International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 08 Issue: 05 | May 2021 www.irjet.net p-ISSN: 2395-0072

Web Application to Analyze Stock Market Results using Machine Learning

S.Annie sheryl[1], D.Thulasiram[2], K.Yogeshwar Reddy[3], R.Roshan[4]

[1]Assistant professor, Dept of Computer Science Engg., Panimalar Institute of Technology, Chennai, India
[2,3,4]Student, Dept. of Computer Science and Engineering, Panimalar Institute of Technology, Chennai, India

---

Abstract - *Deep learning is a part of machine learning in artificial intelligence that has networks able to adapt in learning unsupervised data that are unlabeled or unstructured. This proess is otherwise known as deep neural learning or deep neural networks. The Deep Learning consists of an algorithm called Convolutional Neural Network(ConvNet/CNN) which gets the input image, assign importance (learnable weights and biases) to various aspects/objects in that image and be able to distinguish one from the other. The pre-processing required in a ConvNet is much lesser as compared to the other classification algorithms. In Deep Learning primitive methods filters are hand-engineered that comes with enough training, Convolutional Networks have the ability to learn these filters/characteristics..*

1. INTRODUCTION

The stock market is a broad array of investors and traders who gets and trade stock, hiking the price up or down. The prices of stocks are managed by the demand and supply principles and the ultimate goal of buying shares the trade is to make money by buying stocks in companies whose discern value (i.e., share price) is expected to rise. Stock markets are firmly linked with the chain globe of economics. The hike and fall of share prices could be traced with of some Key Performance Indicators (KPI's) .Here the opening stockprice (`Open') ,intra-day peak price (`High'), intra - day low price (`Low'), end-of-day price (`Close'), and total volume of stocks traded during the day (`Volume')are the five most commonly used KPI's**.**

Economics and stock prices are mainly relying upon subjective views about the stock market. It is near impossible to foretell stock prices to the T, due to the volatility of components that play a crucial role in the movement of prices. Yet, it is possible to make an inculcate estimate of prices. Stock prices never differ in confinement: the movement of one inclined to have an avalanche effect on several other stocks as well [2]. This point of stock cost movement can be used as an crucial tool to predict the prices of many stocks at a time. Due to the utter volume of money intricate and number of money exchanges that take place every minute, there comes a trade-off between the accuracy and the volume of predictions made; as such, most stock prediction systems are implemented in a distributed, parallelized fashion [7]. These are some of the considerations and challenges faced in stock market analysis.

2. EXISTING SYSTEMS AND THEIR DRAWBACKS

Traditional addresses to stock market analysis and stock price prediction including fundamental analysis, which peers at a stock's past performance and the general credibility of the organization itself, and statistical analysis, which is solely disquited with number crunching and identifying the specific patterns in stock price variation. The latter is commonly achieved with the assisting of Artificial Neural Networks (ANN's) or Genetic Algorithms (GA), but these fail to capture relationship between stock prices in the form of long-term temporal dependencies. Another prime affair with using simple ANNs for stock prediction in the phenomenon of blowing up / vanishing gradient , where the mass of a large network either become heavy or too small (respectively), exceedingly slowing their convergence to the suitable value. This is constantly caused by two factors: mass are initialized randomly, and the mass closer to the end of the network also alternate to change a lot more than those at the beginning. .

In a possible approach to stock market analysis is to diminish the dimensionality of the input data [2] and apply feature discretion algorithms to shortlist a core set of features (such as gold price, oil price, inflation rate, etc.) that have the greatest knock on stock prices or money exchange rates across stock markets. However, this method won't be consider long- term trading strategies as it fails to take the entire history of trends into account; furthermore, there is no provision for outlier detection.

3. PROPOSED SYSTEM

We prefer an online studying algorithm for forecasting the end-of-day price of a given stock (see Figure 2) with the assist of Long Short Term Memory (LSTM), a type of Recurrent Neural Network (RNN).

Batch versus online learning algorithms

Online and batch learning algorithms varies in the way in which they work. In an online algorithm, it is feasible to stop the enhancement process in the mid of a learning run and still train an effective model. This is specifically useful for very large data sets (such as stock price datasets) when the convergence can be sustained and learning can be stopped early. The hypothetical learning pattern is ideally used for online learning because the model is trained on every data point — each limitation update only uses a single randomly chosen data point, and while vacillations may be observed, the weights eventually converge to the same optimal value. On the contrary, batch algorithms keep the system masses continuous while computing the error correlated with each sample in the input. That is, each parameter improvement involves a scan of the whole dataset this can be highly time- consuming for large-sized stock price data. Speed and precision are equally important in predicting trends in stock price movement, where prices can alter wildly within the span of a day. As such, an online learning process is finer for stock price prediction.

