Trend Aware Investment Strategy for Stock Market based on a Classifier System and on Intelligent Optimization Techniques

Pedro Esteves

***Abstract*—In this work, it is proposed a hybrid system that combines the use of Support Vector Machines with Genetic Algorithms to create trading strategies based on the current conditions of the market. This system uses a classifier based on the Support Vector Machine to analyze the current trend and select a specific trading strategy, previously optimized by a Genetic Algorithm for the corresponding trend. The system requires an initial training phase of the classifier, which will then be used to split the data into groups of different trends. Each of these groups is optimized separately based on technical analysis rules in order to achieve a tailored trading strategy for each type of trend. For this experience, it is used four important indexes from distinct countries, namely, the SP500, the DAX, the NIK225 and the IBEX35. The data used covers the market data since September 2001 up to September 2018 and the system is tested for each year since September 2008. This system is compared against a similar system without the trend classification component for evaluation purposes. The results for the three markets suggest a possible increase in performance of the strategy optimization method if it is implemented a classifier with the intent of strategy segregation. The data generated during the training and testing phase imply that the proposed system is capable of generating strategies better adapt to the present trend.**

***Index Terms*— Genetic algorithms, Machine learning, Support vector machines, Stock markets.**

I. Introduction

THE prediction of market trends and the creation of automated trades are hot topics in financial markets. There is a long history of Machine Learning (ML) research in this area, starting with White's [1] neural network approach. Before that, statistical methods were the typical approach [2]. However, due to the complex non-linear behavior of the markets and the lack of correlation with statistic indicators, those approaches proved to be inefficient. Markets have a highly noisy nature, which results on systems to be prone to act on false signals and thus producing investment losses.

This work focuses on the creation of a system composed of a market classifier that is capable of distinguish different market conditions, using machine learning techniques and a strategy optimizer. The classifications are used to choose different strategies optimized for the respective conditions. Thus, producing a system that is able of quickly adapt to various environments. A common practice when analyzing the evolution of a market asset is to use market indicators [3]. To do so, several indicators are computed from the available asset data. These indicators are pointers of different market characteristics, but many evaluate the same parameter in a very similar way leading to a possible multicollinearity. Creating a system capable of analyzing the current market trend in a larger time frame is desired. This approach is going to help reduce the susceptibility of trades to local noise and have tailored strategies optimized to each of the possible market contexts.

II. State of Art

Predicting market price changes, with the best accuracy possible, is desired by traders in general. This prediction is fundamentally a pattern recognition task. Independently of the theoretical approach, whether using technical analysis, fundamental analysis or sentimental analysis, in every one of these cases it is necessary to look for correlations in past data. For that reason, in available works there are several important decisions that lead to completely different paths.

The first step starts by choosing the type of data that will serve as input for the system. It is possible to use fundamental analysis using financial indicators or reports. As well as, technical analysis which can be a simpler approach since the calculation of indicators can be done in any time frame, as long as data is properly available.

The second stage is to create a system capable of correlating the received data with a price move. The process of choosing a market analyses technique is vital, and it can be done through a classifier that look at trends as classes or by using a regression method to predict future price values or even an optimization technique applied to trading strategies.

Finally, an evaluation methodology should be considered to examine the performance of the proposed system. A simple approach can be used by simply comparing the return on investment (ROI) of the system in comparison with traditional techniques. However, it is also possible to do a study regarding the drawdown of the system and even an analysis of the success rate of trades.

A. Simple Classifiers

There is a clear dominance of two techniques, the Support Vector Machine (SVM) and the Artificial Neural Network (ANN). These two are the two most developed single classifiers [4] [5]. Stankovic [6] uses a prediction model based on Least Squares Support Vector Machine to predict the trend of the stock market indices in emerging markets. The model took as inputs the most commonly used technical indicators and simulated trading in different periods of time. The results showed that the model outperforms simple TA rules based on a single TA indicator, using popular indicators. Rajashree Dash [7] combines a computational efficient functional link artificial neural network (CEFLANN) with a set of defined rules in order to generate buy and sell signals on stock trading. As input the CEFLANN receives a set of popular technical indicators and the output is a continuous signal used in a defined algorithm that represents the trading rules. This model shows a superior return when compared with frequently used models such as SVM, KNN, Naïve Bayesian and decision tree.

