Stock Market Price Prediction Using Machine Learning Algorithms

Names	Marius Kinderis, Monika, James Barry, Thomas Legrand
Student Numbers	16211820,16210539, 16212754, 16115821
Programme	MCM
Module Code	CA684
Assignment Title	Machine Learning
Submission date	20/03/2017
Module coordinator	Qun Liu

I declare that this material, which I now submit for assessment, is entirely my own work and has not been taken from the work of others, save and to the extent that such work has been cited and acknowledged within the text of my work. I understand that plagiarism, collusion, and copying are grave and serious offences in the university and accept the penalties that would be imposed should I engage in plagiarism, collusion or copying. I have read and understood the Assignment Regulations set out in the module documentation. I have identified and included the source of all facts, ideas, opinions, and viewpoints of others in the assignment references. Direct quotations from books, journal articles, internet sources, module text, or any other source whatsoever are acknowledged and the source cited are identified in the assignment references. This assignment, or any part of it, has not been previously submitted by me or any other person for assessment on this or any other course of study.

I have read and understood the referencing guidelines found recommended in the assignment guidelines.

Name: Marius Kinderis
Monika
James Barry
Thomas Legrand

Date: 21/04/2017

Stock Market Prediction using Machine Learning Algorithms

Marius Kinderis, James Barry, Monika Chaudhary, Thomas Legrand

April 21, 2017

1 Introduction

The prediction of stock market price movements represents a challenging task. It has become increasingly known that the movement of stock prices are not random, indicating the need for robust models to conduct technical analysis. First generation studies have focused on linear statistical methods to forecast stock market price movements. Financial literature since the late 1980s has highlighted the role of Artificial Intelligence in stock market prediction. This paper aims to test the efficacy of using two Machine Learning techniques to conduct stock market price prediction: a Recurrent Neural Network in the form of a Long Short-Term Memory Network and a Support Vector Machine. The two models are evaluated within the context of predicting the direction of the Dow Jones Industrial Average Index.

2 Problem Definition

The aim of this project is to formulate a number machine learning models which will attempt to predict the direction of the Dow Jones Industrial Average (DJIA) Index. Long Short-Term Memory networks (LSTMs), are a specific type of Recurrent Neural Network which possess the ability of learning long-term dependencies. The task in hand is the prediction of time-series data, and the LSTM model is chosen as the model is designed to avoid long-term dependency problems. A Support Vector Machine is also studied as a comparison.

3 Existing Methods

3.1 Statistical

There exist a wide array of forecasting models which have been used to predict the direction of stock price movements. Traditional methods to predict stock prices include least squares regression, the Capital Asset Pricing Model (CAPM) and the Arbitrage Pricing Theory (APT). Other statistical methods include the use of Autoregressive Integrated Moving Average (ARIMA) models developed by Box and Jenkins (1970). Statistical forecasting methods represent the first generation models which have been designed for stock market prediction. It must be noted however, that more recent research by Chen and Leung (2004) has shown that standard econometric models were unable to produce significantly better predictions than the Random Walk model proposed by Malkiel (1985). Meanwhile, a study by Huang et al. (2005) has shown that nonlinear models are more suited to simulate volatile stock markets and are better at predicting results than traditional linear models.

3.2 Machine Learning

3.2.1 Artificial Neural Networks

Literature since the late 1980s has highlighted the role of Artificial Intelligence in stock market prediction. Inspired by the effectiveness of neural network modelling techniques in other fields involving pattern recognition and nonlinear forecasting, researchers such as White (1988) posited the question whether such techniques could also be applied to extracting nonlinear regularities from financial data.

Numerous studies indicate the success of artificial intelligence in predicting stock market behaviour. Refenes et al. (1994) modelled stock returns with ANNs in the framework of the APT and compared the outcome with regression models. Their study showed that even the most simple neural networks, such as Back Propagation (BP) neural networks, outperform the best practice of regression models which adopt the APT for stock ranking. ANNs are an attractive tool for predicting stock price movements due to their ability to find accurate solutions from data which is mostly complex, noisy and chaotic (Deboek, 1994).

Despite the success which ANNs have experienced across a wide range of financial prediction tasks, Lawrence (1997) mentions that a neural network may not be the best prediction device because the factors in a highly complex system like the stock market are too complex to be understood for a long time. Further to this, Yu et al., (2009) point out that in some applications of ANNs, local minima and overfitting are often encountered in the model. Overfitting occurs when the model tries to fit all points in a dataset and therefore, becomes too dependent on the training data and doesn't capture the general trend. In such cases, the model is unable to generalise when new data is added and subsequently may encounter problems related to local minima.

