Stock Price Indicator

Investment and Trading
Capstone Project

Guilherme Augusto Camara

March 28, 2017

1 Definition

1.1 Project Overwiew

The future value of a company stock over a period of time can be predicted using a methodology known as *Stock Market prediction*[1]. Given some metrics over a period, such as opening stock price, volume of stocks traded and highest stock price traded, predictions can be done using an algorithm. The analyses of all possible inputs are carried out, in order to predict the Adjusted Close stock price (in other words, the future value of the company).

Accurate future value prediction is an objective pursued by many companies. This project will not be sensible to all data available, trying to perform data-mining in twitter and other tools to predict if a company will have a drastic fall (or rise) on its stock price. I intend to investigate the financial market, understanding how it works and the variables responsible of its changes, not to create the best predictive tool on the market.

1.2 Problem Statement

The stock price determines the value of a company in the market. Many companies and people invest their money on stock shares from companies. Buying and selling stocks in the right moment can make an average-income person become a millionaire, as well as make somebody go bankrupt.
The main issue is to determine the right moment in which to buy or sell stock shares in order to achieve the highest profit as possible.

Many companies release daily reports about the best options on the stock market. They determine from which companies you should buy or sell shares, considering the short and the long term.
For example, if you are able to predict that a company will grow 150% in the next two years, it would be wise of you to invest all of your money in this company's stocks. There is no investment in the market that guarantees this kind of return with no risk the money invested. However, two years is a long period and there are various things which can happen to this company in this time. Therefore, it is necessary to acquire the ability of predicting if this will become a bad decision, as well as of determining the right time to sell the stocks.

An increasing number of open source tools can currently provide a significant amount of information about the stock market. Thus, historical information can be downloaded in order to analyze how the stock price of a company behaved over the past years. All the data is based on dates and prices. Therefore it is quantifiable, measurable and replicable, given that data from the past can be analyzed. The problem stated is to determine the *adjusted close stock price*, and it will be done through an analysis of historical and recent data from stock values.

1.3 Metrics

Once a model is developed to fit the data and make predictions, an evaluation method to measure its performance is necessary. There are different types of evaluation methods for different types of datasets. The dataset that will be studied and used for this project consist in a continuous output. Therefore, the metric used to measure the performance of this model should consider a continuous model.

The *Coefficient of determination*, know as *R squared* is a metric largely used to regression models, because it can indicate the proportion of the variance in the dependent variable that is predictable from the independent variables[3]. The $R^2 score$ is a statistical measure of the closeness between the data and the fitted regression line [2]. The result will be between 0 and 1, where 0 indicating that the model doesn't explain the data, and 1 indicating that the model completely explains the data.

2 Analysis

2.1 Data Exploration

From the *Yahoo Finance* website some features can be achieved from each stock company. It is provided the same features to every company listed. These are the features:

- **Open Price:** Is defined as the price of the first trade of the day. It is common that the opening price is not identical to the last closing price. This occurs as effect of the investors' expectations outside the trading hours.

- **High price** and **Low price:** The high price is defined as the highest price at which a stock traded over the day, while the low price is the lowest price. Analyzing these values it is possible to find gaps or sudden jumps in a stock's price when there was not trading between those two prices.

- **Volume:** Is defined as the number of transactions occurred in a day. The volume is an important measure of strength of the prices. If the volume is higher than average, it is a sign that a particular stock price is getting stronger. Therefore, analyzing the volume high helps to understand the price movement.

