Delft University of Technology

Computer Science Project EWI3615TU

Group 21

Stock prediction

Authors:
Sander Delfos (4317262)
Dennis Schoonhoven (4601173)
Victor Guillet (4488636)
Marten Verbree (4629639)
Fazia Ghuman (4364554)

Supervisor:
Dr. Ir. Bart H.M. Gerritsen

January 16, 2019

Evaluation

In this evaluation one can find a summary of our project.

Motivation

The main aim of the project was to develop an algorithm capable of trading at a profitable level autonomously. Choosing this as topic of research presented a number of advantages, including the fact that a large amount of data being readily available, that this would give us the chance to learn more about the concepts of trading and that it was something that might be genuinely applicable in our future lives.
Finally Twitter sentiment analysis seemed a good and efficient option to implement for fundamental analysis, but because it was not certain whether this actually had any relationship with the stock values, this was seen as a part of our research.

Objective

The main objective for this project is to build a trading algorithm capable of picking up on a number of trends. Based on these trends, a trading bot should then decide to buy, sell or do nothing. One of the sub-objectives was to find whether Twitter sentiment was a useful indicator for trading. Another sub-objective was to find the relationship between fundamental indicators and technical indicators, along with their relative importance.

Plan

The initial concept revolved around having a general machine learning tool capable of aggregating the information returned from a number of different sources. On one hand a Twitter bot would scrape the web and recover tweets posted (which would then be filtered according to a number of parameters). A sentimental analysis would then be run and fed back in the general machine learning tool. On the other hand, a technical analysis would be run over the values of the stocks over that time period, an these results would then be returned, allowing for the main algorithm to generate educated buy or sell decisions.

Google trends would be used for fundamental analysis. With Google trends one can see how many times the terms 'apple' (company), iPhone', iPad' and other relevant terms are searched on Google. It was not yet decided what type of machine learning model would be used, but one of the ideas was a neural network. It was found that information about the company Apple was readily available, therefore it was decided to use Apple as the company that would be analysed.

Program

The program we created can be divided in four parts, a fundamental analysis analysis, a technical analysis, a data collection method and a trade bot protocol. Apple's stock values were used to create the signals for the technical analysis. The indicators that were used are the RSI, a long- and short-term SMA and volatility.

It was decided not to use data from Google trends, because it doesn't hold any information about whether the increase of popularity is a positive or a negative thing. This meant that only Twitter sentiment analysis was used. The Twitter bot fetched all the tweets with the keyterm 'apple', 'ipad' and 'iphone'. The tweets were filtered on language using the python library langdetect, and filtered on words like juice or pie so that the amount of irrelevant tweets was heavily decreased. This was done because it was much simpler and time efficient compared to writing a machine learning algorithm for this. The sentimental analysis was performed with

the python library Textblob, and for each day this would calculate the average sentiment. This was done because of the lack of training data available for the sentimental analysis, and the complexity of writing a machine learning algorithm for this. This was our only signal for the fundamental analysis, so this signal represents the fundamental analysis as a whole.

Once completed, the results of both the technical and fundamental analysis were combined together (a weighted addition was performed to emphasize on the importance of certain patterns, improving the overall model accuracy). The result was fed to a trade bot which then followed a very strict trading protocol. This allowed to get an estimation of the effectiveness of the overall analysis.

Results

Even though some parts of the original plan were not implemented (mainly due to time issues), or because they were not as relevant as originally though out to be, the results that were achieved are convincing. The software solely relying on technical analysis showed that it's able to make money by picking up trends in a time period relatively fundamental-news free. Adding in the fundamental analysis component allowed to begin to compensate for that, with the extra information allowing to pick up key fundamental moments. The trade bot also presented limitation; due to a lack of time it was not possible to implement a trading protocol which allowed it to take advantage of downtrends, missing out on the opportunity to significantly increase profit. The trade bot trading strategy would also require buy itself a significant amount of research as performance could be drastically improved by making smarter use of the analysis output.

Conclusion

Looking back at the objectives that were set out, one can conclude that this project has been successful. The software was able to pick up trends with technical indicators good enough so that, theoretically, money could have been made. The main draw back of the technical analysis still remain its disability to account for fundamental news. Evidence was found that a Twitter sentiment analysis is indeed a useful indicator for a fundamental analysis. As with the technical side, the fundamental side is not perfect. Most of the time, the sentiment is quite uneventful, people are generally not going to radically change their opinions about a company like Apple, and place tweets about this. It is when some big news comes out or an important event happens that people tweet about it, and then the sentiment will show a significant change. Combining the two into one master signal that can make buy and sell decisions is still a big challenge. It can be done with manual tweaking, but this takes a lot of time, which is undesirable. Using machine learning to tweak the parameters instead of doing it manually would be a better solution, but unfortunately there was no time left to implement this.

Contents

1 Introduction

Large amounts of data are now readily accessible on the internet for anyone with a minimum technical knowledge. This along with the ever more powerful tools available to the general public opens up possibilities, giving the chance for individuals to effectively compete in the trading sector more effectively than ever before.

This document contains a proof of concept of a potential trading algorithm concept, capable of collecting data from various sources, basic trend-spotting through a combination of technical and fundamental tools, and a preliminary automated buy-and-sell decision system based on the model created. The model is designed to allow for effective and easy expansion, allowing for an extremely wide of range of information and data analysis to be added. The model was also built in a way which allows for a number of optimization methods to be applicable, including a number of different machine learning techniques (such as Genetic algorithms).

The paper is structured as follow: section 2 contains the development plan, along with the goals of the project, the context, requirements, risk and constrains faced. Because not everything from the development plan is actually implemented, section 3 describes what was changed from the original plan. Section 4 focuses on the design and section 5 follows with how it was implemented. Section 7 presents and discusses the results and finally the conclusion is contained in section 8.

2 Development plan

This chapter will describe the original development plan, which was made at the beginning of the project before any actual coding was done. Therefore, this plan doesn't fully represent the actual finished product, but rather the thought process at the start of the project. This chapter will be followed up by section 3, which will cover the differences between the plan and what was actually done.

