Project: Stock Predictor Based Correlation

Student: Itay Avisar

Abstract

In this project, we propose a system to predict a daily gain or loss of a stock. The predictor architecture based on algorithm we developed using Artificial Intelligence Neural Network (ANN). The network features will combined finance analytics features with signal processing technics. The main topic in this project is how other stocks are influence the value of another one. We will use cross-correlation technic between various stocks' features, and estimate the impact of a feature using feature selection.

The question we investigate in the framework is:

- **How does the cross-correlations (CC) between stocks may be a reliable indicator for daily gain/loss of the close price?**

At the end of the project, we will choose the parameters, which obtain the best results. Along all the research, we will compare our results with a trivial predicator regarding our best and worst results.

Table of content

1 Introduction

The stock market can make profit of millions to one who can predict wisely the market. On the other hand, the market can be surprising and in the same way cause for many big loose.

A stock price derivate from the company worth and influenced by the many exchanges in the stock market. A various finance indicators may assist to predict in how manner the stock is desirable and so about its daily gain.

In this project, we would like to inspect some **indicators across companies**. We assume that the stocks of competitive companies may behave in opposite correlation, while stocks of co-operative companies will correlate well. In the opposite of this assumption, stocks may correlate by influence from the whole market gain/loss.

We can see bellow, stocks of QCOM and INTC, which can be count as competitive firms, but still have resemblance in the general orientation.

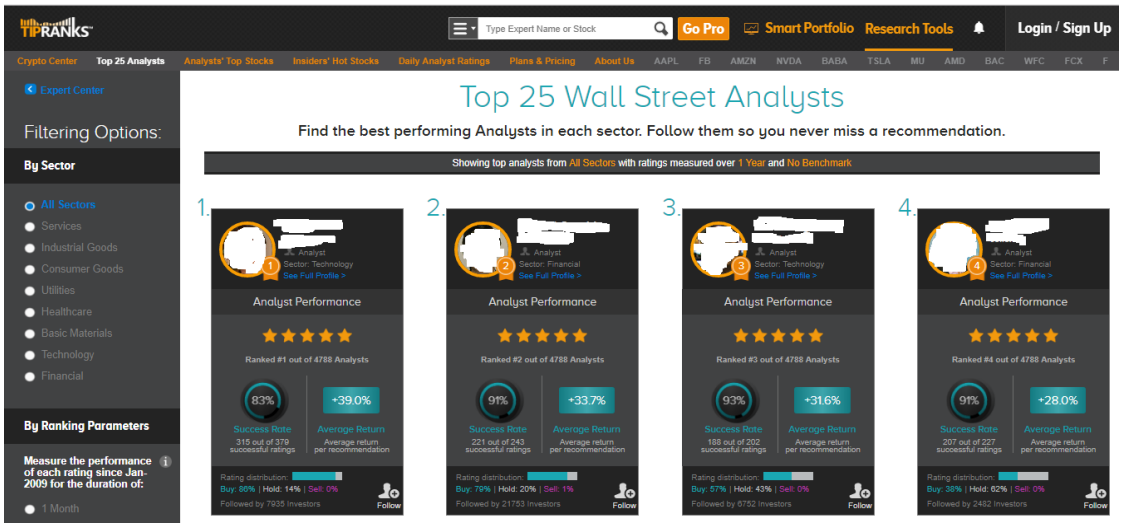
Figure 1.1:

https://finance.yahoo.com/quote/QCOM?p=QCOM

https://finance.yahoo.com/quote/INTC?p=INTC

There are many stock analysts in the market, which traders lean on when they trades. Some of them have strong reliability and some less. Therefore, a trader may take in to account only the best rank analysts. We will use this idea by ranking our features.

Figure 1.2:



https://www.tipranks.com/analysts/top

In order to examine the stocks impact between each other over time, we use the finite cross-correlation function. This function operates on two signals: (a, b) and defined to be:

$$Cross\_Correlation[k; a, b] = \sum_{i=0}^{n} a[i] \cdot b[i + k]$$

This function helps to make resemblance between two signals with a lag. As much as the function have higher value, mean the two signals are more resemble. The point, k, which obtain the maximum of the function, indicates the lag of the two resemble signals.

In general, $Cross\_Correlation[k; a, b] \neq Cross\_Correlation[k; b, a]$. Signal $'a'$ may resemble most to signal $'b'$ by shifting it with $k_0 > 0$ steps. So we expect that $'b'$ is resemble most to signal $'a'$, by shifting $'b'$ with a lag of $-k_0 < 0$.

In this way, we obtain the formula:

$$Cross\_Correlation[k; a, b] = Cross\_Correlation[-k; b, a].$$

Related works

1) There are many related works, which handle stock prediction with a various methods/classifiers. One we inspired most was by "***buffalo team***" which won 1st place Through the Boston Data Festival at 10th November 2013.

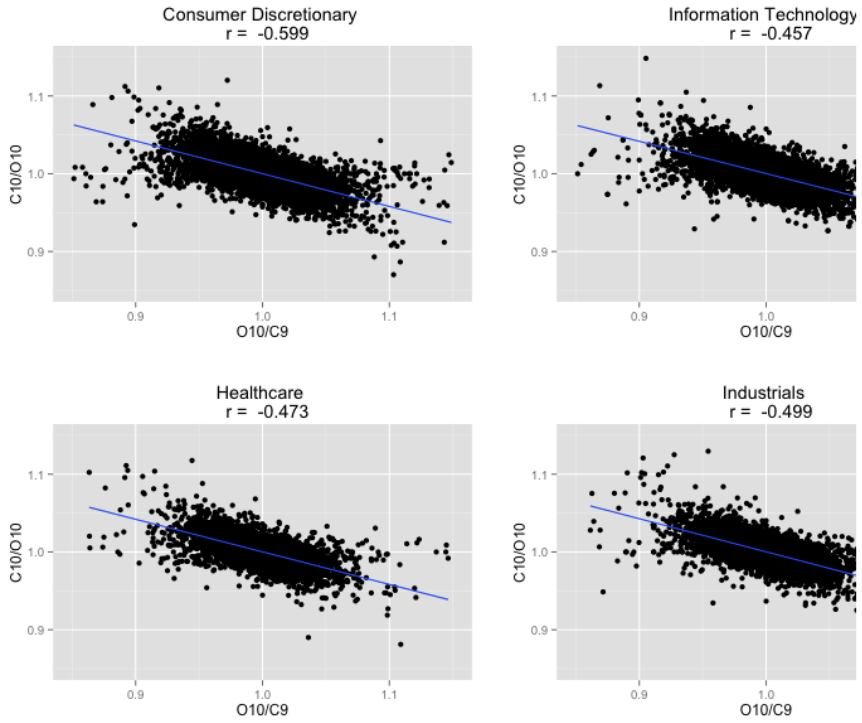
The main question the team handled was:

- How much success can obtained for predicting the directional movement of stock, given historical data for past 9 days and its opening price on the same day.

We impressed from the algorithms they used: Random Forest and Bootstrap-Aggregation. Inspired by these algorithms, we developed our own algorithm, which lean on the same guideline of independent votes.

The results shown by the team was very high accuracy with Area Under Curve (AUC) of ~0.94. However, we were not impressed much because –

a. The data they used was old data from ~1990 and was less relevant for modern data.
b. The results was evaluated only by 1 prediction of 94 stocks
c. We tested some of their assumptions on our own data, and found these assumptions to be irrelevant. (as relation between $\frac{close\ same\ day}{open\ same\ day}$ to $\frac{open\ same\ day}{close\ prev\ day}$)



link –

https://sites.google.com/site/predictingstockmovement/home

2) Another related paper named ***"Stock Prediction Based on Financial Correlation"*** proposed a prediction in the Korea Stock Market. The prediction used genetic-neural-network, and it build a buy/sell strategy, unlikely this project, which predicts a binary value (gain/loss). From this paper, we referenced features for our predictor as the moving-average and some more. Although the title sounds promising, we found this paper less interesting from the first one and concentrated on the first.

2 The Proposed Solution:

2.1 Dataset

Our dataset based on the historical data for each stock and collected from **Yahoo-finance** for about the 10 past years. The historical data contains the following samples for each day:

1. Date – the date of the specific day
2. Open – the open value at the same day (in USD)
3. High – the highest value at the same day (in USD)
4. Low – the Lowest value at the same day (in USD)
5. Close - the close value at the same day (in USD)
6. Volume – the number of the stock trades on the same day

Yahoo finance is one of the most common use and convenient API for prediction projects. In order to obtain this data fast, we use the ***"yahoo_historical"*** python package.

Most of our stocks are been collect from the S&P500. The S&P500 stock itself (symbol ^GSPC) are influenced by the 500 firms that the group combined. Therefore, we expect S&P500 stock to gain or loss simultaneously with some of it combined firms' stocks. These kind of stocks may obtain correlations with no lag (the maximum of the CC located at k=0). These correlations are not be valuable, because our prediction relies on past day's data and not simultaneously. We collected the data from 141 different stocks:

List of symbols –

'AIV','AMAT','ADM','AIZ','T','ADSK','ADP','AN','AZO','AVGO','AVB','AVY',
'BLL','BAC','BK','BXLT','BAX','BBT','BDX','BBBY','BRK.B','BBY','BLX','HRB',
'BA','BWA','BXP','BSX', 'BMY', 'BF.B', 'CHRW', 'CA', 'COG', 'CPB', 'COF', 'CAH','CSCO','GOOGL',
'GOOG', 'HSIC','HP','MU', 'MSFT','MLNX','WYN', 'WYNN', 'XEL', 'XRX',
'XLNX','QCOM','IBM','CERN', 'INTC','^GSPC','AAPL','ABT', 'ABBV', 'ACN', 'ADBE', 'AAP', 'AES',
'AET', 'AFL','AMG', 'ARE', 'APD', 'AA', 'AGN', 'ALXN', 'ALLE', 'ADS', 'ALL','MO', 'AMZN', 'AEE',
'AAL', 'AEP', 'AXP', 'AIG', 'AMT', 'AMP', 'ABC', 'AME','AMGN', 'APH', 'APC', 'ADI', 'AON', 'APA',

'AIV', 'AMAT', 'ADM', 'AIZ', 'T', 'ADSK','ADP', 'AN', 'AZO', 'AVGO', 'AVB', 'AVY', 'BLL', 'BAC', 'BK','BAX', 'BBT', 'BDX', 'BBBY', 'BBY', 'BLX', 'HRB', 'BA', 'BWA', 'BXP', 'BSX','BMY', 'BF.B', 'CHRW', 'CA', 'COG', 'CPB', 'COF', 'CAH', 'CSCO', 'GOOGL', 'GOOG', 'HSIC', 'HP', 'MU','MSFT', 'MLNX', 'WYN', 'WYNN', 'XEL', 'XRX', 'XLNX', 'QCOM', 'IBM', 'CERN', 'INTC', '^GSPC'

With this data, we will generate some known financial indicators (i.e. MA, RSI…) and some correlation interpolation between the signals. We will feed a classifier using this type of features and examine the impact of the interpolated signals features. Each of the signals, which will be feed into the classifier, will be eventually normalized by the formula

$$normalize\big(f(t)\big) = \frac{f(t) - E\big(f(t)\big)}{std\big(f(t)\big)}$$

Our training set generated from the above data at the #X past days, when #X is limited window size as a parameter. Our test set is a single sample, from the day that queries to be predict. We run this prediction over several iterations while keeping the train set and the test set separately and obtain the accuracy from these iterations.