LSTM – an overview

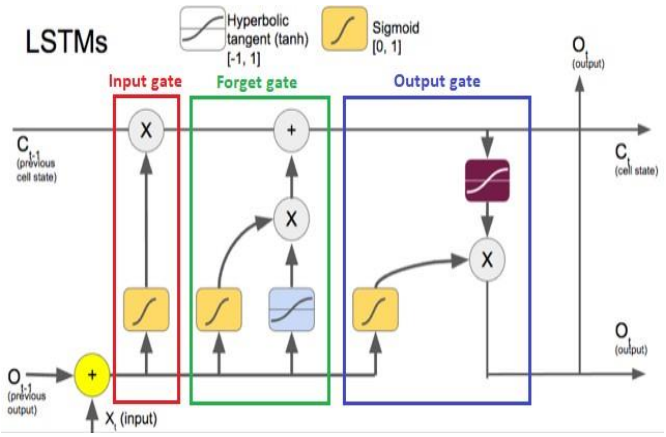

Fig - 1: An LSTM memory cell

LSTM's are a distinctive subset of RNN's that can capture context-specific temporal reliances for long periods of time. Each LSTM neuron is a memory cell that can retain other information i.e., it maintains its own cell state. While neurons in usual RNN's merely take in their previous invisible state and the present input to output a new hidden state, an LSTM neuron also takes in its old cell state and results its new cell state.

An LSTM memory cell, as depicted in Figure 1, has the following three components, or gates:

1. **Forget gate**: the forget gate chooses when specific portions of the cell state are to be replaced with more recent information. It outputs values close to 1 for parts of the cell state that should be retained, and zero for values that should be ignored.

2. **Input gate** : based on the input (i.e., previous output o(t-1), input x(t), and previous cell state c(t-1)), this section of the network learns the conditions under which any information should be stored (or updated) in the cell state

3. **Output gate**: depending on the input and cell state, this portion decides what information is generated forward (i.e., output o(t) and cell state c(t)) to the next node in the network.

Thus, LSTM networks are ideal for exploring how variation in one stock's price can affect the prices of several other stocks over a long period of time. They can also decide (in a dynamic fashion) for how long information about specific past trends in stock price movement needs to be retained in order to more accurately predict future trends in the variation of stock prices.

Advantages of LSTM

The main advantage of an LSTM is its ability to learn context- distinct temporal dependence. Each LSTM unit remembers information for either a long or a short period of time (hence the name) without directly using an activation function within the recurrent components.

An important fact to note is that any cell state is multiplied only by the output of the forget gate, which differs between 0 and 1. That is, the forget gate in an LSTM cell is accountable for both the weights and the activation function of the cell state. Therefore, information from a previous cell state can proceed through a cell unchanged instead of increasing or decreasing exponentially at each time-step or layer, and the weights can converge to their ideal values in a reasonable amount of time. This allows LSTM's to resolve the vanishing gradient problem – since the value saved in a memory cell isn't iteratively modified, the gradient does not vanish when trained with

backpropagation.

Additionally, LSTM's are also relatively not considerate to gaps (i.e., time lags between input data points) compared to other RNN's.

Stock prediction algorithm

Algorithm 1: LSTM stock prediction algorithm

Input: Historical stock price data

Output: Prediction for stock prices based on stock price variation

1. Start
2. Stock data is taken and stored in a numpy array of 3 dimensions (N,W,F) where :
   - N is number of training sequences,
   - W is sequence length
   - F is the number of features of each sequence.
3. A network structure is built with [1,a,b,1] dimensions, where there is 1 input layer, *a* neurons in the next layer, *b* neurons in the subsequent layer, and a single layer with a linear activation function.
4. Train the constructed network on the data
5. Use the output of the last layer as prediction of the next time step.
6. Repeat steps 4 and 5 until optimal convergence is reached.
7. Obtain predictions by providing test data as input to the network.
8. Evaluate accuracy by comparing predictions made with actual data.
9. End

Fig - 2: Stock prediction algorithm using LSTM

Terminologies used

Given below is a brief summary of the various terminology relating to our proposed stock prediction system:

1. **Training set** : subparagraph of the original data that is used to train the neural network model for predicting the output values
2. **Test set** : part of the original data that is used to make predictions of the output value, which are then differentiate with the actual values to judge the performance of the model
3. **Validation set** : section of the original data that is used to tune the framework of the neural network model
4. **Activation function**: in a neural network, the activation function of a node shows the output of that node as a weighted sum of inputs.

