An Exploration of the Predictive Power of Social News Sentiment on Stock Market Prices.

Dylan McDonald

MSc in Data Analytics

*CCT*

*Dublin, Ireland*

sbs23038@student.cct.ie

***Abstract*—The influence of social media platforms has revolutionized information dissemination and significantly impacted financial markets. This paper delves into the intriguing relationship between sentiment expressed in social news and subsequent stock market price movements. Through a comprehensive exploration, we investigate the extent to which sentiments conveyed through social media can predict stock market price fluctuations.**

**We begin by establishing a robust methodology for sentiment analysis, leveraging advanced natural language processing techniques to extract sentiments from a dataset of social news source. To assess the predictive power of social news sentiment, we employ deep learning methods. The findings from this study shed light on the dynamic between social news sentiment and stock market dynamics. Understanding the predictive capabilities of social news sentiment holds immense potential for investors, traders, and financial analysts in making informed decisions, thus shaping the future landscape of financial markets and investment strategies.**

I. INTRODUCTION

This paper takes a primary focus on the predictability of language, particularly through social media blog posts, in future stock market price movements and whether the development of sophisticated neural networks, comprised of market sentiment, can provide a return for investors. The motivation behind this research focus was to test the efficient market hypothesis and factor in market sentiment as a potential determinant of price movement. The three types of Neural Networks explored in this research paper are Artificial Neural Networks (ANNs), Recurrent Neural Networks (RNNs), and Convolutional Neural Networks (CNNs).

II. LITERATURE REVIEW

The purpose of this literature review is to summarize and comment on existing literature in the areas of; applications of different neural network architectures, deep learning for sentiment analysis, and methods and tools for big data storage. Following a critical evaluation of sources, the aim is to outline a primary motivation for research, assess deep learning model architectures related to a specific application field, analyse the data storage requirements of this project and to identify any potential research gaps in these fields.

Sentiment is associated with different attributes and, as such, the literature points towards no universal definition. Schleifer and Summers (1990) define sentiment as an investors overall attitude toward financial markets, DeLong et al. (1990) defines it as investors’ formation of beliefs about future cash flows and investment risks that are not justified by existing facts, and Brown and Cliff (2004) are of the opinion it represents the ‘expectations of market participants relative to a norm: a bullish (bearish) investor expects returns to be above (below) average, whatever average may be. A myriad of research papers criticizes the efficient market hypothesis (EMH), the rational behaviour of investors, questioning the ‘random walk of future asset prices’ and whether or not other factors can act as future indicators for stock price movements. Behavioural finance argues that sentiment proxies may be candidates for explaining subtle pricing anomalies. Lee et al. (1991) argues that sentiment may be a candidate explanation for different asset valuations across investors. Trading strategies, such as the one suggested by Zhang and Skiena (2010) have been developed utilizing natural language processors (NLPs) on blog/news data to generate consistent returns for investors. Katayama and Tsuda (2020) investigated whether a trading strategy based on a sentiment analyser using deep learning can be profitable using Nikkei Telecom from the Nikkei Shimbun morning edition. Consequently, it is worthwhile to investigate whether deep learning models based on market sentiment can predict future stock price movements.

Deep learning (DL) is the subfield of artificial intelligence that focuses on creating large neural network models that are capable of making accurate data-driven decisions, particularly suited to contexts where the data available is complex and large (Kelleher, 2019). There have been three waves of developments in DL: cybernetics in the 1940s-1960s, connectionism in the 1980s-1990s and the current resurgence beginning in 2006 (Goodfellow et al., 2016). DL has attracted much attention from the academic community due to its state-of-the-art performance in many research domains such as speech recognition (Dahl et al., 2012), (Hinton et al., 2012) collaborative filtering (Salakhutdinov et al., 2007), and computer vision (Cireşan et al., 2010.),(Zeiler et al., 2011). Artificial/Feed-forward Neural Networks (ANNs) consist of an input layer, an output layer and a specific number of hidden layers. Neurons are contained in the input and output layers related to the number of input and output parameters. It is then an objective to determine the number of hidden layers and the number of neurons in each hidden layer. Activation functions are applied by each neuron in the hidden and output layers to the weighted sum of inputs. A Recurrent Neural Network (RNN) is an extension to the traditional ANN except that it is designed for processing sequences of data. The output from the neurons in the current layer are used as feedback to the neurons of the previous layer. RNNs are used to solve problems related to; time series data, text data and audio data. Convolutional Neural Networks (CNNs) are mainly used for image and video processing and are a subtype of Deep Neural Networks (DNNs) inspired by the visual cortex of animals. There have been applications of CNNs for the ordering of text for particular categorizations. Johnson and Zhang (2014) apply CNN to high-dimensional text data for use in classification.