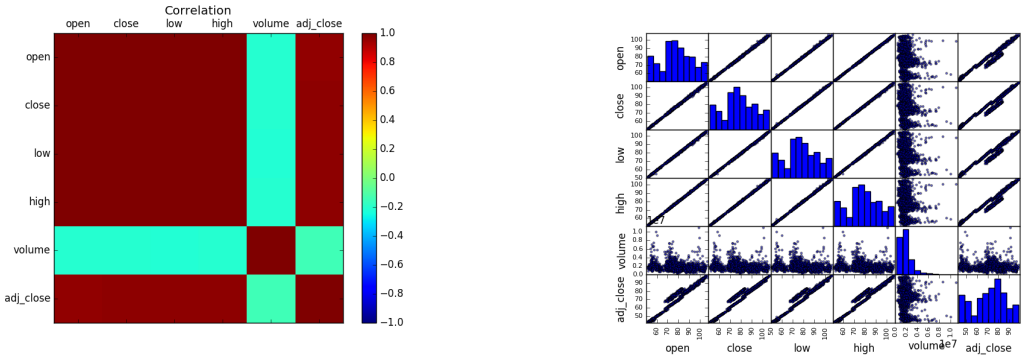
- **Close Price:** Is defined as the price of the last trade of the day. Although it does not mandatorily reflect the next's day opening price, this value is important for investors to analyze changes in stock prices over time. Comparing closing prices, investors can measure the market sentiment over a period.

- **Adjustment Closing Price:** Is defined as a stock's closing price that has been amended to include all corporate actions, such as stock splits, dividends/distributions and right offerings, that occurred at any time before the next's day opening price. This marker gives analysts an accurate representation of the company's equity value beyond the simple market price. This feature is often used to study the company's historical performance.

To develop this project one company needed to be chosen. In order to understand better all the data available, and study the best algorithms, some points about a company were defined: The company should be on the market for the least five years and, in this period, demonstrated some soft price variation. This condition was based on the objective of this project: It will not to determine when some company will break or rise. It will only predict for some period if the company stocks will be valorized or not. In order to determine the sentiment of a company to predict if it will break or not, would be necessary data from other sources, such as twitter. However, this project will remain as an open source code and eventually will be developed to work with new features. Therefore, considering all the points stated before, the company **NVS - Norvatis AG Common Stock** was chosen and the following results will be from it.

2.2 Exploratory Visualization

The first step is to understand the dependency between the features. Plotting the correlation between the data, the following images were generated:



(a) Correlation graphic

(b) Scatter matrix graphic

Figure 1: Correlation between features of the dataset

Analyzing the Figure 1a and the Figure 1b, it is clear that all the features, except to the Volume, are all correlated. For this project, this means that can be used just one of them to build the model. The Volume feature does not seems to help with the model, so this feature will be excluded. In the Figure 2 three features are plot together, in order to understand their correlation.