B. Hybrid Systems

There is an extensive collection of recent proposals using hybrid derivations of neural networks or support vector machine algorithms and the majority achieve a better prediction accuracy when compared to more classic approaches. However, a growing trend is noticeable in the use of ensemble systems with the objective of minimizing the drawbacks of some ML techniques, resulting in a superior performance and thus beating all solo approaches. Asad [8] created a prediction system that combines SVM, Relevance Vector Machine and ANN, by generating decision values based on a majority voting mechanism. This architecture was tested on Istanbul Stock Exchange and presented around 70% accuracy on prediction capabilities. Another example of an ensemble system is Petropoulos approach [9] of developing a model that uses a set of machine learning algorithms, namely SVM, random forests, Bayesian autoregressive trees, NN and Naïve Bayes. The signals generated by those techniques are managed through a combination of majority voting, genetic algorithm optimization and regression weighting. This is used to predict the exchange rate of a set of currency pairs and the experiments displays an annual return as high as 18%.

C. Market Time Frame Study

Some papers alarm to the importance of the time frame chosen and the associated forecast horizon, since there is an impact in accuracy when leveraging these factors. Shynkevich [10] published a paper that extensively examines the effect of shifting the input window length for the creation of technical indicators, and even explores the combination of that change with the forecast horizon. Traditional machine learning techniques (SVM, ANN and kNN) are used to generate predictions. The evaluation was based on prediction accuracy, average return per trade and risk/reward ratio. The evaluated measures show a visible dependency of the combination between the input window length and the forecast horizon. The highest prediction performance is achieved when both the parameters are roughly equal. Zhang [11] proposes an approach called status box method that instead of only using local turning points, creates boxes containing specific patterns. This way, making predictions in bigger time frames is possible, which results in a better assimilation of the stock market information. Long-term trends are easily obtained by comparing status boxes to theirs neighbors and there is a reduction on trading noise. AdaBoost-GA-PWSVM was the classifier implemented and the results show a better generalization performance when the status box method is used.

III. Proposed Solution

In Fig. 1, it is represented a diagram with the overall architecture of the proposed system, where three main modules, represented by the round blocks, divide the system. The *Dataset* block represents the market data and the two square blocks represent data transformations. The main idea is to use a classifier based on the SVM to classify the market in three different behaviors, namely uptrend, downtrend and sideways. The result from this classification allows the aggregation of candles according to their trends. For each group of candles, it is employed a genetic algorithm process to optimize the trading strategy used. For each iteration of the algorithm it is used the Strategy Simulator module to calculate the fitness of each strategy. This segmentation of optimized strategies intends to maximize the profitability of the system by making it more robust to changes on the market conditions.

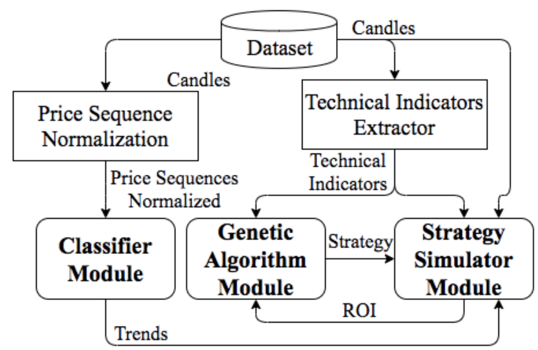

Fig. 1. Schematic of the proposed system architecture.

The information of daily candles of a certain asset is translated in order to produce a collection of price sequences that are readable by the SVM classifier. The data produced consists of a sequence of normalized closing prices. Each sequence contains a trend pattern that will be classified by the Classifier module. The classifier needs to be trained with labeled data, and for that reason it is used a method for the manual classification of trends.

Initially, the SVM classifier needs to receive labeled data in order to be trained. This data is used to perform a grid search algorithm on the classifier parameters using a cross-validation method. The best combination of parameters is used to train the SVM model that will be used to classify the trends on a market. These trends are used later in the system to optimize different strategies for each group of trends. The universal trait of trends implies that a trained classifier should be able to classify trends the trends of a market independently of the period or asset used.