A plethora of research has been carried out applying sentiment and financial market trading activities. Most studies have utilized traditional methods such as conventional classification algorithms and Pearson correlation coefficient. Loughran and McDonald (2011) developed a word list using a sample of the United States’ Security and Exchange Commission (SEC) 10-K filings from 1994 to 2008 categorizing words into six groups; positive, negative, litigious, uncertainty, model strong and model weak. They found significant relations between the categories and file date returns, trading volume, subsequent return volatility, standardized unexpected earnings, and two separate samples of fraud and material weakness. Wang et al. (2015) investigate the quality and impact of content on social investment platforms against historical performance of relevant stocks. Datasets are constructed from 9 years of SeekingAlpha articles and 4 years of StockTwits messages. Pearson’s correlation is computed between the article sentiment and a stock’s future price change. Wang et al. ignores the strength of sentiments and the magnitude of price movements, which will be explored in this paper. Da Silva et al. (2014) uses an ensemble of classifiers to label tweets as positive or negative concerning a query term. Multinomial Naïve Bayes, a Support-Vector Machine (SVM), Random Forest and Logistic Regression form the ensembles. The paper demonstrates that the use of ensembles of multiple base classifiers, combined with scores obtained from lexicons can improve the accuracy of tweet sentiment classification.

The emergence of deep learning techniques has allowed for large unstructured datasets to become very powerful in numerous scientific fields. Big data is defined as a collection of massive and complex data set and volume including huge quantities of data. Big data is described by ‘5 Vs’; Volume – the amount of data available, Velocity – the speed at which data is created, Variety – the different structures of data compiled together, Veracity – the quality of the data captured, and Value – the usability and usefulness of the data captured (Ishwarappa & Anuradha, 2015). The volume of the source data used in this research paper is quite large and therefore an effective means for storing and processing the data had to be utilized in order to use in the development of a predictive model. The data used in this paper was stored using the Hadoop Distributed File System (HDFS). HDFS is designed to handle very large datasets, be deployed on commodity hardware, exhibit a high level of fault tolerance, and enable streaming access to filesystem data. The files are broken into block sized chunks which are then stored as independent units. HDFS works off a master/slave architecture where namenodes are the master and datanodes act as slaves. The datanodes manage the data storage, handle request from the namenode and perform tasks such as block creation, deletion, and replication according to the instructions of the namenode. The data used in this paper was processed using Apache Spark. Apache Spark is a Java Virtual Machine (JVM) that can distribute data processing tasks across multiple computers, either on its down or in tandem with other distributed computing tools. A Resilient Distributed Dataset (RDD) is a concept representing an immutable collection of objects where operations can be split across a computing cluster. Spark is often compared to Hadoop’s MapReduce. The main difference between these two is that Spark processes and keeps the data in memory for subsequent steps resulting in dramatically faster processing speeds (IBM). In this paper, Spark transforms the large text file into a dataframe consisting of a table of rows and columns. Each row represents an entry into the redddit community in the form of a JSON object. Spark is accessed via python and once the required data is extracted it is exported and stored in HDFS.

III. METHODOLOGY

*A. Data Collection & Preparation*

The data used for the purpose of this research paper takes the form of titles of threads in the popular social news website, Reddit. The specific thread titles are collected from the ‘r/Economics’ community. The community is the largest for economics and serves as a central forum for users to read, discuss, and learn about topics related to the economic discipline. It allows for users to post opinions about current affairs such as immigration, trade, protectionism etc. The data was obtained from a specific thread in the ‘r/pushshift’ community as a torrent. Torrents allow for the download of larger files faster than traditional downloading methods. Once downloaded, the file is in a format called zstandard compressed ndjson and is extracted using the 7-Zip software to decompress the file. The file is decompressed into a 525MB text file with separate ‘JSON’ objects on each line of the text file.

