation: 0.2
Generations	Stopping Criteria
Selection parameter	• Tournament size (TS): 3 • Hyper-selection: TS – 5
Crossover	• Two - Cut – Point • Crossover prob.: 0.8
Random Immigrants	50%
Sliding Window	1 month

b) Evolutionary Computation VS Benchmark Analysis

The following section compares the benchmarks against the *Static GA* and the *Proposed System* (with the *DGA* approach) and the results obtained take into account the leverage introduced by the gene of the chromosome that addresses leverage levels.

The proposed solution, illustrated on figure 39, exhibits a better *Return on Investment* (*ROI*), compared to the approaches described in section 4.1. The proposed system outperforms the other strategies, although this solution is still a bit heavy on computation, and only breaks the other approaches in the end, duo to its steady growth, it rarely looses positions with considerable losses. As it is possible to observe, the *S&H* and the *SGA* approaches, reach higher returns at certain points, but they are very volatile, and the objective is to get a steady growth throughout time. The histogram of the *ROI* described in figure 40, shows that for the proposed approach, 12% of the runs obtained negative values, 36% were values near zero or results that are somewhat similar to the other approaches, but still, for 76% of runs, the proposed system outperforms the other approaches, giving a high confidence for the proposed strategy. Applying the *Random Walk* approach is clearly worse in every run, compared to the rest of the approaches. This method shows that in 90% of times, the results were negative. For the *B&H* and the *S&H* the results clearly depended on the type of market they were in. For *B&H,* 100% of the runs, the results were below the proposed solution, and in *S&H*, 88% of the runs were also below the proposed solution.

The proposed solution is also put against a *Static GA*, *memoryless*, without *hyper-mutation*, and *hyper-selection* features. The results confirm the assumptions made for this work, where it is possible to conclude that dynamic approaches enhance the performance of the *GA*, especially the proposed approach of introducing the *SVM Module* in the algorithm, making it as said before more stable and less exposed to great losses. In figure 40, it is possible to see that the proposed solution outperforms the *SGA* in 66% of the runs.

Table 26 presents a more detailed statistical analysis of the average results over the 50 runs. Moreover, the *Drawdown* obtained for every approach described, shows that the *Proposed System* outperforms the other solutions, indicating, as it was concluded in Case Study C, that the *Proposed System* is a stable and a steady growth approach. It also shows that the proposed solution has low reactivity, which is a positive sign that this method works in real-time investments.

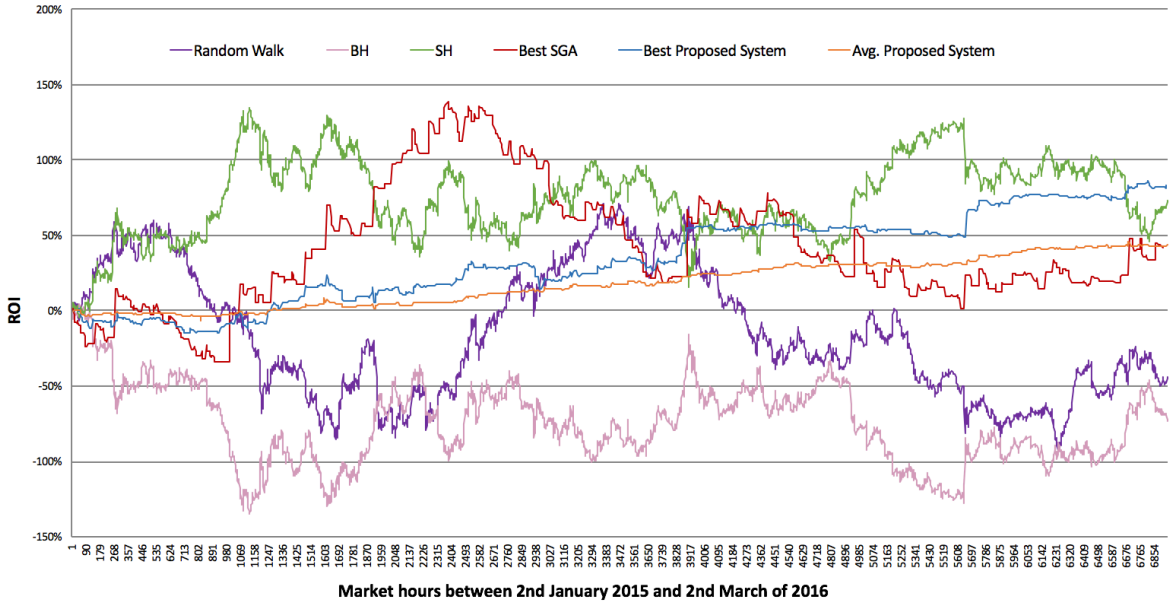

Figure 39 – Returns with leverage obtained during test period for the different approaches

