Tribhuvan University

Institute of Science and Technology

Stock Market Forecasting with LSTM and Sentiment Analysis

Dissertation

Submitted to

Central Department of Computer Science and Information Technology

Kirtipur, Kathmandu, Nepal

In partial fulfillment of the requirements for the Master's Degree in Computer Science and Information Technology

<u>Submitted By</u>

Ashish Shrestha

T.U. Registration No.: 5-2-1078-0002-2011

T.U Examination Roll No.: 304/073

January, 2020

<u>Supervisor</u>

Mr. Tej Bahadur Shahi.

Tribhuvan University

Institute of Science and Technology

Central Department of Computer Science and Information Technology

Student's Declaration

I hereby declare that I am the only author of this work and that no sources other than the listed here have been used in this work.

....

Ashish Shrestha

Date: January, 2020.

TRIBHUVAN UNIVERSITY

Central Department of Computer Science and Information Technology

Kirtipur, Kathmandu

Supervisor Recommendation

I hereby recommend that this dissertation prepared under my supervision by **Mr. Ashish Shrestha** entitled **"Stock Market Forecasting with LSTM and Sentiment Analysis"** be accepted as partial fulfillment of the requirements for the degree of M.Sc. in Computer Science and Information Technology. In our best knowledge this is an original work in computer science.

____

Supervisor

Asst. Prof. Tej Bahadur Shahi

Central Department of Computer Science and Information Technology

Tribhuvan University,

Kirtipur, Kathmandu, Nepal

Tribhuvan University

Institute of Science and Technology

Central Department of Computer Science and Information Technology

LETTER OF APPROVAL

This is to certify that we have read this dissertation and in our opinion it is appreciable for the scope and quality as a dissertation in the partial fulfillment for the requirement of Master's Degree in Computer Science and Information Technology.

Evaluation Committee

……………………………………………..	…………………………………………….
Asst. Prof. Nawaraj Paudel	**Asst. Prof. Tej Bahadur Shahi**
Head of Department	Central Department of Computer Science
Central Department of Computer	and Information Technology
Science & Information technology	Tribhuvan University,
Tribhuvan University	Kirtipur, Kathmandu, Nepal
Kirtipur	

……………………………………………..	…………………………………………....
(External Examiner)	**(Internal Examiner)**

Acknowledgment

First of all, I would like to express my sincere gratitude to my respected teacher as well as my dissertation supervisor, **Mr. Tej Bahadur Shahi**, Assistant Professor, Central Department of Computer Science & Information Technology (CDCSIT), Tribhuvan University for his cooperation, encouragement and strong guidelines throughout this thesis work. With his expertise knowledge and ideas, he always provides me necessary guidelines and motivations to tackle the problem raised during preparation of this work.

I am also indebted to the Head of Central Department of Computer Science & Information Technology, **Asst. Prof. Nawaraj Paudel** for his suggestions encouragement, valuable directions and for providing me favorable environment in conducting the research.

I would also like to acknowledge and extend my heartfelt gratitude to everyone for their support and continuous encouragement in this dissertation. Any kind of suggestions or criticism will be highly appreciated and acknowledged.

Abstract

Stock market is an industry where lots of data is generated daily and benefits are reaped on the basis of accurate prediction. Many people invest in stock market having some prediction and more luck. A decision in stock market plays an important role in the investor’s life. Also, stock market is a very complex system and non-linear in nature. So, then it is very difficult to analyse all the impacting factors before making a decision. Making decision with traditional techniques may be time consuming and may not ensure the reliability of the prediction.

Data from stock market is a time series data and different variations of neural networks are widely being used for stock forecasting and prediction problems. Among various architectures of deep neural network, LSTM is one of the design that supports time steps of arbitrary sizes and is free of vanishing gradient problem.

Furthermore, sentiment analysis is becoming popular in predicting the stock market behavior based on investors reactions. This research work studies the usage of LSTM networks on stock market prediction. Also, assuming that news articles have impact on stock market, this is an attempt to study the relationship between news and stock trend.

From the result of the experiment carried out during this research work showed that the financial news do affect the stock market. Sentiment scores from financial news in addition to the stock indices do make the prediction or forecasting process more predictable.

