Capstone Project

Machine Learning Engineer Nanodegree

Ivan Landim Frota Leitão Junior
September 27th, 2016

I. Definition

The stock market is full of opportunities, every day billions of reais are negotiated in the market. Over than half a million citizens have money invested in Bovespa with 120 billion reais of assets inside Bovespa, only in the hand of citizens, excluding companies owning assets. The amount of money and opportunity are big but also are the risks, plenty of people lose their money on the market, some of them losing large amounts.

In this scenario, our objective is to create a machine learning algorithm that would be able to predict the day a stock will go up. With this in hand, we would be able to increase the probability of ours intraday trades. The idea is to improve the trades that are done in the stock for just one day. This opens the possibility to invest with leverage (lend money) for free, only considering the risk of the operation. (In this report leverage will not be explained, check this link for more information.)

Project Overview

The algorithm will predict a day that the stock price will move up. The objective here is not to predict the amount it will go up but the day it will happen. Other studies can try to predict the better moment in these days to enter in the market or to predict the percentage it will move up, using this study as a starting point.

The chosen stock to be analyzed in the study is the ITSA4 (ITAUSA INVESTIMENTOS ITAU S.A. - Preferred stock). The reason for choosing this stock is because it has a long historical data and the negotiation is substantial. Other stocks can be analyzed later using the same algorithm just fitting their data.

The data used is raw data of the negotiations from 2006 to 2015, it will be over 2000 days of negotiation and it will be gathered from the Bovespa website. The data gives the price variation of the stocks in each day (open, high, low, close) and volume of the negotiation in the day.

As a bonus analysis, we will verify if buying in the open price and selling in the close price will give money if we follow the algorithm prediction. It is possible that the losses in the days it misses are bigger than the gains in the days it is right, this is the main reason to have it as a bonus analysis.

Problem Statement

We invest in a market that moves up and down, the probability that a given day it moves up is lower than 50%, in the last 10 years. Our intention is to have a higher probability in the days we decide to invest.

We will break the main objective into steps: - Gather and prepare the data for analysis - Include technical analysis indicators - Clean the data for usage in the Machine Learning - Split the data in training and test data - Prepare the machine learning with Decision Tree, K-neighbors and Random Forest - Evaluate the performance and pick the best model - (Bonus analysis) Check the stock performance of the chosen model in the test data

Metrics

To evaluate the performance of the model we have decided to use the precision score. The main objective of our analysis is to be in the market when there is a higher probability of gaining money. There is no problem that the market goes up when we are outside of it. We want to avoid the market going down when we are inside it.

For this reason, precision score was the best score in our point of view. We try to maximize the amount of true positive (positive trades that we considered positives) and to reduce the false positives (negative trades that we considered positive).

The precision formula is: precision = true positive / (true positive + false positive)

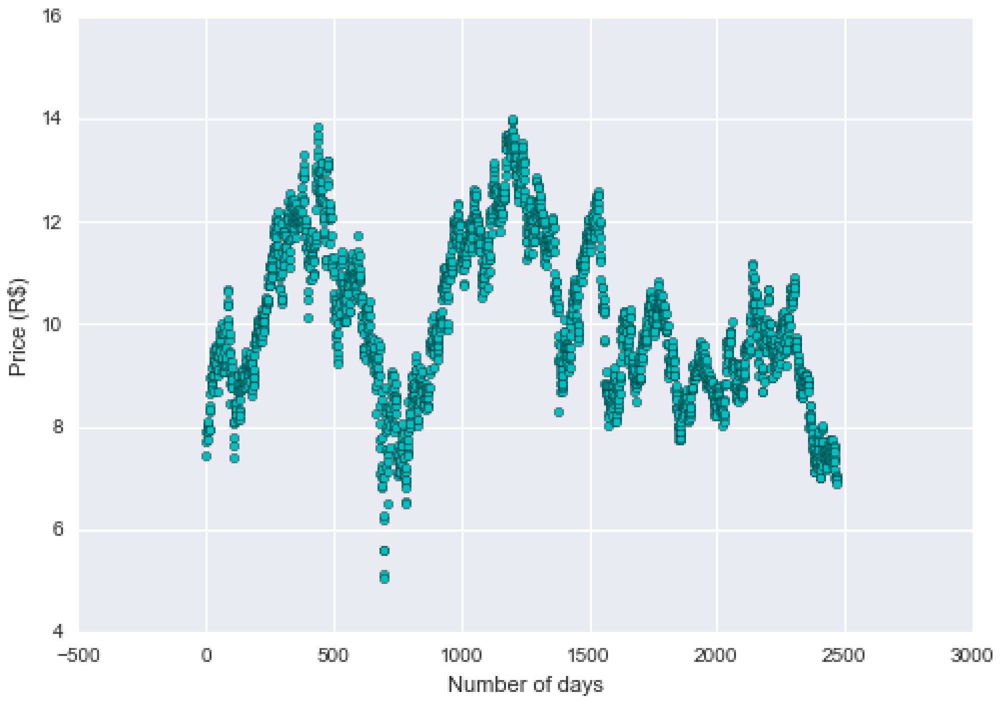
As bonus metric we considered the stock performance considering we bought the stocks in the opening of the day we expected to move up and sold the stock in the close of the day. In this case we considered we bought one stock each time and evaluate the profit we got in the end of all operations.