The daily candles are also used to calculate technical indicators that will be used by the Strategy Simulator module and manipulate by the Genetic Algorithm module.

The optimization of strategies based on trends requires a trained classifier that receives the normalized price sequence of each candle of that market data and outputs the respective trend. The candles are then divided into trend groups. For each trend group it is used the Genetic Algorithm and Strategy Simulator modules to derive the best strategy. The result is three different trading strategies, each one of them optimized for a specific market condition.

The system can also use the genetic algorithm to optimize a single trading strategy using the entire collection of candles. This is useful since it allows an easy assessment of the impact of trend consideration when deploying a trading strategy.

A. Dataset

The dataset consists of the price action since the year 2001 up to 2018 for four of the most traded indexes. The Standard&Poor's 500 (SP500), consisting of the 500 largest publicly traded companies in the United States of America. The Deutscher Aktienindex (DAX), that represents 30 of the largest German companies. The Nikkei Heikin Kabuki 225 (NIK225), which comprises of the 225 largest publicly traded companies of Japan. And the Índice Bursátil Español 35 (IBEX35), that represents 35 of the largest Spanish companies. This data was retrieved from Yahoo Finance and it is comprised of daily values containing information of the open, close, high and low prices, as well as the volume for the corresponding day. The system consumes this information and it creates the technical indicators for the strategy module and the normalized price sequences for the classifier.

The system allows for the manual process of labeling sequences of candles with respect to the present trend. This labeling process involves choosing random candles from the dataset to be classified. For each candle that goes under manual trend classification, it is evaluated the pattern of the sequence of candles prior to the candle at hand. After this process, two files are created corresponding to the training and testing files for the Classifier module and their data is separated in time in order to get a clean training and testing process.

The normalized price sequences are generated through a process that starts by calculating the length between the highest and lowest closing price observed during the window of candles considered. Then, for each closing price it is subtracted the lowest closing price and divided by the length previously obtained. The result is a sequence of numbers on a scale of one to zero that try to represent the pattern of the closing prices during the period represented in the window. This approach allows us to compare similar patterns for different price ranges, as well as for markets that typically have different percentage changes on price but follow the same trend shapes.

The technical indicators used are computed using a well-known Java library for technical analysis called ta4j [12].

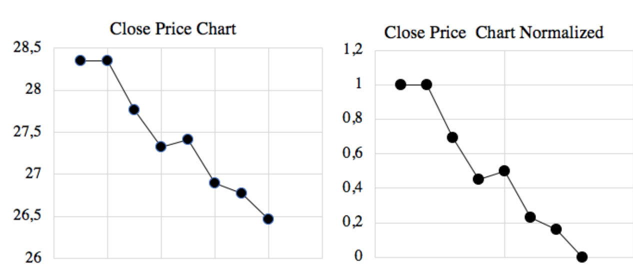

Fig. 2. Example of close price sequence and normalization.

B. Classifier Module

The classification of the market trends is performed on this module. The implementation of the SVM relies on the use of LIBSVM [13], which provides a strong and well tested library for the implementation of the algorithm originally invented by Vapnik [14].

For the classification of the trends, it is used, as input for the SVM, a sequence of closing price values that describe the trend detected at a certain point in time. Each array of values represents the trend observed at the current candle being classified and the number of elements in that array is what defines the size of the window. For this work, three types of labels were used, namely uptrend normally represented by continuous higher highs and lows, downtrends represented by continuous lower highs and lows, and sideways which represent all the other types of patterns.

The SVM classifier needs to be initially trained with labeled samples. It receives as input data a collection of normalized price sequences with the correct trends that will be used to train the classifier. The training process requires a set of parameters, where Kernel, C and Gamma parameters are related with the SVM, and the K parameter is related with the cross-validation method used. To find the optimal configuration for the SVM it is used a grid search method for the parameters for each kernel. Generally, the values chosen in the grid search increment in a logarithmic scale since the spectrum of possible values is wide and for each solution it is necessary to test the classifier accuracy. The search process starts with the linear kernel that modules the simplest patterns and from that it iterates through the polynomial and radial basis function (RBF) kernels, that are able of classifying more complex spaces. The parameter C is present on all kernels and it defines the penalty of the decision boundary, if a greater value of C is used, the decision curve can handle larger errors. For the Polynomial and RBF kernels, the Gamma parameter exists, and its increment allows for a curvature increase of the decision boundary which results on a higher adaptation to the training data.