2.2 Cross-Correlation assumptions

Our main tool for choosing a matching stock, which suspicious to give most useful information about the stock we are query, is the cross-correlation between these two signals. We define a signal of a stock as any daily data sequence, which interpolated only by that specific stocks' data. A simple stock signal may be the sequence of the stocks' close prices. More sophisticated stocks signal may be as the percentage change, which can be interpolate by the close and open prices. Further discussion in the features in section 3.4.

In order to obtain a reliable correlation, there is highly importance to choose wisely the cross-correlation parameters. Because we are handling finite signals, taking different sizes of the signals may lead for a different lag between the same signals. , choosing the correlated stock is not trivial, and need to take optimum size of the signal to be correlate.

Handling a finite signals cross-correlation may obtain bias, and causing wrong estimated lag between these two. For better understanding of this impact, we distinguish three technics to calculate cross-correlation for finite signals:

1) **Cross-correlation of the original signal with a limited window –**

This technic calculate the CC with limited steps, using the original whole signal. A big disadvantage of this method is that we obtain **big bias** from the edges of the signal, which may lead to wrong lag between the CC

Example:

$$a = [1,2,3,4,5,6,7,6,5,100]\ , b = [3,4,5,6,7,6,5,100,100,100]$$

Both of the signals have the same length, and it apparent that signal $'b'$ is the same as signal $'a'$ with a lag of 2 steps ($i.e\ \ a[i+2] = b[i]$). If we will take a window size = 4, we will see that for each sub-signal within the original signal, started from index #base, with length=4, the cross-correlation:

$$CC(k) = \sum_{i=\#base}^{\#base+4} a[i+k] \cdot b[i]$$

Let take for our example #base = 3, then

$$CC(0) = [4,5,6,7] \cdot [6,7,6,5] = 107$$

$$CC(1) = [5,6,7,6] \cdot [6,7,6,5] = 144$$

$$CC(2) = [6,7,6,5] \cdot [6,7,6,5] = 146$$

$$CC(3) = [7,6,5,100] \cdot [6,7,6,5] = 614$$

We can see that the edge of the cross-correlation (sample = 100) cause a bias of the lag, and we can think the lag is 3, although the real lag is 2. The problem occurs during high changes and for small signals. The additional value from the edge, caused the CC to obtain significantly higher value.

2) **Cross-correlation window with padding zeros:**

This technic overcome the bias problem by padding zeros all the samples out from the window. For the above example, we can see that:

$$\tilde{a}_4 = [\dots 0,0,4,5,6,7,0,0, \dots]$$

$$\tilde{b}_4 = [\dots 0,0,6,7,6,5,0,0, \dots]$$

$$CC(k) = \sum_{i=0}^{4} \tilde{a}_4[i+k] \cdot \tilde{b}_4[i]$$

$$CC(0) = [4,5,6,7] \cdot [6,7,6,5] = 130$$

$$CC(1) = [5,6,7,0] \cdot [6,7,6,5] = 114$$

$$CC(2) = [6,7,0,0] \cdot [6,7,6,5] = 88$$

$$CC(3) = [7,0,0,0] \cdot [6,7,6,5] = 49$$

The problem with this method is that for small signals, the maximum at most be with lag = 0. This because in this lag, there is less of 0s and more values to multiply and sum. With large signals and well correlates, the zeros may be negligent in compare to the correlation value. We define this phenomenon of the edges as **edge effect.**

3) **Binary-correlation-**

Because we are handling only directional movements of the stock, we can rather check the correlations by a binary signal. This method cutoff the bias effect. It apparent to be efficient for the exact same signals, and less leading wrong on small signals. In the binary signal, we represents '1' as gain of value, and '0' as loss of value.

Example for the above-

$$\tilde{a}_4 = [\dots 0,0,4,5,6,7,0,0, \dots]$$
$$\tilde{b}_4 = [\dots 0,0,6,7,6,5,0,0, \dots]$$
$$\tilde{a}_{4_{bin}} = [1,1,1,1]$$
$$\tilde{b}_{4_{bin}} = [1,0,0,0]$$
$$CC(0) = [1,1,1,1] \cdot [1,0,0,0] = 1$$
$$CC(1) = [1,1,1,0] \cdot [1,0,0,0] = 1$$
$$CC(2) = [1,1,0,0] \cdot [1,0,0,0] = 1$$
$$CC(3) = [1,0,0,0] \cdot [1,0,0,0] = 1$$

Here all the signals have the same peak. The lag of the cross-correlation is not clear, but it does not miss-lead to other lag. For larger signals, this method bring the CC to obtain it maximum on the exact lag.

It seems that the two last technics are acceptable for our problem.

Just bring up to note, there is also cyclic cross-correlation, which is mainly helpful for periodic signals. We assume our signals in the stock market are not periodic and therefore we prevent this technique from discussion. We will examine the correlations values from padding zeros technic, using with/without the binary correlations.

2.3 Accuracy Methodology

In order to evaluate our predictor we need to define accuracy methodology. There are 4 evaluations we were willing to use:

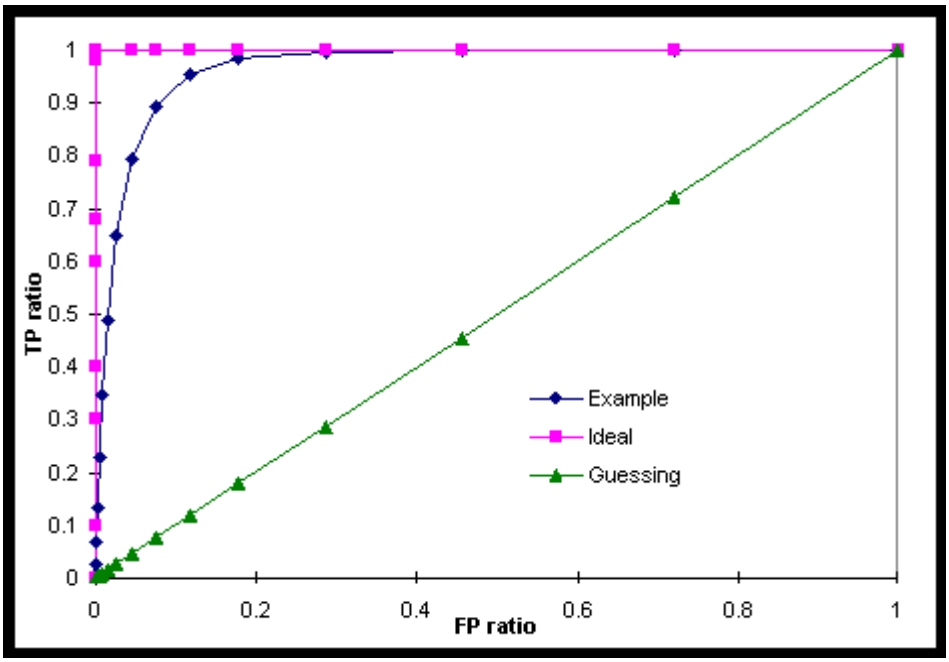
1. Percentage error\error rate:

   The simplest and intuitive accuracy assessment which measured by the formula:

   $$percentage\ error = \frac{\#wrong\ classifies}{\#total\ classifies}$$

   This methodology assume nothing about the distribution of the target. Hence, when we have target which most of the time classifies to be '1', the percentage error will be high even for a trivial predictor which predict constant '1'.

2. Receiving operating characteristic (ROC) curve:
   In this method, the evaluation get better as the data distribution more separated. The curve is a plot of "True Positive Rate" (TPR) over "False Positive Rate" (FPR). The curve is a collection of points over all various thresholds, which distinguish true/false sample. This curve represented by the following graph:



   From this curve one can study, how much the data distribution is separated.

3. Distance of the percentage error from trivial predictor:

An intuitive trivial predictor from the stock market would be to predict as the last target value. Such as if a stock was gain yesterday, probably it would gain as well for today. As an amateur trader, we must say it sounds reasonable. Therefore, we will evaluate our percentage error by the distance from the trivial percentage error.

4. Random prediction:
   A random prediction of a binary vector always have 0.5 error rate.

2.4 Agents

Our propose solution inspired by real world amateur trading strategy- review over professional analysts and lean on their advice. In our project we use "Agent" to represents a professional analyst. This agent eventually vote his opinion about the prediction. The agent will be generate from a specific single stock. By this, we are equivalent the agent impact as the stock impact over the query stock.

2.4.1 How Agent Votes:

The agent's votes based on the lag correspondent to the peaks of the CC values. Eventually the directional values we attend to predict is the close prices. Therefore, naturally CC would be on the close prices signal. Another interesting CC the agent use:

- Percentage change signal using the close signal – big change in one stock may indicate a change in other stock
- Difference between open on the same day vs. the close of the previous day – the movements at the close hours of the market, may imply the orientation of the coming day. Such as a big jump at the close hours implies a gain in the coming day.
- Volume of trading – it is common that big change in volume come with rising of the stock price. Investors, which believe the Agent's stock price will rise, may believe also in another stock, and may invest on the query stock after delay of response time.
- Moving Average(MA) signal using the close signal – the average of a stock over some period may ignore the noise from high frequencies changes. Therefore, the MA may be efficient when we correlates orientations.

The agent works in several modes, which we would like to investigate:

1) Single lag, single CC:
   This is the simplest agent. First the agent calculate the Close signal CC between his represented stock with the query stock. Second, the agent calculate the lag obtained by the peak of the CC function. Finally, the agent will vote as his own stock tag at the same lag, and it CC value to indicate his rank.

2) Single lag, multiply CC:
   The agent will calculate CC for multiply signals-
   a. CC for close prices
   b. CC for the percentage change signal
   c. CC for the volume signal
   d. CC for the MA signal

   Then according to the highest CC, the Agent will calculate the corresponding lag to the peak of the relevant CC. finally the agent will vote the same tag that was at the lag of the correlated stock, and it CC value to indicate his rank.