3.2.2 Support Vector Machines

Having experienced some potential limitations with ANNs, namely the propensity to experience local minima and overfitting problems, more recent literature has focussed on the use of Support Vector Machines (SVMs). An SVM is a machine learning algorithm which finds a maximum margin hyperplane that separates two data classes. The use of SVMs in financial prediction gained popularity after the seminal work of Vapnik (1995).

There exist two advantages of using an SVM over an ANN. Firstly, Huang et al. (2005) outline that a key property of an SVM is that training an SVM involves solving a linearly constrained quadratic programming problem so that the solution of SVM is always unique and globally optimal, unlike a neural network's training which requires non-linear optimisation with the danger of getting stuck at local minima. Secondly, the SVM minimises the risk of overfitting by choosing the maximum margin hyperplane in the feature space. Here, the hyperplane finds an optimal solution by maximizing the distance between classes and only chooses the data points which lie closest to the separating hyperplane. Additionally, if the data is not linearly separable, an SVM can transform the data into a higher dimensional space through a process called the Kernel function. For these reasons, the SVM models have gained popularity (Kim, 2003).

3.2.3 Long Short-Term Memory RNN

Unlike regression modeling, time-series predictive modeling adds a further degree of complexity due to the presence of a sequence dependence between input variables (Brownlee, 2016). One species of models which can control this sequence dependence are Recurrent Neural Networks (RNNs). RNNs are a type of Neural Network which possess a chain of repeating modules of a neural network (Olah, 2015). A specific type of RNN is the Long Short-Term Memory Network (LSTM) which was developed by Hochreiter & Schmidhuber (1997) in order to remember sequences of information for long periods of time. Their superior ability to remember long sequences of information is why

they are considered an excellent model for prediction tasks such as predicting financial time-series data.

The main benefit of the LSTM is that if there is a gap between relevant information, the LSTM will not forget relevant data whereas a regular RNN will not be able to connect information if the relevant data is too far apart. In this way, they possess an ability to maintain memory over long distances.

The LSTM is trained using Backpropagation through time. Instead of regular neurons, LSTM networks have memory blocks that are connected through layers. The blocks have components which are capable of memorising recent sequences, which makes it smarter than a traditional neuron. The block consists of gates which control the block's state and output. A block operates on an input sequence and each gate within the block uses the sigmoid activation units to control whether they are triggered or not. In this way, the change of state and the allowance of information flowing through the block is conditional. The gates of the units have weights that are learned through the training of the model.

The three types of gate within a unit are: The forget gate - which decides what information to remove or 'forget' from the block. The input gate - which decides what values from the input will update the memory state. The output gate - which decides on what to output based on the input and the memory of the block.

4 Methods

4.1 Intuition

As alluded to in the Existing Methods section, there exist a number of shortcomings with traditional statistical methods and even Artificial Neural Networks. For these reasons, it was decided to use a state-of-the-art LSTM-RNN model for our time-series prediction task. The LSTM model should be more suited to this task due to its ability to connect long sequences of information. The Support Vector Machine model is also included as it is a very good model for tasks such as stock market price prediction.

4.2 Dataset

We gathered the dataset from the Yahoo Finance website. We chose to study the Dow Jones Industrial Average Index between the 1st of January 2002 (to avoid direct consequences from the events of September the 11th 2001) and the 7th of April 2017. We load the csv file with Pandas library and only keep closing values. Indeed it is not needed to keep dates as time between two values is supposed to be the same. The next important step is preprocessing the data. We begin with converting the integer values into floats, as it is a better output for Neural Networks. Indeed matrix multiplication is one of the most common mathematical operations for neural network programming, and it has been proven that using floats leads to better performance. We also have to standardize the data using Scikit-Learn MinMaxScaler function to minimize the effect of the scale on the predictions. When it is done, it is time to split the data into a training and a testing set. Knowing that with time series data, the sequence of values is important, we won't randomly choose but rather take the first two-halves of the dataset as the training data and the remaining part as the testing data. Finally, instead of having a 1-dimensional dataset, we decide to reshape it into a 2-dimensional dataset, where the first column represents the value at a time t and the second column as the value as the time t+1. Now the data is ready to be used in our models.

4.3 Question

To what extent can we predict the price of the Dow Jones Industrial Average Index using Machine Learning models?

5 Modeling and Experiments

For the assignment, we followed Crisp-DM model which is Cross Industry Standard process for Data Mining.