II. Analysis

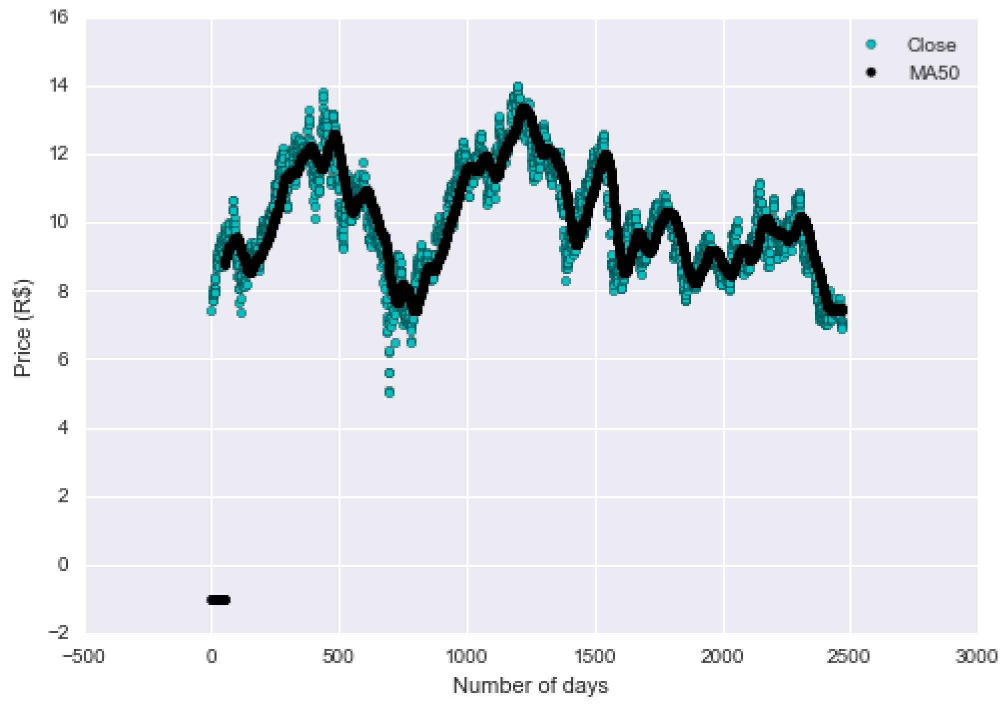
Data Exploration

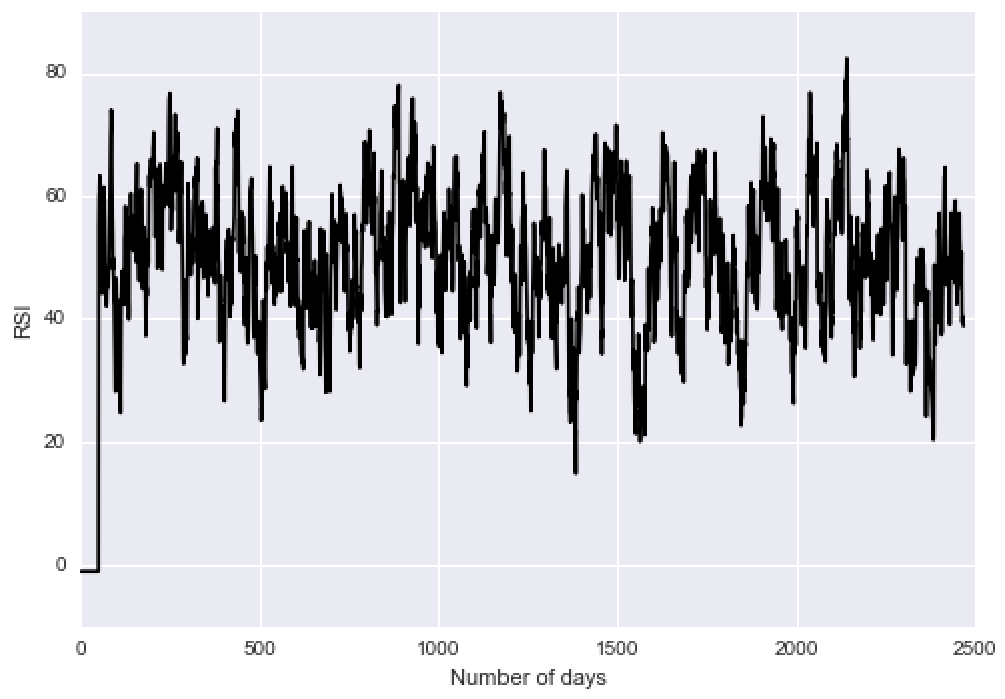
(Last reviewer complained that in the graphs for Down/Up there was number also, I couldn't do histograms with categories. If you know how to do it, feel free to explain, if not, please accept it. I don't think correcting every graph in Paint will improve the exercise in the end.)

The data used in this project is the historical prices and volumes of the stock for 10 years. The close price of the stock is in the graph above.

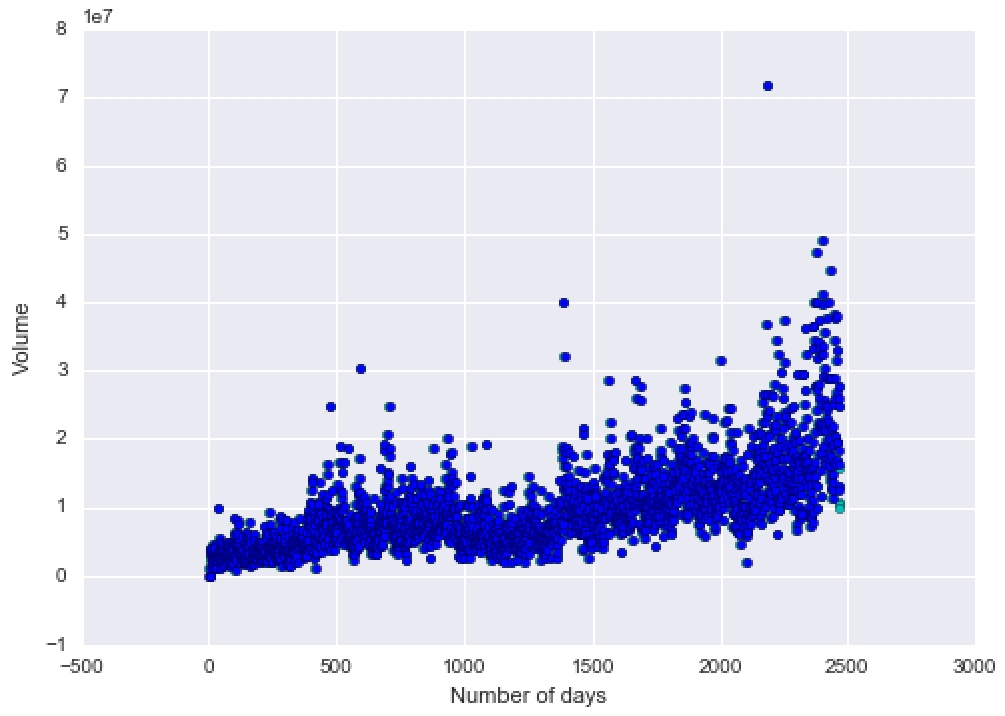


To improve the data we have added some new features using the technical analysis indicators as Moving Average and Relative Strength Index, in the images below. The detailed information about them are in the Jupyter notebook.