Each line of the ‘Economics-submissions.txt’ file contains information such as the author’s username, the title of the thread, the UTC timestamp it was posted, the number of upvotes etc. To achieve the purpose of this research paper, the title of the thread and the UTC timestamp were the components of interest to extract from the text file. Due to the size (525MB) of the text file containing the relevant information for the subreddit threads relating to economics, the HDFS was utilized and interacted with ‘PySpark’, the python library for Apache Spark. The file was stored in HDFS on a virtual machine and was loaded as a ‘PySpark DataFrame’ with a single column, ‘value’. Each observation in ‘value’ represents one of the ‘JSON’ objects of the ‘Economics-Submissions.txt.’ file. After loading in as a dataframe, the titles and UTC timestamps were extracted by user-defined python functions used to transform the ‘value’ column into json objects, which creates key/value pairs. Two separate pyspark dataframes were created for the UTC timestamps and titles and were exported to HDFS. These dataframes were then allowed for sharing between the virtual machine and local machine on a shared drive established between the two, allowing for the utilization of the powerful ‘tensorflow’ library for building neural networks.

The stock price index used for the purpose of this research was the closing price of the S&P 500. The S&P 500 is widely regarded as the best single gauge of large-cap U.S equities. The index includes 500 leading companies and covers approximately 80% of available market capitalization (S&P, 2023). The data was downloaded via a Bloomberg terminal using the ticker ‘SPX’ for the relevant time.

A graph with blue line

Description automatically generated

Fig. 1. S&P 500 Index time series.

The dataframes shared from the virtual machine were then loaded into python for data preparation. The two independent dataframes obtained from the reddit text file and the closing prices of the S&P 500 index were merged, resulting in a reddit thread title, and closing price pairing for each date. The resulting dataframe is dated from January 2008 to December 2022. This data forms the dependent variable, the output of various neural networks. The target is the one-day stock price movement direction which will either be ‘Up’ or ‘Down’. Thus, this research paper explores a binary classification problem.

*B. Sentiment Analysis*

The Natural Language Toolkit (NLTK) package is imported into python which provides easy-to-use interfaces to over 50 corpora and lexical resources such as WordNet, along with a suite of text processing libraries for classification, tokenization, stemming, tagging, parsing, and semantic reasoning, wrappers for industrial-strength NLP libraries, and an active discussion forum (NLTK.org). The Valence Aware Dictionary and sentiment Reasoner (VADER) is used as a lexicon and rule-based sentiment analysis tool that is specifically attuned to sentiments expressed in social media (Shelar & Huang, 2018). VADER utilizes a mix of lexical highlights marked by their semantic direction as either positive or negative. It produces four sentiment measurements from each word grading, a positive, neutral, negative, and compound score. The compound score is a normalized score describing the total amount of the lexicon grades (Bajaj, 2021). These compound scores are aggregated for each date by taking the average score and then a sentiment is assigned to a particular date from the group of three based upon the below logic.

1. VADER Score cateogires

| **VADER Score Classification** | Compound Score | Sentiment |
| --- | --- | --- |
| >= 0.05 | Positive |
| <= -0.05 | Negative |
| -0.05 < Score < 0.05 | Neutral |

The sentiment assigned to the scores becomes an independent variable that will feed into various neural networks to assess predictability with stock market price movements. Two forms of the sentiment variable are considered as an input, one where the sentiment the day before a stock price movement and another the day of a stock price movement. The one-day lagged stock price market movement direction is also included as an input to decipher whether previous price movements can be attributed to future price movements. Therefore, we have three independent variables to consider as inputs when developing neural networks.

*C. Artificial/Feed-Forward Neural Network*

ANNs have been successfully applied to many diverse fields such as pattern classification, image processing and stock market predictions and thus are very attractive to use for learning on any nonlinear problem. These networks, due to their usability are often referred to as Universal Function Approximators and are particularly useful when dealing with tabular data. Shulka, 2019, provides informative detail on the set of requirements to consider when developing neural networks.