2.1 Background

Since mankind discovered the profit possibilities of trading a market, a race of understanding and predicting is persisting, with traders all around the world seeking to out-perform traditional buy-and-hold investing in an attempt to take maximum advantage of the movements of the market. For most of the last few decades, traders were limited to making most of their decisions themselves based on the news, a number of indicators and mathematical formulae, and their own "feeling" of the market. However as computing power increased, computing cost decreased, and large amounts of data becomes more an more readily available, a new method of trading has been on the rise; algorithmic trading.

Algorithmic trading involves writing a number of programs analyzing data collected from various source and attempting to spot trends, allowing to make predictions and forecasting a market's movement. The indicators and data types fall in one of two different categories: technical analysis and fundamental analysis. Technical indicators focus on looking at a number of values/parameters calculated and attempt to detect trends in value evolution to give insight on the state of the market. Fundamental analysis on the other hand focuses on analyzing the impact of specific news on the prices. Both are required for a complete analysis to be achieved, as otherwise certain fluctuation can simply not be explained effectively.

The action of buying and selling can also be handled by the computer, allowing for even further automation of the process, effectively of solving some trading's biggest issues; human error and around-the-clock work.

2.2 Objectives

The main objective for the software will be to give advice to the end user on whether or not to invest in stocks for a company along with handle in a completely automated way buy-and-sell decisions. This project will focus on predicting the future for stock value of Apple. The company has a long and varied history, making it suitable for testing both the technical analysis side of the model and the fundamental side (having been the subject to a number of impact-full events fully fundamental driven on a number of occasions). The objective can be split up into multiple sub-objectives:

- Collecting from possible data sources:
  - Apple stock market values
  - Dow Jones index values
  - Tweets from Twitter relevant to Apple
  - Google Trends with search terms relevant to Apple
- Data process:
  - Process stock market/Dow Jones values
  - Generate a number of indicators from the market values obtained

  - Sentiment analysis from tweets
- Trend spotting
- Forecasting (optional)
- Connection to trading API (optional)

A successful implementation of the above mentioned items should provide the user with clear insight on the market's situation at a specific time. This would allow for a better, more educated, (and possibly even completely automated) decision process to be achieved when it comes to interacting with the market, and making buy and sell decisions. If successful the algorithm could then be connected to a trading API.

A simplified visual representation of the process can be seen in Figure 2.

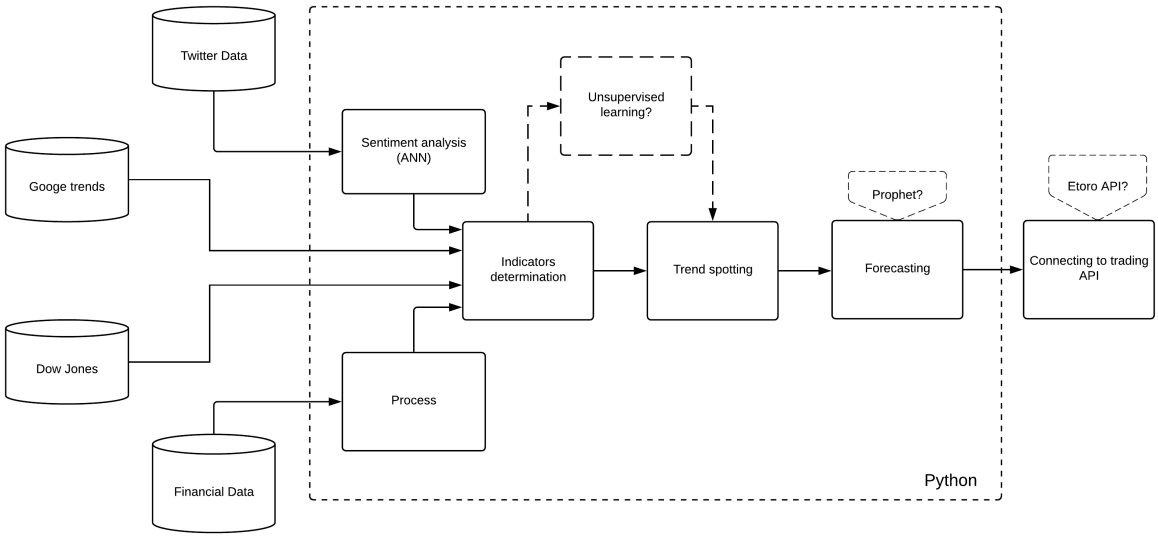

Figure 1: *Graphical representation of the followed method.*

2.2.1 End goal

The end-goal of this project is to achieve a working prototype which can generate and pickup a number of trends. Given that data for the stock market is readily available from multiple sources, reliability should not be a problem in the technical analysis section. Therefore, trend spotting based on historical trends in Apple stock market value will be the first goal. The other data sources (Twitter, Google Trends) might be less reliable and thus there is some uncertainty in what can be achieved with these. More on this can be found in subsection 2.4. The data retrieved from those sources will also most likely not be perfect, the main focus will hence be on adapting the methodology according to our data to take this in account effectively.

2.3 Requirements

2.3.1 Acceptance criteria

The final goal for the output includes the following points:

- An analysis combining both technical and fundamental data analysis
- A (graphical) overview that predicts the potential evolution of stock value with good accuracy

- A capacity to account for both long and short term trends
- A decision-making tool based on the predicted stock value evolution capable of generating and outputting effective buy-and-sell decisions.

2.3.2 Technical requirements

Live data from the stock markets and Twitter is needed. This means that time will be an issue. Therefore a neural network or reinforcement learning technique could be used, as this type of AI is best capable of dealing with time issues.
Data mining mechanisms will be needed for this. For Twitter this would mean a Twitter bot that gives us the information from Twitter about the subjects we want it to follow, in this case Apple. For the stock market this would be a bot that takes the stock market data from the last five years, and keeps track of the live values of the stock market.
A tweet needs to be analysed for the sentiments attached to it. This means that an AI must focus on positive or negative words in the tweet. Also it should look at the hashtags used, as these can put extra emphasis on the positive or negative aspects of the tweet. It should be able to categorise the tweets that are neutral to an output that is neutral as well, because more tweets about Apple also means more interest in Apple and therefore a higher chance of movement in the stocks because of it.
For trend spotting the stock market will be analysed in such a way that historic trends will be taken into account. This can be done by training a neural network on old data, and applying what it has learned to current financial data.
To be able to predict the future stock values, the trends that have been spotted need to be implemented. This can be done by implementing a code like Prophet from GitHub, but only if the trend spotting works.