For each combination of parameters, it is used a cross-validation method based on the K-Fold technique during the grid search in order to get a better indication of how well the classifier will behave on new data. The data is divided into K sets and it is trained K times. For each time, a different set is used for testing and the remaining sets are used for training. At the end of the process, it is computed the average accuracy of the model in all the iterations. This method generates a less biased measurement of the classifier performance.

In the presence of a trained classifier model, it is possible to use the Classifier module to classify unlabeled market data. It receives as input data a collection of normalized price sequences of a specific market and it outputs the corresponding trends through the classifier.

C. Strategy Simulator Module

The Strategy Simulator module is responsible for the implementation of trading strategies and the corresponding simulation. This implies the creation of trading rules that will be used to generate trading signals during the simulation. This module has two modes of operation, it can simulate for a specific type of trends, or it can simulate for all candles regardless of their trend classification. The output of a simulation is a list of trading signals that yield the corresponding ROI. When doing a simulation based on trends, it is necessary to run the simulation for each type of trend and get the final ROI from the combined signals of these simulations.

The trading rules are conditions used to evaluate the market through technical indicators. These conditions are used to decide if the simulator should enter a long or short position or if it should close an existing position. Since the objective is to support our strategies in technical analysis, the proposed rules use a set of different technical indicators. In order to get a clear picture of the market it is used a set of indicators that focus on distinct market aspects. For trend information, it is used the crossing of two moving averages to measure the direction and strength of a trend or a reversal. Regarding momentum, it is used the Relative Strength Index (RSI) to assess the velocity of price movement by examining the sequence of the most recent closing prices. Lastly, with respect to volatility, it is used the Bollinger Bands (BB) to estimate the recent level of price fluctuation. In Fig. 3, it is displayed the proposed rules for the implemented system.

TECHNICAL INDICATOR	SIGNAL RULE	TYPE OF SIGNAL	PARAMETERS
SMA	$(Z)SMA_X < (Z)SMA_Y$	Long	X=[5,10,15,20] Y=[-1,1]
	$(Z)SMA_X > (Z)SMA_Y$	Short	Y=[2*x,3*x,4*x,5*x]
RSI	$RSI_{14} < X$	Long	X=[0:100]
	$RSI_{14} > Y$	Short	Y=[0:100]
BBW	$BBW_{20,2} < X$	Long	X=[0:1]
	$BBW_{20,2} > Y$	Short	Y=[0:1]
BB	$(Y)BB_{20,X\ Upper\ band} > (Y)Close$	Long	X=[1,2,3,4]
	$(Y)BB_{20,X\ Lower\ band} < (Y)Close$	Short	Y=[-1,1]

Fig. 3. Proposed rules for each technical indicator.

During the strategy simulation, at each candle iteration, several signals will be generated, and it is expected to observe conflict between the various indicators. To cope with the inconsistency of the technical indicators, it is implemented a system that equally distributes the voting power between the signals. This voting system aims to boost the confidence of the system trades and avoid over trading. The Fig. 4 presents a state diagram that illustrates the voting mechanism implemented.

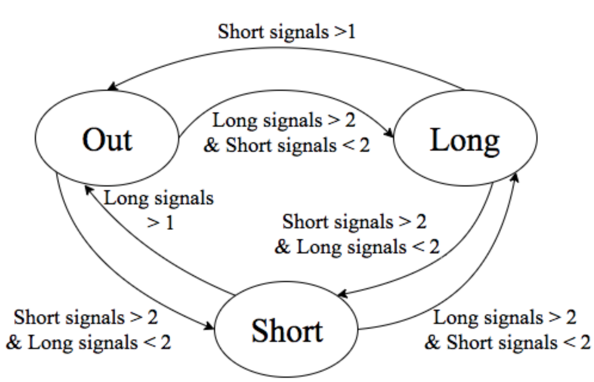

Fig. 4 - State diagram of the implemented strategy.