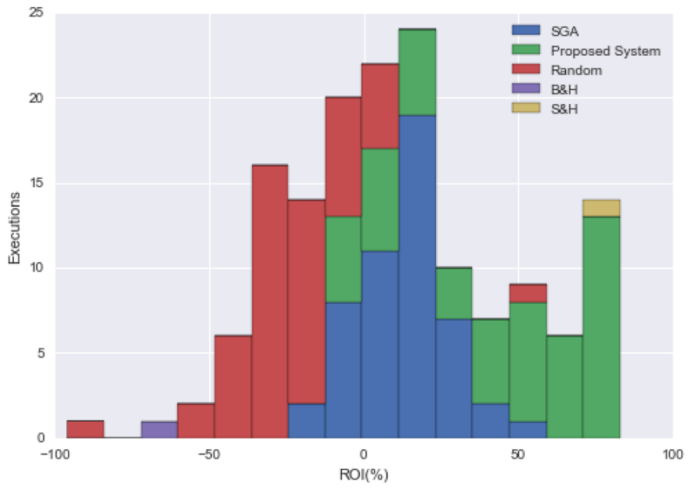

Figure 40 - Histogram with the ROI for the 50 executions

Table 26 - Comparison between the proposed system and the benchmark approaches

Parameter	Random	B&H	S&H	SGA	Avg. Proposed System	Best Proposed System
ROI (%)	-22.3	-72.2	72.2	12.5	43.9	83.5
ROI without Leverage (%)	-5	-7.22	7.22	8.1	8.9	12.8
Profitable Positions (%)	26	31	69	39	57	63
Std. Dev. of ROI (%)	26.9	17.7	17.7	15.4	17.3	30.2
Number of days with negative ROI(%)	68.1	100	0	10.3	15.4	16.5
Max. Drawdown (%)	200	160	20	60	9	14

To conclude this section, it is important to refer that, the solutions shown for the different approaches might loose more money than the initial investment. This situation can be explained by the fact that the representation is only mathematical, and due to leverage it is possible to loose more money than the initial investment, as shown in “Concept of Leverage” (Section 2.3).

c) Proposed System VS State-of-the-Art

When compared the *Best Proposed System* with the leverage approach made by Hirabayashi et al. [7], it is possible to see that the *Best Proposed System* shows a steadier behaviour and less risk exposure when trading EUR/USD. Regarding the non-leverage approach, the *Proposed System* shows higher risk exposure, due to the fact that it is more volatile, when compared with Hirabayashi et al. [7]. Again, these cannot be direct comparisons since the work made by Hirabayashi et al. [7]does not focus on the EUR/USD market and the test period is different, nonetheless, the work made by Hirabayashi et al. [7] is a good benchmark to prove that the *Proposed System* provides promising results.

Although the work developed by Sermpinis et al. [26] does not show a *ROI* evolution in a chart, the results provided, regarding *Drawdown* are of great importance. Table 26 shows that the *Best Proposed System* has a *Max. Drawdown* of 14%, and the results obtained for the EUR/USD in the Sermpinis work show very similar results, of 13.63% for the *GA-SVM* approach, 14.81% for *GA-SVR* approach and 14.38% for *RG-SVR* approach. Although the Test Period is not the same, the results obtained by Sermpinis et al. indicate that the Drawdown obtained for the *Best Proposed System* is a successful result.

Finally, both the works that were compared with the Proposed System proved that is possible to use Evolutionary Computation in the FX market.

4.2.6 Case Study E – Studying the Impacts of Leverage

The following case study analyses the impacts that leverage has, while investing in the FX market. As it was explained in section 2.1.1 c, leverage can bring great returns but also great losses. One of this work’s objective is to implement leverage to get high returns but at the same time, control the risk associated by adding leverage. Further on, this case study, compares the *Proposed System* with leverage against the *Proposed System* without leverage and a strategy where the leverage level is set at 10 against the proposed strategy of optimizing the value of leverage using the GA chromosome. The experiment was made with 50 executions within the parameters set in table 27.

Table 27 – Case Study configuration

Parameters	Value
Market	Forex – EUR/USD
Training Period	01/01/03 – 01/01/2015
Real Test Period	02/01/2015 – 02/03/2016
Number of executions	50
Number of Generations	Proposed Stopping Criteria
Population Size	200
Evaluation Functions	• Return On Investment (ROI) • Drawdown
Leverage	[2 , 10]

a) Studying the Impacts of Leverage – Proposed System

This section analyses the experiment where the *Proposed System* is tested when leverage is applied. The results are put against the *Proposed System* without leverage. A behaviour analysis is made, together with a result analysis in order to prove why using leverage in the *Proposed System* does not present a greater increment in risk exposure.

The *Proposed System* takes advantage of the *GA* chromosome to optimize the leverage level, that is described in the chromosome representation section. This approach outperformed the approach of having a leverage level on a fixed value.

Figure 41 compares the *Proposed System* with leverage against the same *Proposed System* without leverage. The approach with leverage clearly outperforms the approach without leverage, showing that the *Proposed System* makes profitable positions along the test period. Although the leveraged *Proposed System* introduces more volatility and increases the *Drawdown* measure, when compared to the *Proposed System* without leverage, it provides a solution with a steady growth and very few cases where the *ROI* decreases considerable. Nonetheless, those decreasing points do not affect the investment in a way that makes it impossible to recover, Figure 41 shows, for the *Best Proposed System* that, despite of the fact of showing some losses in particular periods, the algorithm always recovered from those losses and ended with a *ROI* in its maximum value.