2.4 Risks

During projects in which big data is used numerous things can go wrong. Big data projects can for example struggle with the way the data is stored; doing this incorrectly can end up costing a lot of money. It is important to look at those risks beforehand to ensure a prosperous project. Data storage should not be a risk for this small project, but the acquisition of data, however, should be taken into consideration. Multiple sources of data are used as stated before. The acquisition of data related to the stock market is not a problem: this information can reliably be found on the internet. More problematic might be the acquisition of reliable data from Twitter. Especially since tweets from 2 months back might be needed to train the program. The program therefore should be able to deal with for example missing tweets as input. This as well illustrates the advantage of using multiple sources for data: bad or missing data in one category should have a low impact on the result. Also the combination of sentimental and technical analysis helps with this risk. After training the program, it gathers live data. This means that there is a larger chance of data not being useful. To fix this, filters are used that look at data and check for example if the tweet is about Apple, the company, or apple, the fruit. On Twitter the risk of getting fake news is also large. However, it was decided that this was not a risk as this will also be seen as positive or negative news by real people. Another risk would be that there might be little to no connection between the sentiment of tweets about Apple and the stock values. This will be avoided by making it possible for the AI to set the weight of the sentiments of the tweets, so that the AI could ignore the tweets if there is no relationship between the tweets and the stock data.
The largest risk is that the trends spotted are not accurate enough, which means that forecasting and connecting to a trading API is useless. This means that trend spotting is the most risky link in the system, and it therefore needs to be tested thoroughly.

Finally a risk during any project is simply getting stuck. Methods from the Software Engineering Methods course in previous quarter will be applied to assure a smooth development process.

2.5 Constraints

It is also important to consider the limitations of the software.
The initial goal is set to make the program predict whether the stock price of Apple is going to rise or to fall in the near future. The output is not more extensive. The program does not operate autonomously and the investor is still in charge at any time to buy or sell stock. If the process is coming along well, the output of the program could for example be expanded by adding graphs indicating the future stock price or making the program give substantiated suggestions, like 'buy stock' or 'sell stock'.
But there are as well more fundamental questions, relevant to the software. Is it possible to identify all factors influencing the stock price? Is the data selected to predict the stock price at all related to the stock price? For example, messages posted on Social Media are taken into account by the software when predicting the stock prices, although it might very well be possible that these messages are most of the time hardly related to the actual stock price. The last, and maybe most important question: is it at all possible to predict stock prices and, if so, up to an accuracy so that it is possible to make profit? During the project it will be interesting to study these questions and push the software to its limits.

3 Actual Development

Here the development plan in chapter section 2 is reevaluated with respect to how the project was actually implemented. Along comes a validation of the changes made. Only the sections of the original development plan where something changed are discussed.

3.1 Objectives

The project in the end only used the following data sources:

- Apple stock market values
- Tweets from Twitter relevant to Apple

Some of the initial ideas were thus left out. The main reason for this is simply a lack of time and these ideas could certainly be implemented later on.
Considering the goals set for data processing, these ones actually have been implemented:

- Generate a number of indicators from the market values obtained
- Forecasting

These 2 are essential in order to be able to make successful buy or sell decisions and thus have been prioritized.

3.2 Requirements

Due to the fact that the final system does not farm data live, it was unnecessary to use neural networks or reinforcement learning techniques. The same applies to the sentiment analysis, instead of using AI for this a simple Python module suffices. Furthermore instead of using a neural network to spot a trend, the plan is to use a genetic algorithm to combine various indicator signals so a trend can be acquainted. The reasoning is that a genetic algorithm is perfect for this kind of optimization problems.

4 Design plan

In this section, the design plan will be described, which consists of the objectives made in terms of design concepts. This plan outlines the strategy followed to obtain the design objectives and the considerations that had to be taken in order to achieve these objectives as were set out. Further on, this plan will describe the additional risk factors that were involved and had to be taken into account. These design principles will help in further implementation of our software.

4.1 Design objectives

The design objectives for the development of this software are mainly to achieve robust software by minimizing the use of bad data. Another important objective is designing flexible software functionality that will easily allow to change certain functionality or adding new functionality. Also re-usability of code is an equally as important design requirement, which makes it possible to minimize dependency between parts. Our main objectives can be divided into several subsections:

- Robustness:
  - Minimal use of bad input data
  - Ability to handle unusual conditions such as bad data, programming error or environmental conditions
- Flexibility:
  - Extensibility
  - Loose coupling of software components
- Re-usability of functions and classes

A simplified visual representation of the design of the technical analysis can be seen in Figure 2.

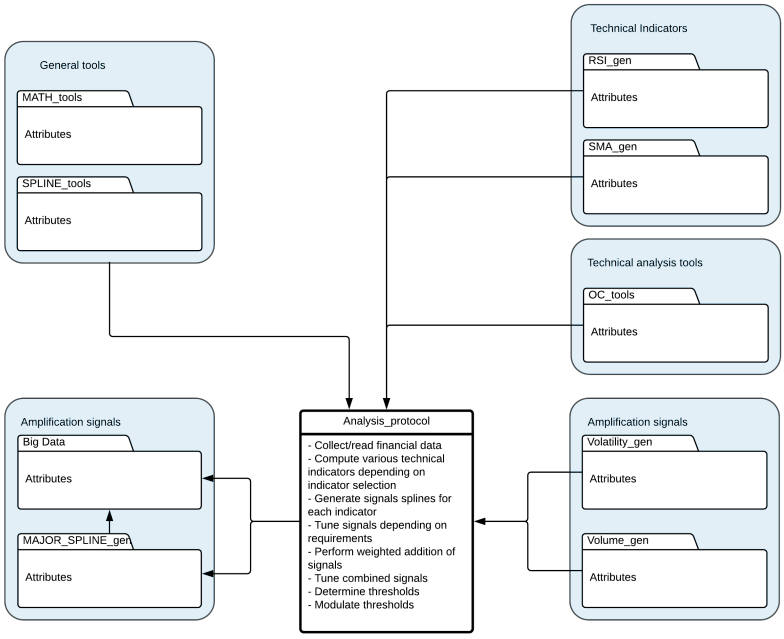

Figure 2: *Graphical representation of the design for technical analysis*

The diagram for the technical analysis part shows how different parts of the implementation are connected together. As is shown, the generate tools, the technical indicators, the technical analysis tools and amplification signals are used by the analysis protocols to calculate big data amplification signals.