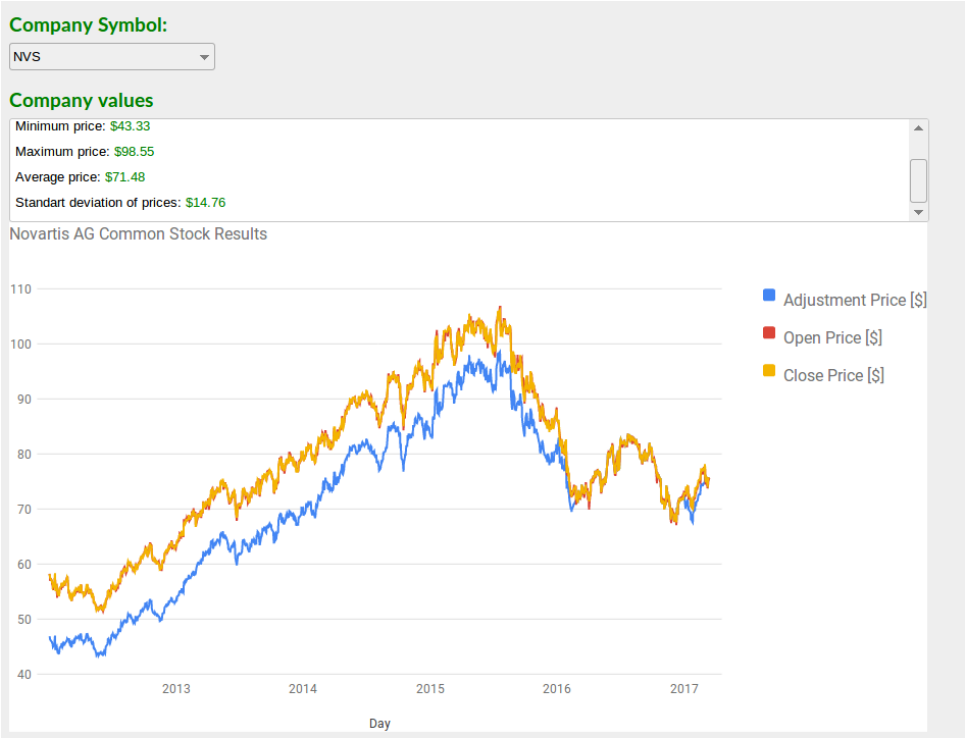


Figure 2: For the past five years, the features Open, Close and Adjustment Price are displayed.

The Figure 3 presents the beginning of the data set and how it is structured. Folowed by its statistics.

	name	symbol	date	open	close	low
0	Novartis AG Common Stock	NVS	2012-01-03	58.24	58.18	57.85
1	Novartis AG Common Stock	NVS	2012-01-04	57.9	57.93	57.48
2	Novartis AG Common Stock	NVS	2012-01-05	56.92	57.46	56.76
3	Novartis AG Common Stock	NVS	2012-01-06	57.37	57.31	57
4	Novartis AG Common Stock	NVS	2012-01-09	57.12	57	56.72
5	Novartis AG Common Stock	NVS	2012-01-10	56.86	56.93	56.82
6	Novartis AG Common Stock	NVS	2012-01-11	56.15	56.49	55.85
7	Novartis AG Common Stock	NVS	2012-01-12	56.56	56.73	56.23
8	Novartis AG Common Stock	NVS	2012-01-13	55.3	55.8	55.11
9	Novartis AG Common Stock	NVS	2012-01-17	56.27	56.28	56.14

Figure 3: Structured head of the Dataset.

Minimum price: $43.33
Maximum price: $98.55
Mean price: $71.50
Median price $73.43
Standard deviation of prices: $14.72

2.3 Algorithms and Techniques

The result that this algorithm will predict is continous. This is a carctheristic of a linear problem, therefore, regression algorithms will be used to develop this model. However, there are few different methods for linear regression. In order to choose one to work with, the following *map* can be analyzed, represented on Figure 4.

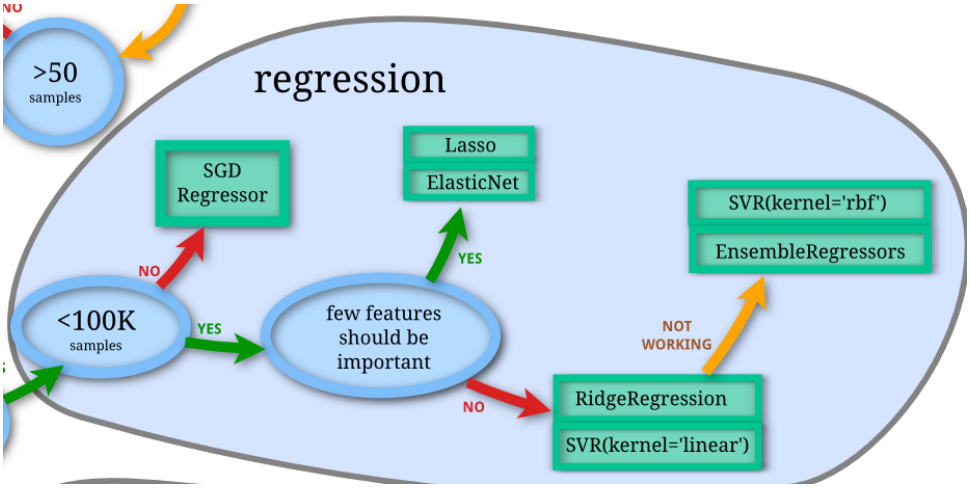


Figure 4: Scikit-learn algorithm cheat-sheet for regression.