3) Multiply lag majority votes:
   For each of 1) and 2) methods, after calculating the lag, the agent will vote as the majority of the tags in a window. We take this window to be with size 3. This help to avoid noise in a clearly orientation. In addition, the agent will vote the CC value obtained to indicate his rank.

4) Probability vote:
   For each of the above methods, the agent's binary vote will be multiplication of the binary vote (as for each method) with the CC peak value. Also here the additional vote of the CC value will come as is.

2.4.2 Agent Principles

1) Independence - the agent shall be independent to other stocks, and need to vote only using the data of the:
   a. represented stock he owns,
   b. The input stock.

2) Circumstantial Agent – if the lag obtained from the peak of the CC is negative, so the query stock are influence the agent value instead from the opposite.

3) Informative – the agent lag obtained from the peak of the CC should be greater than 0. Otherwise, the directional movements on both of the stocks are simultaneously and that is not inform any benefit for prediction.

4) Agent reliability- the higher of CC value so the higher of the reliability of the agent. An agent with very low CC represents as an amateur agent. This agent may not take into account for predictions.

5) Big lag wrong tag – if the lag obtained for the agent is too big from some threshold (let say 30 days) then the correlation may not be a circumstance correlation, and might be casual. Therefore, we may want to ignore these agents.

6) Agent effectiveness change over time – as the CC values between stocks changed during time, the agents effectiveness is changes. Hence, on each iteration we should re-generate new agent according to this timestamp.

2.5 Learning Algorithms

As we already said, we rather to implement our system as much as similar to the real trader in market. This is by ranking better or worse analysts as agents, and follow the leading ones. As in the real world, there are sometime miss-lead analysts, which we will call them as "liars". We developed an algorithm for our system, based and inspired by the following learning algorithms:

2.5.1 Random Forest

Random Forest is type of ensemble learning. In this kind of algorithms, there is an independent committee, which votes for the same target. The finally vote will be after consider some function over all the votes. As in most of the cases, our algorithm choose the majority votes. We are not use these committees as decision trees. We use it in two places:

1) Before agent are voting, it generate CC from several signals (in the multiply CC mode). Each of the signals have independent data. Then our committees is the tags according each lag obtain from the corresponding CC signal. The agent finally vote as the majority votes.
2) Having several votes from several agents lead us to choose what is the prediction from all votes. Therefore, the agents' votes are our committees and

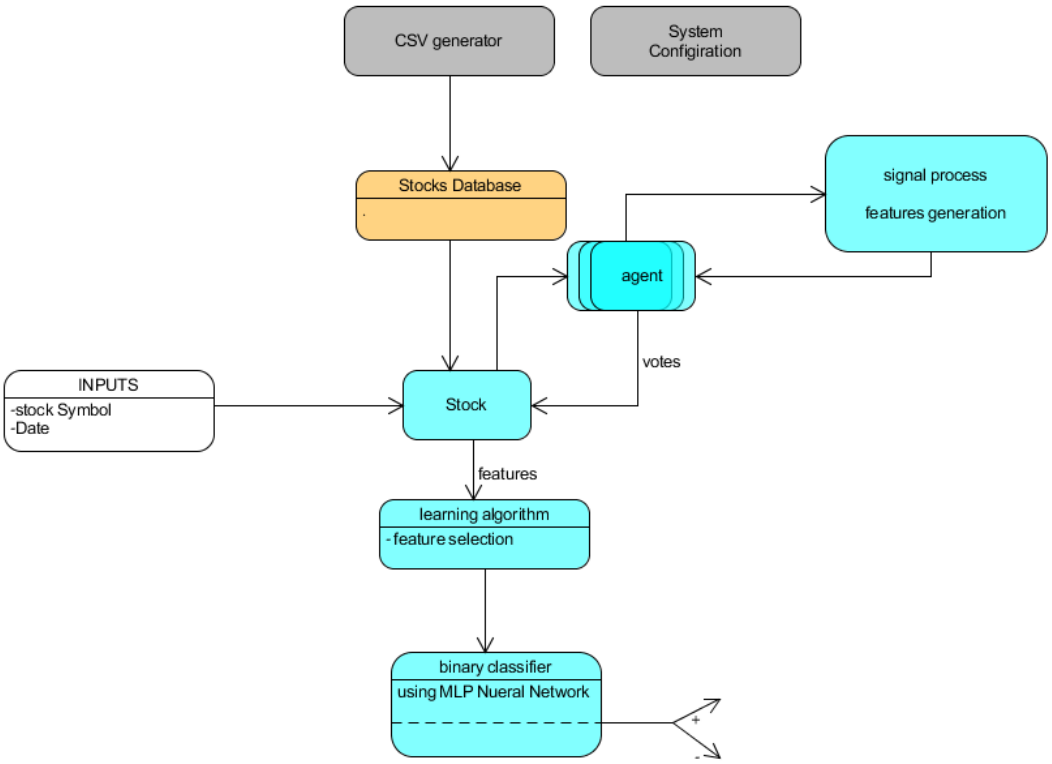
after fed this votes to Artificial Neural Network (ANN) we obtain the output from the votes.

2.5.2 Random Sample Consensuses (RANSAC)

The RANSAC algorithm used to exclude outliers samples from our data. It does it by choosing a random sub-datasets and then evaluate the outputs. The algorithm does it over several iterations and finally choose the best samples. We use this algorithm in our system in order to ignore "liars" agents. This algorithm used as an implementation for feature selection.

3 The Proposed System

Our system is combined by 7 units and additional configuration unit. Each unit have its own inputs/outputs and described separately in section 3.2. The whole system described as:



3.1 Inputs:

- Share's symbol.
- Date: DD/MM/YYYY.

3.2 Outputs:

- +/- Indicator if whether the share will gain or loss at the day after.

3.3 System Units

3.3.1 CSV Generator

Generates CSV files of the historical data from yahoo finance for a list of stocks

Inputs:

- List of symbols to generate their CSVs

Output:

- CSV files of the input symbols

Interactions:

- Stock Database unit reads from this CSVs

3.3.2 Stocks Database

This unit responsible to generate and holding the raw data of all the stocks. The unit reads csv files as it shown in the historical data from yahoo finance. Every access for the absolute raw data will be through this unit.

Inputs:

- Csv files of the historical data of all the relevant stock. Data from yahoo-finance.

Output:

- Enable access to extract the raw data (Close, Open, High, Low, Volume, Date)

Interactions:

- Signal processor and feature generator extract the raw data and generate the relevant signals

3.3.3 Signal Processor and Feature Generator (SPFG)

The unit process the relevant features from the raw data of each stock. It responsible to calculate the CC values, and the financial indicators.

Input:

- Raw signals data to process.

Output:

- Processed signals and CC values.

Interaction:

- Stock – access the signal processor unit in order to obtain the stock signals
- Stock Database – the holder of the data, which the unit use.

3.3.4 Stock

Represents the stock and hold all relevant data only for the specific stock. The relevant data is the raw data in the active window for the prediction.

Inputs:

- Raw data from Stocks Database unit (i.e. Close,Open…).

Outputs:

- Financial features as MA, RSI, GCR,DCR, Momentum ,Change percentage - discussed further in section 3.4 on suggested features

Interaction:

- Stocks database – access to extract its own relevant raw data
- Signal processor – in order to obtain the relevant features from the raw data
- Agent – agent use the stocks data in order to decide what does he votes.

3.3.5 Agent

Represents a specific stock analyst, which eventually decide a vote to predict.

Inputs:

- Stock to represent as an agent
- Stock to query and predict his tag

Outputs:

- Binary vote whether the stock to predict will gain or loss

Interaction:

- Classifier – feed the classifier by its own vote
- Stock – obtain all stock feature for two stocks. The represented stock and the stock to predict.

3.3.6 Binary Classifier

An Artificial Neural Network, which feeds by the agents votes and classify whether if the stock gain or loss. The architecture of the ANN built by the rule of thumb:

$$\#parameters \cdot 10 \sim \#samples \cdot \#features$$

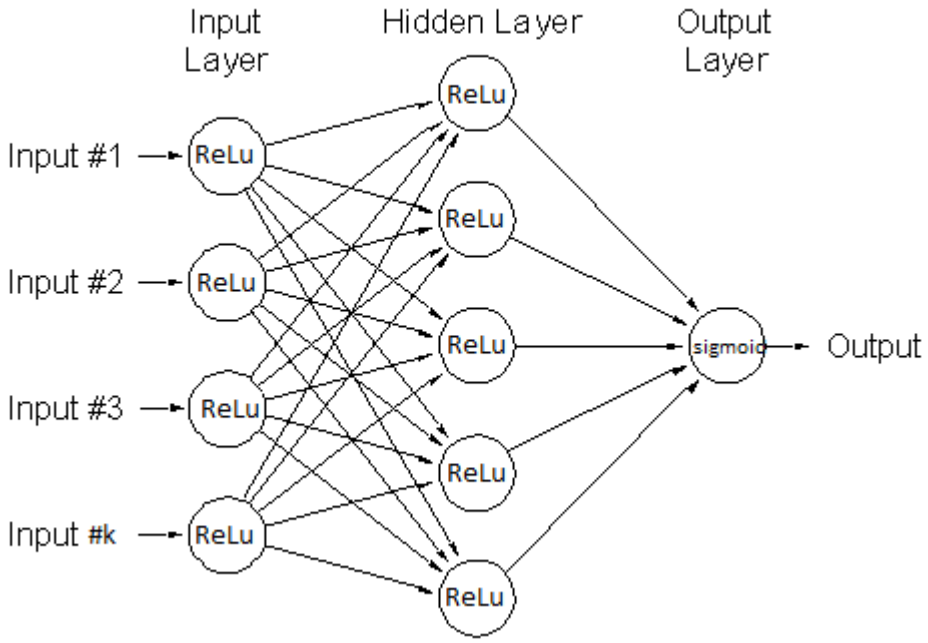
Because the agent reliability changed during time, we may filter different agents' votes during time. This causing the architecture to change together with the features size along the process.

The input layer will have size as the feature vector size. Then we have two hidden layers with

$$\#parameters \ = \ \#InL \cdot \#hidL1 + \#hidL1 \cdot \#hidL2 + \#hidL2 \ \cdot \#outL$$

Where $InL = input\ layer, hidL* = hiddel\ layer*, outL = output\ layer$

Each of the neurons in the input and hidden layers will use ReLu activation. The output layer neuron will use sigmoid activation in order to have convenient conversion to binary value.