4.2 Strategy

The main underlying strategy for the design process is start designing the software in two different parts separately, which are the technical analysis part and the sentiment analysis part and merge these parts together afterwards. This parallel way of designing ensures that two completely independent parts are designed. The pursued strategy has therefore been mostly a function oriented approach, in which the software is composed of a set of interacting units with each a clearly defined function. For example, in Figure 2 it is clearly shown that the technical analysis part is also divided into several parts with by function to keep them independent. The advantage gained from this strategy is that the two parts are merged only after both perform well and enables a closer insight into the parts that are not functioning correctly, which makes it easier to spot the bugs. The followed technique for designing software is called modularization, in which the modules are designed in such a way, that they can be executed or compiled separately. Other advantages modularization offers is that components in our software with a high level of cohesion can be re-used, which is exactly what our design objective was. Apart from division based on functional components in the software, this way of designing enables easier maintenance due to having smaller components, exactly like how it is shown in the diagram. These design techniques are used in order to satisfy the set design objectives.

At the start of the implementation of this software, the biggest focus point was the retrieval of useful data which would strongly determine the way our software is implemented. Initially, the way for filtering was set out differently than was actually applied. This was mostly because the original data sources turned out to be less useful and other ways of fetching data were more applicable to the current situation. After the retrieval of data and successfully cleaning the data, the actual techniques for data analysis was thoroughly thought through and applied based on the type of available data. In the next step, output signals were obtained and the decision was made to invest more time into making both parts perform well. Moreover, the strategy was continuously adapted based on the current state of the development process.

4.3 Critical features

To reach the design objectives as set out, some critical features are needed. As said, the software should be flexible and re-usable. This translates into the need of having independent parts of software. The two main parts are the technical and fundamental sides, but the software inside these parts should be coded independently as well. Therefore, classes were used (in some parts of the software), and everything was built into as much as possible independent functions that were also as short as possible, making them easier to re-use.
The code should be easily extendable, which in this case means preparing the software so that more technical and/or fundamental indicators can be added in. For example, if in the future a new news feed is added as a fundamental indicator, the output of this new indicator should be easily fed into the main signal and only needs to be changed in the main analysis protocol class.

In order to ensure the robustness, flexibility and re-usability it highly important that cohesion is applied as much as possible in the modules. This implies that our classes, their methods and their data inside our modules belong together with a high degree. In Figure 2 it also shows how

our cohesion is applied. Also, the software becomes easier to maintain, test or understand and therefore this must be a critical feature in the design of software.

4.4 Risk-factors and to-avoids

The design of software can bring many advantages, however trying to achieve the set objectives can be quite a challenge in terms of managing complexity. Many components are developed independently based on function, nevertheless there are always some components that are naturally more coupled and can cause over-design as consequence of complexity. This is one of the risks that are involved and need to be avoided, because coupled components can be hard to reuse and tend to be fragile. Fragile design can lead to software that is easy to break, thus also needs to be avoided as much as possible.
Another downside is the needless repetition of code in the classes when implementing functions that resemble each other. This repetition of code should be avoided by reusing functions and implementing functions in a reusable way, thus by simplifying the functions.

5 Implementation

In order to take a maximum advantage of different analysis, three big questions need to be addressed; what is the optimal way to interpret and format the information provided, how can this information then be combined effectively, and how is the final output used for making buy and sell decisions? The section below describes how the analysis were performed, along a slightly unconventional yet effective way of exploring the last two questions, and provides solutions based notably on concepts borrowed from classical Physics.

The implementation section is split up into several subsections, namely subsection 5.1 which covers the fundamental analysis side, subsection 5.2 which covers the technical analysis side, and finally how both analyses were combined in subsection 5.3. This layout follows the order the several parts have been implemented and thus should give a clear overview of the process.

5.1 Fundamental analysis

The program uses data from Twitter to conduct the fundamental analysis. The plan is to farm tweets from Twitter containing the word apple and perform a sentiment analysis on the tweets.

The first step is to get the tweets from Twitter. The initial solution was to use Tweepy, which uses the official Twitter API. Using Tweepy it is possible to get tweets with the word 'Apple' in it and add even more specifications, like for example the user. The big drawback however is that Tweepy does not support retrieving tweets older than approximately 2 weeks. Since this project would require tweets from years back to successfully implement a fundamental analysis, it was decided to not use Tweepy. Instead the Github project named GetOldTweets-python [1] was used. This project uses web-scraping to be able to retrieve older tweets. Now tweets with the word apple from up to multiple years back could be retrieved. These tweets come with the user-ID, date, amount of favourites and amount of retweets.

The next step consisted of filtering the tweets. Using all the tweets including apple for the fundamental analysis is a bad idea. A tweet stating that apple liquor is tasty, is probably not related to the stock price of apple. In order to solve this category of tweets, a list with forbidden words was made, for example: tree, pie and juice. If a tweet contained a forbidden word, the tweet is deleted. All non-English tweets had to be filtered as well. This is because later on a sentimental analysis is applied on the content of the tweet, however only English tweets can be analysed. To filter out non-English tweets the langdetect Python module is used; this module contains a function which returns the language of an input string. The fast majority of remaining tweets now is English. The final step in the filtering process is to retrieve more relevant tweets. The GetOldtweets Python module has the option to filter based on top tweets. By enabling this option the top tweets from each day containing the word apple are retrieved. These tweets contain information with more impact than the average tweet, since the information in it is meaningful and read by more people. Fetching the tweets takes quite a lot of time. A Twitter webpage is quite big with lots of images, resulting in the downloaded webpage being quite a large file, with not that many actual tweets. Fetching the tweets therefore is restricted on internet speed. Also, after fetching a certain amount of tweets, the software would either freeze or not find any tweets for any day. This is most probably due to Twitter blocking the connection, as it was solved by either switching to another network or using a VPN. After the filter the tweets are written to separate json files based on their date.