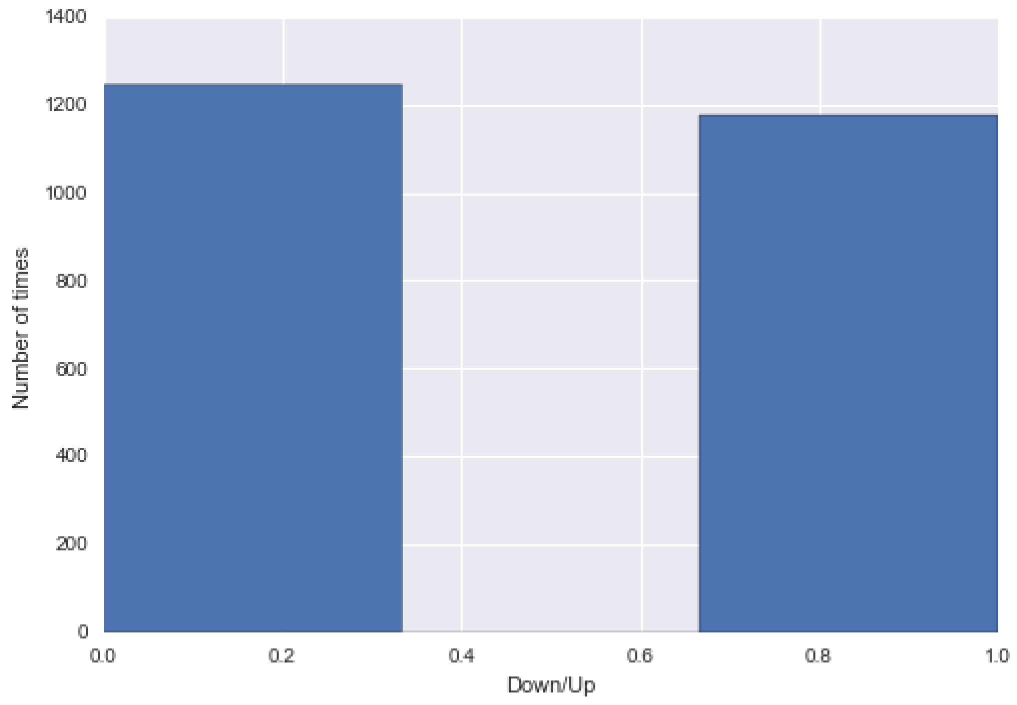




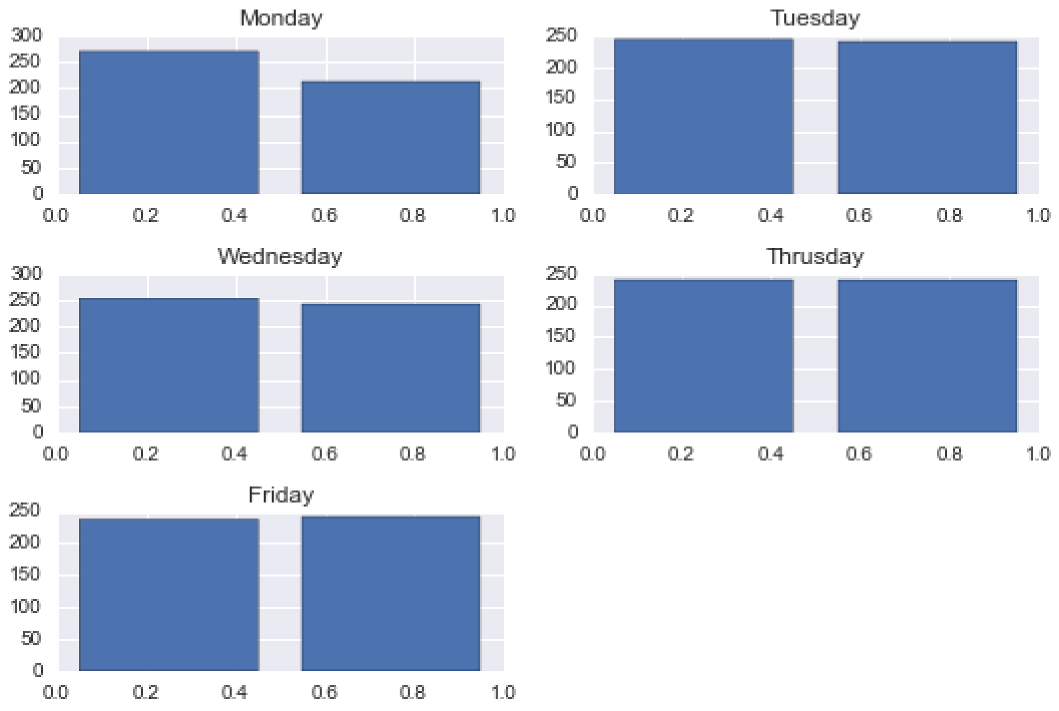
The volume of trades in each day are also presented here.

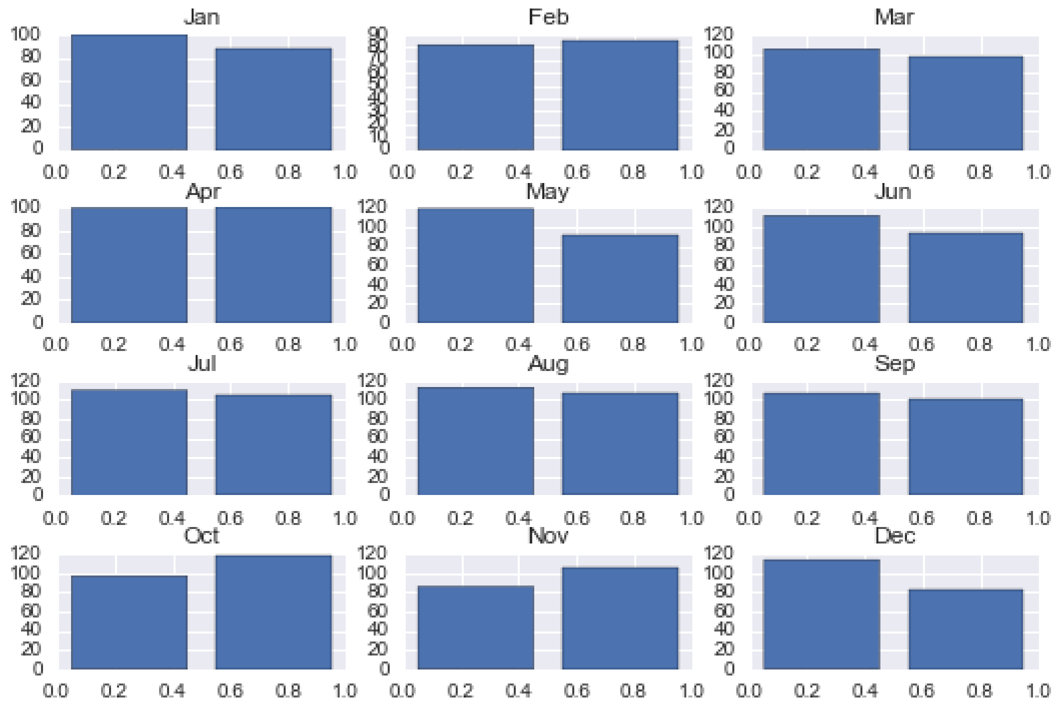


Verifying the amount of days it moved up, we can see that it is less than half of them.



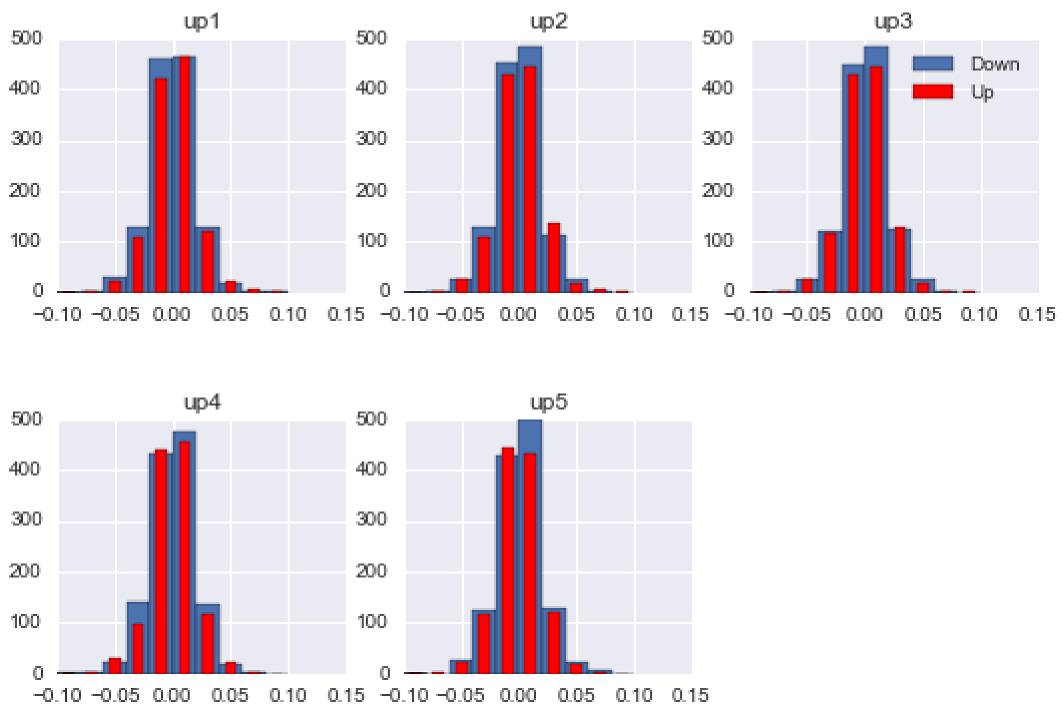
We also analyzed it depending of the day of the week and the month of the year.