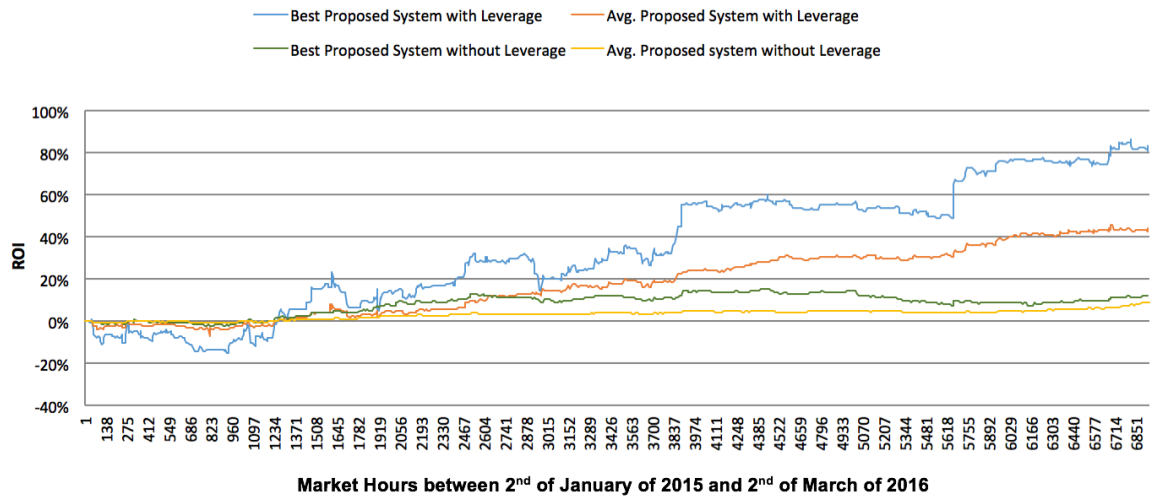

Figure 41 - Comparison the *Proposed System* with leverage against the same *Proposed System* without leverage

The *Proposed System* without leverage obtained a *ROI* of 12.8% and 8.9% against 83.5% and 43.9%, for the *Best Proposed System* and *Avg. Proposed System* respectively. The results show that the leverage introduced in the *Proposed System* increased the performance of the solution, making it possible to obtain higher returns than the variations of the *FX* market, without exposing the investment to excessive risk, as it is possible to see, the *Best Proposed System*, stays negative for 16.5% of the test period. Although the other approaches remain negative for a similar percentage, the *Proposed System*, without leverage, obtained fewer losses. This is an expected result, because leverage might expose the algorithm to higher losses. Nonetheless, the leveraged solution was able to recover,

showing that the leverage level, introduced in the positions where the algorithm looses money, is lower than the leverage levels introduced in the profitable positions.

Regarding the *Avg. Proposed System*, it does not present considerable losses during the 50 executions, which means that, the *Proposed System*, was able to decide with success where to introduce high levels of leverage.

Finally, the *Best Proposed System* with leverage shows that it is possible to take high profits from the small FX market variations, as it can be seen in the chart, where the profitable positions in the *ROI*, from the *Best Proposed System* with leverage, clearly outperformed the same positions from the solution without leverage.

After comparing this two approaches, it is important to describe the behaviour of the *GA* when it comes to choose the level of leverage for the best population. Figure 42 describes the optimization made for the three types of markets addressed in this work. For 50 executions, the *GA* obtained the different optimal solutions for the best population, depending on the type of market, the Y label shows the results for the distributions obtained during the 50 executions, for each market, it is shown the level of leverage optimized for that specific execution.

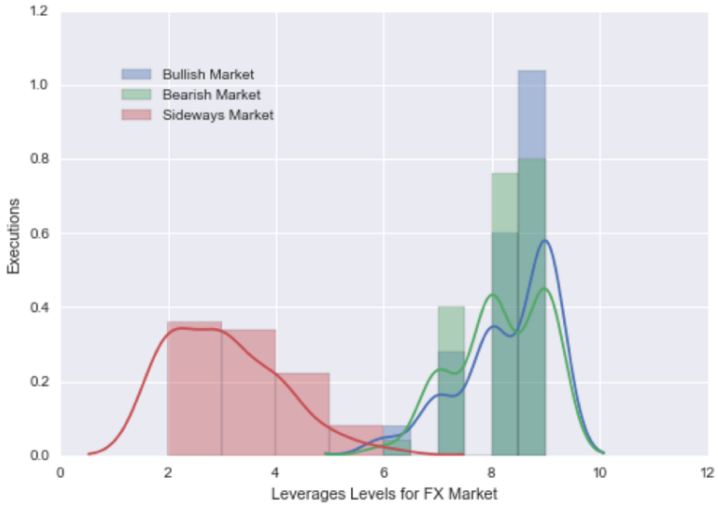

Figure 42 – Leverage levels obtained for the three different markets in 50 executions

As it is possible to see, for both *Bearish* and *Bullish* markets, that are, traditionally, easier to describe and to define, the leverage level optimized is in a great majority of time, a level 9. This result is rather promising because the *GA* can successfully introduce leverage in the higher levels, meaning that, profitable positions are being well chosen for this two types of markets. As for the *Sideways* market, results show that for this type of market, the leverage level obtained with higher density was the level 2 and 3, showing that this type of market is, as it was said before, harder to define and thus harder to make investments in those periods, also, this market shows a wider distribution when compared to the *Bullish* and *Bearish* market, indicating that this two types of markets are more profitable to invest than the *Sideways* market. Results are in line with the theory of *FX* markets, proving that investing in a *Sideways* market is more volatile, due to the fact that, there is no defined trend.