The final step is to conduct a sentimental analysis on the acquired tweets. To this end the Textblob Python module was used. This module contains a function which accepts a string as input and returns a value between -1 and +1. This returned value represents the sentiment of

the tweet. A negative value corresponds to a negative sentiment, the closer the value to -1 is, the more negative the sentiment and likewise for positive values. By reading out the tweets from the json file of a day and accumulating the sentimental values of all these tweets, the sentiment towards Apple on Twitter can be estimated for specific dates. If this is done over a longer time period, the sentimental values can be plotted versus the corresponding dates and so a signal is made representing the sentiment towards Apple on Twitter. This finishes the implementation of the fundamental analysis.

5.2 Technical analysis

The data required for the technical analysis was definitely easier to find and collect than for the fundamental analysis. Algorithmic trading becoming more and more popular, a large number of data source are available to the public, with up-to-date, accurate value of the market provided for free. Among all the possible providers, Yahoo! finance was retained. Its excellent python support and extensive data base (going back up to 30 years for certain companies) made it the logical first choice for prototyping.

Now on to the analysis, technical analysis tools such as the RSI (explained in subsection A.2) have traditionally been used in the following way. If the value of the RSI went past or bellow a certain value (or threshold), the RSI was effectively considered as sending out a "buy" or "sell" trigger signal. This principle holds for most technical indicators, such as the SMA (explained in subsection A.3), with SMA sending out trigger signals when for example a long term SMA crosses a short term one. This method however presents a clear flaw. While the trigger signals are effective in locating trigger points, no information in between these is collected and taken advantages of, and the information returned by the technical indicators is discreet

Example of a signal obtained: ["Hold","Hold", "Sell","Hold","Hold", "Buy","Hold","Hold"]

It is possible however to derive a continuous signal from these indicators. For the RSI, if instead of looking at instances of the signal crossing the threshold, the distance separating the thresholds from the signal is kept track of. This results in a continuous signal of values representing the proximity (as so the likeliness) of the signal to hit the thresholds. This way, each indicator stops simply returning "Trigger points" but an evaluation of the bullish/bearish (explained in subsection A.1) tendencies of the market.

5.3 Combining the information gathered, the "Affectum signal"

Now that a continuous signal has been derived for each indicator and analysis, the question of effectively combining them comes about. A possible answer lies in classical physics, and the fact that "signals" were obtained from the different indicators. After performing a simple normalization on the signals obtained (effectively ensuring that their max and min amplitude is always 1 and -1 for example, with 1 representing a perfect bullish tendency and -1 a bearish one), it becomes possible to add up and take the averages of the signals. This method of signal addition results in the signals combining, forming constructive interferences where all the indicators reflect similar market tendencies, and destructive interferences when returning a different market appreciation. The final signal obtained then reflect the overall "feel" of the market, generated from the aggregate analysis of the different indicators.

This first step enables for an effective and optimized information collection method to be followed, but can still be improved significantly. This method so far gives equal importance to the different indicators and information included in the signal. It should however be kept in mind that all information is not equally relevant at all times, and should hence be pondered

accordingly. A simple, yet effective way of achieving this is through performing a weighted addition of the signals. Giving a different weight to each signal, and then dividing the main signal obtained by the sum of the weights enables controlling and modulating the importance and impact of the information in the signal.

Even further information can be included in the signal, by modulating it's amplitude with the daily volume of stock traded for example. A large volume of transaction in a market suggest a much larger probability of significant changes in market values, and a small one the opposite. Including this provides the signal with with even more insight in the market's situation.

As this signal represents the overall "feeling" of the market at a specific instance, it will be referred to as the "Affectum signal".

5.3.1 Making decisions based on the Affectum signal

Once a solid feel of the market has been obtained through selecting correct parameters for the different indicators, it is possible to perform buy and trade decisions based on the value/strength of the Affectum signal. A number of features can be monitored such as the gradient of the signal, or simply its highs and lows. For this research the signal strength going past a set upper or lower threshold triggered a buy or sell decision (similarly to the RSI). The results of this method can be seen in **??**.

5.4 Version management

Git was used to keep track of the current version of the project. One version of the program was continuously adapted to get towards the final goal. The combination of these 2 ensured a smooth process.

6 Testing

Testing has been done directly after a function has been written. Most of the functions are tested manually.
All functions have been tested in two ways, because it was decided that the following two questions are the most important for every function:

- Does the function return what it needs to return? Also on edge cases?
- Does the function have undesirable side effects?

At first the result of the function was looked at. If this was incorrect, it meant that the code was wrong. The debugger mode in PyCharm was used to step through the function. This was done to find where the function was wrong. This process was repeated until the bug was fixed. Initially it was thought that this was enough testing, but with this way of testing there was still a chance that the result was right, but the method was partially or completely wrong.

This is why every function was analysed, even when the result was correct. The question "Does the function have undesirable side effects?" was answered by using this method. One discovered side effect was that starting dates in the weekend could not be handled by the program, due to the fact that the stock market is closed in the weekends. The workaround was to use the closing value of the stock price on previous Fridays.

Validation

The tests also need to be validated, because the exact output for a lot of functions was unknown. A certain expected output was created by setting a range for the correct output. It was checked whether this expectation was met. An example for this is the testing of the sentimental analysis function. Here the function was given a negative string, a slightly negative string, a neutral string, a slightly positive string and a positive string. As the outcomes were not exactly known, a range was set for the test strings, for example: from 0.1 to 0.5 for the slightly positive string or from -0.1 to -0.5 for the slightly negative string.

Some tests were also automated, for which PyTest was used. For example, a test was written to check the filtering of the tweets, as the desirable output for the filters could easily be checked. With this it was discovered that words that contained capital letters were not filtered out, and this was fixed immediately.