We choose our network configurations as most common in a binary classification. The configurations:

1. uniform initializing for each of the parameters
2. reduce the loss function of the binary cross-entropy
3. metrics are binary accuracy
4. optimizer- stochastic gradient decent (SGD)

Inputs:

- Several agents votes for a same query stock

Outputs:

- Binary vote whether the stock to predict will gain or loss

NOTE: this is the output of the whole system

Interaction:

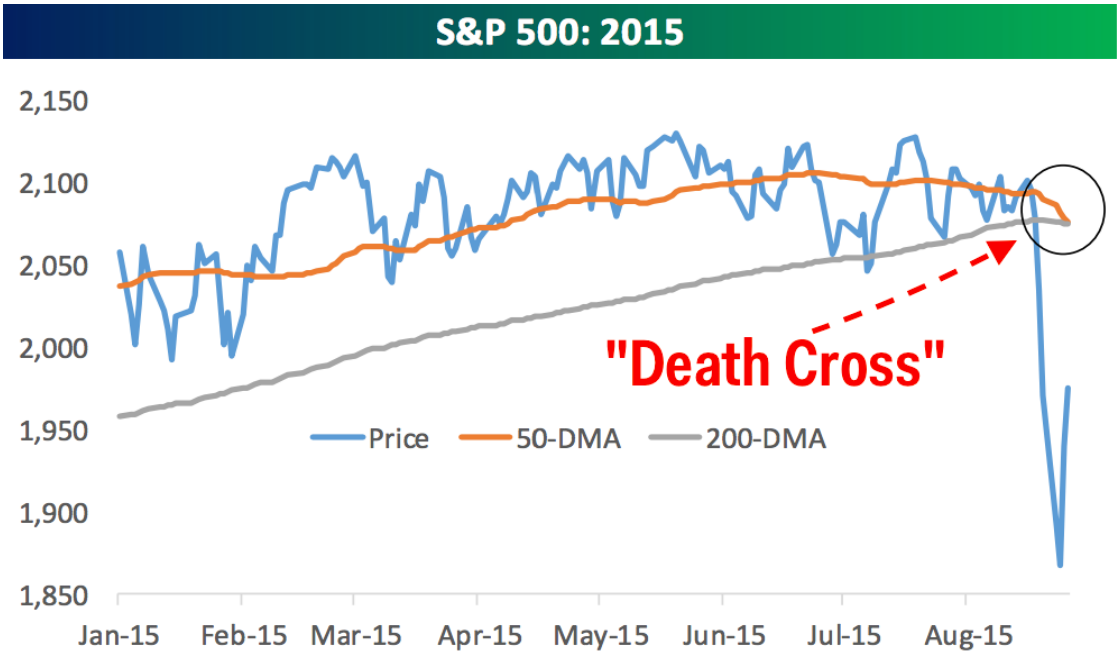
- Agent - the ANN feature is actually all the agents' votes.

3.4 Suggested stock Features:

3.4.1 Financial features:

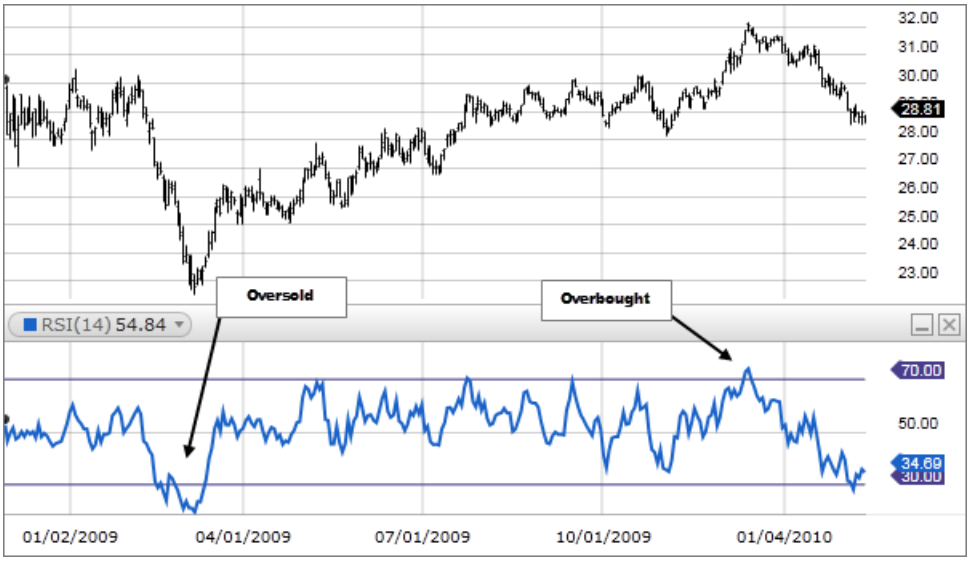
- **Moving Average (MA) –** Average over recent 90 days of the close prices ~ (over recent quarter of the year).
- **Golden-Cross (GCR) –** represents when a short term MA cross **higher** from a long term. We use short term MA to be 14 days and long term MA to be 200. The GCR is a binary indicator, which indicate if happened GCR.
  GCR may strongly indicate about a bull market, which have higher probability to gain.

- **Death-cross (DCR) -** represents when a short term MA cross **lower** from a long term. We use short term MA to be 14 days and long term MA to be 200. The DCR is a binary indicator, which indicate if happened DCR.
  DCR may strongly indicate about bear market, which have higher probability to loss value.



- **Relative strength index (RSI) –** The RSI may indicate whether if a stock is overbought/oversell, which directly imply gain/loss correspondently. It use the average loss and the average gain of the stock in order to calculate a "strength" of the stock. We are using averages over the past 14 days as advised by Welles Wilder, the developer of this indicator. The RSI obtain a value at the scope [0,100]. When the average loss = average gain then the RSI will be exactly 50. A significantly gain over loss will obtain value ~100, and non-gain at all will obtain RSI=0 .The exact formula:

$$RSI(t) = 100 - \frac{1}{1+\frac{avg_{gain}(t)}{avg_{loss}(t)}}.$$



- **Momentum -** periods change of 10 days of the close prices. formula:

$$M = x(t) - x(t - 10)$$

- **Current open VS previous close-** the change between the current open price and the previous day close price.

- **Percentage change (First derivative)-** the percentage change of the close prices. Formula:

$$x'[t] = \frac{x[t] - x[t-1]}{x[t-1]}$$

3.4.2 Attributes as features

- **Close signal** – the close prices signal which need to be predicted
- **Volume signal** - The volume of shares trading.

3.5 Algorithm

1. Step 1- collecting data:
   Collecting the historical data and organize it in a csv files

2. Step 2 – configurations:
   2.1 Date: = the date to predict
   2.2 Current stock: = stock to predict (all data past from Date)
   2.3 Initialize statistic parameters
   2.4 Choose agents modes

3. Step 3 – training:
   3.1 For each date in the past 40 days:
      3.1.1 Tags vector := append to the tags vector the tag of for the current stock at the current date
      3.1.2 for each stock != current stock:
         3.1.2.1 generate agent which represent the stock at the current date
         3.1.2.2 Add the agent vote to vector of features of the current date
      3.1.3 [Optional] Run RANSAC on the set of features with subset of 5
   3.2 Train the binary classifier using the tags vector, and the features vector

d. Step 4 – classify:
   4.1 for each stock != current stock:
      4.1.1 generate agent which represent the stock at the query date
      4.1.2 Add the agent vote to vector of features of the current date
   4.2 feed the binary classifier with the vector of features
   4.3 round the classifier output to a binary '0'/'1' and that is the system prediction

3.6 Scalability

Our system use agents in order to vote according each stock. In particular, we investigate the impact of the CC over stocks and that is the main principle when an agent vote. Therefore, the system is very flexible by changing the agents' configurations and it easy to add more properties to agents such that the agents vote will processed as a user wish. In that way, our system have high scalability and may extend the system using more agents and more properties for each agent.

4 Experiment Methodology

On every experiments, we first use our system trying to predict a same stock with a specific lag. To do so, we copied the stock of AAPL, and shifted it samples by 10 days to generate a new stock name AAPL_10Lag. Then we expected our system to have significant low error rate when predicting AAPL_10 stock using the AAPL stock.

Each experiment tested by:

1. Choose a window of sequential dates that wished to be predicted. Eventually, the evaluation of the predictor will based on these dates.
2. Each of predicted date in the window will be one iteration.

3. Building training set by the data of 50 past days from the date on the current iteration.
4. The test set will be only one sample of the current date.
5. On each iteration, we run the algorithm from section 3.5 to predict a query date.
6. After collecting the prediction errors from all dates in window, we evaluate the performance as described in section 2.3 on accuracy methodology

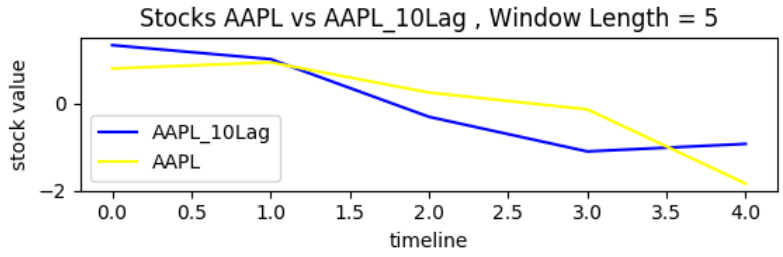
Our experiment will take specific stock, but we are willing to check how does it works on various more stocks.

4.1 Experiments Description

4.1.1 Tuning CC window length

First thing shall be done in our experiments sessions is to find the best CC window length. In order to do so, we examined the CC value and lag between the stock AAPL, and AAPL_10Lag, which it is the same with retard of 10 samples.