The dataset available is relatively small, it is smaller than one hundred thousand samples. The number of features are small, there is only six features available. The data that this algorithm will work also can not be defined as regression threes, since new data will be predicted - the next stock prices week. The behavior of the data can not be estimated with a simple line or polynomial model. And, after analyzing all these points, the model selected to be used is the **Support Vector Machine - Support Vector Regressor**.

The *Support Vector Regression* algorithm is a good fit, because of the kernel trick. As others regression models, the *SVR* present a method where it always try to fit a line in between the points, and this line is designed to be always as far from the points as possible. But, this model also presents kernel options. Kernel is essentially a mapping function, which is used to transform a given space in some other. There are few default kernel options available for SVR: Linear, Polynomial, RBF, Sigmoid, Precomputed or custom. And the one that fits better this project is the RBF - Radio Basis Function Kernel. Also, the *SVR* algorithm presents some others tune options. Once the model is defined, the easiest way to defined the best options is to try each one and define which combination is better. There are a few tools available to perform it, consuming more cores of the processor, GPU, less RAM, and so on. It was implemented the *Pipeline* method, consuming 4 cores. The *pipeline* is used to assemble several steps that can be cross-validated together setting different parameters.

2.4 Benchmark

The result can be easily evaluated, since it will be quantifiable and compared to the already existing value. Therefore, the evaluation of the success of the model can be done by comparing the final predicted value of the *Adjusted Stock price* with the real one. The benchmark value, considering the $R^2 score$ (section 1.3) for this project, that will be pursed initially will be of 0.3. Since every day the results can be updated in order to correct the predictor, this benchmark is referred to the next market day, where the data would not be available yet. The objective is to determine if the predictor can identify the tendency of the market, in order to provide a prediction better than a random guess.

3 Methodology

3.1 Data Preprocessing

The first step is to acquire the data. Using the *Yahoo Finance* tool for python, the data is easily fetched with using the company stock symbol and the historical period. After it, an object containing the stock prices and some other informations will be acquired. In order to avoid fetching the same data every time the code in run and to avoid store everything on the RAM memory, there was developed a database to organize and store the data. This way, only new data will be fetched and stored. This database is designed to store data for any number of companies.

The database was organized in three tables:

- **Companies:** Stores the Symbol and the Name of all listed companies.
- **Period:** Stores the period of the data stored of each company.
- **Data:** Store all the useful data of all companies.

The Figure 5 shows the organization of the database.

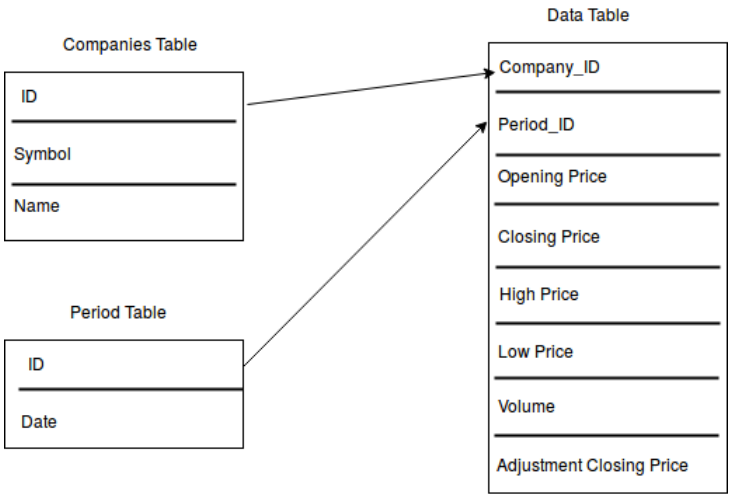


Figure 5: Organization of the tables.