7 Results

In this section various acquainted Affectum signals will be discussed to show the findings of the project.
Figure 3 compares the sentiment on Twitter to the actual stock prices.

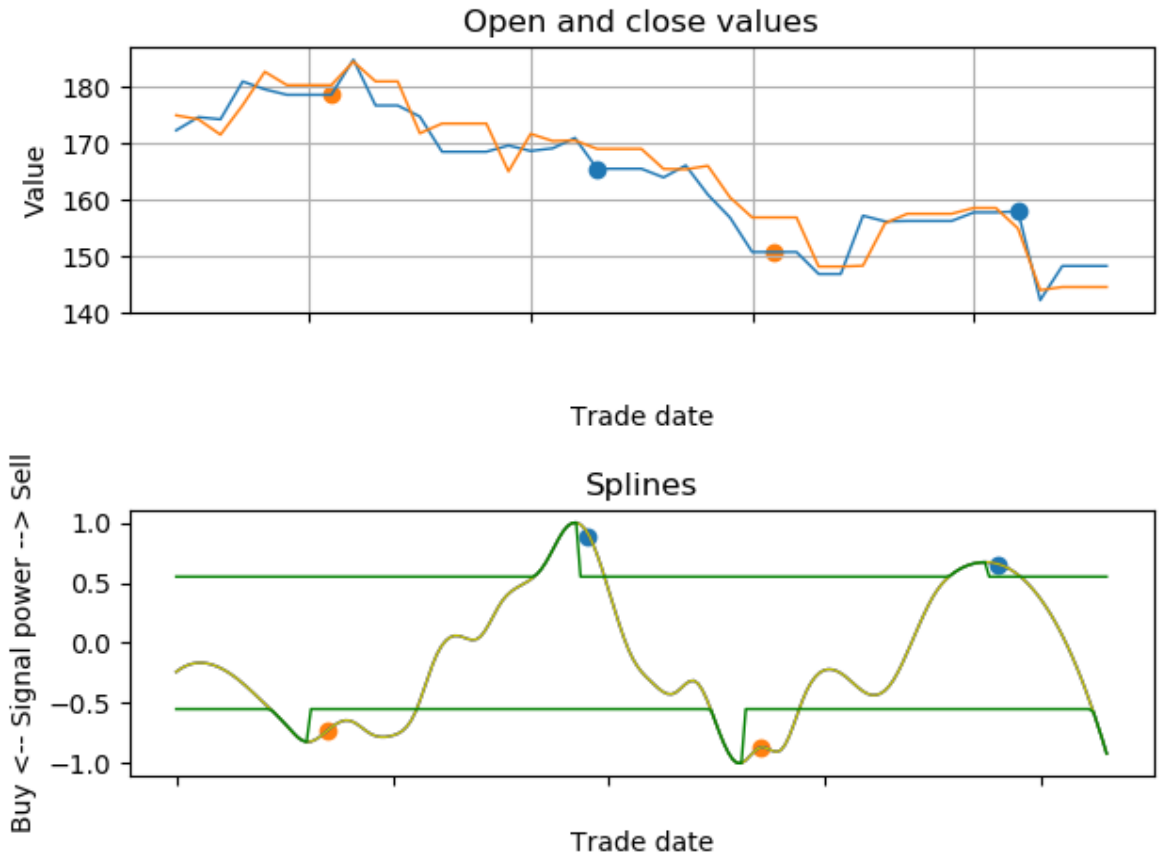

Figure 3: Fundamental analysis results, with open/close prices at the top, the fundamental signal at the bottom and trigger points plotted over both

In the graph on the top the opening and close values of Apples stock are plotted versus the time. The signal in the graph on the bottom merely represents the sentiment towards Apple on Twitter. It is the inverse of the signal described in subsection 5.1. This means that negative values represent a positive sentiment on Twitter and positive values a negative sentiment. Once the signal gets negative, stocks should be bought and, vice versa, once it gets positive stocks should be sold. Finally 2 horizontal lines can be seen in this bottom graph, they are the upper and bottom threshold. Once the signal crosses these thresholds an action should be taken.
By observing Figure 3 there seems to be a relation between the sentiment on Twitter and the actual stock prices. Observe for example the last positive spike of the sentiment: the signal is steadily rising in the few days prior to the moment the stock drops drastically in the top image. The same relation holds for the first positive spike of the inverted sentiment: the stock price remains the same for a while and then drops. There are of course as well moments, the sentiment does not predict stock prices in a desirable way, take the first dip of the sentiment analysis: the stock price does increase a lot subsequently. However the stock price does drop very fast afterward. The sentiment signal alone, would probably not be a good indicator to make decisions at this point. This remarks the importance of combining various indicators to be able to make good decisions.