Here, the sigmoid and ReLU (Rectified Linear Unit) ac$Activation function = \Sigma(Input * weights) + Bias$ize the prediction model

a. **Sigmoid** – has the following formula

$$y = 1/(1+e^{-x})$$

and graphical representation (see Figure 3)

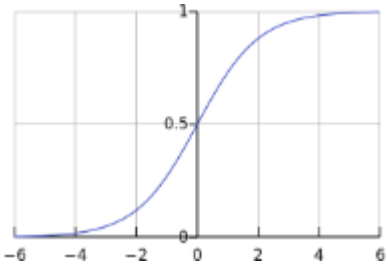

Fig - 3: sigmoid activation function

b. **ReLU** – has the following formula

$$y = max(0,x)$$

and graphical representation (see Figure 4)

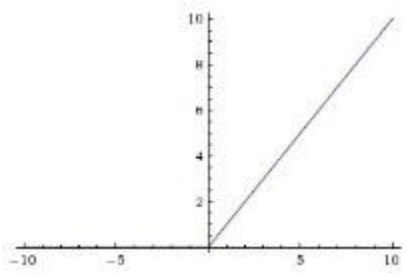

Fig - 4: ReLU activation function

5. **Batch size** : number of samples that must be processed by the model before renovating the weights of the parameters
6. **Epoch** : a complete pass through the given data file by the training algorithm
7. **Dropout**: a approach where randomly selected neurons are neglected during training i.e., they are "dropped out" randomly. Thus, their allowance to the activation of downstream neurons is secularly removed on the forward pass, and any weight updates are not applied to the neuron on the flip pass.
8. **Loss function** : a function, defined on a data point, prediction and label, that calculates a penalty such as square loss which is mathematically explained as follows –

$$l(f(xi), yi) = (f(xi) - yi)^2$$

9. **Cost function**: in addition of loss functions upon the training set . An example is the Mean Squared Error

(MSE), that is mathematically explained as follows:

$$MSE() = \Sigma Ni = 1(f(xi) - yi)^2/N$$

10. **Root Mean Square Error (RMSE)**: measure of the variation between values predicted by the model and the values clearly observed. It is calculated by obtaining the summation of squares of the differences between the predicted value and actual value, and dividing it by the number of given samples. It is mathematically expressed as follows:

$$\sqrt{\frac{\Sigma(y_{predicted} - y_{actual})^2}{N}}$$

In general, smaller the RMSE value, greater the accuracy of the predictions made.

4. OVERALL SYSTEM ARCHITECTURE

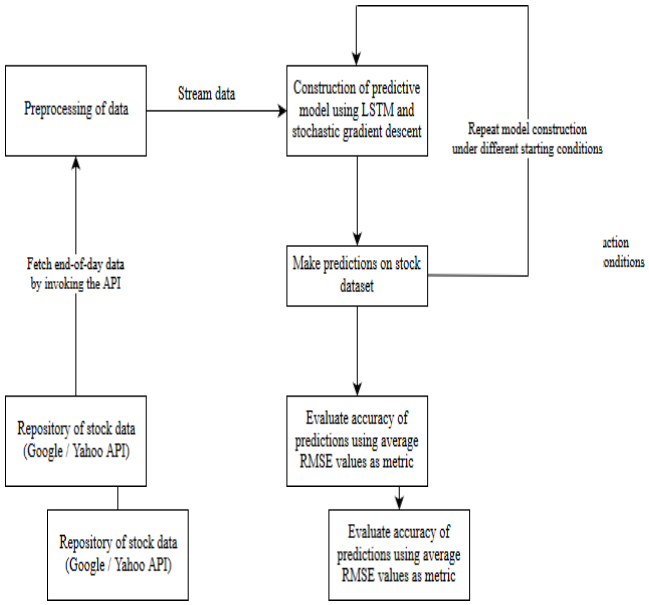

Fig – 5: LSTM-based stock price prediction system

The stock prediction system depicted in Figure 5 has three main components. A brief explanation of each is given below:

Obtaining dataset and preprocessing

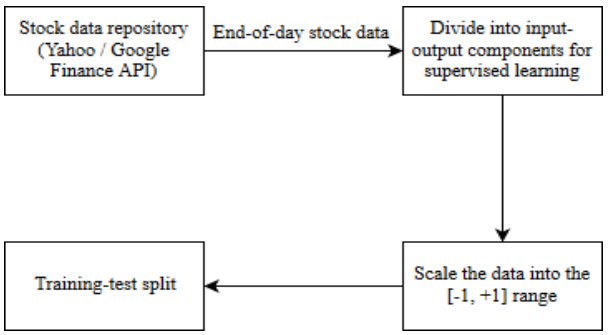

Fig – 6: Data preprocessing

Benchmark stock market data was extracted from two primary sources: Yahoo Finance and Google Finance. These two websites offer URL-based APIs from which historical stock data for various companies can be acquired for various companies by simply specifying some limitations in the URL.

The obtained data contained five features:

1. Date: of the observation
2. Opening price: of the stock
3. High: highest intra-day price obtained by the stock
4. Low: lowest intra-day price obtained by the stock
5. Volume: number of shares or contracts bought and sold in the market during the day
6. OpenInt i.e., Open Interest: how many futures contracts are presently The obtained data contained five features:

The above information is then transformed (Figure 6) into a format suitable for use with our prediction model by executing the following steps:

1. Transformation of time-series data into input-output components for directlearning
2. Scaling the data to the [-1, +1] range

Construction of prediction model

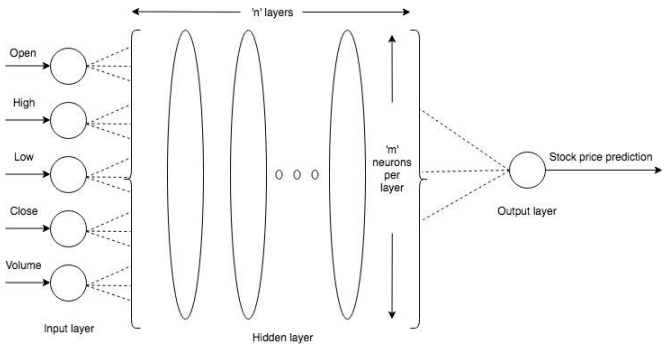

Fig – 7: Recurrent Neural Network structure for stock price prediction

The input data is divided into training and test datasets; our LSTM model will be placed on the training dataset, and the precision of the fit will be tested on the test dataset. The LSTM network (Figure 7) is builded with one input layer having five neurons, 'n' invisible layers (with 'm' LSTM memory cells per layer), and one output layer

(with one neuron). After placing the model on the training dataset, hyper-parameter adjusting is done using the evaluation set to choose the optimal values of parameters such as the number of hidden layers 'n', number of neurons 'm' per invisible layer, batch size, etc.

Predictions and accuracy

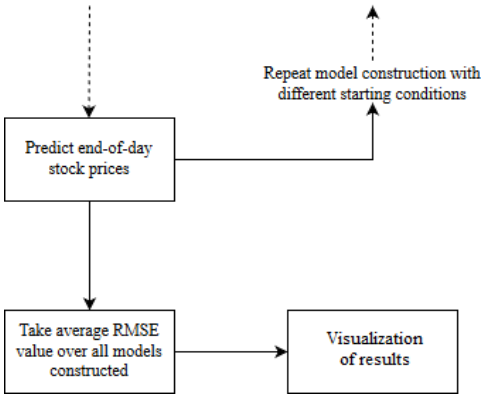

Fig – 8: Prediction of end-of-day stock prices

Once the LSTM model is fit to the training data, it can be used to foretell the end-of-day stock cost of an arbitrary stock.

This prediction can be performed in two ways:

1. Static – a easy, less accurate procedure where the model is placed on all the training data. Each new time step is then foretelled one at a time from evaluation data.
2. Dynamic – a complex, more precised approach where the model is refit for every time step of the test data as new observations are made available.

The precision of the prediction model can then be determined robustly using the RMSE (Root Mean Squared Error) metric. This is due to the fact that neural networks in general (including LSTM) tend to give various results with different starting conditions on the same data.

We then repeat the model construction and foretelling for a number of times (with different starting conditions) and then take the average RMSE as an indication of how well our configuration would be anticipated to perform on unseen real- world stock data (figure 8). That is, we will compare our predictions with actual trends in stock price ups and down that can be inferred from historical data.