Figure 4.1.1.1:



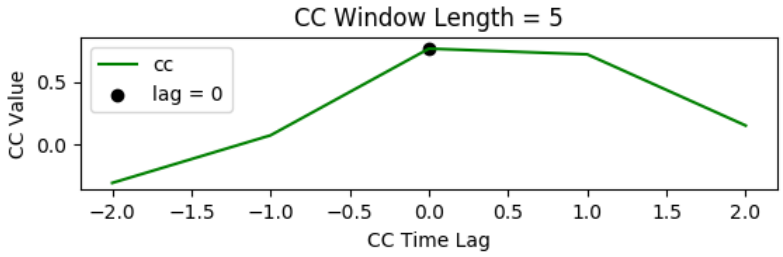


Figure 4.1.1.2:

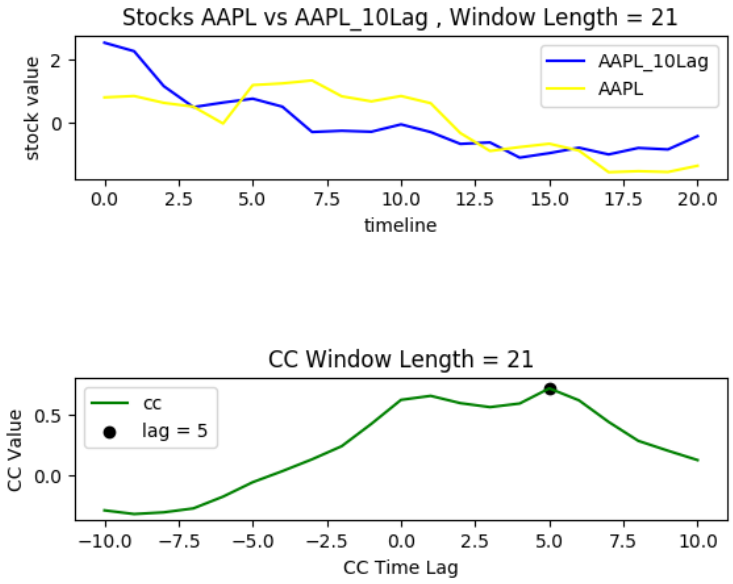


Figure 4.1.1.3:

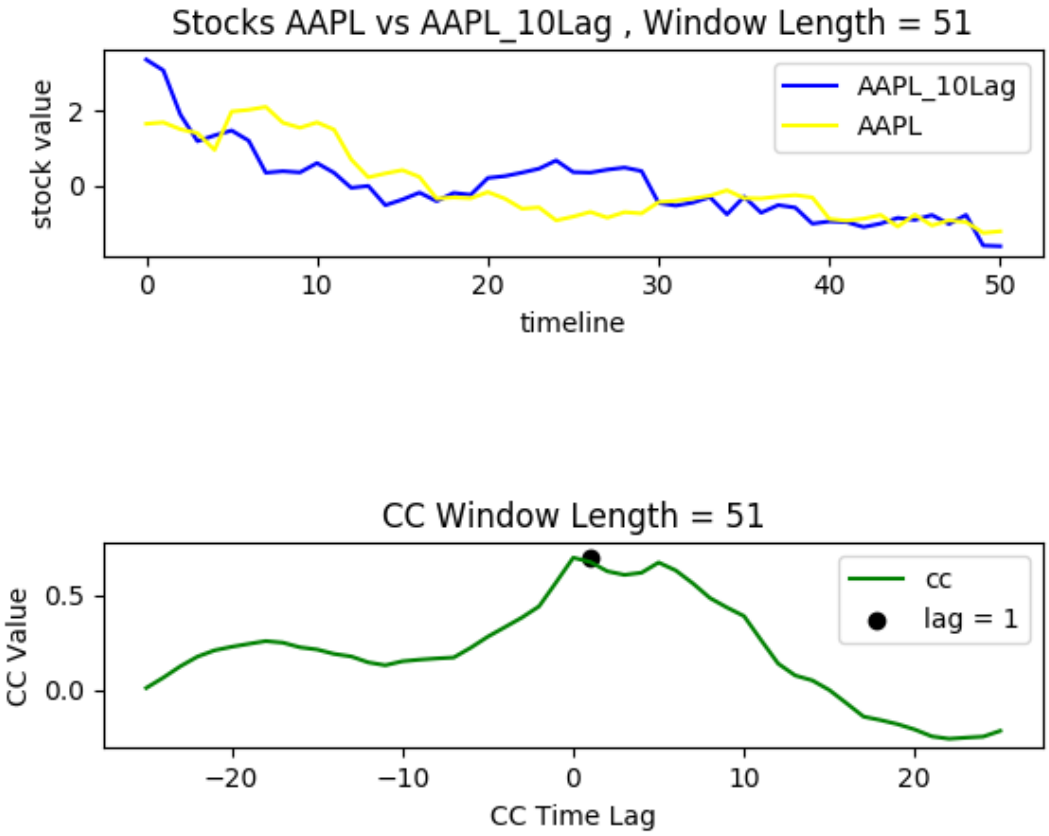
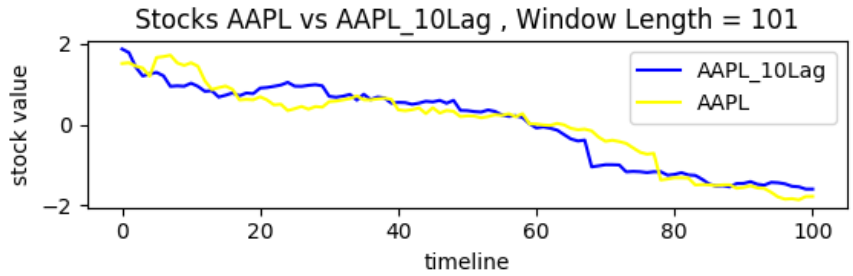
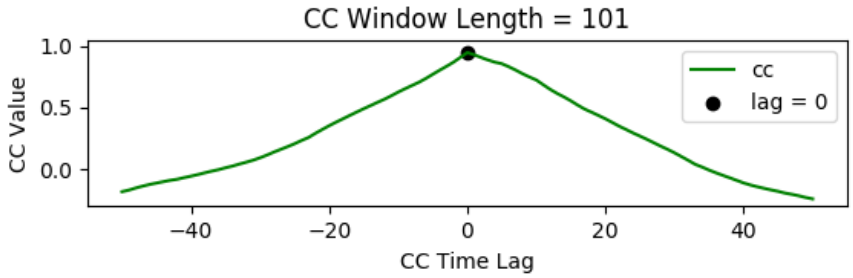


Figure 4.1.1.4:





We can see that with any window length, we could not succeed to get on the real lag.

In figure 4.1.1.1, we can see that the two signals are not seems to be the same. It is obvious, because the length of the signals are less from the lag itself. In that case it is clearly understand that we can't catch the real lag.

In figure 4.1.1.2, the window size is big enough to perceive a correlation, but due to edge effects as discussed on section 2.2, the maximum of the CC value getting closer to no lag (lag=0) from the real lag (lag = 10).

We can see in figures 4.1.1.3 and 4.1.1.4 that as the window length grows so the lag getting closer to no lag. This is happening, because with a longer signals, the edge effect have more impact, and have a bigger tail of zeros which obtain lower CC value.

To overcome this problem we tried to run CC with binary signals of the directional movements:

Figure 3.1.1.5:

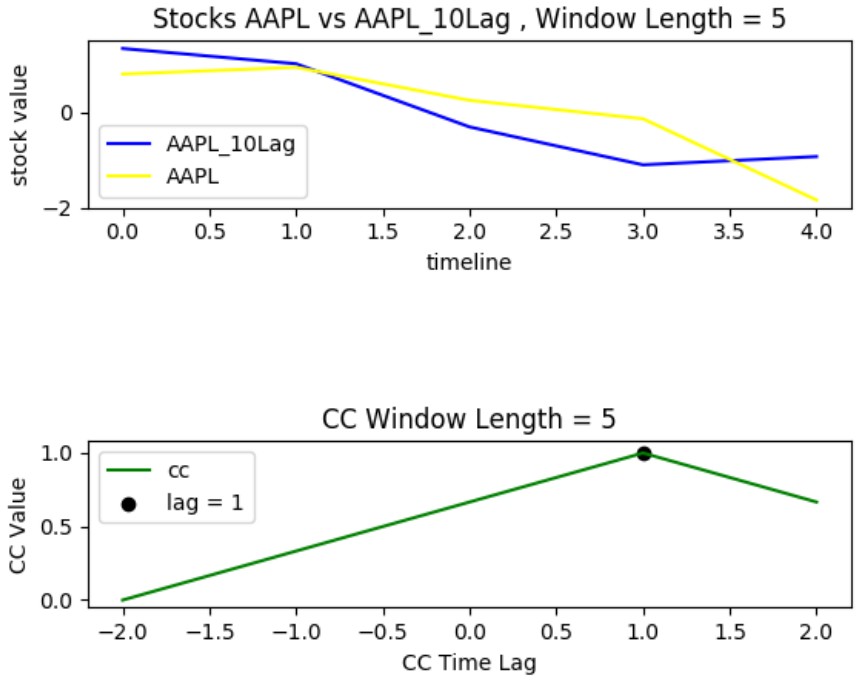


Figure 4.1.1.6:

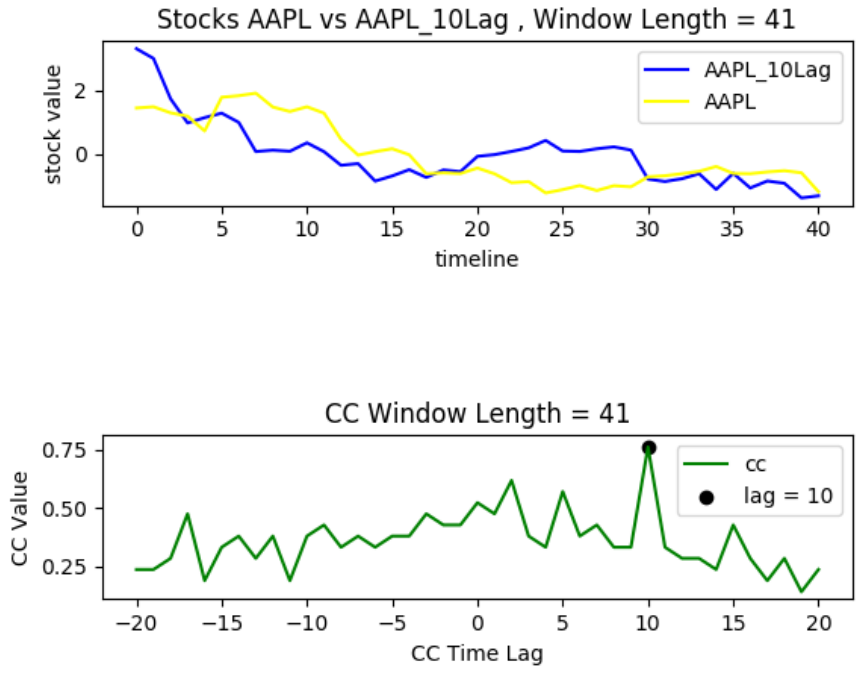
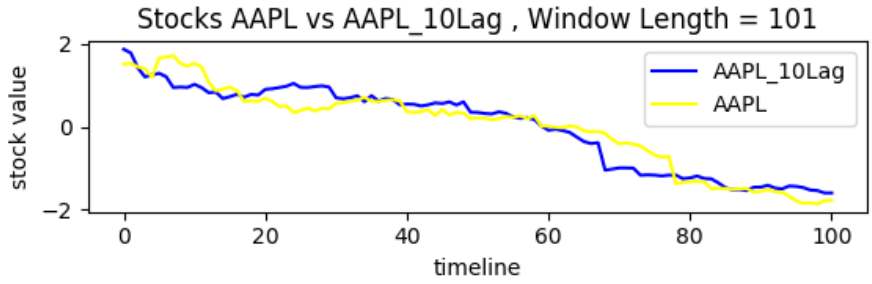
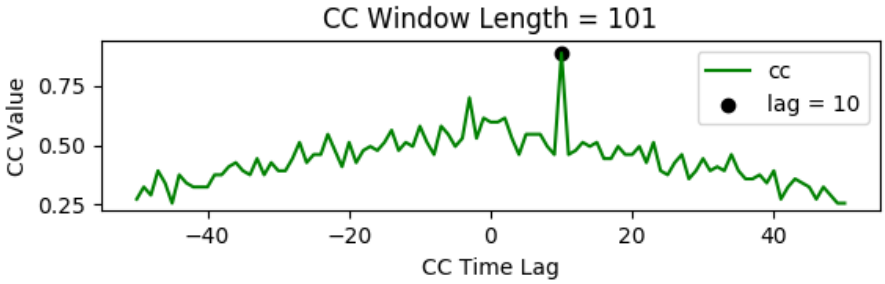


Figure 4.1.1.7





Here we can see much better results. The lag is clearly located by the CC maximum value. In addition, also here it obvious that smaller window length from the real lag, will not perceive the lag. However, as long as the window length is bigger, then we still obtain the real lag. As explained in section 2.2, then binary CC bring the edge effect with lower impact and enable to obtain clearly the real lag.