To conclude this section, one can see that the *Proposed System* with leverage outperforms the *Proposed System* without leverage. A very positive outcome is that both results return positive *ROI*s, meaning that the *Proposed System* is well designed and does not return profit only because it has leverage. Introducing leverage provided a great improvement in the results, showing that this tool is very important in a FX market. nonetheless, it is necessary to use it with care, since it can turn against the investor if the exposure to risk is to high, being the main reason to choose levels of leverage that are smaller than the usual used in the investment world. By using high levels of leverage in an evolutionary system, showed that is possible to get high returns without exposing the system to high levels of risk, thus showing that the *Proposed Syste*m is a very promising algorithm to invest in *FX* markets.

b) Studying the Impacts of Leverage – Chromosome Representation VS Fixed Leverage Levels

This section studies the impact of two different strategies when implementing leverage, while the chromosome representation strategy aims to be a more dynamic approach, since the leverage levels are optimized to a specific type of market, the second approach only differs from the first in a single aspect, both have to make the same decision of introducing leverage in a given position, but this method only applies a fixed level of leverage with a value of 10. Figure 43 shows the results obtained for the *Proposed System*, comparing the *GA Chromosome Representation* against the *Fixed Level of Leverage*.

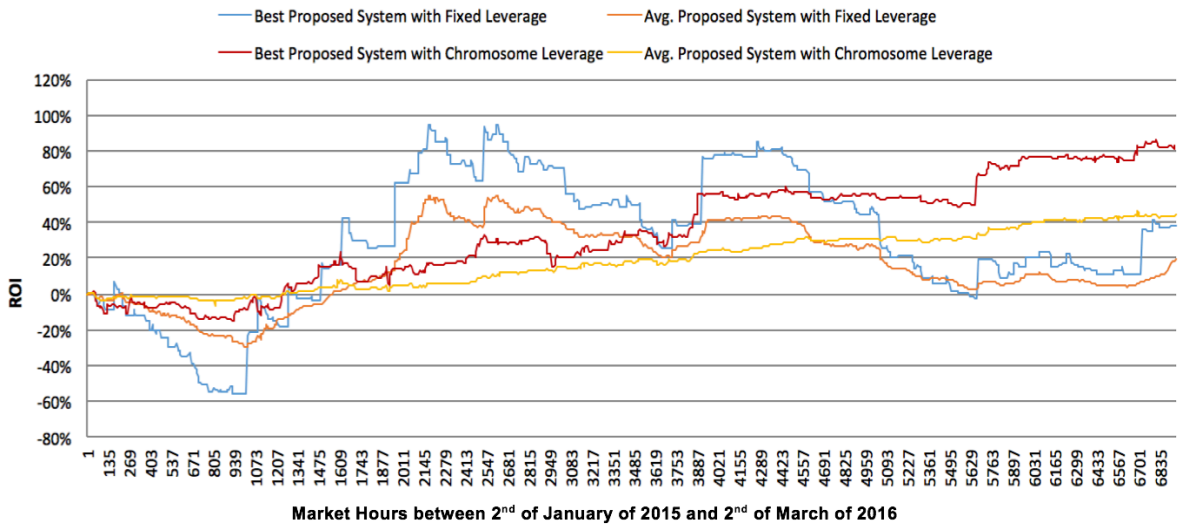

Figure 43 - Results obtained for the *Proposed System*, comparing the *GA Chromosome Representation* against the *Fixed Level of Leverage*

Results show that the *Best Proposed System*, with a varying level of leverage, outperforms the approach with fixed leverage, indicating that, using different levels of leverage produces a steadier solution, rather than using a fixed level of leverage, that naturally, exposes the algorithm to higher risk and provides a more volatile evolution of the chart. The *Chromosome Representation* provides a higher level of leverage for profitable positions and lower levels of leverage for non-profitable positions,

proving, why the *Proposed System* obtains high returns and a reasonably low *Drawdown* that, as it was show in Case Study C, it is very similar with the *Drawdowns* obtained in the Sermpinis et al. [26] work. The *Chromosome Representation* approach obtained a *ROI* of 83%, against a ROI of 37.5% obtained by the *Fixed Leverage Approach,* while the *Drawdown* obtained for the *Best Proposed System* is 14%, the *Drawdown* for the *Fixed Leverage* is 110%, showing that this approach is highly risky and highly volatile. Table 28 provides the results obtained for the *ROI*, *Drawdown* and the percentage of negative days that the *ROI* was negative.

Table 28 – Results for the two approaches experimented

Parameter	Avg. Proposed System with Fixed Leverage	Best Proposed System with Fixed Leverage	Avg. Proposed System With Chromosome Representation	Best Proposed System With Chromosome Representation
ROI (%)	19.3	37.5	43.9	83.5
Number of days with negative ROI(%)	57.3	51.2	15.4	16.5
Max. Drawdown (%)	105	110	9	14

At this point, it is possible to state that the *Chromosome Representation* outperforms the *Fixed Leverage* approach, but a question still remains, what levels of leverage does this approach use? Figure 44 exhibits the variation of the leverage levels during the test period, showing that, the levels of leverage depended on the type of market, more specifically in this execution the Bullish Market obtained a leverage of 9, the Bearish Market a leverage level of 8 and the Sideways Market a leverage level of 2, note that, when the proposed system does not use leverage, the level is always 1. Moreover, figure 44 shows that for well defined markets the leverage level is higher than the rest and even more, the leverage level is higher in the situations where the FX market presents higher price variations. Finally, the leverage levels presented are not actual columns, but a very oscillatory variation of the levels it self, making it look like a histogram.