The simulator has the possibility of being long, short or out of a trade. For each point in time during the simulation, a state change can be made by analyzing the number of short and long signals generated in that moment.

D. Genetic Algorithm Module

The Genetic Algorithm module is responsible for the optimization of strategies based on the returns yield by those strategies. The individuals created by the genetic algorithm are translated to trading rules that form a trading strategy. The Strategy Simulator module returns the ROI of these strategies to the Genetic Algorithm module, working as the fitness function of the algorithm. For the implementation of the genetic algorithm it is put to use the JGAP library [15], which is a sophisticated genetic programming library written in Java.

The chromosome defined for the genetic algorithm consists of several values that control the parameters of the strategy rules used in the Strategy Simulator module. The rules presented in Fig. 3 have variables (X, Y and Z), that are chosen using the genetic algorithm. These variables can represent threshold values, define characteristics of the technical indicators or change the logic implemented in the trading rule. The values present on the chromosome directly affect the trading signals and thus influencing the trading decision. In Fig. 5 it is represented the chromosome used to parameterize the rules and optimize the trading strategy.

Short SMA	Long SMA	SMA Long Short Direction	RSI upper bound	RSI lower bound	BBW upper bound	BBW lower bound	BB Standard Deviation	BB Long Short Direction
[5,10,15, 20]	[2x,3x,4 x,5x]	[-1,1]	[0 : 100] Step: 5	[0 : 100] Step:5	[0 : 1] Step: 0.05	[0 : 1] Step: 0.05	[1,2,3,4]	[-1,1]

Fig. 5. Chromosome representation.

The genetic algorithm uses the return on investment of the strategy simulation as fitness function, which means that for each genome created it is necessary to run a strategy simulation correspondent to the specific genome. During the optimization phase, the pool of genomes will evolve in the direction of higher returns.

The genetic algorithm uses a selection method that consists on electing a fixed number of individuals that were able to show the best fitness results. These individuals will be considered for the creation of the next generation. The threshold was set to 90%, meaning that the best 90 individuals from the population of 100 are used to produce the next population. The population size is kept constant and it is allowed to have duplicated chromosomes. The initial population is set to 100 and it is created with random values for its genes.

The crossover process chooses two random chromosomes from the individuals that pass the selection method and use them to generate a new chromosome. The crossover rate is set to 0.35, which means that there is a 65% chance that the new chromosome is a clone from one of its parents and a 35% chance that the new chromosome inherits genetic information from both parents. This mechanism of combining two chromosomes involves swapping genes between the two initial chromosomes and thus creating a new distinct chromosome.

The mutation rate is another important definition that allows the algorithm to maintain genetic diversity by introducing random changes to the genes values of a chromosome. It is important to notice that a high mutation rate could lead to an aimless arbitrary search. The mutation rate was set to 1/12, meaning that on average one in every 12 genes is mutated.

IV. Results

A. Evaluation Metrics

The computation of the evaluation metrics for the classifier can be done using the confusion matrix displayed in Fig. 6. This confusion matrix is shown as the relation of events classified as positive or negative for a given class and the relation between the predicted and actual values. The four types present on the matrix are the basis for computing the evaluation measures of the classifier.

		Predicted	
		Negative	**Positive**
Actual	**Negative**	True Negative (TN)	False Positive (FP)
	Positive	False Negative (FN)	True Positive (TP)

Fig. 6. Representation of the Confusion Matrix.

Considering that the market being analyzed can have predominant types of trends, the evaluation of the model based purely on the corresponding accuracy is not a good procedure since it can lead to false interpretations.

The formula for the accuracy is:

$$Accuracy = \frac{\mathrm{TP} + \mathrm{TN}}{\mathrm{TP} + \mathrm{TN} + \mathrm{FP} + \mathrm{FN}}$$

As a result, the model is evaluated regarding its precision and recall, and thus providing a more robust evaluation methodology. A classifier that always outputs a specific type of trend regardless of the input can have a high accuracy if the market analyzed contains primarily that specific trend. For that reason, the precision and recall can be used to measure the efficiency of the model in detecting less frequent types of trends.