An ANN consists of an input layer, a specified number of hidden layers and an output layer. The input layer contains a specific number of neurons related to the number of features used by the network to make predictions. As this paper considers three independent variables, the specified dimension used in the design of the input layer for the ANN is three. The number of hidden layers and neurons within these layers is dependent on the problem, however, generally one to five hidden layers will suffice. For this paper, one hidden layer will be used and the number of neurons in this layer is tuned using the ‘GridSearchCV’ hyperparameter tuning method. Following tuning, the optimal number of neurons was set to 8. An activation function is defined in the hidden layer and is the function used to transform the input layer to the output (Sharma & Athaiya, 2020). For ANN this is optimally tuned to the sigmoid function. The sigmoid function is the most widely used activation function as it is a non-linear function. It is continuously differentiable and a smooth S-shaped function. For binary classification one output neuron is used per positive class and the sigmoid activation function is used to limit the output between zero and one. Binary cross-entropy is the default loss function for binary classification problems as it is the preferred function under the inference framework of maximum likelihood. The Root Mean Squared Propagation optimizer is tuned to decide the weights and learning rates of the neural network.

1. ANN HYPERPARAMETERS

| **ANN Hyperparameters** | Argument | Value |
| --- | --- | --- |
| Activation | Sigmoid |
| Batch Size | 10 |
| Epochs | 10 |
| Optimiser | RMSPROP |
| Hidden Units | 8 |

*D. Recurrent Neural Network*

RNN techniques have been applied to a wide variety of problems. Simple partially recurrent neural networks were introduced in the late 1980’s by several researchers including Rumelhart, Hinton, and Williams to learn strings of characters (Rumelhart et al., 1986).

The architecture of a RNN can range from fully interconnected to partially connected nets. A fully connected network does not have distinct input layers of nodes, and each node has input from all other nodes. A recurrent neural network is designed in a way where feedback to the node itself is possible. In a partially connected network, some nodes are part of a feed-forward structure and other nodes provide the sequential context and receive feedback from other nodes. The weights assigned to input nodes are processed using backpropagation (Medsker & Jain, 2001). Rumelhart, Hinton and Williams describe the backpropagation algorithm as repeatedly adjusting the weights of the connections in the network so as to minimize a measure of the difference between the actual output vector of the net and the desired output vector. This can lead to problems such as the vanish and exploding gradient problem, which should be factored into the architecture of an RNN.

The ‘Keras’ library of tensorflow is used in python to develop the architecture for a simple RNN used to predict the price direction of the S&P 500 index. The ‘SimpleRNN’ function calls in the input layer where each cell takes one data input and one hidden step that is passed from one time step to the next. The input dimensions used in the development of an RNN is different to that of an ANN. Whereas the input array used in an ANN corresponds to the number of features of a dataset, the input array used in an RNN incorporate the time-steps and the number of features. For the purpose of this research paper, three time-steps are considered as an arbitrary point to compare the predictive power of the two different neural networks. Hyperparameter tuning is also utilized when developing the RNN for key parameters such as; the number of neurons in the layer, the activation function, the optimization algorithm, the number of epochs and the batch size. The number of neurons tested are 50 and 100. This determines the capacity of the model to learn complex patterns from the input data. The activation functions tested are the sigmoid and relu functions and the optimization algorithms tested are adam, stochastic gradient descent and root mean square propagation. The hyperparameters used in the RNN architecture are found in the table 3.

1. RNN HYPERPARAMETERS

| **RNN Hyperparameters** | Argument | Value |
| --- | --- | --- |
| Activation | Relu |
| Batch Size | 20 |
| Epochs | 10 |
| Optimiser | SGD |
| Units | 100 |

IV. Findings

The performance of the neural networks described in the previous section are assessed under the metrics of accuracy, precision, and recall. These metrics are specific to classification problems and are obtained from confusion matrices which report true/false positives/negatives. A true positive is a correctly predicted event, a true negative is a correctly predicted non-event, a false positive is an incorrectly predicted event, and a false negative is an incorrectly predicted non-event. The below figure represents a confusion matrix.

A diagram of values and negative values

Description automatically generated

Fig. 2. Confusion Matrix.