We used a few different machine learning algorithms for our experiment: SVM and LSTM-RNN.

5.1 SVM - Support Vector Machine

On a first approach, SVM seems to be a suitable method to predict time series data. Thus we used the Support Vector Regressor from the Scikit-Learn library. It is important to determine the right kernel between linear, polynomial, gaussian and sigmoid as well as tuning the hyperparameters. For that purpose we used the GridSearchCV function from Scikit-Learn, which enables us to test a set of possibilities. We are also using a 3 K-Fold cross validation to decrease the importance of the pick of the training and testing datasets. It just so happens the best parameters are a linear kernel with a penalty parameter of 1.25.

5.2 LSTM-RNN

The previous result is quite satisfactory, but it is interesting to know if a Neural Network could outperform. LSTM seems to be the better choice for predicting times series data. For this section, we will use the Keras library, based on Tensor Flow, which makes it easier to build networks. To begin with, we need to reshape the data in order to put them as inputs which will suit Keras requirements, hence the shape [samples, time steps, features].

Our network is made of a visible layer with 1 input, a hidden layer with 4 LSTM neurons, and an output layer that makes a single value prediction. All the neurons implement the default sigmoid activation function. Then we compile it by using Adam optimizer, which is an algorithm for first-order gradient-based optimization of stochastic objective functions.

The number of epochs will be 10, because evidence shows that it quickly converges, and the batch size to train the network is set to 1, meaning that the method will be stochastic. According to Kesker et al. (2016), using a larger batch size results in a significant degradation in the quality of the model. Additionally, by using a smaller batch size, it helps us to avoid encountering problems related to local minima. Smaller batch sizes are also recommended for partially random problems, which is the case concerning stock prediction because despite every day being linked, some external events could have a big impact on the Close value.

5.3 Window Method

After trying to predict the value at the next day's (t+1) stock price, based on today (t), we will now try to use 3 days before (time = t + 1). We are using the same workflow as before, except that the LSTM has a different input shape but we are keeping the same structure.

6 Result and Analysis

6.1 SVM

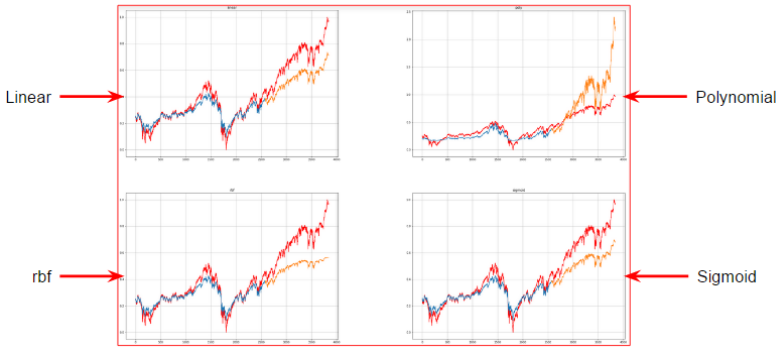

Figure 1: Different SVM kernels.

Here we have displayed different predictions of SVM algorithms. The red line depicts the actual price on the specific day. Blue fluctuation represents training data and orange fluctuation represents the testing data. Just by looking at the graphs, we can see that the best predictions were made using the linear model.

6.2 LSTM RNN

6.2.1 Simple

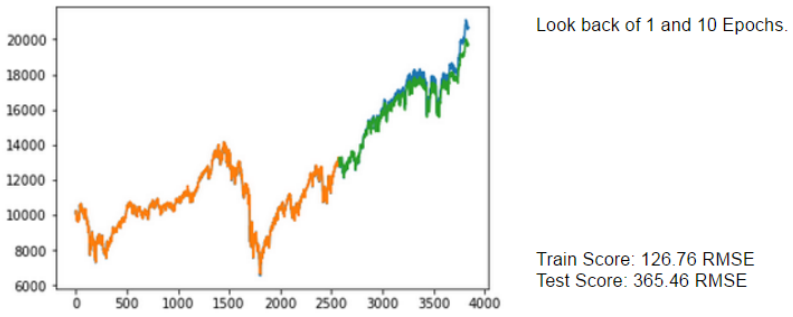

Figure 2: LSTM RNN with 1 day of Look back and 10 Epochs.

Here we have displayed different predictions of LSTM RNN algorithm. The blue line depicts the actual price on the specific day. Orange fluctuation represents training data and green fluctuation represents the testing data. Just by looking at the graphs we can see that the predictions are quite accurate. The RMSE looks good. We know that at the end of the prediction we expect a bigger error, which is due to a the randomness of the sales in the market.