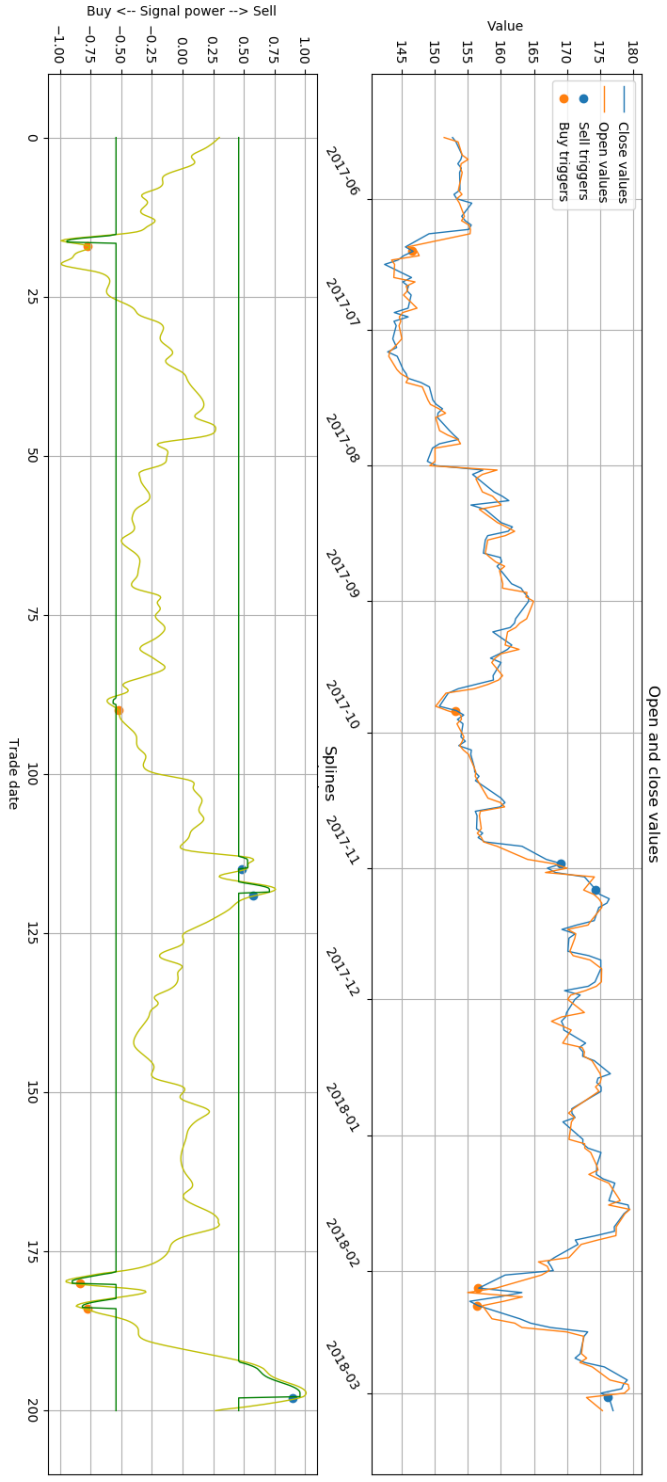

Figure 4: Technical analysis results, with open/close prices at the top, the Affectum signal generated at the bottom and triggers points plotted over both

In Figure 4 the graph on the top again contains the open/close prices of Apples stock versus the time, albeit this time over a bigger time span. This time the bottom graph contains the signal built up out of all technical indicators combined. Again 2 horizontal lines are present, representing the upper and lower threshold. Once the signal passes the lower threshold, it should be a good time to buy stock according to the indicator and once the signal surpasses the upper threshold it is time to sell, as indicated on the y-axis. Comparing the times the signal drops below the lower threshold to the stock prices at those periods, it is visible that the program tends towards buying at periods when the stock price is in a low peak. The reverse holds for the moments the program tends towards selling. But there are also more questionable moments. Looking at the month January in 2018, the signal did not manage to cross the upper threshold before the stocks would drop drastically. Instead the program decided on selling at the start of November 2017, when the stock prices stopped rising. Would it not be better to have the program sale stocks closer to the drop in price? The decision making progress of the program is ultimately based upon the weights given to individual indicators and of course the indicators used. Maybe the program would make better decisions in the above scenario if the sentimental analysis from Twitter was taken into account as well. Besides, the set of weights used for the different indicators could be further optimised as well. An interesting idea is to optimise the height of the thresholds at the same time as well.
Figure 5 shows the net-worth versus the time if the advice of the program in the bottom graph of Figure 4 would have been followed. The net-worth is initialized at a 1000 units. In between November 2017 and February 2018 the net-worth remained the same due to the fact that all the stock was sold back at November. As can be seen, over half a year the program manages to make a profit of about 10%, which is good news. The profit is probably the most important indicator of the programs performance. Being able to express the performance in a single number is important if a genetic algorithm is to be used for weight optimization, see subsection 9.1.