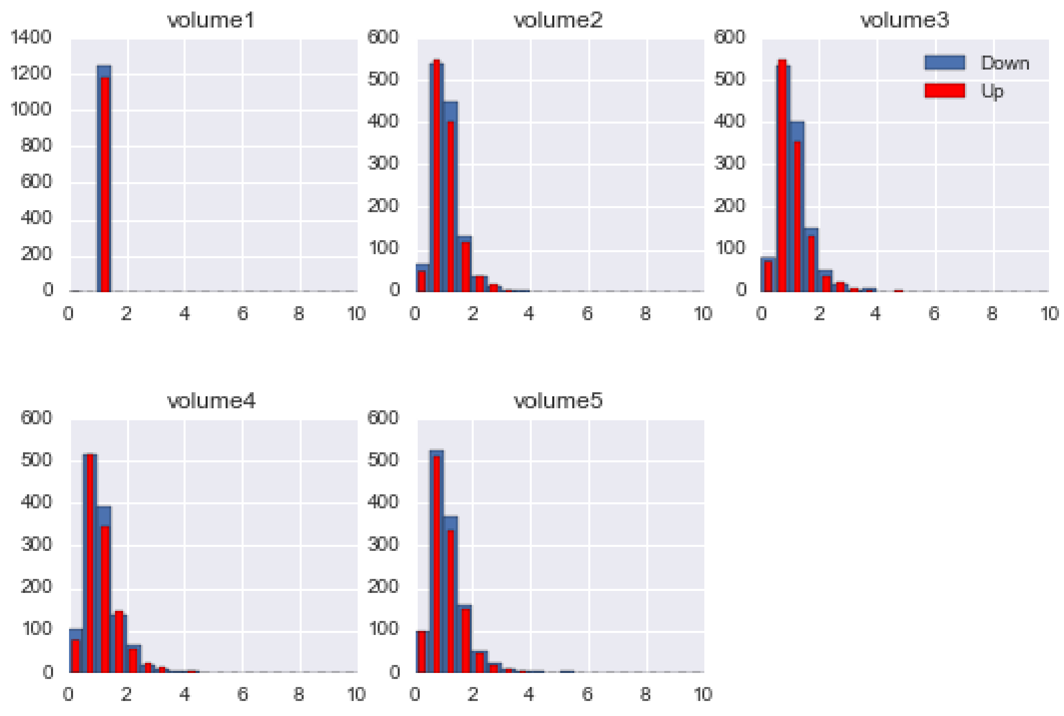


Exploratory Visualization

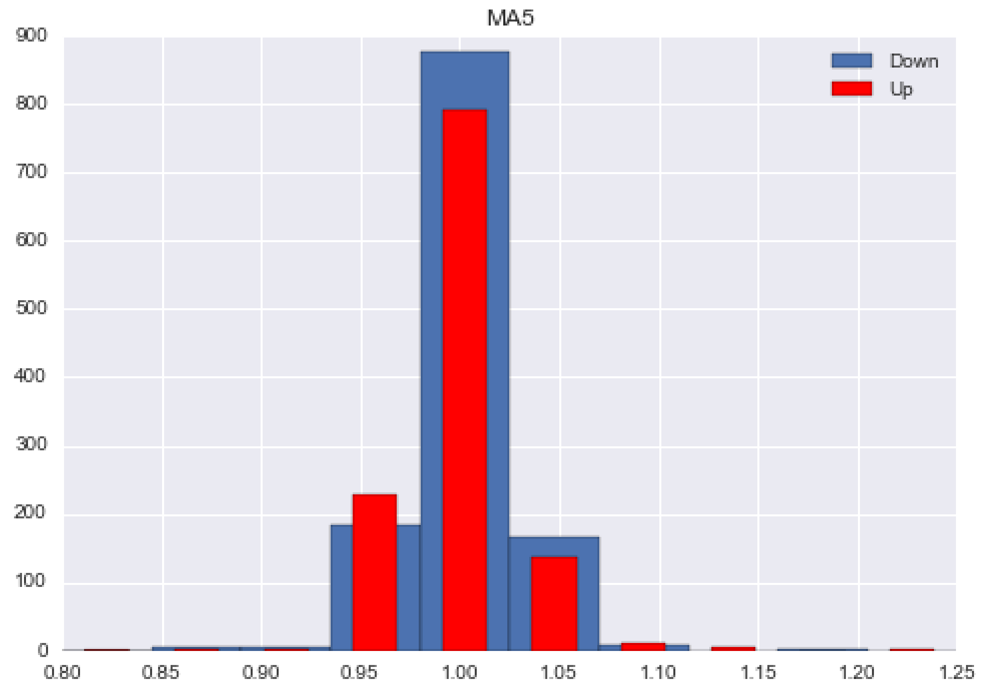
We also had performed tests checking if moving up in the previous 5 days changed the probability of moving up today.

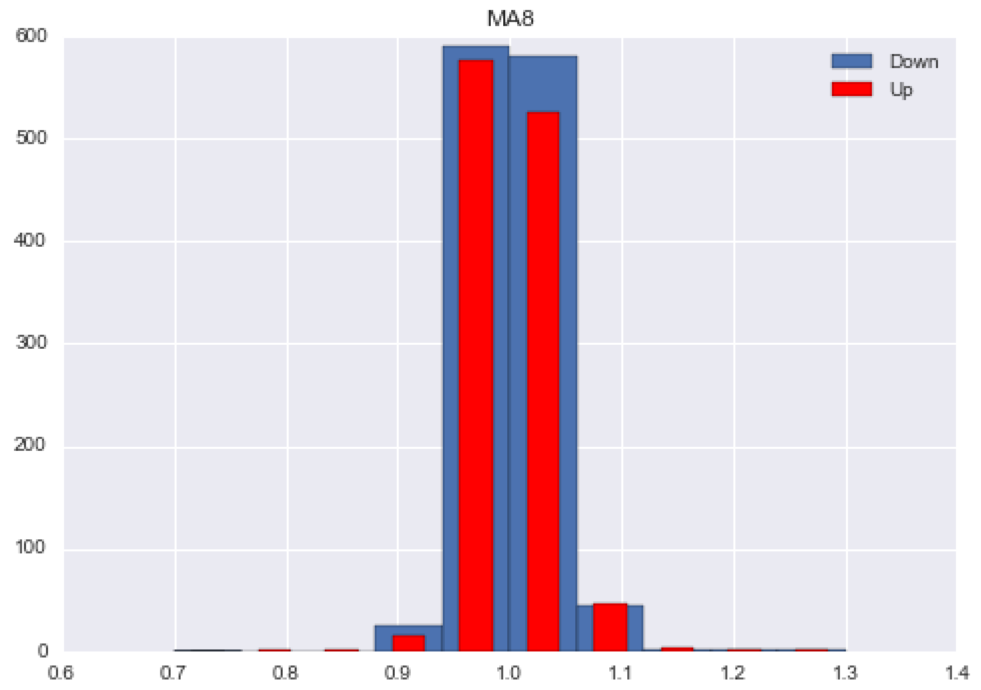


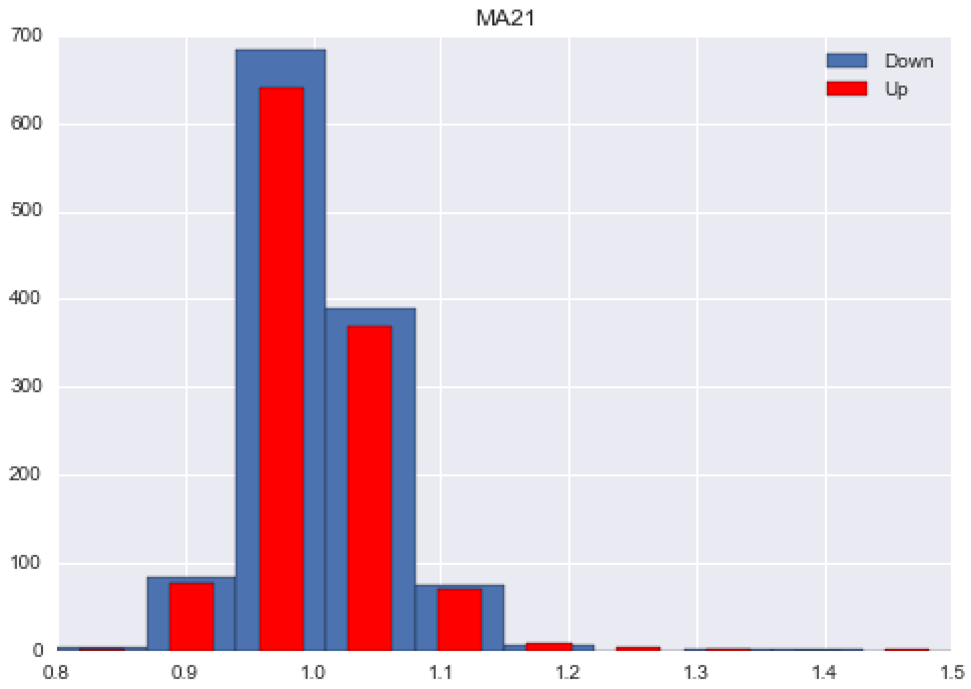
From the volumes we could analyze the last volumes in relation of the previous volumes.