6.2.2 Window Method

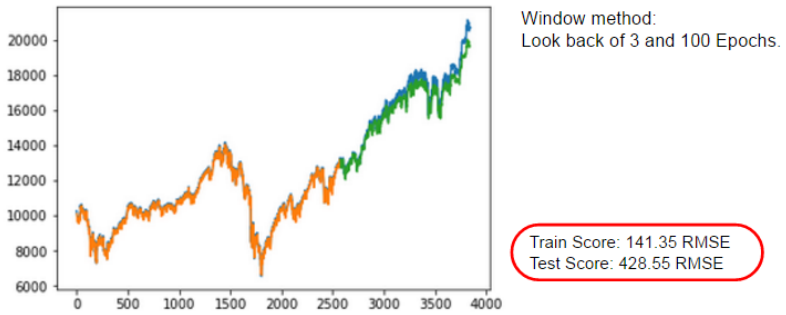

Figure 3: LSTM RNN with 3 days of Look back and 100 Epochs.

In this graph we have tried to increase the number of look backs to 3 days and use 100 epochs instead of 10. We cannot notice a huge difference in graphs but if we look at RMSE, we see that the error has increased.

We can deduce that we have overfitted our model and that our model has some difficulty to adapt to the fast fluctuation due to the randomness of the market.

7 Conclusion

As a conclusion, predicting stocks is not an easy task for a machine learning model but it has still better results than human intuition. The CRISP-DM approach was used to manage this project. First, we were able to understand the business surroundings, then we obtained our data and spent some time to understand it. We did not need to clean our data which led us directly to the modeling part where we have explored 2 approaches of dealing with the problem: SVM and LSTM RNN models where we have obtained satisfying results. During this project we have learned how to deal with time series data which would help us a lot in the future as such data is not limited to financial markets but also in a lot of other important fields. This project gave us an opportunity to experiment what it feels to be a Machine Learning practitioner.

The whole project can be found on GitHub:

https://github.com/mariuskin/

References

G. Deboeck, Trading on the edge: neural, genetic, and fuzzy systems for chaotic financial markets. John Wiley & Sons, 1994, vol. 39.

R. Lawrence, “Using neural networks to forecast stock market prices,” University of Manitoba, 1997.

B. G. Malkiel, A random walk down Wall Street: including a life-cycle guide to personal investing. WW Norton & Company, 1999.

Keskar, N.S., Mudigere, D., Nocedal, J., Smelyanskiy, M. and Tang, P.T.P., 2016. On large-batch training for deep learning: Generalization gap and sharp minima. arXiv preprint arXiv:1609.04836.

W. Huang, Y. Nakamori, and S.-Y. Wang, “Forecasting stock market movement direction with support vector machine,” Computers & Operations Research, vol. 32, no. 10, pp. 2513–2522, 2005.

L.Yu, H.Chen, S.Wang, and K.K.Lai, “Evolving least squares support vector machines for stock market trend mining,” IEEE Transactions on evolutionary computation, vol. 13, no. 1, pp. 87–102, 2009.

Box, G.E. and Jenkins, G.M., 1970. Time Series Models for Forecasting and Control. San Francisco.

A.-S. Chen and M. T. Leung, “Regression neural network for error correction in foreign exchange forecasting and trading,” Computers & Operations Research, vol. 31, no. 7, pp. 1049–1068, 2004.

White, H., 1988. Economic prediction using neural networks: The case of IBM daily stock returns.

Cortes, C., and V. Vapnik. "Support vector machine [J]." Machine learning 20.3 (1995): 273-297.

K.-j. Kim, “Financial time series forecasting using support vector machines,” Neurocomputing, vol. 55, no. 1, pp. 307–319, 2003.

Brownlee, Jason (2016):Time Series Prediction with LSTM Recurrent Neural Networks in Python with Keras - Machine Learning Mastery: http://machinelearningmastery.com/time-series-prediction-lstm-recurrent-neural-networks-python-keras/. [Accessed 07 Apr.2017]

Colah.github.io. (2017). *Understanding LSTM Networks -- colah's blog*. [online] Available at: http://colah.github.io/posts/2015-08-Understanding-LSTMs/ [Accessed 22 Apr. 2017].

Hochreiter, Sepp, and Jürgen Schmidhuber. "Long Short-Term Memory | Neural Computation | MIT Press Journals". Mitpressjournals.org. N.p., 1997. Web. 18 Apr. 2017.