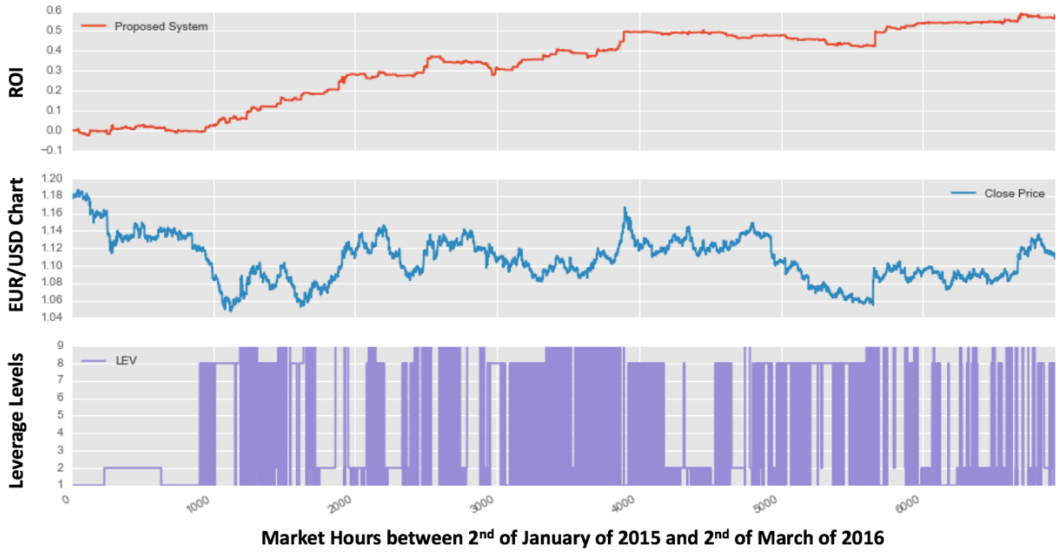

Figure 44 - Variation of the leverage levels during the test period

c) Conclusions of case study E

The experiment made for this case study shows that the Proposed System with leverage outperformed the one without leverage in every aspect, the leverage approach that obtained the best results was the *Chromosome Representation*, proving that a dynamical approach is better than a static one (Fixed Level approach), also, it was proven that the *GA* was capable of optimizing a value for leverage, even though, this gene does not contain any information about the FX prices, like Technical Indicators do.

4.3 Chapter Conclusions

This chapter analysed five case studies, that include the most important experiments developed in this thesis, providing results and some reflections about the approaches taken into account. Moreover, the experiments prove that a Dynamic Genetic Algorithm outperforms a Static Genetic Algorithm in dynamic environments, also, it is proven that the SVM classifier brings an important contribution for this work, improving the performance of the *Proposed System.*

Case Study A showed, for the *SVM Model*, the RBF kernel outperformed the other kernels, for both approaches when comparing which features to use, where the *Price Sequence* approach proved to be better than the *Technical Indicator* approach.

As for Case Study B, where it is experimented the best configuration for the Genetic Algorithm, the solution obtained shows that the 2XPop-PSC, can be used, even though it is a little bit heavier on computation.

Case Study C, provides a comparison between the *Static GA* approach and the *Dynamic GA*, proving, as it was stated before, that a *DGA* outperforms a *SGA* in dynamic environments, providing a steadier and more adaptable solution. Regarding Case Study D, where a comparison with other benchmarks and a lighter comparison with two works from the literature was made, showing that the *Proposed System*, outperforms the usual benchmarks and, when compared with other works, it is possible to conclude that, presents promising results to be applied in real-time trading.

Finally, Case Study E, studies the impacts of leverage applied in the *Proposed System*, implementing a new and different approach from the works seen in the literature. The experiment concluded that, implementing leverage can produce higher returns, without exposing the investment to high levels of risk.

5. Conclusions and Future Works

5.1 Conclusions

This work proposes a viable solution to automatically invest in Forex Markets, using technical indicators and price sequences to predict entry and exit points, although, it is highly recommended to use hardware that has good performance measures, so that is possible for the algorithm to work in real time. The recommendation is made because, the more information about the FX market, the bigger the populations and the more price sequences the *SVM* is provided to improve the training module, the better the classification of the market is. Also, by studying the *SVM* approach, it is possible to conclude by the results, that the sideways market is more difficult to describe than the others as it was seen in chapter 4, where the label "0" showed poorer quality in the classification, due to the fact that the Sideways market is difficult to be classified even by a human eye. Both *Bullish* and *Bearish* markets were classified with very successful scores of precision, recall and accuracy. By the results obtained in the test period it is possible to conclude that the classification a market is highly important, as an environment as well as a tool that allows the implementation of leverage. The environment classification passed from the *SVM* onto the three different *GA*s obtained very promising results, since the *Proposed System* presented a steady growth with a low *Drawdown*. In a more technical matter, the *SVM* proved to be a reliable classifier, said that it is important to address the following conclusions:

- The way data is organized, is the key to get a good *SVM* classifier, i.e., for most problems, and especially the one addressed in this thesis, combined a great amount of data that is not suitable to use directly from the index. An organized and well structured dataset improves the performance and the quality of the results that are obtain by the *SVM* model.
- The quality of the training set is also a very important issue, since this data set is the foundation on which the *SVM* learns. If the data set is not well balanced between the different labels, the metrics may return ill-defined results that do not translate the reality of the classification. Another measure for data set quality is whether the training set is correlated with the testing set, e.g., if the training set only has data from a *FX* chart that dates from 1999, and the test set is from the year 2015, then probably, the classification will be ill-defined since the data is not highly correlated anymore, since the *FX* market has changed a lot in 16 years.
- The cross-validation used is a very important tool to assess if the training set is not over fitted before testing the model on the test set. This tool proved to be very useful to develop the model, in order to ensure that the model, did not "leak" information between the training and test set, i.e., if the *SVM* model has any information about future actions, then the results cannot be reliable.

A final conclusion about the *SVM*, this ML approach provides a reliable method that not only was able to classify, with high metrics, the different markets, but also, reached a good performance that did not create a struggling point in the architecture, while providing the information to the *GA* module.

Regarding the *GA* module, one showed good adaptive response, and a good trade-off between diversity and quality of a given population, also the memory approach, improved the performance of the proposed solution, and thus, being very important so that the algorithm could adapt to different environments during the evolution of the price sequences. The technical indicators provided to this work, showed that it is possible to invest in EUR/USD using technical analysis, moreover, the strategy that optimized the parameters of the technical indicators proved to be very useful, to also introduce diversity in the populations. This approach provided a wide variety of individuals because the technical indicators did not use the usual standard values. Nonetheless, the intervals used in those parameters must not be very wide, being advisable to choose a window around the standard values that are commonly used in the market.

The use of dynamic methods clearly increased the performance and improved the behaviour of the ROI. Using adaptive methods provided new solutions in the search space that usually are not possible with a static approach. The *Associative Memory* method used, for the three different GAs, proved that is advantageous to train each population for a specific environment, rather than training one population for a broader environment. This method also provides a more adaptive solution, since the *Proposed System* is capable of shifting to a different population when the environment changes.

5.2 Future Works

For future works, it is important to keep in mind that, although the *Evolutionary Computation* plays a fundamental role to provide the best solutions, the investment rules used to entry or exit the market can be perfected with more elegant rules and strategies that are used by professional traders. In a more technical point of view, it is advisable to explore the following mechanisms:

- Explore unsupervised learning methods to classify different types of markets;
- For classifying purposes, a volatility prediction to help the *SVM* to improve the market type classification is important, not only to classify but also to make investment decisions and provide information for the leverage tool, i.e., this work classified three types of markets, but it is interesting to develop the idea that within those three types there are more types of markets;
- Introduce new mechanisms of multi-population schemes, that help the algorithm to search in parallel different types of solutions;
- Studying the impact of introducing a back-end *GA* to optimize the parameters used in the *SVM*, rather than use a *Grid Search* algorithm;
- Comparing different *Stopping Criteria* methods, such as introducing measures of variance in the population, or in the fitness function. The Stopping Criteria can influence the search of new solutions, thus being a very important topic in the study of *Genetic Algorithms*;

References

[1] M. Mitchell, "Genetic Algorithms: An Overview," Complexity, Vol. 1, No. 1, pp. 31–39, Sep. 1995.

[2] A. Neubauer, "Theory of The Simple Genetic Algorithm with α-Selection," In Proceedings of The 10th Annual Conference On Genetic and Evolutionary Computation (GECCO ’08), Atlanta, USA, New York: ACM, 2008, pp. 1009–1016.

[3] P. K.-F. Man, K. S. TANG, and S. Kwong, Genetic Algorithms: Concepts and designs, Illustrated ed. Springer Science & Business Media, 2012.

[4] J. Holland, "Adaptation in Artificial and Natural Systems." Ann Arbor: The University of Michigan Press, 1975.

[5] A. E. Drake and R. E. Marks, "Genetic Algorithms in Economics and Finance: Forecasting Stock Market Prices and Foreign Exchange — A Review," Genetic Algorithms and Genetic Programming in Computational Finance, pp. 29–54, Jan. 2002.

[6] S. Cirillo, S. Lloyd, And P. Nordin, "Evolving Intraday Foreign Exchange Trading Strategies Utilizing Multiple Instruments Price Series," Corr, Nov. 2014.

[7] A. Hirabayashi, C. Aranha, And H. Iba, "Optimization of The Trading Rule in Foreign Exchange Using Genetic Algorithm," In Proceedings of The 11th Annual Conference On Genetic and Evolutionary Computation (GECCO ’09), Montreal, Canada, New York, USA: ACM, 2009, pp. 1529–1536.

[8] A. Gorgulho, R. Neves, And N. Horta, "Applying a GA Kernel On Optimizing Technical Analysis Rules for Stock Picking and Portfolio Composition," Expert Systems with Applications, Vol. 38, No. 11, pp. 14072–14085, Jan. 1407.

[9] R. W. Colby and T. A. Meyers, The Encyclopedia of Technical Market Indicators. New York: Irwin, 1988.