With this process, the data need to be achieved before the processing. However, while the data from one company is being downloaded into the database, the next algorithm can be performed to the data of a storage company. Also, the data of all wished companies can be downloaded into the database before starting. The positive point about this, is that RAM memory is saved by storing data in hard disk. However, this database can grow as the hard disk supports, therefore it is important to not try to download all the data from every company in the stock market. Once the data is organized in the database, the Preprocessing starts.

In order to deal with the outliers points, the solution used was to implement the *SavitzkyGolay filter*, which is a digital filter used to smooth the data[4]. This solution is easy to implement and works fine with this dataset, however, the challenge is to define the best parameters. It is common to conclude that the best parameters would be the ones that result in the best *Rscore*. But, this would not smooth

the curve, just fit it perfectly. Therefore, the best parameters should be the ones that describes the data, but exclude the outliers. The result of this filter application can be visualized in Figure **??**:

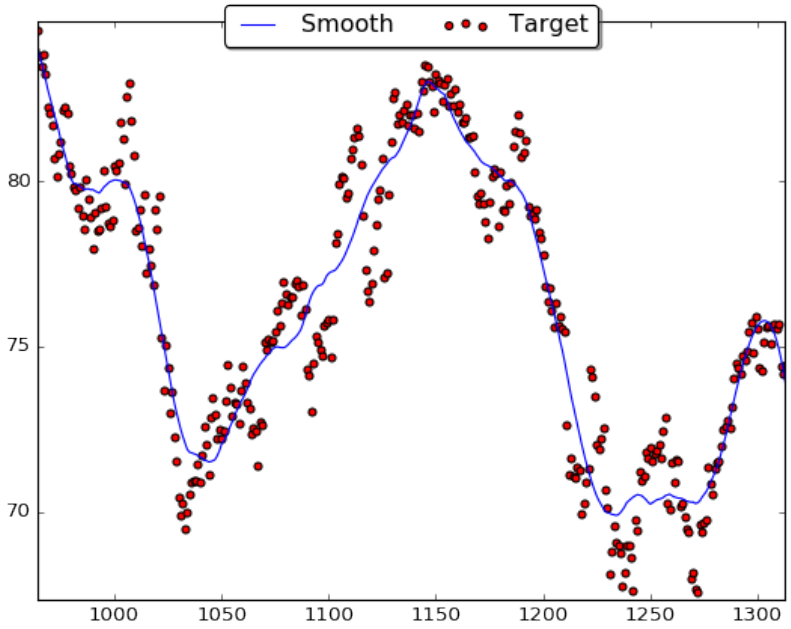


Figure 6: Comparison between data points and the smooth plot

3.2 Implementation

Given all the features available: *Open, Close, High, Low , Volume and Adjustment Close Price*, the firs step is to define which data will be used. The *Adjustment Close Price* is the label of this project, or, in other words, the data that this algorithm will try to predict. As studied in the Subsection 2.2, all the remaining prices, except for the *Volume*, are fully correlated. Therefore, *Open, Close, High or Low* should perform as features well. However, any of this values are defined before the next business day starts. The consequence of it is that the predictor would only work when, for example, the open price is given in the beginning of the day (as explained in 2.1, the open price is not always the same as the last close price). Therefore, in order to perform this regression, the feature used will be the time.

Once all the data to be used was defined, the next step is to split the data into train and test data. The method utilized for this project was the sklearn *train_test_split*. This method split arrays or matrices into random train and test subsets. The subset test was defined as 20% of the whole data.

Using the *Support Vector Regressor*, the next step is to define the best regressor. The *SVR* presents some tuning options, and in order to find the best option the method used was the *Pipeline* algorithm as defined in Subsection 2.3. In order prevent excessing process, the first thing was to define the kernel, and The *Radio Basis Function (RBF)* kernel was the best fit. After it, all others parameters were left as variables to be defined in the process.