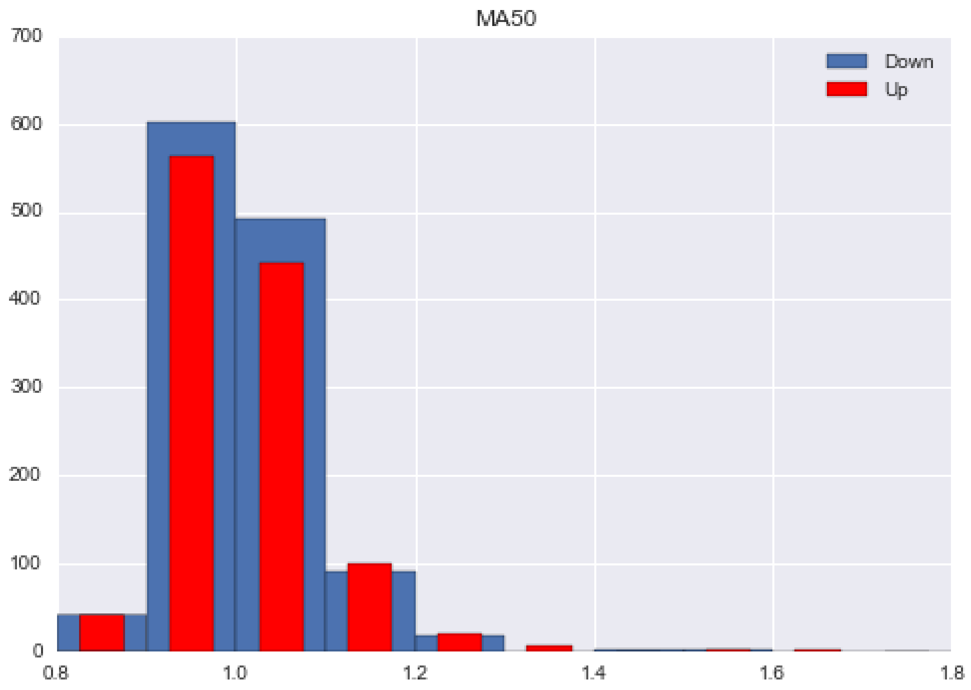


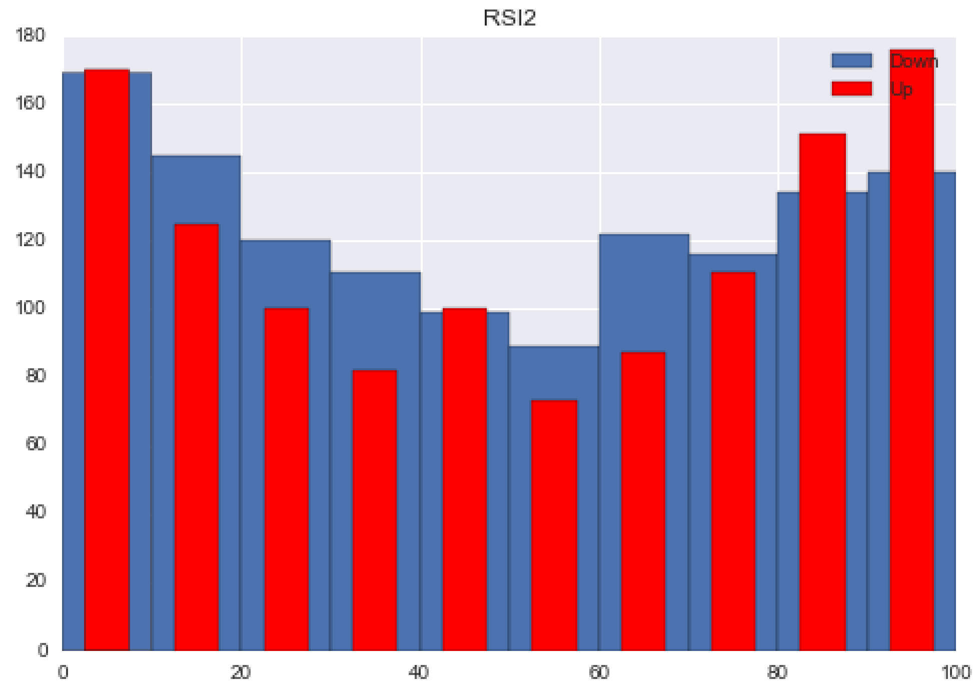
We also verified it for different Moving averages (in relation of the last price) and Relative Strength Index.

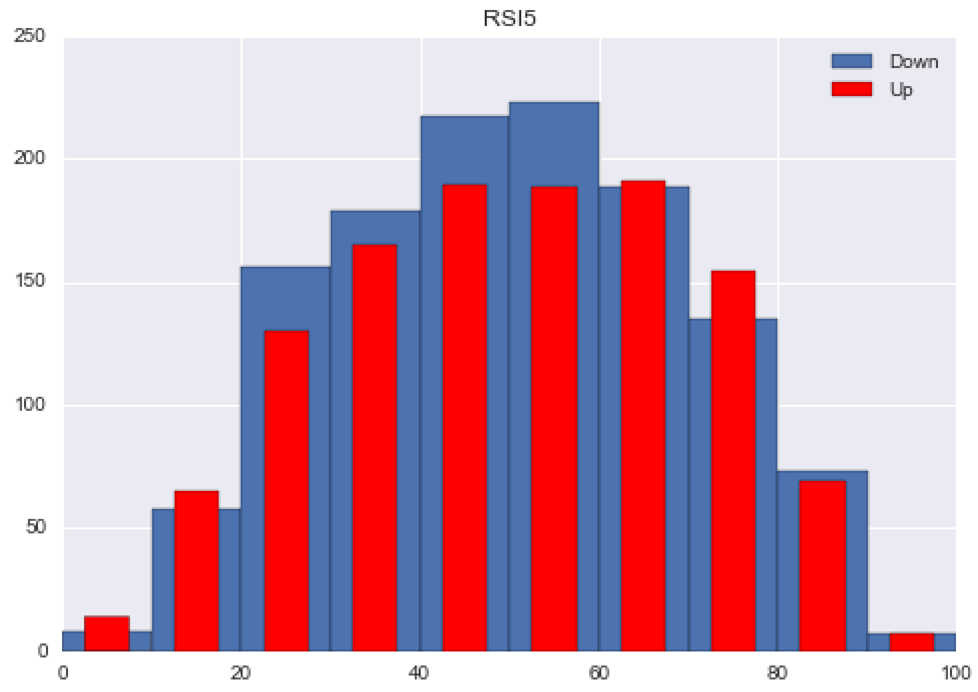

MA5
Down
Up
900
800
700
600
500
400
300
200
100
0
0.80
0.85
0.90
0.95
1.00
1.05
1.10
1.15
1.20
1.25

