CS229 - Project
Long-Short Strategy Using Bank Analysts Recommendations

Thomas AYOUL, Alexandre ELKRIEF, Nicolas MEILLER
Mentor: Youssef Ahres

December 11, 2015

Abstract

In this project, we aim at trying to predict stock prices of the Eurostoxx50 index using machine learning techniques applied to bank analyst recommendations. Based on those predictions, we create a long-short trading strategy. We then back-test this strategy and use different validation tools to improve the model. We investigate two types of logistic regression models with different sets of features. We then build strategies using sliding training sets and try other classifiers such as SVM or Random Forests. Finally, we add trend following and mean reversion features.

1 Introduction

Banks and equity research houses have analysts covering and publishing research on stocks, in particular blue-chips stocks such as the ones forming the Eurostoxx50 index. Aside from publishing research, they provide a rating to predict the future performance of the stock and a price target. These analysts are industry sector specialists and cover on only a small set companies. As a result, the ratings they publish take into account fundamental economic factors, historical results of the companies as well as their specific experience. We want to aggregate all these ratings which summed up all these factors using machine learning techniques.

2 Data Acquisition and formatting

Banks and stock broker's analysts publish their ratings, target prices and if they would advise to buy or sell the stocks on Bloomberg. For each feature, we use their average which can also be find on Bloomberg.

The first step of the data acquisition process was pulling up the data from Bloomberg. In the absence of a working python Bloomberg API on campus, we had to use excel to download the data. For each company of the Eurostoxx50 index, we used the historical ratings from each equity analyst covering the stock, the price target as well as the average rating and price target across all analysts. Finally, we downloaded the end of day prices. We used historical data from the January 1st 2000 to June 30th 2015.

The output from Bloomberg being very poorly formatted, we started by writing a VBA script to reformat it. We then temporarily put the data in python dictionaries to allow us to move forward with the modelling work. The sparsity of the ratings which are not published on a varying basis lead us to put the data in data-frames, which are much more convenient in order to deal with the concatenation of dates.

3 Related Work

Financial times series predictions to build trading strategies is a field driving a lot of innovation in the financial industry in particular in the hedge fund world. However, given the nature of their business model, a lot of the work happening in this field remains private. Machine learning driven algorithmic trading

strategies include PCA, Deep Neural Networks (Boltzman Machines), GBM and all forms of regressions. Another point that is even more opaque than the machine learning techniques used is the features chosen to build the model, which are a key factor in whether the strategy is going to be successful or not. It is therefore difficult to compare our performance with others. However, a strategy is generaly considered to perform well if it achieves a success rate of above 60%. Other key metrics to evaluate a trading strategy include its maximum draw-down and its Sharpe Ratio:

$$S = \frac{\text{Portfolio return} - \text{risk free rate}}{\text{Portfolio standard-deviation}}$$

4 Models

Model 1 : Naive Logistic regression

Creating a long-short strategy requires buy and sell signals. Therefore, in order to fit the binary nature of the outcomes we are trying to predict, we chose logistic regression as a first model. In this model, we assume that all stocks behave the same way and have independent increments. We then model them as a single response vector, i.e. we are assuming no difference between the stocks and put them all in the training set.

The features we wanted to use are the analyst ratings and target prices but we had to compute everything in return terms over a given interval t to have a comparable scale for all stocks. This gives the following features matrix:

$$\mathbf{X} = [\ StockReturn_t,\ AnalystTargetReturn_t,\ RatingReturn_t,\ AverageRating_t,\ EurostoxxReturn_t\ ]$$

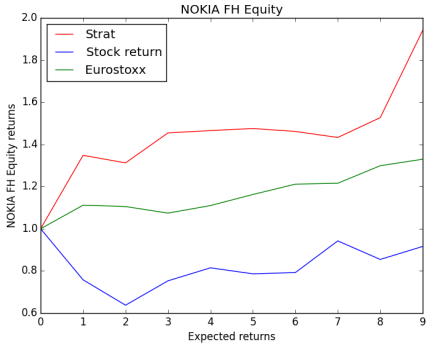
where the return for a quantity S is computed as $\frac{S_t - S_{t-n_{day}}}{S_{t-n_{day}}}$.