[10] Investopedia.com, "Leverage Definition," Investopedia, 2003. [Online]. Available: http://www.investopedia.com/terms/l/leverage.asp?layout=orig.

[11] P. H. Winston, "Lecture 13: Learning: Genetic Algorithms," MIT Open Courseware. [Online]. Available:http://ocw.mit.edu/courses/electrical-engineering-and-computer science/6-034-artificial-intelligence-fall-2010/lecture-videos/lecture-13-learning-genetic-algorithms/.

[12] J. Branke, Evolutionary Optimization in Dynamic Environments, Vol. 3. Springer Science \& Business Media, 2012.

[13] L. Martí, "Luis Martí," 2014. [Online]. Available: http://lmarti.com/genetic-algorithms-algoritmos-geneticos-eng1456.

[14] T. Park, R. Choe, And K. Ryel, "Dual-Population Genetic Algorithm for Non-stationary Optimization," In Proceedings of The 10th Annual Conference On Genetic and Evolutionary Computation (GECCO '08), Atlanta, USA, New York, USA: ACM, 2008, pp. 1025–1032.

[15] S. Yang, "Evolutionary Computation for Dynamic Optimization Problems," In Proceedings Of The Companion Publication of The 2015 Annual Conference On Genetic and Evolutionary Computation (GECCO '15), Madrid, Spain, New York, USA: ACM, 2015, pp. 629–649.

[16] H. G. Cobb and J. J. Grefenstette, Genetic Algorithms for Tracking Changing Environments. Naval Research Lab Washington Dc, 1993.

[17] Y. Shengxiang and T. Renato, "Hyper-Selection in Dynamic Environments," In IEEE World Congress On Computational Intelligence., Hong Kong, IEEE, 2008, pp. 3185–3192.

[18] Y. Shengxiang and R. Hendrik, "Hyper-Learning for Population-Based Incremental Learning in Dynamic Environments," In IEEE World Congress on Computational Intelligence, Trondheim, IEEE, 2009, pp. 682–689.

[19] T. T. Nguyen, S. Yang, And J. Branke, "Evolutionary Dynamic Optimization: A Survey of The State of the Art," Swarm and Evolutionary Computation, Vol. 6, pp. 1–24, Oct. 2012.

[20] P. H. Winston, "Lecture 16: Learning: Support vector machines," MIT OpenCourseWare. [Online]. Available: http://ocw.mit.edu/courses/electrical-engineering-and-computer-science/6-034-artificial-intelligence-fall-2010/lecture-videos/lecture-16-learning-support-vector- machines/.

[21] J. Yao and C. L. Tan, "A Case Study on Using Neural Networks to Perform Technical Forecasting of Forex," Neurocomputing, Vol. 34, No. S 1–4, pp. 79–98, Sep. 2000.

[22] C. Evans, K. Pappas, And F. Xhafa, "Utilizing Artificial Neural Networks and Genetic Algorithms to Build an Algo-Trading Model for Intra-Day Foreign Exchange Speculation," Mathematical and Computer Modelling, Vol. 58, No. S 5–6, pp. 1249–1266, Jan. 1249.

[23] S. Deng and S. Akito, "Foreign Exchange Trading Rules Using a Single Technical Indicator from Multiple Timeframes," Advanced Information Networking and Applications Workshops (WAINA), 2013 27th International Conference On: IEEE, 2013, pp. 207–212.

[24] L. Mendes, P. Godinho, And J. Dias, "A Forex Trading System Based on a Genetic Algorithm," Journal of Heuristics, Vol. 18, No. 4, pp. 627–656, Aug. 2012.

[25] M. P.B and B. Adam, "Evolutionary Algorithm in Forex Trade Strategy Generation," in Computer Science and Information Technology (IMCSIT), Proceedings of The 2010 International Multiconference on, IEEE, 2010, pp. 81–88.

[26] G. Sermpinis, C. Stasinakis, K. Theofilatos, and A. Karathanasopoulos, "Modeling, Forecasting and Trading the EUR Exchange Rates with Hybrid Rolling Genetic Algorithms - Support Vector Regression Forecast Combinations," European Journal of Operational Research, Vol. 247, No. 3, pp. 831–846, Dec. 2015.

[27] S. Kei, D. Shangkun, and S. Akito, "Prediction of Foreign Exchange Market States with Support Vector Machine," In Machine Learning and Applications and Workshops (ICMLA), 2011, 10th International Conference On, IEEE, Vol. 1, pp. 327–332.

[28] R. J. Elliott, W. C. Hunter, and B. M. Jamieson, "Drift and Volatility Estimation in Discrete Time," Journal of Economic Dynamics and Control, Vol. 22, No. 2, pp. 209–218, Feb. 1998.

[29] A. Rossi and G. M. Gallo, "Volatility Estimation Via Hidden Markov Models," Journal of Empirical Finance, Vol. 13, No. 2, pp. 203–230, Mar. 2006.

[30] C.-Y. Yeh, C.-W. Huang, and S.-J. Lee, "A Multiple-Kernel Support Vector Regression Approach for Stock Market Price Forecasting," Expert Systems with Applications, Vol. 38, No. 3, pp. 2177–2186, 2011.

[31] S. Pang, L. Song, and N. Kasabov, "Correlation-Aided Support Vector Regression for Forex Time Series Prediction," Neural Computing and Applications, Vol. 20, No. 8, pp. 1193–1203, Nov. 2010.