3.3 Refinement

In order to obtaing better results, studying the code was possible to find two options to improve it, in the pre-processing and in the learner. In the preprocessing, as explained in subsection 3.1, a filter was implemented to perform a smooth in the data. There are two options to tune this filter, the window that the algorithm will consider to perform the regression and the function's degree used in this regression. The metric used to evaluate how close was the new curve to the data, was used the *Rscore*. So, in order to discover which pair of parameters would be better, it was simulated the learner with three scores: The worst score, a mean score and the highest score. The results showed that the mean score was the right solution, therefore a pair of parameters that gave a score of 0.9942 was chosen.

The other part of the algorithm needed to be improved was the learner. As explained in subsection 3.2, the *Pipeline* method was implemented to obtain the best parameters to the learner. Using the *RBF* kernel, there was a couple of options to tune:

- C: Penalty parameters
- Gamma: Kernel coefficient

Once the *Pipeline* process is implemented, the *GridSearchVC* will be responsible to find the best set of parameters simulated, using the *Rscore* as value of decision.

3.4 App development

Before implementing any new method or algorithm, efforts were made to create an interface to visualize and interact with the process. In order to implement something simple before moving to new high-end solutions, the solution was the framework python Web, with Javascript and HTML. The structure defined was: An algorithm in python responsible to launch the Web server and perform the back end of the project, which is achieve data and perform the predictions. The HTML is rendered by the python to be visualized, and the javascript is responsible to create the visual, graphics, interacting with the HTML and Python. The visualization was defined in three steps: Select the company and the period, select the algorithm to be used and the period that will be predicted.

To the visual part of this web solution was used *jQWidgtes*, that can be downloaded on http://www.jqwidgets.com/, and *Google Charts*, that can be downloaded on https://developers.google.com/chart/ . These frameworks were defined after previous experience using them. And this solution was implemented to organize better the code and visualize each step of the process. As explained before, the following three steps were defined:

- Choosing Company: A list of available companies symbols is listed in order to be chosen. After choosing one, the Figure 2 is displayed. The objective of this visualization is to understanding the stock prices over a period. This is just historical data, no preprocessing were made until this point.
- Selecting Algorithm: A list of available algorithms is listed. After a choice is made, a graph is displayed presenting a comparison between the historical data and the result of the prediction to all the data. In this step, when using the *SVR* method, the pipeline algorithm will be used to find the best parameters. A box containing information about the regressor used and the **Test Score** will be showed. Illustrated on Figure 8.
- Select Period to predict: A list of available days to predict will be displayed. It will have options from one day to forty five days. When the period is selected, a graph will be displayed showing the behavior of the stock prices over this period. Illustrated on Figure 9.

4 Results

4.1 Model Evaluation and Validation

There are two different scores to be considered as result, the score given by the training and test data, which performed only over known data, and the score resulted by future predictions. These scores will be defined as **Test Score** and **Prediction Score**.

About the **Test Score:**
After considering the aspects explained on subsections 2.2 and 3.2, the main model developed to perform the predictions use the *Support Machine Regressor* algorithm. Simulating the results of several companies, the $R^2 score$ reached for everyone was higher than 0.9. All the results was achieved using the kernel *RBF* with *Pipeline* in order to obtain the best possible parameters. However, it is necessary to display the results in a understandable and efficient visualization. This objective was reached using *google's chart API* for HTML. This solution allows the user to interact with the graph in an easier form, understanding better what is being predicted.

This model is robust enough for this problem and performs with a high accuracy. Small changes are being rightly predict, but, as explained in the section 1, the scope of this project is not to predict when a drastic fall (or rise) of stock shares will occur. Therefore, in order to understand better the stock prices of some companies and to predict if it might be a good option to buy or sell shares in a short term, the algorithm implemented is robust. The graphs generated by this algorithm will be displayed in the next subsection.

Until the final model was no decided, some others model were stressed in order to obtain great results. These models are available, when using the *app* developed for this project, to be selected and executed. An algorithm of *deep learning* is implemented, however, it is not optimized to work with different data, which makes it unstable to be used.

About the **Prediction Score:**
Using the learner acquired by last simulation, was separeted the last 10 days of the dataset, to visualize how the predictor would perform. The score achieved was of 0.2229. The Figure 7 compares the data of these 10 days, the smooth curve used in the predictor and the predictor.