Accuracy is described as the percentage of cases that are correctly classified in a dataset. A high accuracy figure represents that a neural network is correctly predicting class labels for the given inputs. Precision denotes the proportion of predicted positives that are correctly categorized as positive. It can be referred to as the true positive accuracy. A high precision values proves that a neural network avoids false positives. Recall or sensitivity is the proportion of real positive cases that are correctly predicted as positive. A high recall value accurately captures positive class labels and avoids false negatives. The formulae for the above measures are found in the table 4.

1. Evaluation metrics formulae

| **Evaluation Metrics** | Metric | Formula |
| --- | --- | --- |
| Accuracy | (TP+TN)/(TN+TP+FN+FP) |
| Precision | TP/(TP+FP) |
| Recall | TP/(TP+FN) |

A comparison between an ANN and a RNN is found in table 5 based on the aforementioned evaluation metrics. Precision and accuracy is the exact same for the ANN on both the training and testing sample. This is similar to the RNN. The ANN slightly outperforms the RNN on the training sample. However, the highest instance of accuracy or precision for both neural networks is slightly above 55%. These results are slightly more accurate than a random guess. Very high recall values are observed meaning that the model is very efficient in predicting upward price movements. However, a trade-off is encountered between precision and recall and we observe higher recall as a consequence of low precision.

1. Evaluation metrics formulae

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Performance** | **Accuracy** | | **Precision** | | **Recall** | |
| Train | Test | Train | Test | Train | Test |
| **ANN** | 55.69 | 53.66 | 55.69 | 53.66 | 100 | 100 |
| **RNN** | 55.69 | 51.32 | 55.69 | 51.32 | 100 | 100 |

V. Conclusion

Stock market prices have traditionally fallen under the efficient market hypothesis, in that all readily available information is reflected in the current price. This paper explored the possibility that population perception of current affairs through a reddit community might perhaps reflect market prices. Deep learning, in particular neural networks designed to process text data, has the power to powerfully observe patterns between market sentiment and future prices and provide sizeable returns for investors. The results of this paper did not prove that the experiment would provide sizeable gains however a potential follow-up to this study would be to include a number of further exogenous variables with predictive power for prices and potentially pruning methods for neural networks in order to speed up development time.

REFERENCES

Shleifer, A. and Summers, L.H. (1990). The Noise Trader Approach to Finance. *Journal of Economic Perspectives*, 4(2), pp.19–33. doi:https://doi.org/10.1257/jep.4.2.19.

De Long, J.B., Shleifer, A., Summers, L.H. and Waldmann, R.J. (1990). Noise Trader Risk in Financial Markets. *Journal of Political Economy*, 98(4), pp.703–738.

Brown, G.W. and Cliff, M.T. (2004). Investor sentiment and the near-term stock market. *Journal of Empirical Finance*, 11(1), pp.1–27. doi:https://doi.org/10.1016/j.jempfin.2002.12.001.

LEE, C.M.C., SHLEIFER, A. and THALER, R.H. (1991). Investor Sentiment and the Closed-End Fund Puzzle. *The Journal of Finance*, 46(1), pp.75–109. doi:https://doi.org/10.1111/j.1540-6261.1991.tb03746.x.

‌

Zhang, W. and Skiena, S. (2010). Trading Strategies to Exploit Blog and News Sentiment. 4(1), pp.375–378. doi:https://doi.org/10.1609/icwsm.v4i1.14075.

Katayama, D. and Tsuda, K. (2020). A Method of Using News Sentiment for Stock Investment Strategy. *Procedia Computer Science*, 176, pp.1971–1980. doi:https://doi.org/10.1016/j.procs.2020.09.333.

Kelleher, J.D. (2019). *Deep learning*. Cambridge, Ma: The Mit Press.

Goodfellow, I., Bengio, Y. and Courville, A. (2016). *Deep Learning*. Cambridge, Massachusetts: The Mit Press.

Dahl, G.E., Dong Yu, Li Deng and Acero, A. (2012). Context-Dependent Pre-Trained Deep Neural Networks for Large-Vocabulary Speech Recognition. *IEEE Transactions on Audio, Speech, and Language Processing*, 20(1), pp.30–42. doi:https://doi.org/10.1109/tasl.2011.2134090.