The formulas for the precision and recall are:

$$Precision = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FP}}$$

$$Recall = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}}$$

One way of evaluating both the precision and the recall in a single score is through the F1 score, which consists of the harmonic average between the precision and the recall measurements.

The formula used on its calculation is:

$$F1 = 2 \times \frac{\mathrm{Precision} \times \mathrm{Recall}}{\mathrm{Precision} + \mathrm{Recall}}$$

The score computed for a specific classifier will then be the average F1 score of the three labels considered.

The formula for the classifier score is:

$$Score = \frac{F1_{uptrend} + F1_{sideways} + F1_{downtrend}}{3}$$

Regarding the evaluation of the strategies, it is used a simple ROI metric, which measures the outcome of the strategy but not the risk associated with it.

The ROI is a percentage measurement and its formula is:

$$ROI = \frac{Gain\ from\ investment - Cost\ of\ Investment}{Cost\ of\ Investment}$$

B. Case Study – Classifier Time Frame Analysis

In this case study, the SVM classifier is implemented with different time windows in order to analyze the effects of the window size on the classification of trends. For each size considered, it is applied a grid search algorithm to find the best combination of kernel and parameters. For each iteration in the grid search it is used a cross-validation process based on the K-fold method. This process reduces the possibility of overfitting the training data. The value used for K is 5, meaning that the training data is divided in 5 parts and the K-fold algorithm is applied to those 5 blocks. To evaluate the performance of the classifier it is calculated the average of the F1 scores of the three labels through the formulas presented previously. This type of evaluation provides the desired information since it distributes the relevance of each type of trend evenly, regardless of an existing unbalance of trend types on the data set. After the grid search, the classifier is tested on the testing data using the parameters that yielded the highest scores for the three different kernels.

Regarding the lengths of the windows it was chosen three different sizes. A large window of 100 candles since it constitutes a good representation of a long-term trend. A small window of 20 candles, which is close to the minimum size possible for the materialization of a trend. And finally, a medium window of 50 candles that represents a compromise between the large and small window.

In this case study it is used the market data of the IBEX35 index from 1999 up to 2018, since it contains several examples of the three types of trends considered.

For each window size it was used 1000 samples of manually labeled trends to train and test the classifier. The training and testing data are divided using a typical ratio of 70% for the training data and 30% for the testing data, which means that there are around 700 samples for training and 300 samples for testing for each window size. For the grid search method, this study considers three different kernels, namely the linear, the polynomial and the RBF kernel. The grid search process is computed 20 times in order to generate more robust results. The

Fig. 7 summarizes the considerations made for this case study.

CONSIDERATIONS

WINDOW SIZE		[100, 50, 20]
ITERATIONS		20
SAMPLES	**100 Window**	1000
	50 Window	1000
	20 Window	1000
GRID SEARCH	**Kernel**	[“linear”, “polynomial”, “rbf”]
	C	[0.1,0.5,1,5,10,50,100,500,1000]
	Gamma	[0.001,0.01,0.1,1,10,50,100,500,1000]
	Method	K-Fold
	K	5
DATA	**Market**	IBEX35
	Duration	21/09/1999 – 21/09/2018
DATA RATIO	**Training Data**	70%
	Testing Data	30%

Fig. 7. Considerations for the *Classifier Time Frame Analysis* case study.

The Table I presents the best scores achieved for each window and kernel during the training phase. The best scores were chosen through the grid search method. It is possible to notice that the RBF kernel achieves the highest score for the three different sizes. Furthermore, the grid search method for each window, showed that the RBF has a distribution of scores higher than the other three kernels. Regarding the size of window used, it is visible that the 100-window size achieves higher scores. This can be explained by a more frequent overlapping of data between trend samples since the span of the trend is bigger or simply by the existence of more clear trends.