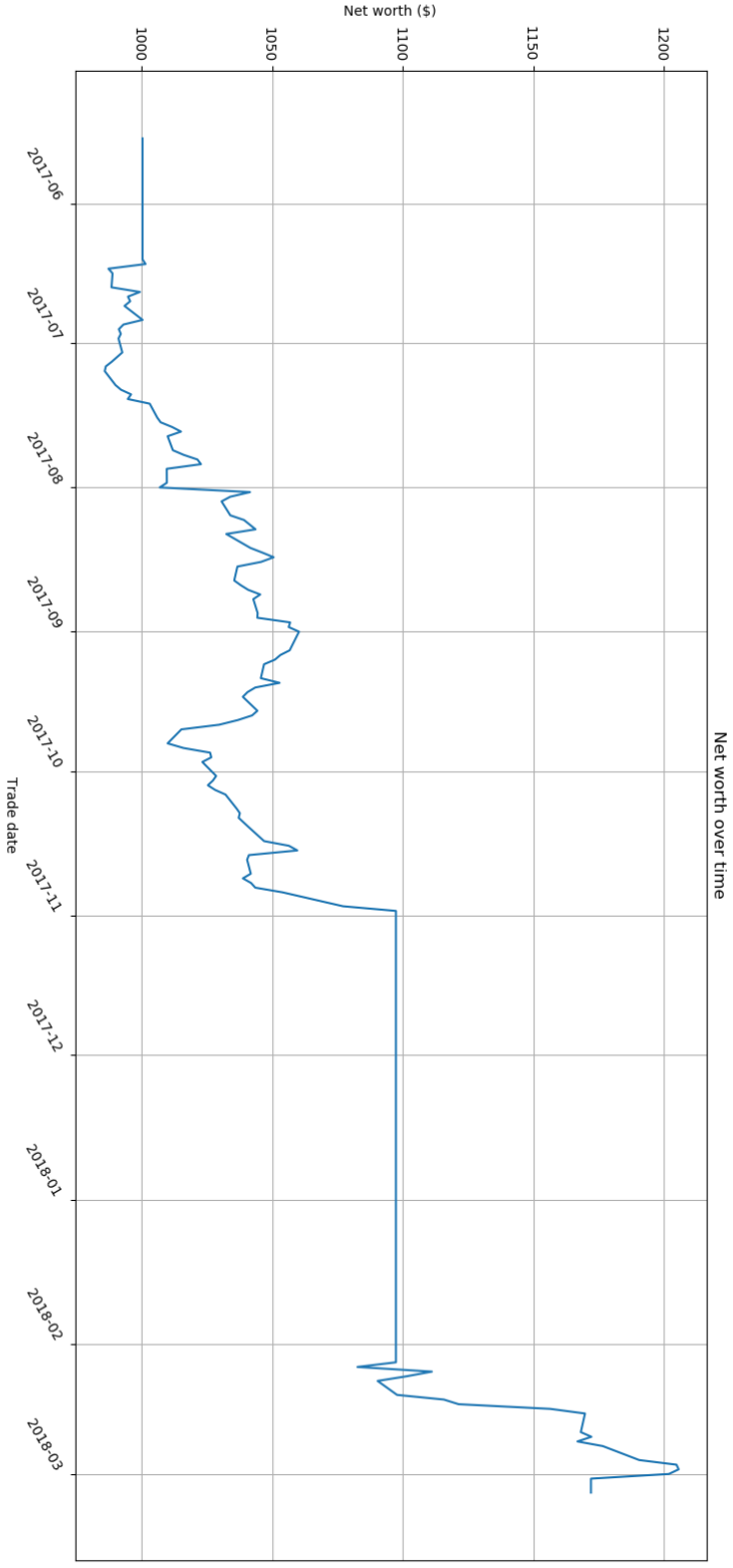

Figure 5: Profit generated by the technical analysis over time

8 Conclusion

Looking back at the objectives that were set out, it can be concluded that this project has been successful. In this section, a conclusion is drawed and the drawbacks of our software will be analyzed along with a more efficient solution for future analysis.

Technical analysis results

The software was able to pick up trends with technical indicators well enough so that, theoretically, money could have been made. The software was able to locate and predict clear dips and ups in the evolution of the values, effectively achieving two digit percentage profit on a significant number of occasions (the profit achieved also greatly depended on the time period and the time period length). The main draw back of the technical analysis still remain its disability to account for fundamental news, resulting in small unpredictable patterns which ultimately, confused the signal, limiting its accuracy to large trends only. A possible fix for this would be determining what events lead the values evolution, allowing for it to be taken in account in the analysis performed and in the weights of the Affectum signal.

Fundamental analysis

Evidence was found that a Twitter sentiment analysis is indeed a useful indicator for a fundamental analysis. As with the technical side, the fundamental side is not perfect. Most of the time, the sentiment is quite uneventful, people are generally not going to radically change their opinions about a company like Apple, and place tweets about this. It is when some big news comes out or an important event happens that people tweet about it, and then the sentiment will show a significant change. Combining the two into one master signal that can make buy and sell decisions is still a big challenge. It can be done with manual tweaking, but this takes a lot of time, which is undesirable. Using machine learning to tweak the parameters instead of doing it manually would be a better solution, but unfortunately there was no time left to implement this.

9 Future Plans

This section provides a brief overview of concepts that have not been implemented, but are the next logical steps to further development.

9.1 Genetic Algorithm

The genetic algorithm is the most prominent, and probably most impactful concept. Allowing a Genetic algorithm to optimize the parameters used by the various indicators and weights in the weighted signal addition would effectively allow for "tuning" of the model and Affectum signal. This would in turn lead to a significant increase in profits as the accuracy of the model increases drastically.

9.2 Fundamental indicators

The Dow Jones Text Feed could be an important addition. It is a news feed provided and purposely made readable for computers. A large advantage of this indicator is that it does not need filtering, as the only information that is send is about the Dow Jones index. However, this is only useful when looking at American companies, as the Dow Jones index is only based on American stock Markets.
The CNN fear and greed index would usually be a very good addition. However, for the last few months this indicator had been stagnant. Therefore it was not very useful during that period in time. Since 2019 has started, this index has become a lot less stagnant. Therefore it is worth to be added in the future.

9.3 Technical indicators

*add Technical indicators here that we want to add!!!

9.4 Multiple Trading bots for portfolio management

An especially interesting addition to this trading algorithm, is trading with multiple trading bots, monitoring different companies. This way if one of the stock value's are not favorable for buying, your money is not stagnant but is invested in another stock that is favorable. Your money could also be invested in multiple stocks at the same time. This decreases risk significantly, as less money will be invested in just one company. This in turn should allow for a more optimized portfolio management, leading to an increased profit over time.

References

[1] J. Henrique, "Getoldtweets-python," https://github.com/Jefferson-Henrique/GetOldTweets-python, 2019.

A Glossary

A.1 Bull / Bear

A Bull (or Bullish behaviour) is the opposite of a Bear (or Bearish behaviour).

Bull: A bull is an investor who thinks the market, a specific security or an industry is poised to rise. Investors who adopt a bull approach purchase securities under the assumption that they can sell them later at a higher price. Bulls are optimistic investors who are attempting to profit from the upward movement of stocks.

Bear: A bear is an investor who believes that a particular security or market is headed downward and attempts to profit from a decline in stock prices. Bears are typically pessimistic about the state of a given market.

A.2 Relative Strength Index RSI

The Relative Strength Index - RSI is a momentum indicator that measures the magnitude of recent price changes to analyse overbought or oversold conditions. It is primarily used to attempt to identify overbought or oversold conditions in the trading of an asset. The relative strength index (RSI) is calculated using the following formula:
$RSI = 100 - \frac{100}{1+RS}$

Where RS = average gain of up periods during the specified time frame / average loss of down periods during the specified time frame.

Traditional interpretation and usage of the RSI is that RSI values of 70 or above indicate that a security is becoming overbought or overvalued, and therefore, may be primed for a trend reversal or corrective pullback in price. An RSI reading of 30 or below is commonly interpreted as indicating an oversold or undervalued condition that may signal a trend change or corrective price reversal to the upside.

A.3 Simple Moving Average - SMA

The Simple Moving Average - SMA is a average over a certain time period. It is calculated by adding up the closing prices of a security for certain amount of days and then dividing this by the amount of days:

$SMA(date) = \sum_{n=date}^{date-days} \frac{closing\_price(n)}{days}$

A short term SMA has a relatively low amount of days it calculates the average over, long term SMA has a relatively high amount of days it calculates the average over. Therefore, a short term SMA follows the security price quite well, while long term SMA captures more of the long term movement of prices.