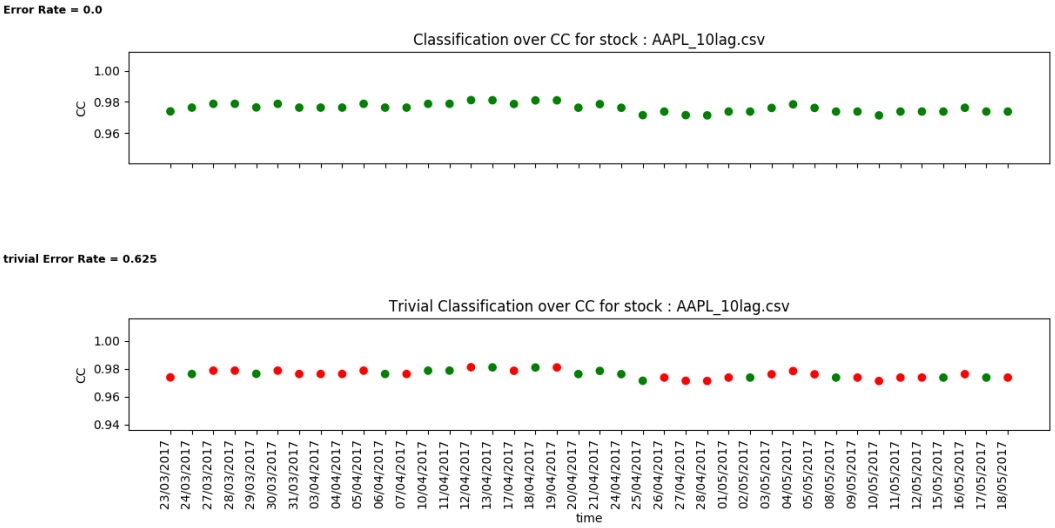
We estimate that stock which comes with recently correlation (smaller lag), are more circumstance. Therefore, we investigate the correlations only with lag smaller from quarter. That mean we are willing to reach for lag that not greater then ~90 samples. For that we are finally choose to make all our CC with window length of 401.

NOTE: we are taking odd numbers in order to shift properly the CC signal by same size in the negative and positive axes.

4.1.2 Predicting same stock with constant lag

Following the previous section, we decide on choosing the CC configurations to be binary CC with window length of 401. The next step is to see whether our predictions of the same stock with a constant lag is successful. Here is the results for predicting AAPL_10Lag stock by AAPL stock:

Figure 4.1.2.1:



We can learn from the green scatters that all predictions was correct and receive no error at all. In addition, we can see that all predictions obtain CC value grater then 0.96.

This results uses feature mode to be single best agent, which is obvious was the agent represent the AAPL stock. In this mode, we are feeding the features only by the best agent vote. Now we are ready to check our performance with other stocks in the market.

4.1.3 Close Prices Cross-Correlation Table

In order to investigate well the correlations impact on the prediction we should first investigate a stock that comes well correlated with others. Therefore, we build Cross-Correlation table crossing each of the stocks close price. Then we will choose the desirable stock with highest correlated and valid lag with other specific stock. This is because it would be irrelevant to investigate a stock, which have undervalued cross-correlation compared to others.

We generated a table of correlations between each two stock in file 'correlations.csv'. Here is part of the table:

Figure 4.1.3.1:

ADSK.csv	ADS.csv	ADP.csv	ADM.csv	ADI.csv	ADBE.csv	ACN.csv	ABT.csv	ABC.csv	ABBV.csv	AAPL_10lag.csv	AAPL.csv	AAP.csv	AAL.csv	AA.csv	
['0.64', 0]	['0.62', 0]	['0.63', 0]	['0.60', 0]	['0.62', 0]	['0.62', 0]	['0.58', 0]	['0.60', 0]	['0.52', 5]	['0.55', 6]	['0.57', -10]	['0.58', 0]	['0.53', 3]	['0.62', 0]	['1.00', 0]	AA.csv
['0.67', 0]	['0.66', 0]	['0.64', 0]	['0.60', 0]	['0.66', 0]	['0.64', 0]	['0.60', 0]	['0.62', 0]	['0.58', 0]	['0.58', 0]	['0.59', -10]	['0.61', 0]	['0.56', 0]	['1.00', 0]	['0.62', 0]	AAL.csv
['0.57', 0]	['0.59', 0]	['0.64', 0]	['0.59', 0]	['0.58', 0]	['0.58', 0]	['0.61', 0]	['0.59', 0]	['0.52', 0]	['0.55', 3]	['0.54', -10]	['0.56', 0]	['1.00', 0]	['0.56', 0]	['0.53', -3]	AAP.csv
['0.65', 0]	['0.65', 0]	['0.64', 0]	['0.61', 0]	['0.71', 0]	['0.64', 0]	['0.61', 0]	['0.64', 0]	['0.57', 4]	['0.59', 0]	['0.97', -10]	['1.00', 0]	['0.56', 0]	['0.61', 0]	['0.58', 0]	AAPL.csv
['0.63', 10]	['0.63', 10]	['0.62', 10]	['0.60', 10]	['0.69', 10]	['0.62', 10]	['0.60', 10]	['0.62', 10]	['0.57', 0]	['0.58', 10]	['1.00', 0]	['0.97', 10]	['0.54', 10]	['0.59', 10]	['0.57', 10]	AAPL_10lag.csv
['0.66', 0]	['0.62', 0]	['0.61', 0]	['0.61', 0]	['0.60', 0]	['0.63', 0]	['0.62', 0]	['0.71', 0]	['0.66', 0]	['1.00', 0]	['0.58', -10]	['0.59', 0]	['0.55', -3]	['0.58', 0]	['0.55', -6]	ABBV.csv
['0.62', 0]	['0.62', 0]	['0.61', 0]	['0.56', 0]	['0.59', 0]	['0.61', 0]	['0.59', 0]	['0.63', 0]	['1.00', 0]	['0.66', 0]	['0.57', 0]	['0.57', -4]	['0.52', 0]	['0.58', 0]	['0.52', 11]	ABC.csv
['0.66', 0]	['0.68', 0]	['0.68', 0]	['0.58', 0]	['0.67', 0]	['0.68', 0]	['0.68', 0]	['1.00', 0]	['0.63', 0]	['0.71', 0]	['0.62', -10]	['0.64', 0]	['0.59', 0]	['0.62', 0]	['0.60', 0]	ABT.csv
['0.68', 0]	['0.65', 0]	['0.75', 0]	['0.62', 0]	['0.69', 0]	['0.69', 0]	['1.00', 0]	['0.68', 0]	['0.59', 0]	['0.62', 0]	['0.60', -10]	['0.61', 0]	['0.61', 0]	['0.60', 0]	['0.58', 0]	ACN.csv
['0.74', 0]	['0.71', 0]	['0.73', 0]	['0.61', 0]	['0.69', 0]	['1.00', 0]	['0.69', 0]	['0.68', 0]	['0.61', 0]	['0.63', 0]	['0.62', -10]	['0.64', 0]	['0.58', 0]	['0.64', 0]	['0.62', 0]	ADBE.csv
['0.71', 0]	['0.70', 0]	['0.69', 0]	['0.62', 0]	['1.00', 0]	['0.69', 0]	['0.69', 0]	['0.67', 0]	['0.59', 0]	['0.60', 0]	['0.69', -10]	['0.71', 0]	['0.58', 0]	['0.66', 0]	['0.62', 0]	ADI.csv
['0.65', 0]	['0.67', 0]	['0.62', 0]	['1.00', 0]	['0.62', 0]	['0.61', 0]	['0.62', 0]	['0.58', 0]	['0.56', 0]	['0.61', 0]	['0.60', -10]	['0.61', 0]	['0.59', 0]	['0.60', 0]	['0.60', 0]	ADM.csv
['0.67', 0]	['0.70', 0]	['1.00', 0]	['0.62', 0]	['0.69', 0]	['0.73', 0]	['0.75', 0]	['0.68', 0]	['0.61', 0]	['0.61', 0]	['0.62', -10]	['0.64', 0]	['0.64', 0]	['0.64', 0]	['0.63', 0]	ADP.csv
['0.69', 0]	['1.00', 0]	['0.70', 0]	['0.67', 0]	['0.70', 0]	['0.71', 0]	['0.65', 0]	['0.68', 0]	['0.62', 0]	['0.62', 0]	['0.63', -10]	['0.65', 0]	['0.59', 0]	['0.66', 0]	['0.62', 0]	ADS.csv
['1.00', 0]	['0.69', 0]	['0.67', 0]	['0.65', 0]	['0.71', 0]	['0.74', 0]	['0.68', 0]	['0.66', 0]	['0.62', 0]	['0.66', 0]	['0.63', -10]	['0.65', 0]	['0.57', 0]	['0.67', 0]	['0.64', 0]	ADSK.csv
['0.59', 10]	['0.58', 2]	['0.61', 0]	['0.60', 0]	['0.57', -9]	['0.62', 0]	['0.63', 0]	['0.59', 0]	['0.56', -10]	['0.56', 0]	['0.57', -8]	['0.57', 2]	['0.57', 0]	['0.56', 9]	['0.55', -5]	AEE.csv
['0.59', 2]	['0.58', 2]	['0.62', 0]	['0.61', 0]	['0.58', 8]	['0.61', 0]	['0.64', 0]	['0.56', 2]	['0.56', 2]	['0.56', 8]	['0.57', -8]	['0.58', -16]	['0.56', 0]	['0.58', 0]	['0.57', -20]	AEP.csv
['0.62', 0]	['0.63', 0]	['0.66', 0]	['0.63', 0]	['0.63', 0]	['0.64', 0]	['0.67', 0]	['0.59', 0]	['0.54', 0]	['0.55', 6]	['0.56', 8]	['0.57', 0]	['0.55', 0]	['0.59', 0]	['0.60', 0]	AES.csv
['0.63', 0]	['0.64', 0]	['0.66', 0]	['0.59', 0]	['0.59', 0]	['0.64', 0]	['0.60', 0]	['0.63', 0]	['0.64', 0]	['0.65', 0]	['0.55', -10]	['0.57', 0]	['0.58', 0]	['0.56', 0]	['0.56', 0]	AET.csv
['0.67', 0]	['0.67', 0]	['0.74', 0]	['0.65', 0]	['0.66', 0]	['0.69', 0]	['0.69', 0]	['0.68', 0]	['0.61', 0]	['0.62', 0]	['0.61', -10]	['0.62', 0]	['0.59', 0]	['0.63', 0]	['0.64', 0]	AFL.csv
['0.60', 0]	['0.62', 0]	['0.61', 0]	['0.56', 0]	['0.56', 0]	['0.61', 0]	['0.59', 0]	['0.63', 0]	['0.65', 0]	['0.64', 0]	['0.55', -5]	['0.55', 5]	['0.54', -2]	['0.57', 0]	['0.51', -3]	AGN.csv
['0.67', 0]	['0.70', 0]	['0.71', 0]	['0.63', 0]	['0.69', 0]	['0.67', 0]	['0.69', 0]	['0.65', 0]	['0.67', 0]	['0.61', 0]	['0.63', -10]	['0.65', 0]	['0.58', 0]	['0.64', 0]	['0.62', 0]	AIG.csv
['0.63', 0]	['0.59', 0]	['0.67', 0]	['0.63', 0]	['0.56', -6]	['0.63', 0]	['0.66', 0]	['0.58', 0]	['0.58', 0]	['0.57', 0]	['0.56', -8]	['0.56', 2]	['0.55', 0]	['0.57', 0]	['0.56', 14]	AIV.csv