Hinton, G., Deng, L., Yu, D., Dahl, G., Mohamed, A., Jaitly, N., Senior, A., Vanhoucke, V., Nguyen, P., Sainath, T. and Kingsbury, B. (2012). Deep Neural Networks for Acoustic Modeling in Speech Recognition: The Shared Views of Four Research Groups. *IEEE Signal Processing Magazine*, 29(6), pp.82–97. doi:https://doi.org/10.1109/msp.2012.2205597.

Salakhutdinov, R., Mnih, A. and Hinton, G. (2007). Restricted Boltzmann machines for collaborative filtering. *Proceedings of the 24th international conference on Machine learning - ICML ’07*. doi:https://doi.org/10.1145/1273496.1273596.

Cireşan, D.C., Meier, U., Gambardella, L.M. and Schmidhuber, J. (2010). Deep, Big, Simple Neural Nets for Handwritten Digit Recognition. *Neural Computation*, 22(12), pp.3207–3220. doi:https://doi.org/10.1162/neco\_a\_00052.

Zeiler, M.D., Taylor, G.W. and Fergus, R. (2011). *Adaptive deconvolutional networks for mid and high level feature learning*. [online] IEEE Xplore. doi:https://doi.org/10.1109/ICCV.2011.6126474.

Johnson, R. and Zhang, T. (2015). Effective Use of Word Order for Text Categorization with Convolutional Neural Networks. *North American Chapter of the Association for Computational Linguistics*. doi:https://doi.org/10.3115/v1/n15-1011.

Arnold, T.M., McDonald, B. (2011). When Is a Liability Not a Liability? Textual Analysis, Dictionaries, and 10-Ks. *CFA Digest*, 41(2), pp.57–59. doi:https://doi.org/10.2469/dig.v41.n2.20.

Wang, G., Wang, T., Wang, B., Sambasivan, D., Zhang, Z., Zheng, H. and Zhao, B.Y. (2015). Crowds on Wall Street. *Proceedings of the 18th ACM Conference on Computer Supported Cooperative Work & Social Computing - CSCW ’15*. doi:https://doi.org/10.1145/2675133.2675144.

da Silva, N.F.F., Hruschka, E.R. and Hruschka, E.R. (2014). Tweet sentiment analysis with classifier ensembles. *Decision Support Systems*, [online] 66, pp.170–179. doi:https://doi.org/10.1016/j.dss.2014.07.003.

Ishwarappa and Anuradha, J. (2015). A Brief Introduction on Big Data 5Vs Characteristics and Hadoop Technology. *Procedia Computer Science*, [online] 48(48), pp.319–324. doi:https://doi.org/10.1016/j.procs.2015.04.188.

S&P Dow Jones Indices (2023). *S&P 500® - S&P Dow Jones Indices*. [online] www.spglobal.com. Available at: <https://www.spglobal.com/spdji/en/indices/equity/sp-500/#overview>.

Shelar, A., Huang, C-Y. (2018). Sentiment Analysis of Twitter Data. *2018 International Conference on Computational Science and Computational Intelligence (CSCI).* doi: <https://www.researchgate.net/publication/333191309_Sentiment_Analysis_of_Twitter_Data>.

Bajaj, A. (2021). *VADER Sentiment Analysis | NLP Sentiment Analysis Using VADER*. [online] Analytics Vidhya. Available at: <https://www.analyticsvidhya.com/blog/2021/06/vader-for-sentiment-analysis/>.

Lavanya Shukla (2019). *Designing Your Neural Networks*. [online] Medium. Available at: <https://towardsdatascience.com/designing-your-neural-networks-a5e4617027ed>.

Sharma, S., Athaiya, A. (2020). ACTIVATION FUNCTIONS IN NEURAL NETWORKS. *International Journal of Engineering Applied Sciences and Technology*, 4(12), pp.310-316*.* doi:https://www.ijeast.com/papers/310-316,Tesma412,IJEAST.pdf.

Rumelhart, D.E., Hinton, G.E. and Williams, R.J. (1986). Learning representations by back-propagating errors. *Nature*, [online] 323(6088), pp.533–536. doi:https://doi.org/10.1038/323533a0.

L Medsker and L Jain (1999). *Recurrent Neural Networks Design and Applications*. Crc Press.

**contain guidance text for composing and formatting conference papers. Please ensure that all template text is removed from your conference paper prior to submission to the conference. Failure to remove template text from your paper may result in your paper not being publish**