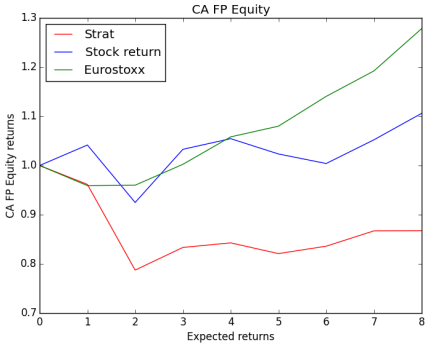
We are trying to predict whether a given stock price is going to raise or fall in the next n_{day} days. As a result the label associated will be:

$$Y = \begin{cases} 1 \text{ if } StockReturn_{t+n_{day}} > 0 \\ 0 \text{ otherwise} \end{cases}$$

Comments and Results

In the first iteration of the model we use 75% of the data as training set, with no consideration of the chronology of events. The model is then tested on the remaining 25% of the data. This model achieves an error rate of 43% which isn't quite sufficient to build a successful trading strategy. Indeed, as we can see in the figures below, the resulting long-short strategy can perform very well or very poorly, depending on the stock.





Modelling all stocks identically in a single response vector is a very strong assumption, it assumes that stocks have no sector or industry specific components and that they are independent which doesn't seem to be performing well. The way we trained our model also does not take into account the time series dimension of stock prices. Training of 75% of the data randomly means that we could be using prices target for the year 2003 to predict 2014 which is a clear flaw. We therefore move on to building a second model to consider the stocks as a whole in order to take correlations as well as the chronological component of stock prices into account.

Model 2 : Greedy Logistic Regression

In the second model, as we could not predict each stock individually, we try to predict the return of the Eurostoxx50 over the next n_{day} days. The response variable is therefore a buy or sell binary signal on the index, defined as :

$$Y = \begin{cases} 1 \text{ if } EurostoxxReturn_{t+n_{day}} > 0 \\ 0 \text{ otherwise} \end{cases}$$

Similarly to Model 1 everything is computed in terms of returns to to have a comparable scale across all stocks. As we take all the stocks as a whole, every training sample will have information on all the stocks and we will have a feature matrix composed of the same components as the one from the first model, except that we separate the features in stock specific vectors. For stock i we have 3 features:

$$X_i = [\ StockReturn_t^i,\ AnalystTargetReturn_t^i,\ RatingReturn_t^i\ ]$$

This gives the following features matrix with 150 columns:

$$\mathbf{X} = [X_1,\ X_2,\ ...,\ X_{50}]$$

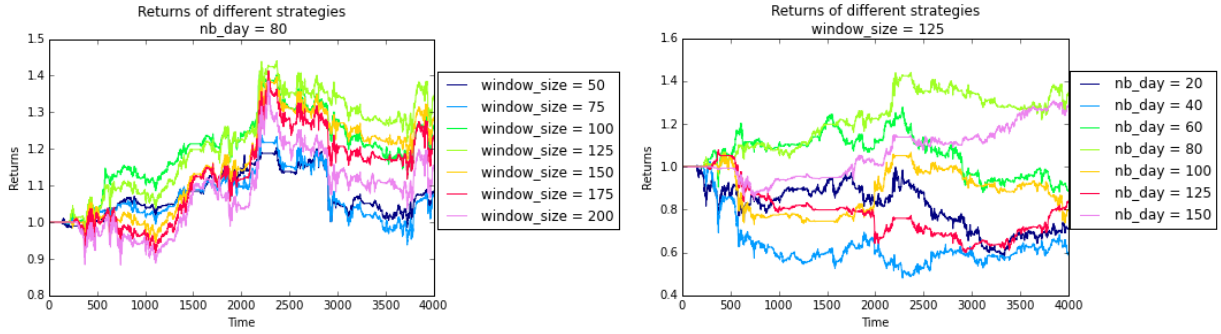
Comments and Results

After optimizing the paramters the best possibl return period appears to be $n_{day} = 40$ days. This means that upon each buy or sell signal from the model, the strategy is to buy the Eurostoxx50 for 40 days and measure the return at the end of this period. In terms of training set, we train on 90% of the data and predict on the remaining 10%. This model achieves an error rate of 33%. We then back-test the strategy giving the results in the figure below. We manage to create a strategy that outperforms Eurostoxx by almost 10% over the past year.