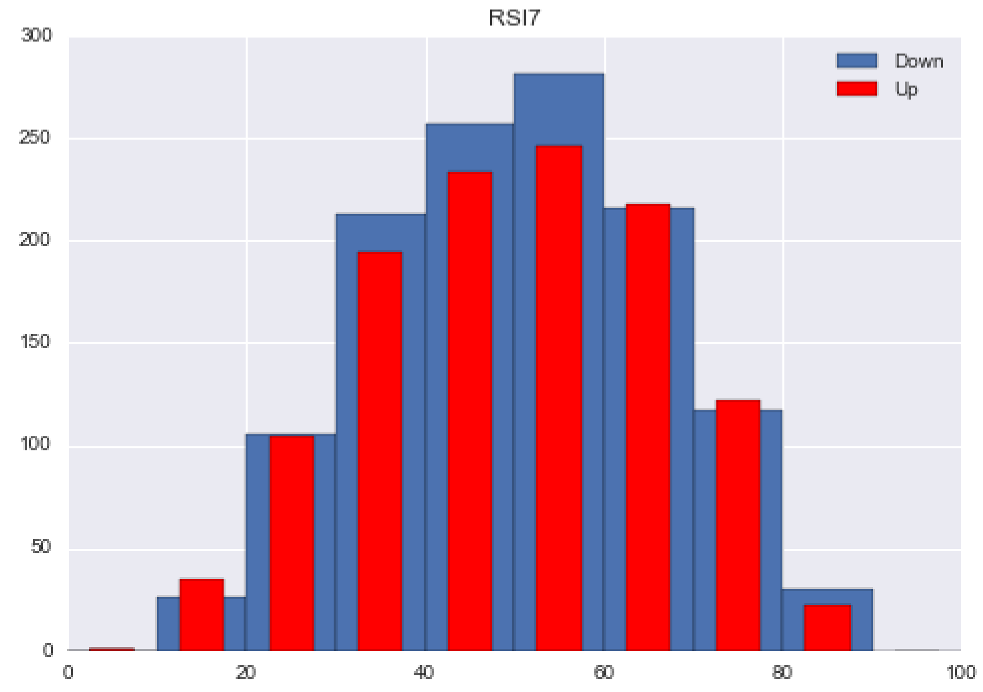

MA8
Down
Up
600
500
400
300
200
100
0
0.6
0.7
0.8
0.9
1.0
1.1
1.2
1.3
1.4

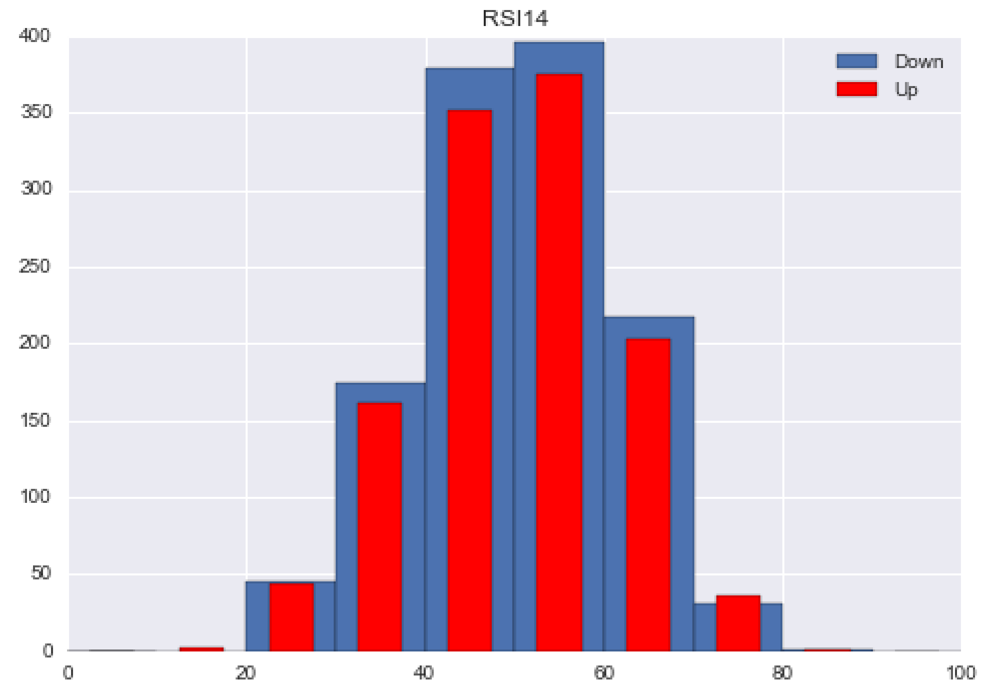

MA21
Down
Up
0
100
200
300
400
500
600
700
0.8
0.9
1.0
1.1
1.2
1.3
1.4
1.5


MA50
Down
Up
0
100
200
300
400
500
600
700
0.8
1.0
1.2
1.4
1.6
1.8


RSI2
Down
Up
180
160
140
120
100
80
60
40
20
0
0
20
40
60
80
100


RSI5
Down
Up
250
200
150
100
50
0
0
20
40
60
80
100



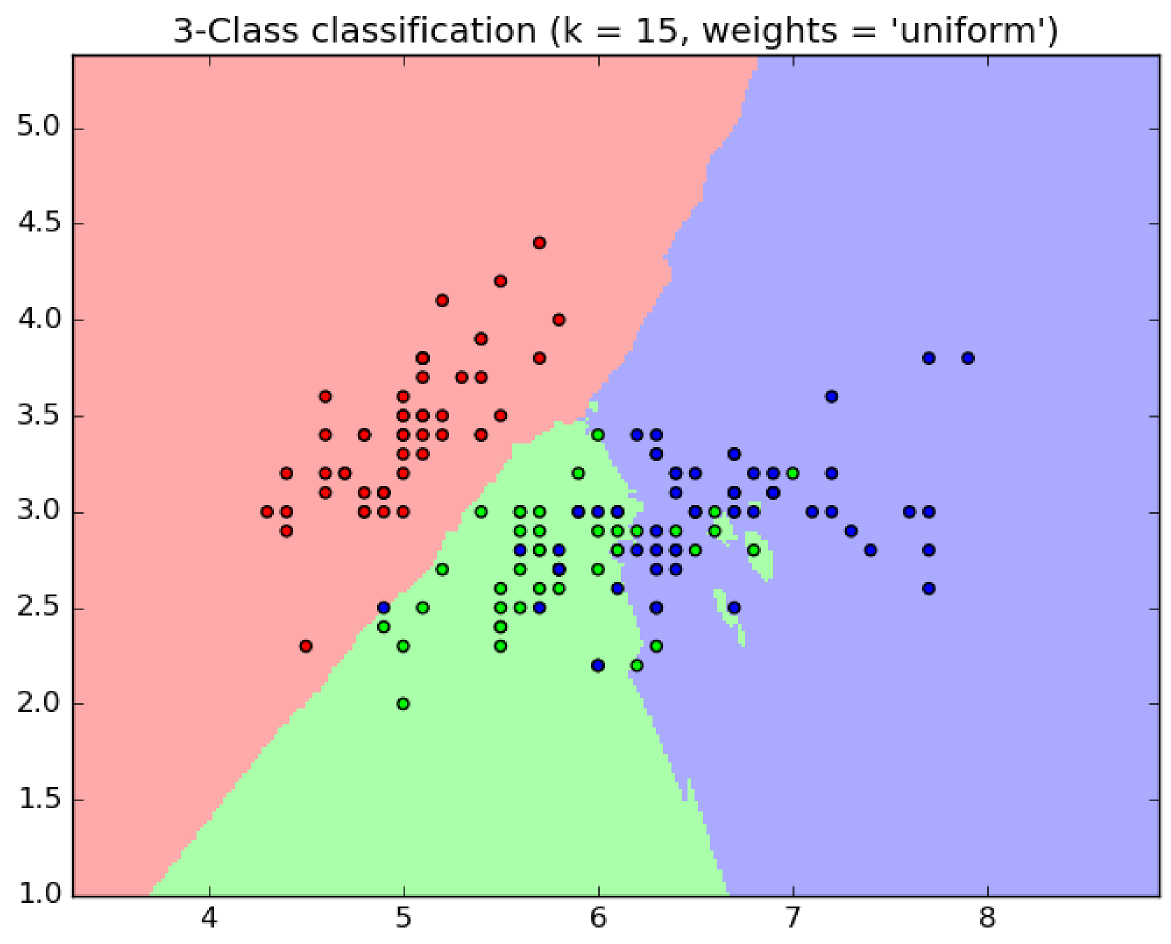


This analysis helped to notice there is some decision tree possibilities to create the model. The algorithms used followed the sklearn map and without this analysis I would not try decision try that was the best model.