TABLE I
TRAINING SCORES FOR CLASSIFIER

Average Score	Linear	Polynomial	RBF
100 window	90.46	91.59	91.75
50 window	87.78	88.83	89.68
20 window	88.81	89.16	89.81

The Table II presents the average F1 scores achieved during the testing phase for the best combination of parameters gathered from the grid search method. This table shows that the results obtained are contained between a score range of 84% to 92%, which proves the success of the classifier regardless of the window or kernel used. However, the RBF kernel manages to outperform the other two kernels for the 100 and 20 window and matches the other kernels scores for the 50-window size. It is also possible to notice that the 50-window size approach underperforms in relation to the other two sizes, which might indicate that trends are less clear for that specific trend span.

The difficulty of achieving higher scores might not be related with technical characteristics of the classifier, but rather with the subjectivity present on the classification of trends. Given the difficulty of defining a clear separation between uptrend and sideways or downtrend and sideways, the results obtained show that the classifier is capable of identifying a trend with a high rate of certainty. And based on the results displayed, the RBF kernel is a good choice for the classification of trends using the SVM classifier.

TABLE II
TESTING SCORES FOR CLASSIFIER

Average Score	Linear	Polynomial	RBF
100 window	90.28	89.57	91.44
50 window	84.38	84.74	84.53
20 window	90.02	89.92	90.44

C. Case Study – Impact of Trends in Strategy Optimization

This case study deliberates the benefits of trend consideration during the process of strategy optimization. This means that it will be compared the system using the classifier and without the use of the.

Taking in consideration the importance of the classifier role, the system is implemented using the three different sizes for the classifier window. Based on the results of the previous case study, it is used the RBF kernel with the respective optimal values of C and Gamma for each window size. The trained classifier, using IBEX35 market data, from the previous case study is used as universal classifier to classify the trends present on all the data used for the three markets since the principles of an uptrend, downtrend or sideways trend are unalterable.

The two systems were tested for a sequence of eleven periods of one year each, starting from 2007 up to 2018. For each period of tested data, the system was trained with all the available data from the year 2001 until the beginning of the testing period. It was chosen a wide and cumulative period of training to ensure that the data used had enough data to create optimized strategies for each trend. It was also assumed that the use of more data would guarantee the creation of strategies that benefit from a better generalization since they would have to perform well in a wider number of years and thus reducing the possibility of overfitting a specific market behavior. The Fig. 8 contains a resume of the considerations decided for this case study.

Markets			SP500, DAX, NIK225
Iterations			10
Classifier	Training Market		IBEX35
	Kernel		RBF
	Window Size 100	C	0.5
		Gamma	0.1
	Window Size 50	C	10
		Gamma	0.1
	Window Size 20	C	5
		Gamma	0.1
Technical Indicators			SMA, RSI, BB, BBW

Fig. 8. Considerations for the *Impact of Trends in Strategy Optization* case study.

For this case study three market indexes with distinctive characteristics were used with the objective of presenting consistent results under various markets through a wide number of years. In order to get results more robust, each simulation was performed 10 times and the results displayed on the following tables consists of the averages of those simulations.

The Table III presents the ROI results obtained at the end of the eleven years using the different strategies for each market. The method using the trends is capable of outperform the trendless approach for the DAX and NK225 markets when it is implemented a 20 or 50 candle window size. However, for the SP500 market, the trendless system yields substantially higher returns with the trendless method. The results also show that the size of the window chosen for the classifier has a severe impact on the system performance.

TABLE III
ROI TESTING RESULTS

ROI	SP500	DAX	NIK225
100 window	0.64	0.18	-0.22
50 window	0.64	0.38	1.28
20 window	0.76	0.33	0.19
Trendless	1.32	0.21	0.15

For the three markets tested, the trend approach using a window of 100 candles, consistently shows worse results in the total returns observed at the end of the eleven years. Yet, for smaller windows, the trend approach surpasses the trendless method returns by 80% and 57% for the DAX market and 850% and 27% for the NIK 225 using a 50 and 20 candle window size respectively. For the SP500 market the trendless approach seems to outperform the proposed system returns by a long margin of more than 200%. One explanation for this phenomenon could be the SP500 price behavior, characterized by low oscillations and a prevalent up trend that compromises the utility of a system that tries to cope with rapid changes on the market conditions.

V. CONCLUSION

This work examines the impact of implementing a classifier in the optimization of trading strategies. For the markets and periods considered during the experiment, it is not possible to conclude with certainty that the classifier increases the system performance. Nevertheless, the results show that it is possible that this system offers more flexibility to changes on the market and thus being more profitable under certain circumstances.