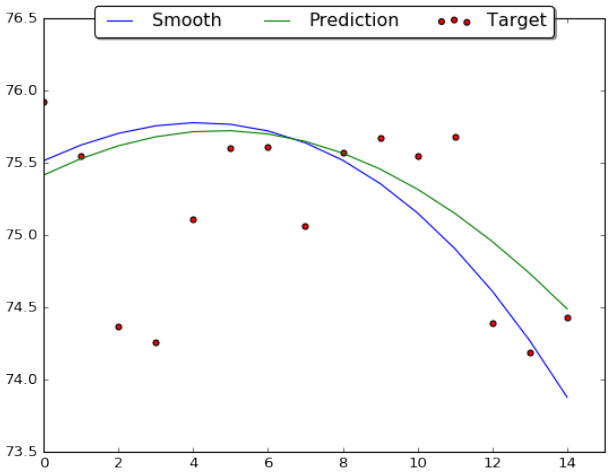


Figure 7: NVS: 10 days of prediction

4.2 Justification

As explained in subsection 4.1, there are two scores considered. Firstly it will be showed this results, and the predictions performed with them, then a comparison between the bechmark and the wanted score.

Test Score
Results achieved for the company *NVS - Norvatis AG Common Stock*. The Figure 8 presents the results in two parts:

- Text: It contains three informations: The specific algorithm used, the score reached with the test data and the parameters used with the model.

- Graph: The historical data and the predict data for the *Adjustment Closing Prices*.

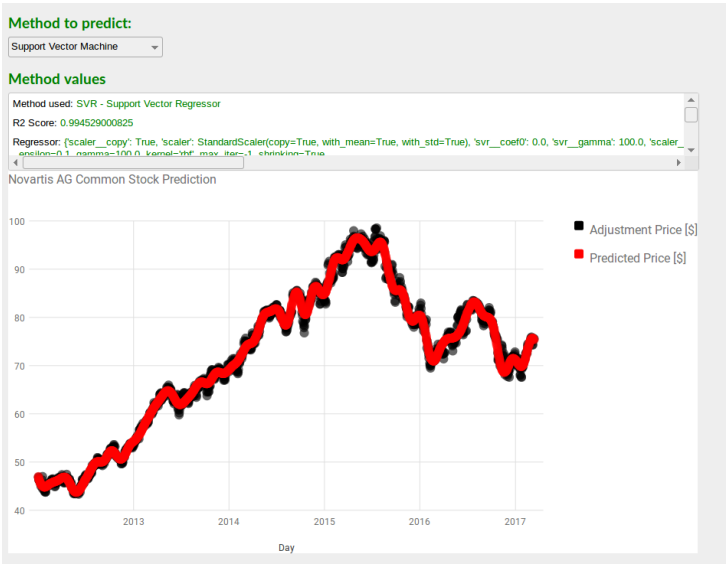


Figure 8: NVS: Prediction Results and Historical Data

The score reached in this simulation was 0.9945, which means that $99,45\%$ of the test subset data was predicted correctly. This score is higher than the wished benchmark in the beginning of this project. The graph, comparing historical data with all the data predicted by the model, is displayed to the user in order to clarify what this score means. After the model is trained, it is possible to predict the next days of the stock prices. The Figure displays the results. Since the Stock Market does not open every day, the sequence of the 45 days predicted in this image is wrongly named. It should be considered on working days.