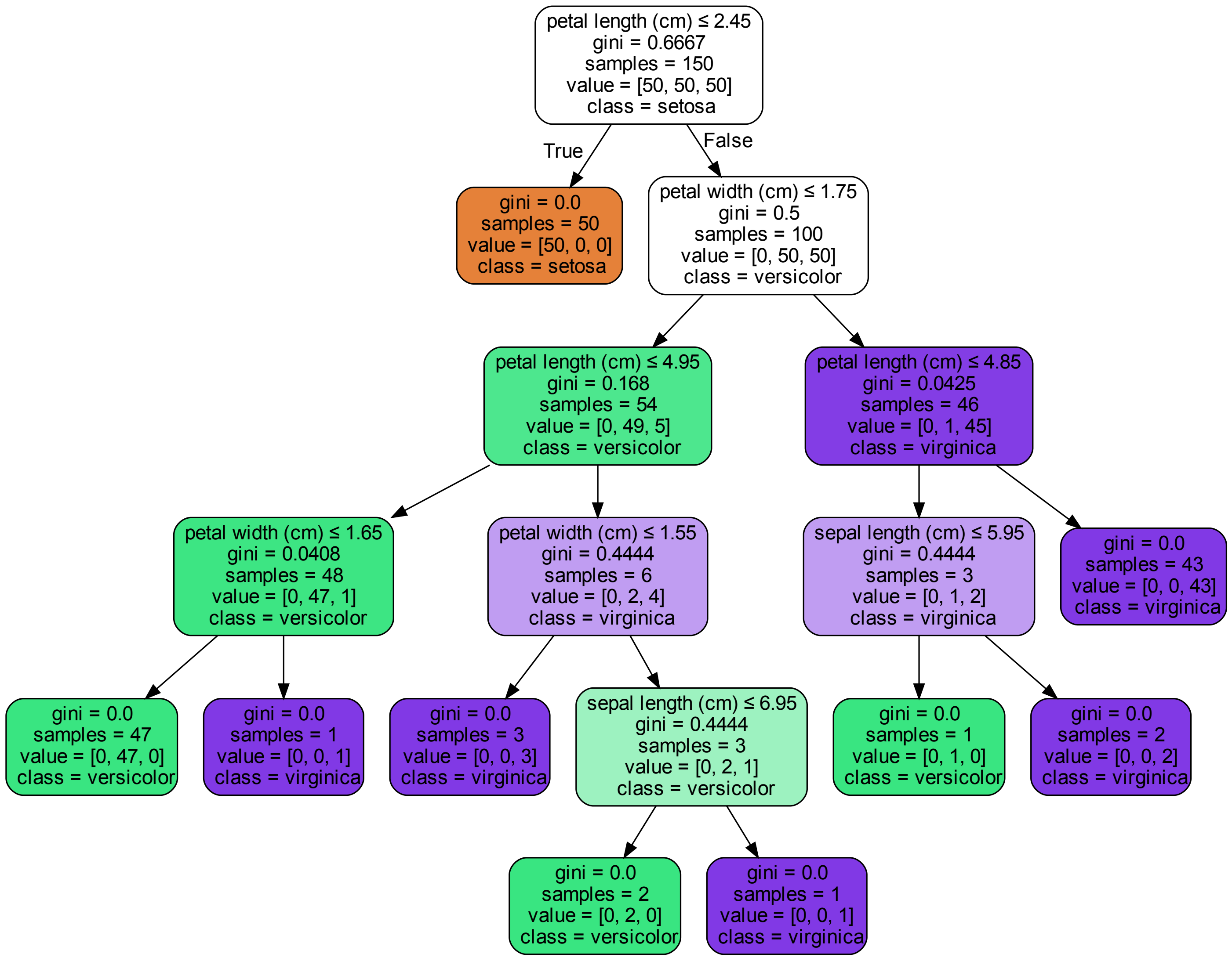
Algorithms and Techniques

Three algorithms were used to model the data: K-Nearest Neighbors, Decision tree and Random Forest (I also tested SVC but it was taking too long to run and AdaBoost but it didn't work well and the last reviewer complain about the short description so I removed it.). For all of them we used some the GridSearch method to improve the parameters. There is a brief presentation of each algorithm below:

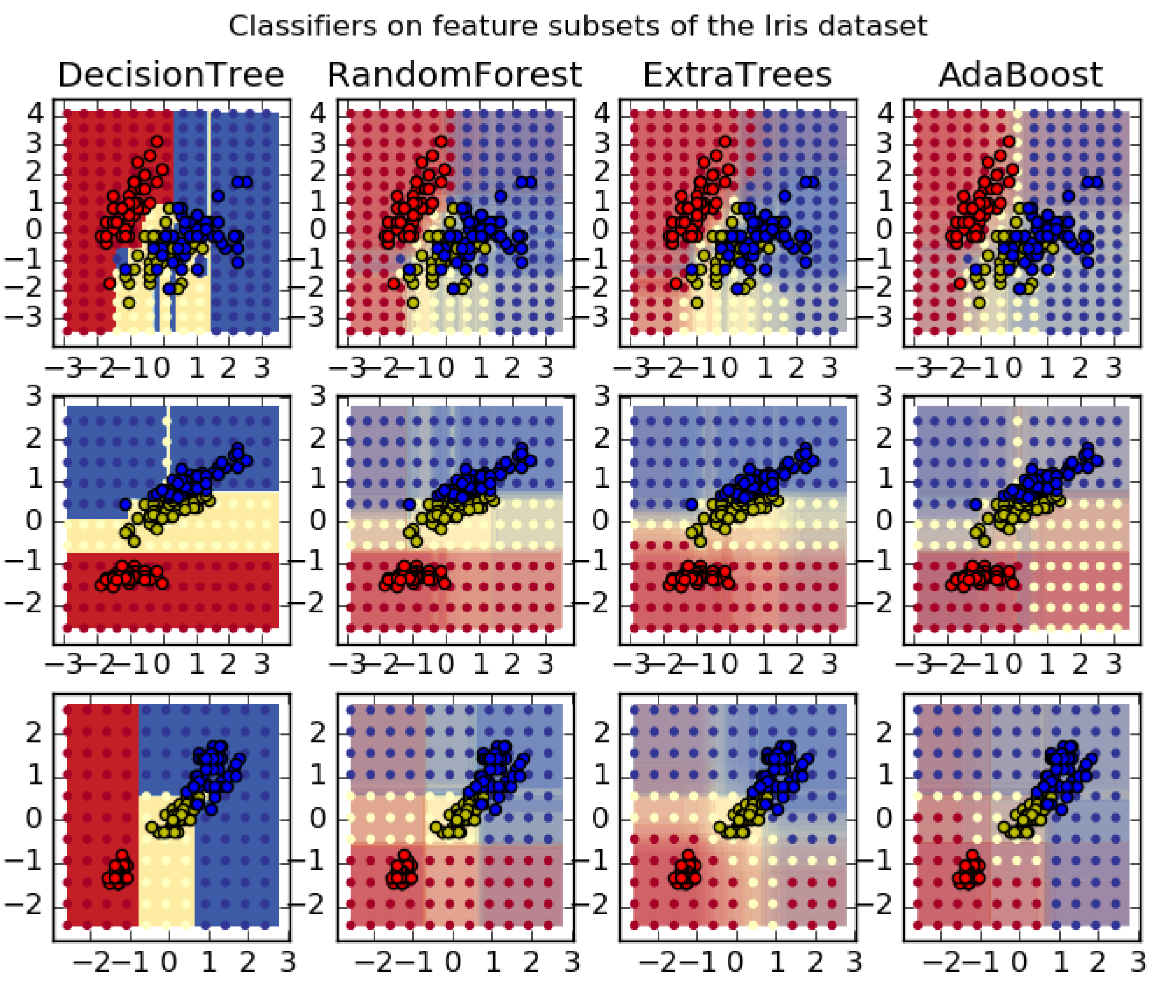
K-Nearest Neighbors: The model is based on check a certain number (K) of nearest neighbors of the point. With the neighbors it considered the point to predict is the same of the K nearest ones. This is the way it creates its predictions.