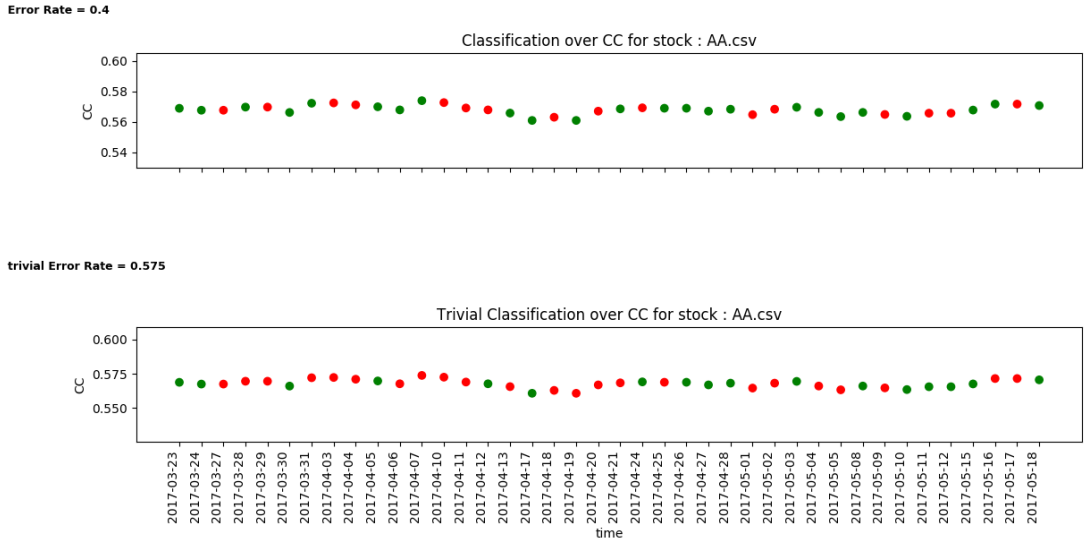
We can see in the table that most of the stocks are correlated with no lag. This is make sense, because most of the time, the stocks response to the whole market. A day with less trading in market causing the market value to loss and that directly affect over all the stocks. Moreover, we can be sure that the "no lag" result not related to the edge effect. That by notice that the AAPL_10Lag stock have constantly lag of -10 samples.

However, there are still stocks, which make as exceptions. This stocks would response for some indicator that stronger from the market value effect, and therefore we investigate if CC is such of this indicator. We can see that most common CC value range is ~ [0.5,0.7].

We decided to investigate stock "AA" due to high correlation with XEL Stock and many more with valid lag.

4.1.4 Single lag, single CC

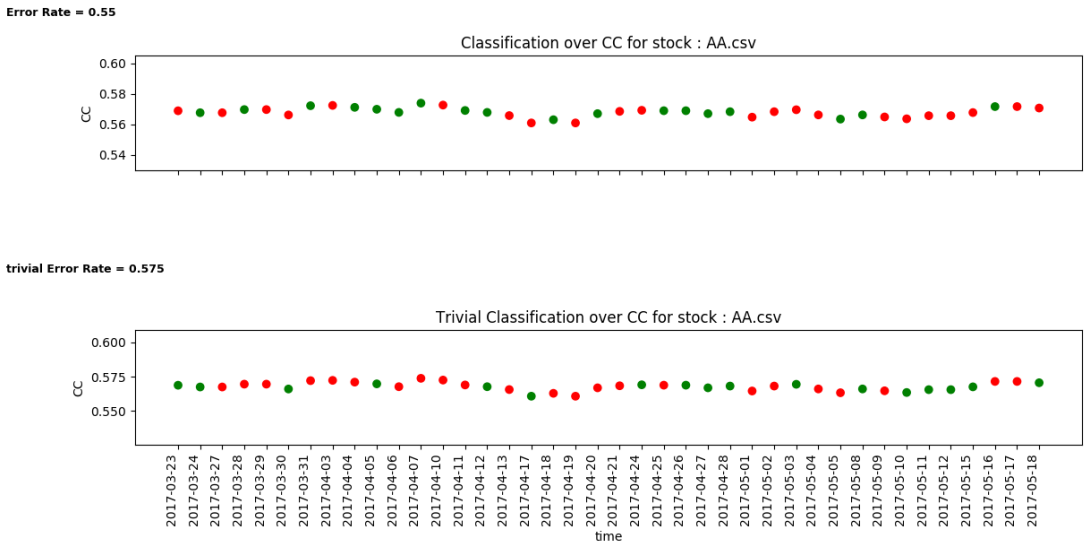
Figure 4.1.4.1:



Her we obtained error rate of 0.4 by predicting AA stock. We can see that the CC values range ~ (0.55, 0.58).
First, we were expect to obtain that predictions with higher CC will be true and lower CC will be false. However, from the scatter, we conclude that predictions with lower CC have more random behavior and therefore there are both true and false predictions. Nevertheless, if we will inspect only predictions that have CC values with higher from 0.568 it is seems to have majority of true predictions.
Another assumption for the errors is that it might be due to noise inserted by multiply irrelevant features. To handle this we will try in the next step to run the classification only by one agent: the best-correlated agent.
In addition, we can see that the trivial predictor obtained percentage error of 0.575, which are pretty higher also for a random prediction.

4.1.5 Single lag, Best Agent (single vote)

Figure 4.1.5.1:

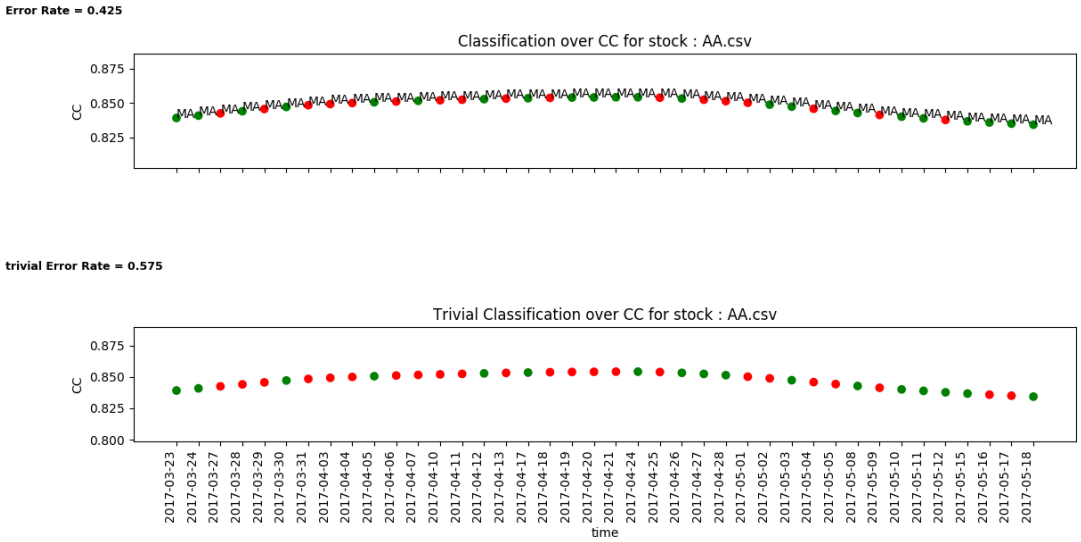


Here we obtained much worse error rate. It can be explained by a lack of features to decision. Although we are feeding the classifier with agents' vote and it CC value, the CC is not high enough for determine a prediction only leaning on this correlated stock past. Therefore, the ANN classifier reach to bad score of loss, and with that score of loss the prediction is almost constant. Because of the lack of features and almost the same input of CC value, the classifier outputs is almost the same. The ANN could not learn anything from the agents' binary vote nor from the CC value.

In the next step we enable agent to choose his vote, not only by the close prices CC, but also with other signals CC. we also decide from now to continue with various agents votes and not only the best agent.

4.1.6 Single Lag, Multiply CC:

Figure 4.1.6.1:



In this method, we can notify that we always choose the MA. This is because we did not use binary vector of the MA directional movement. We used only the normalized vector. Therefore, the agent always see higher CC for such of this signals.