Regarding the classifier, this work presents three different sizes for the collection of candles that form a trend. The results obtained suggest that the size of the window have a low impact on the classification since the average accuracy, recall and precision remain high on the three cases tested. A comparison between the three kernels used by the classifier indicate that the patterns are easy to identify and consequently a simpler kernel is acceptable. The high values obtained for the accuracy, precision and recall, are a sign of the success achieved by the classifier and part of the misclassification of some trends can be blamed on the subjectivity of such classification.

The proposed system showed a better performance when compared to a blind strategy optimization for two of the three markets tested. Yet, it also shown that the size of the window used for the trend classification has a big impact on the system performance. The results show that the medium sized window has the best returns and that a higher window size greatly affects the profitability of the system. The reason for the success of the hybrid system can be explained by the rapid adaptation to the market conditions by shifting between strategies that are better optimized to the specific trend present. The effectiveness of this system can be compromised when under a market that remains steadily under the same market conditions during a long period.

REFERENCES

[1] H. White, Economic prediction using neural networks: the case of IBM daily stock returns, IEEE 1988 International Conference on Neural Networks, 1988, pp. 451-458.

[2] H. V. Roberts, Stock-Market ‘Patterns’ And Financial Analysis: Methodological Suggestions, The Journal of Finance, vol. 17, 1959, pp. 1-10.

[3] Krollner, Bjoern & Vanstone, Bruce & Finnie, Financial Time Series Forecasting with Machine Learning Techniques: A Survey, 18th European Symposium on Artificial Neural Networks, 2010.

[4] W. Lin, Y. Hu and C. Tsai, Machine learning in financial crisis prediction: A survey, IEEE Trans. Syst. Man Cybernetics, Part C, vol. 42, no. 4, 2012, pp. 421–436.

[5] Yelin Li, Junjie Wu and Hui Bu, When quantitative trading meets machine learning: A pilot survey, 2016 13th International Conference on Service Systems and Service Management, 2016, pp. 1-6.

[6] J. Stanković, I. Marković, and M. Stojanović, “Investment Strategy Optimization Using Technical Analysis and Predictive Modeling in Emerging Markets,” Procedia Economics Finance, vol. 19, no. 15, 2015, pp. 51–62.

[7] R. Dash and Pradipta K. Dash, “A hybrid stock trading framework integrating technical analysis with machine learning techniques,” The Journal of Finance and Data Science, vol. 2, no. 1, 2016, pp. 42–57.

[8] M. Asad, “Optimized Stock market prediction using ensemble learning,” 2015 9th International Conference on Application of Information and Communication Technologies, 2015, pp. 263–268.

[9] A. Petropoulos, S. P. Chatzis, V. Siakoulis, and N. Vlachogiannakis, “A stacked generalization system for automated FOREX portfolio trading,” Expert Systems with Applications, vol. 90, 2017, pp. 290–302.

[10] Y. Shynkevich, T. M. McGinnity, S. A. Coleman, A. Belatreche, and Y. Li, “Forecasting price movements using technical indicators: Investigating the impact of varying input window length,” Neurocomputing, vol. 264, 2017, pp. 71–88.

[11] X. dan Zhang, A. Li, and R. Pan, “Stock trend prediction based on a new status box method and AdaBoost probabilistic support vector machine,” Applied Software Computing, vol. 49, 2016, pp. 385–398.

[12] ta4j. (2018). [Online]. Available: https://github.com/mdeverdelhan/ta4j-origins.

[13] Chih-Chung Chang and Chih-Jen Lin, “LIBSVM: A library for support vector machines,” ACM Transactions on Intelligent Systems and Technology, vol. 2, no. 3, 2011, pp. 1-27. Available: http://www.csie.ntu.edu.tw/~cjlin/libsvm

[14] C. Cortes and V. Vapnik, Support Vector Networks, Mach. Learn., vol. 20, no. 3, 1995, p. 273-297.

[15] jgap. (2018). [Online]. Available: https://github.com/martin-steghoefer/jgap.