5. VISUALIZATION OF RESULTS

Figure 9 shows the definite and the predicted closing stock price of the company Alcoa Corp, a large-sized stock. The model was trained with a batch size of 512 and 50 epochs, and the predictions made precisely matched the actual stock prices, as acquired information from the graph.

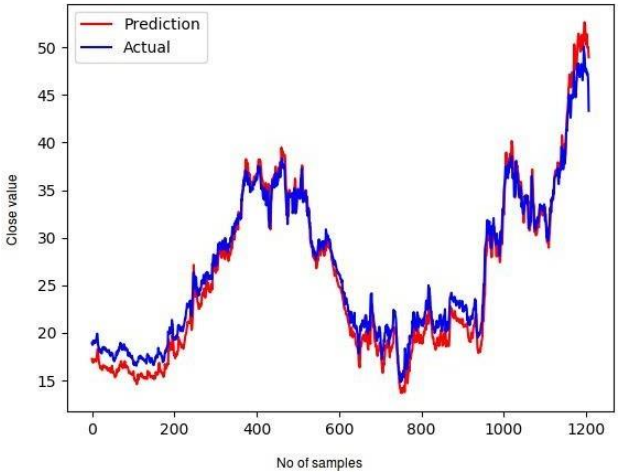

Fig - 9: Predicted stock price for Alcoa Corp

Figure 10 shows the actual and the predicted closing stock price of the company Carnival Corp, a average-sized stock. The model was trained with a batch size of 256 and 50 epochs, and the predictions made closely coordinated with the actual stock prices, as observed in the graph.

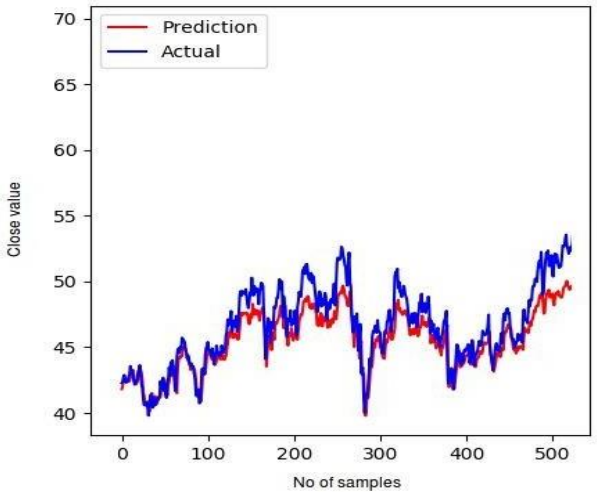

Fig - 10: Predicted stock price for Carnival Corp

Figure 11 shows the definite and the predicted closing stock price of the company Dixon Hughes Goodman Corp, a small-sized stock. The model was trained with a batch size of 32 and 50 epochs, and while the predictions made were fairly accurate at the start, variations will be acquired after some time.

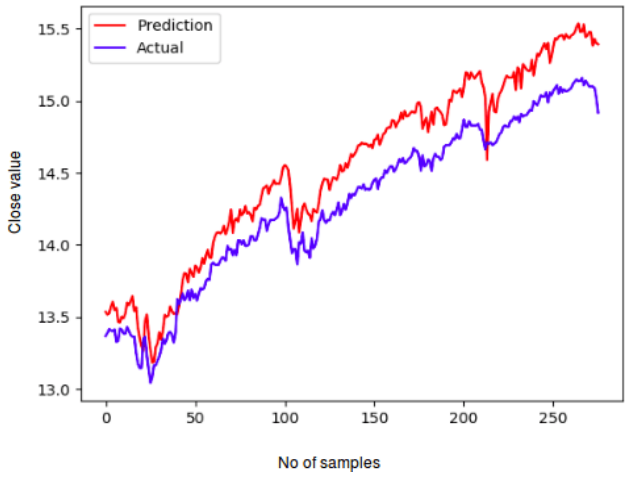

Fig - 11: Predicted stock price for DH Goodman Corp

We can thus infer from the results that in general, the prediction accuracy of the LSTM model improves with increase in the size of the dataset.

5. COMPARISON OF LSTM WITH ANN

The performance of our projected stock prediction system, that uses AN LSTM model, was compared with a straightforward Artificial Neural Network (ANN) model on 5 completely different stocks of varied sizes of information.

Three classes of stock were chosen relying upon the scale of data set. little knowledge set is also a stock that solely concerning ten years of data is out there e.g., Dixon Hughes. A medium- sized knowledge set could be a stock that knowledge up to twenty five years is offered, with examples together with Cooper Tire & Rubber and PNC money. Similarly, a large knowledgeset within which over twenty five years of stock data area unit available; Citigroup and yank Airlines area unit ideal samples of constant.