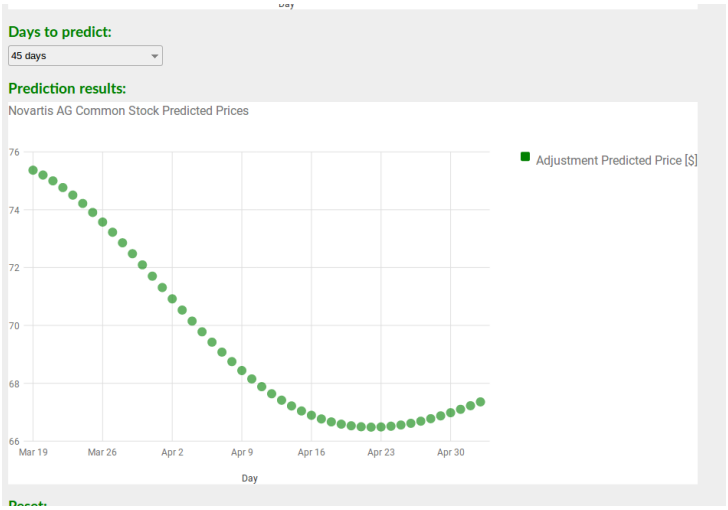


Figure 9: NVS: Prediction Results for 45 days

Prediction Score

After the score achieved with the simulation of the last 10 days, presented in subsection **??**, it is clear that this model is not efficient. Still, this is a better solution than a random guess. As demonstraed in Figure 7, the result tends to follow the prices, however with time it gets higer errors. Therefore, this model could work for a short term prediction, as one or two days, just to understand the value of the shares. The next section will consider the improvements that can be done for this project in order to achieve better results, and explain how they can be implemented.

5 Conclusion

5.1 Reflection

As explained in the Section 1, try to predict the future stock prices is a known problem and there is people trying to make money out of it. And, when there is people studying something, open tools and sources are released. Looking on the internet, you can easily find examples of codes as open source. Also, this code is becoming open source. With the model defined in this project, it is possible to easily visualize every step of the process, and, as explained in Subsection 2.2, the code is completely modular. Implementing new features and visualizations is really fast and easy to fit.

Studying for this project helped to understand better about regression methods, how to optimize them and how to preprocess data to best fit them. The most difficult part was initially to understand how to optimize and visualize all the data available. Also, implementing everything from the beginning was challenging, mostly finding the best way to modularize the code and to implement it with an interface. After visualizing everything and enabling to change parameters without hard-coding it the project took a cleaner form, and everything could be organized.

This application can run in any stock market and any company on it. Analyzing a company in a really short-term, such as in a couple of days, the predictions can reflect the real results. However, as stated on the beginning of this project, this code is not ready to predict a break on the market. For example, if a company is found to be in a corruption scheme, and loses market price, this algorithm will wrongly predict all the results until everything normalizes. Therefore, this application should only be used to understand better the stock prices movement.

5.2 Improvement

There is some improvement space to be done is this application. Starting with the prediction algorithm, there is some good results on the internet with deep learning algorithms. There is a sample of it already implemented, but it is not working fine yet. Using deep learning to implement this kind of prediction would be really interesting. Also, the front end solution presented can be improved with new frameworks, such as Angular 2. With a solution as this, the website would be more interactive with the user and the visualization could become better and cleaner.

However, the most important improvement that should be made, it is to understand the market sentiment in order to predict companies breakdowns and rises. This sentiment could be studied by the movement of the process, as explained in subsection 2.1. Also, performing data mining in websites, such as Twitter, can be a very powerful tool to define the sentiment. Since the code is modular, introducing this kind of feature on the database and in the prediction algorithm could be done, once this feature is defined and achieved. These are the improvements that are intended to be done on this code after this part is done.

References

[1] Wikipedia, Available from World Wide Web at (https://en.wikipedia.org/wiki/Stock_market_prediction), on 15/01/2017 .

[2] Minitab, Available from World Wide Web at (http://blog.minitab.com/blog/adventures-in-statistics-2/regression-analysis-how-do-i-interpret-r-squared-and-assess-the-goodness-of-fit), on 15/01/2017.

[3] Coefficient of Determination definition, Statrek website, Available at (http://stattrek.com/statistics/dictionary.aspx?definition=coefficient_of_determination), on 11/03/2017.

[4] SavitzkyGolay filter description, available at https://en.wikipedia.org/wiki/Savitzky%E2%80%93Golay_filter, on 27/03/2017.