Nevertheless, the error rate results seem to be very sensitive to the percentage of the data on which we train the model. This is logical since the smaller the size of the training set, the bigger the gap between the features we have and predictions we're making. On the contrary, too big learning set takes lot of data that are not relevant with current market.

Model 3 : Sliding Logistic regression

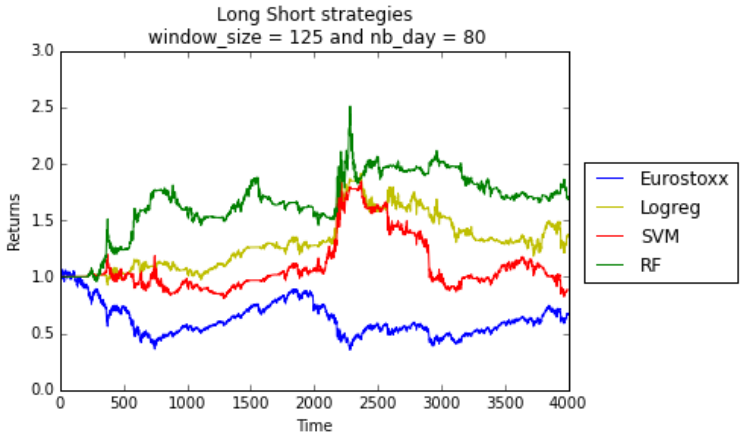
As the size of the training set appears to be a decisive parameter for the previous model and as we did not want to use too old data to predict we decided to train on a sliding window. For instance, at each moment we will only use the information of last year to compute our prediction. It comes from the fact that the markets are changing as well as the stocks prices. It also enables us to test our strategy on several different periods: periods of growth as well as huge draw-downs, such as the 2008 economic crisis. We first of all optimize the size of the sliding window as well as nb_{day}, the horizon of forecasting.



It appears that the optimal number of days is around 80 trading days which corresponds to about 4 months which is the time horizon of forecasting by the analysts.

Does it work with other classifiers ?

After optimizing the parameters we tried other classifiers such as SVM or random forest.



The strategy that seems to perform the best is the one with Random Forest. However, a sharper analysis shows that these strategies behave differently depending on the market fluctuations. These strategies are making profit during huge draw-downs such as the 2008 crisis (for t around 2000 on the graph), which is really good. However, our strategies seem to be performing less well during calm periods. During these periods mean reversion and trend following are performing well generally. As a result, we decided to add these features to our data. However, in our model as they were treated as the other features, there were no real difference. It may be because the time horizon of these approaches are quite different. A point of further research would be to add a second layer to our model to benefit from both approaches.

Moreover these strategies are much less volatile than Eurostoxx. This is a good points for our strategies : we manage to produce bigger returns with smaller volatility, as it can be see in the following table. The Sharpe ratios of our strategies are thus much bigger than the Eurostoxx.

Classifiers	Annualized volatility	Cumulative returns
Logistic regression	9.0 %	35.8 %
SVM	10.9 %	-11.6 %
Random Forest	8.50 %	69.7 %
Eurostoxx	14.5 %	-33.2 %

5 Conclusion and Further Research

Nowadays, a lot of trading strategies are only using mean reversions or trend following approaches. These techniques are usually working quite well in calm periods, but are completely random during crisis. On the contrary, our algorithms manage to "predict" most of the draw-downs.

Indeed the inclusion of bank analysts recommendations which are based on fundamental economic valuations allows for the strategies to incorporate a different and approach than exploiting only the time series data.

The next step would be to develop a second layer that takes into account the study of the recommendation first then compare it to the mean reversion or trend following strategies. Given the good results we got and as students in Computational Finance, we will try to continue this project over the next quarters.

References

[1] Tze Leung Lai, Haipeng Xing. *Statistical Models and Methods for Financial Markets (2008th Edition).*

[2] John C. Hull. *Options, Futures and Other Derivatives (8th Edition).*

[3] Jegadeesh, Narasimhan. *Evidence of predictable behavior of security returns.* Journal of Finance, 45: 881–898, 1990.