[32] A. C and I. H, "Modelling Cost into a Genetic Algorithm-Based Portfolio Optimization System by Seeding and Objective Sharing," In Evolutionary Computation, 2007. CEC 2007. IEEE Congress On, IEEE, 2007, pp. 196–203.

[33] P. Fernández-Blanco, D. J. Bodas-Sagi, F. J. Soltero, And J. I. Hidalgo, "Technical Market Indicators Optimization Using Evolutionary Algorithms," In Proceedings of The 10th Annual Conference Companion On Genetic and Evolutionary Computation (GECCO '08), Atlanta, USA, New York, USA: ACM, 2008, pp. 1851–1858.

[34] A. Gorgulho, R. Neves, and N. Horta, "Using GAs to Balance Technical Indicators on Stock Picking for Financial Portfolio Composition," In Proceedings of the 11th Annual Conference Companion on Genetic and Evolutionary Computation Conference: Late Breaking Papers, Montreal, Canada, New York, USA: ACM, 2009, pp. 2041–2046.

[35] F. Yuan, "Parameters Optimization Using Genetic Algorithms in Support Vector Regression for Sales Volume Forecasting," Applied Mathematics, Vol. 3 No. 10a, 2012, pp. 1480-1486.

[36] H. Zhang; R. Ren, "High Frequency Foreign Exchange Trading Strategies Based on Genetic Algorithms," In Networks Security Wireless Communications and Trusted Computing (NSWCTC), 2010 Second International Conference on, IEEE, Vol.2, pp.426-429, 24-25 April 2010

[37] A. Gorgulho, N. Horta, and R. Neves, Intelligent Financial Portfolio Composition Based On Evolutionary Computation Strategies. Springer-Verlag Berlin Heidelberg, 2013.

[38] S. Yang and X. Yao, "Experimental Study on Population-Based Incremental Learning Algorithms for Dynamic Optimization Problems," Soft Computing, Vol. 9, No. 11, pp. 815–834, Nov. 2005.

[39] S. Yang, "Non-Stationary Problem Optimization Using the Primal-Dual Genetic Algorithm," In Evolutionary Computation, 2003. CEC '03. The 2003 Congress On, IEEE, Vol.3, No., pp.2246-2253 Vol.3, 8-12 Dec. 2003

[40] C. Li and S. Yang, "A Generalized Approach to Construct Benchmark Problems for Dynamic Optimization," Lecture Notes in Computer Science, Vol. 5361, pp. 391–400, Jan. 2008.

[41] J. Branke, "Memory Enhanced Evolutionary Algorithms for Changing Optimization Problems," In Congress on Evolutionary Computation CEC 99, IEEE, 1999, pp. 1875–1882.

[42] S. Yang and X. Yao, "Population-Based Incremental Learning with Associative Memory For Dynamic Environments," In Evolutionary Computation, IEEE Transactions on, IEEE, Vol.12, No.5, pp.542-561, Oct. 2008

[43] C. Li, T. T. Nguyen, M. Yang, S. Yang, and S. Zeng, "Multi-Population Methods in Unconstrained Continuous Dynamic Environments: The Challenges," Information Sciences, Vol. 296, pp. 95–118, Mar. 2015.

[44] S. Yang, "Genetic Algorithms with Memory- and Elitism-Based Immigrants in Dynamic Environments," MIT Press Journals, pp. 385–416, Sep. 2008.

[45] P. A. N. Bosman, "Learning, Anticipation and Time-Deception in Evolutionary Online Dynamic Optimization," In Proceedings Of The 7th Annual Workshop On Genetic And Evolutionary Computation (GECCO ’05), Washington, D.C., USA, ACM, 2005, pp. 39–47.

[46] A. Simões and E. Costa, "Improving Prediction in Evolutionary Algorithms for Dynamic Environments," In Proceedings of The 11th Annual Conference On Genetic and Evolutionary Computation (GECCO ’09), Montreal, Canada, ACM, 2009, pp. 875–882.

[47] "GUI Programming - python Wiki," in Python, 2010. [Online]. Available: https://wiki.python.org/moin/GuiProgramming.

[48] IPython, "Spyder - documentation — Spyder 2.3 documentation", 2009. [Online]. Available: https://pythonhosted.org/spyder/.

[49] "Python data analysis library — pandas: Python data analysis library,". [Online]. Available: http://pandas.pydata.org/.

[50] TA-Lib.[Online]. Available: https://mrjbq7.github.io/ta-lib/func.html.

[51] "Scikit-learn: Machine learning in python — scikit-learn 0.17.1 documentation," in Scikit Learn. [Online]. Available: http://scikit-learn.org/stable/.

[52] 2016, "12.1. Pickle — python object serialization — python 3.5.2 documentation," in Python docs, 2001. [Online]. Available: https://docs.python.org/3/library/pickle.html.

[53] "Precision-recall - scikit-learn 0.17.1 documentation," in Scikit Learn, 2010. [Online]. Available: http://scikitlearn.org/stable/auto_examples/model_selection/plot_precision_recall.html.

[54] "DEAP documentation — DEAP 1.0.1 documentation," in DEAP Documentation, 2009. [Online]. Available: http://deap.gel.ulaval.ca/doc/default/.