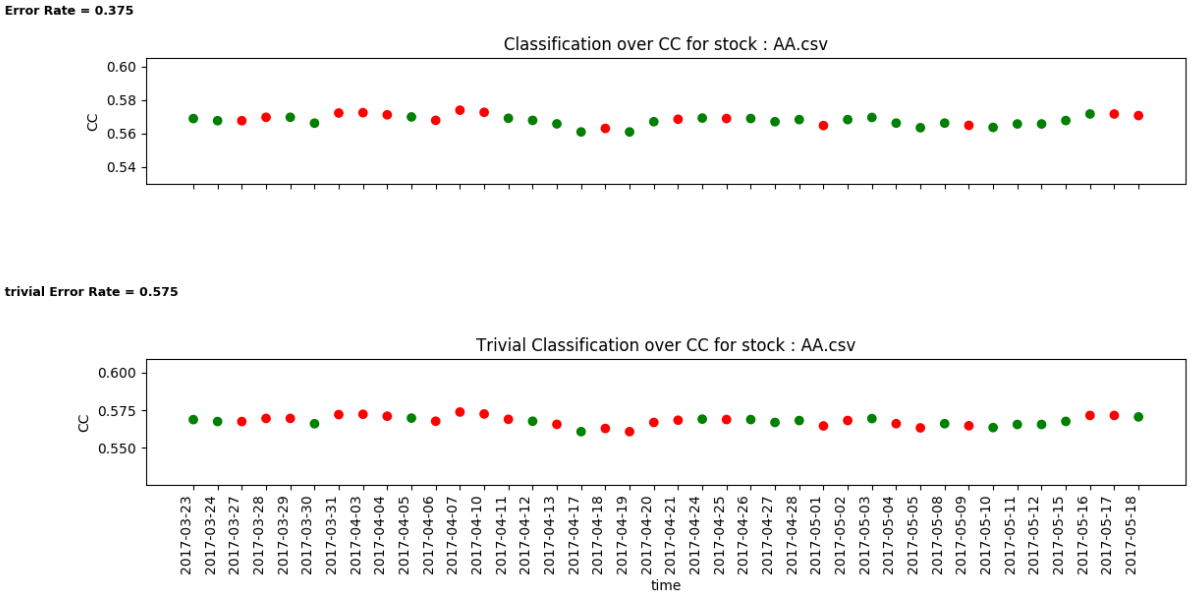
We can see here that the MA signal gain very high CC value, but no predicting much. In addition, the CC rise at the start and then declined with time. That may imply the gain and loss of similarity over time. Therefore, the MA as it is, may be bad indicator. We can see that at the top of the CC values, we obtain more successful predictions then the lower CC.

Although we still achieve more efficient predictor then our trivial one, our prediction is more close to random predictor, which always have the probability of 0.5 success.

4.1.7 Multiply Lag Majority Votes:

In this experiment in each lag achieved, we choose the majority tag in a window size of three samples, centered by the lag distance.

Figure 4.1.7.1:

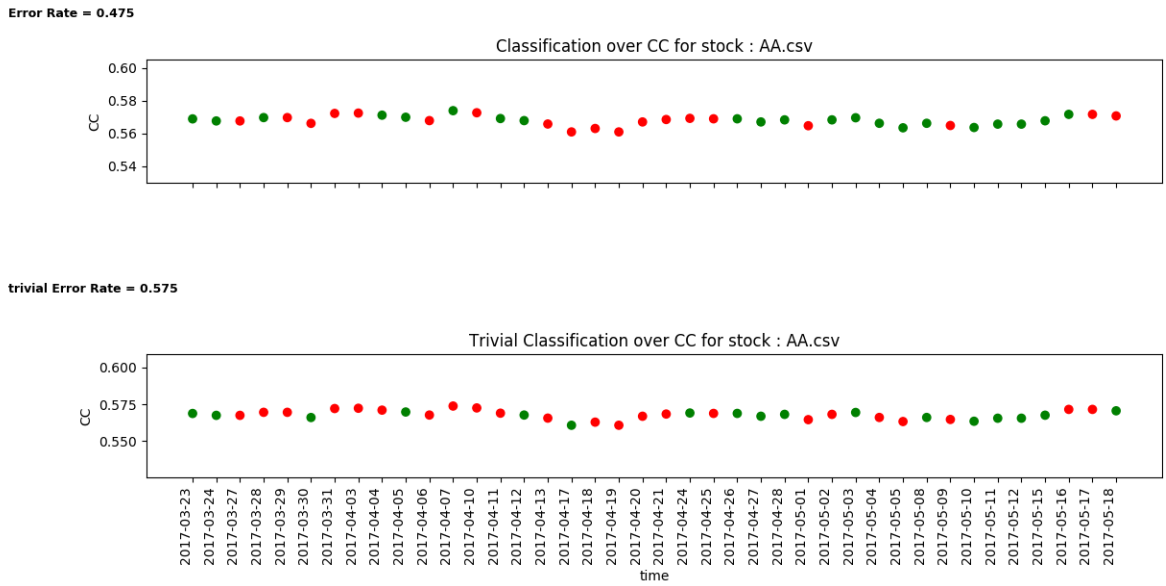


We can see by the graph that we improve our error rate to 0.375. We can notice that most of our success predictions come by bulks. We also see that when the CC changed sharply, we may loss of success prediction. We assume that sharp changes in CC lead to follow new best agent. This new best agent may be a liar and therefore we may wanted to cut his vote/his CC. That is lead us to the following interesting point- as long as the CC value have moderate increase over time, the better the prediction is.

In opposite to prediction by single lag, this method detect directional movement orientation of the correlated stock, and the prediction take to account the orientation rather than the specific exact lag obtained. From the scatter, it can be learn that prediction by orientation may be better from prediction by the exact lag .This, assumption is reasonable, because it is obvious that two different stock are not exact the same, but may response with deferent time delays and different frequencies, and still keep the orientation.

4.1.8 Probability Vote:

Figure 4.1.8.1:



In the figure 4.1.8.1 we ran the system by choosing the agent mode to be multiply lag and single CC. Also here we can see that the success prediction come in bulks. This method obtain worse error rate. We assume that the cause for that is because we have less information from previous method. An agent that want to vote '1' (gain), will vote twice the same value of the CC, which been as additional excess feature. Otherwise, the agent will vote '0' (loss), as he vote in the previous method. Therefore, we didn't gain additional information, and have excess features. This eventually leads for degradation in performance.

5 Summary

In this project, we were exposed to the wide world of predicting market in general and the correlation between stocks in particular. We found the topic of correlate stocks to be very interesting and have found much to research.

From the results, we can learn that the CC is more effective for predictions, when we detecting correlated orientations between stocks. This reflected by better results with multiply lags from the single lag. We can say that single lag may be perfect only for perfectly similar stock, which obviously not exist in real world.

In addition, we can say that the most relevant CC signal, is as expected, the close price signal by it binary formation. This reasonable because this is the signal we are willing to predict.

The results are interesting but it still way far from valuable. The error rate is not much less than the random trivial predictor.

It is implied by the last experiments that the changes in the CC over time may indicate true/false similarity instead of the CC value.

Our framework covers only a fraction of this research and can be expanded much more. Due to computation time, our results have lack of some basis things that should have been check, in order to verify the results.

5.1 Missing In Project

1) Our results reflects only by 40 days predictions. Predictions over much more days may achieve error rate to be closer to a random predictor.

2) Our samples taken from 24/5/2007 to 24/5/2017. The results and the tests we display in this framework was only by the recent 40 samples. In order to promise the results, we should achieve same results in independent windows of time within the period of the taken samples.

3) All the results are demonstrate on a specific stock: 'AA'. We chose this stock because it held with many other stocks high CC and valid lag. Testing on single stock is not sufficient to include the results over all the market stocks.

4) Although we mentioned the finance indicator of the GCR,DCR,RSI, eventually we didn't use them because we understand that they have no valuable CC.

5.2 More To Research

As already mentioned, the correlated stocks is a very wide topic and have much to research. Due to our scalable system, we can propose many researches, even based only on our system. Here is some of them:

1) Leverage agent performance –
   We are highly motivate researches to find better agent performance by adding some more features from the signal processing world. We notes that the CC values of the signals is better effective with the binary form. Therefore, making the agent vote by binaries CC signals, may be more efficient and need to find some more. We propose to investigate the binary signal of the MA.

2) Better preventing from liar agents-

We were prevented from agents, which have no lag, and high CC, by making the agents' votes constant zeros. We would rather to ignore this votes at all and preventing from feeding that classifier with such of votes. We could not manage to do so, because when we collecting the training set, we must have same size of features. In this way, one sample can have bigger size of other sample just because one ignore agent votes, which the other not. We assume that ignoring such of votes and keeping the same size of features along all the training set, can avoid noise and make the classifier more predictive.

3) Checking more parameters -

Our system have many parameters and any change in each of them can affect. We examined, and finally choose the best window size for the signals to be correlated according to our problem. However, we did not really check about different periods of MA, changing window size of the multiply lag prediction.

4) Develop "careful algorithm" –

Our system are always output prediction of gain/loss. There are several predictions that our classifier unit can't predict with high probability. We are rounding the output to be gain/loss even when the ANN outputs a bit upon/below 0.5. We think it may be more productive to evaluate the prediction only when the system are sure about the output by having better marginal values.

5) Prediction of the percentage change –

The predicted gain/loss of the stock are not reflects much the gain/loss trading strategy. This because the directions gain/loss may be not equivalent in value. In that way, one big loss may be greater than several continuous gains. Therefore, a prediction of percentage change together with our system can be powerful tool for traders, and can make some money.

6) Using matching window instead of CC –

This approach raised during the experiment when the edge effect revealed to be significant. We thought to use other indicator for similarity than the CC. this new indicator will be based on finding several windows with varying size of matching between two binary signals. The probability of every matching window with size of n, is $P_{window\ match} = \left(\frac{1}{2}\right)^n$. Hence the bigger window that achieve perfect match, so is the similarity between this stock with the

specific lag is bigger, by defining this similarity to be $\frac{1}{P_{window\ match}}$. In that case, 2 same stocks obtain similarity of:

$$similarity = \frac{1}{P_{window\ match}} =_{window\ length \to \infty} = \infty$$

Although, as we assume, this method have better similarity accuracy, it may be less accuracy with orientation detection from the CC. As we found in section 4.1.7, the orientation detection may be more powerful for such of predictors because in real world there are not same stocks.

6 Resources

- Buffalo capital management
  https://sites.google.com/site/predictingstockmovement/

- Github
- TipRanks
  https://www.tipranks.com/analysts/top

- Stock Prediction Based on Financial Correlation
- P. J. Kaufman. Trading Systems and Methods. John Wiley & Sons, 1998
- Yahoo finance
- Wikipedia
- Investopedia