Decision tree: The decision tree creates several branches and conditions to predict the outcome of a certain model. Using the training data it creates the tree to decide the predicted values.



Random Forest: The random forest is a perturb-and-combine techniques specifically designed for trees. It creates several decision trees considering some randomness and part of the data, so it can perform the predictions in a more precise way. It is really good when you have a lot of data.



Benchmark

We can use two ways to predict a good model.

First, we need to be better than random. The chance of choosing a day that the stock goes up is slightly less than 50%. Any improvement in this performance is positive.

Second, if we buy stocks in the opening of the day and sell in the close we will have positive results. As mentioned before we do not expect to use the model to buy and sell stocks purely. The idea is to use a intraday model in combination with this one but anyway it is a good test.

III. Methodology

Data Preprocessing

From the original data, the days of each negotiation were transformed in days of the week.

The past prices became MAs, RSIs and prices for the last 15 days. All the prices were also divided by the last price, so it became a function of the last price, becoming numbers around 1. Without it we would not have the prices normalized and it would influence the prediction.

The past volumes became the last 5 volumes also divided by the last volume, so we have number around 1. Also normalizing volumes.

We also used a function that selected the K best features to send to the data. The idea here is to decrease the number of features in case the high number causes overfitting.

One last thing is important to be mentioned. We created a second target that was days that the stock closed more than 1% up, instead of only closing up. The reason for it is explained in the next chapter.

Implementation

As previously mentioned the code is documented in the Jupyter notebook with explanation but here we will mention the most important things.

The prediction of the model was really bad until we changed the target function. As mentioned in the Metrics chapter the objective of the model is to improve the chance of the market to move up and not to be always right. Since the models try to be always right we had to change a little the target. The target become to know when the stock would move 1% up. Using this target we were able to increase the chance of predicting a up in the price.

To separate the data between train and test data we had to use the 1% up in the stratified data, in this case we had a good distribution of the days that moved more than 1% up.

For all the algorithms we defined a GridSearchCV with the default crossvalidation of 3-fold. Since the data was already shuffled from the train*test*split we didn't need to worry to shuffle it.

In the beginning we also tested the SVC but it was taking too long to run and it became impossible to use.

It worth mentioning that we used Pipelines and it can be seen in the Jupyter notebook.

Refinement

All change in data done to improve the model were already mentioned previously.

The Gridsearch was used since the beginning in all models so there is not much refinement here.

The Pipelines were implemented later to be able to run the code faster and in a way that run all the possibilities. Before the pipelines the performance of the model was similar to random but being able to run more models faster it is possible to get several results and get the best one.

IV. Results

Model Evaluation and Validation

The chosen model is the Decision Tree.

|Method|Test precision| |------|-------------| |Decision Tree|62%| |K Nearest Nighbors|33%| |Random Forest|54%|

In the test data around 600 days, the decision tree has 26 trades. Decision tree trades 4% of the days what is low but it is a lot better than the ensemble models.

In the training data the decision tree was able to reach a performance of 82%.

In the test data the decision tree had a 62% in the precision score.

We had the idea of preparing a learning curve and a validation curve but since we are training in the days that the market moves 1% up and checking with the days that the market moves up we were not able to perform it. We understand the importance of the curves but it would not be reliable here.

Imagining the learning curves, it is possible that it would be clear that we are still suffering a little bit from variance and more data would improve our model. We can in the future prepare the model to be used in more stocks or get more years so the model improves the performance.

Justification

In comparison with the benchmark the performance of the model is good.

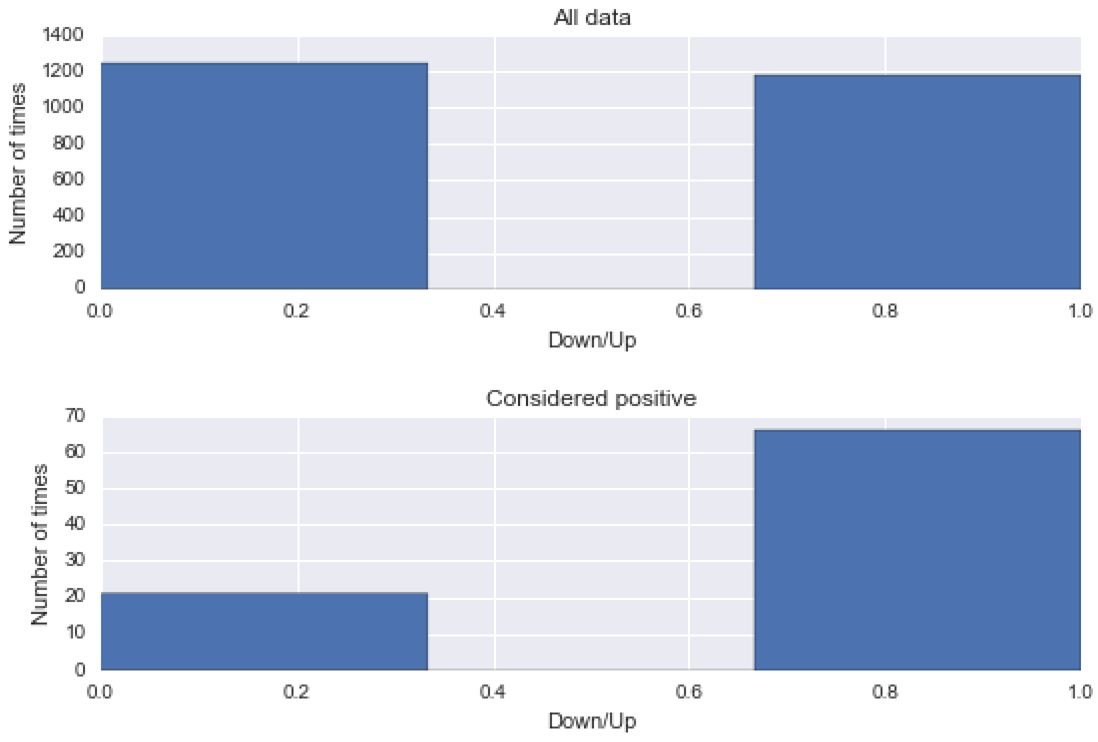
By random the chance of picking a day that would move up is less than 50% and the model got a performance of 62%.

If we used our model to buy one stock in the opening of the market and sell it in the close, we would get out of the market in this test data with a profit of 0.04. It is positive but it is not good and we can consider as 0. But as mentioned before the idea is to improve the performance of other intraday methods and not having a negative result is already good start to improve the other trades.

V. Conclusion

Free-Form Visualization

The main image to show the performance of the model is the total data prediction if it is a positive day or not. The image is shown below and it is the entire data with negative and positive days and the entire data only with the days expected to be positive.



We can clearly see a difference in the probabilities. In here it is showing the training data also what is not perfect but it is a good visualization.

Reflection

The capstone was really good to do. It stretched my understanding about Machine Learning and it showed me it is not easy. Getting a real world problem brings a totally new challenge. Using pipelines brings a total new possibility to test the models and improve the performance of the predictions.

The Machine Learning here was important to get a good solution. The project was harder than expected and the performance was a little below than the desired. Too little trades and/or low precision.

It is interesting to see how hard is to predict the market and its future.

We also tested the problem trying to run as a regression but we were not successful. I didn't test too much but it is one of the branches in the Github project.

I also tested PCA and ICA but it didn't work well. I had it explained here but last reviewer asked to remove it. I really think it should be mentioned but he complained about it.

Improvement

As already mentioned the model could possibly be improve by including more stocks in the trading and getting more historical data.

With more data the ensemble methods and maybe the neural networks could be a possibility for the project. In the case of the neural networks I would need to study more the methodologies, I read a little thinking of implementing them but a have seen they are really useful in Big Data and we didn't have this much data, it could easily overfit as the ensembles.