Parameters just like the coaching split, dropout, range of layers, range of neurons, and activation operate remained a similar for all knowledge sets for each LSTM and ANN. coaching split was set to zero.9, dropout was set to zero.3, range of layers was set to three, range of neurons per layer was set to 256, the elliptical activation operate is employed, and therefore the range of aeon was set to fifty. The batch sizes, however, varied in step with the stock size. little datasets had a batch size of thirty two, medium-sized datasets had a batch size of 256, and massive datasets had a batch size of 512

Table - 1: Comparison of error rates from the LSTM and ANN models

Data Size	Stock Name	**LSTM (RMSE)**	**ANN (RMSE)**
Small	Dixon Hughes	**0.04**	**0.17**
Medium	Cooper Tire & Rubber	**0.25**	**0.35**
Medium	PNC Financial	**0.2**	**0.28**
Large	CitiGroup	**0.02**	**0.04**
Large	Alcoa Corp	**0.02**	**0.04**

The given model RMSE (Root Mean Squared Error) for LSTM model value of 0.04, while the ANN model gave 0.17 for Dixon Hughes. For CEAT Tire & Rubber, LSTM gave an RMSE of 0.25 and ANN gave 0.35. The RMSE values were 0.2 and 0.28 are the PNC Financial values respectively. For Citigroup the LSTM provided an RMSE of 0.02 and ANN gave 0.04, while for the American Airlines the LSTM gave an RMSE of 0.02 and ANN gave 0.04

7. CONCLUSION AND FUTURE WORK

The comparison results between the Long Short Term Memory (LSTM) and Artificial Neural Network (ANN) shows that LSTM has a progressing prediction accuracy than ANN.

Stock markets are difficult to monitor and requires enough of context when trying to interpret the movement and predict its prices. In ANN, each hidden node is finely a node with a single activation function, while in LSTM, each node is a memory cell which stores the contextual information. As such, LSTMs perform good as they can be able to keep track of the context-specific temporal dependency between stock prices for an extensive period of time while performing predictions.

An analysis of the results also indicates that both models can give fine accuracy when the size of the dataset increases. With more amount of data, more patterns can be fleshed out with this model, and the load of the layers can be preferably adjusted.

The stock market is the core for reflection of human emotions. Analysis and Pure number crunching have their own limitations; a possible extension of this stock prediction system would be to augment it with a news feed analysis from social media platforms such as Facebook, where reactions are gauged from the articles. This sentiment analysis can be linked with the LSTM to train further with weights and in improving accuracy.

REFERENCES

[1] Nazar, Nasrin Banu, and Radha Senthilkumar. "An online approach for feature selection for classification in big data." Turkish Journal of Electrical Engineering & Computer Sciences 25.1 (2017): 163-171.

[2] Soulas, Eleftherios, and Dennis Shasha. "Online machine learning algorithms for currency exchange prediction." Computer Science Department in New York University, Tech. Rep 31 (2013).

[3] Suresh, Harini, et al. "Clinical Intervention Prediction and Understanding using Deep Networks." arXiv preprint arXiv:1705.08498 (2017).

[4] Pascanu, Razvan, Tomas Mikolov, and Yoshua Bengio. "On the difficulty of training recurrent neural networks." International Conference on Machine Learning. 2013.

[5] Zhu, Maohua, et al. "Training Long Short-Term Memory With Sparsified Stochastic Gradient Descent." (2016)

[6] Ruder, Sebastian. "An overview of gradient descent optimization algorithms." arXiv preprint arXiv:1609.04747.(2016).

[7] Recht, Benjamin, et al. "Hogwild: A lock-free approach to parallelizing stochastic gradient descent." Advances in neural information processing systems. 2011.

[8] Ding, Y., Zhao, P., Hoi, S. C., Ong, Y. S. "An Adaptive Gradient Method for Online AUC Maximization" In AAAI (pp. 2568-2574). (2015, January).

[9] Zhao, P., Hoi, S. C., Wang, J., Li, B. "Online transfer learning". Artificial Intelligence, 216, 76-102. (2014)

[10] Altinbas, H., Biskin, O. T. "Selecting macroeconomic influencers on stock markets by using feature selection algorithms". Procedia Economics and Finance, 30, 22- 29. (2015).