Keywords: *Deep neural network, Deep learning, LSTM, Neural Network ,Stock Market, Sentiment Analysis, Time Series.*

Table of Contents

List of Figures

List of Tables

List of Abbreviations

ADBL	Agricultural Development Bank Limited
API	Application Program Interface
ARCH	Autoregressive Conditional Heteroskedasticity
ARIMA	Autoregressive integrated moving average
BLSTM	Bidirectional Long Short Term Memory
CNN	Convolutional Neural Network
GARCH	Generalized AutoRegressive Conditional Heteroskedasticity
GRU	Gated Recurrent Unit
HAN	Hybrid Attention Network
HAN-SPL	Hybrid Attention Network- Self Paced Learning
IPO	Initial Public Offering
LSTM	Long Short Term Memory
LTP	Last Transaction Price
MAD	Median Absolute Deviation
MAPE	Mean Absolute Percentage Error
MSE	Mean Square Error
RNN	Recurrent Neural Network
RMSProp	Root Mean Square Propagation
SRNN	Simple Recurrent Neural Network
SDA	Stacked Denoising Auto Encoder
VADER	Valence Aware Dictionary and sEntiment Reasoner

CHAPTER 1: INTRODUCTION

1.1. Stock Market

Stock market is a public market for a company to list their stock and to gather financial resources by trading their company stock with an agreed price. The people who buys these stocks are the stockholders who in return will receive a yearly dividend or bonus from the company benefit. Besides these, stockholders can also trade their stocks in the stock market with an agreed price if they want to earn from the price difference of buy and sell activities. There are two types of stock traded; stocks and shares. Both share and stocks are documents issued by a firm that allows its owner to be one of the company's shareholders. Share is released straight by a business via Initial Public Offering (IPO) or can be bought from the stock market. One can receive a part of the profit of the company called dividend by owning a share. Investor also receives capital gain through purchasing and selling the stocks. Stock exchange is a corporation or mutual organization that offers a marketplace (virtual or actual) for stock brokers and traders, trading stocks and other securities. An index is a statistical composite measure of the movement in the overall market or industry that allows measuring the performance of a group of companies over a period of time [1].

1.2. Sentiment Analysis

Sentiment analysis is contextual text mining that identifies and extracts subjective data in source material and helps a company know its brand, product or service social sentiment while tracking internet discussions. Sentiment Analysis is the most popular text classification instrument analyzing an incoming signal and telling if the fundamental feeling is positive, our neutral negative [2].

Analysis of sentiment is the automated method of understanding an opinion from written or spoken language about a specified topic. Sentiment analysis has become a main instrument to make sense of that information. This has enabled businesses to gain important ideas and automate procedures of all kinds [3] .

Sentiment Analysis, also known as Opinion Mining, is a Natural Language Processing (NLP) field that develops systems that attempt to define and extract views in text. Usually, in addition to defining the opinion, these schemes extract characteristics from the expression e.g.: polarity: if the speaker expresses a favorable or negative view, Subject: the matter being discussed, Opinion holder: the individual or entity expressing the opinion.

Analysis of sentiment can be implemented at distinct range levels:

- Document level sentiment analysis gets the feeling of a full document or paragraph.
- Analysis of the sentiment level of phrase gets the feeling of a single phrase.
- Analysis of sentiment at the sub-sentence stage gets the feeling of sub-expressions within a sentence.

There are many kinds sentiment analysis ranging from systems focusing on polarity (positive, negative, neutral) to systems detecting feelings and emotions (angry, glad, sad, etc.) or identifying intentions (e.g. interested v. not interested) [3].

1.3. Recurrent Neural Network

To perform prediction and classification problems based on the complex training data, deep learning comes into an action. It is recent and advance methodology that shows superior performance. Deep neural networks are applied to process signal, recognize speech, predicting stocks and so on. Deep neural networks utilizes a set of computational layers designed to learn patterns from input data. Each layer is employed to extract a specific type of information. The output of certain layer is the input to the succeeding layer. The input data is fed into the input layer and the target output is ultimately generated by the output layer. One of the technique used to construct deep neural network is recurrent neural network (RNN) which was introduced by Hopfield in 1983 [4]. RNNs are a powerful model for processing sequential data such as sound, time series data or written natural languages. An RNN creates cycles in the network graph to maintain an internal state. RNNs can store information while processing a new input as RNNs are the network with loops that allow information to persist. [5]

RNN can perform the same task for every element of a sequence with the output being depended on the previous computations. [5]

RNN don’t only have neural connections in a single direction i.e. neurons can pass data to a previous or the same layer which is differentiating factor that differentiates RNN and feed-forward networks [6].

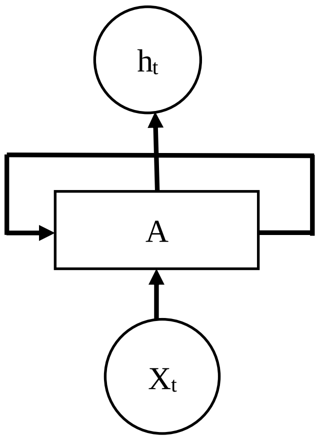


Figure 1 Recurrent Neural Unit

Above diagram shows a chunk of neural network. Here A, looks at some input x_t and outputs a value h_t. A loop allows information to be passed from one step of the network to the next.

A recurrent neural network can be thought of as multiple copies of the same network, each passing a message to a successor [7].

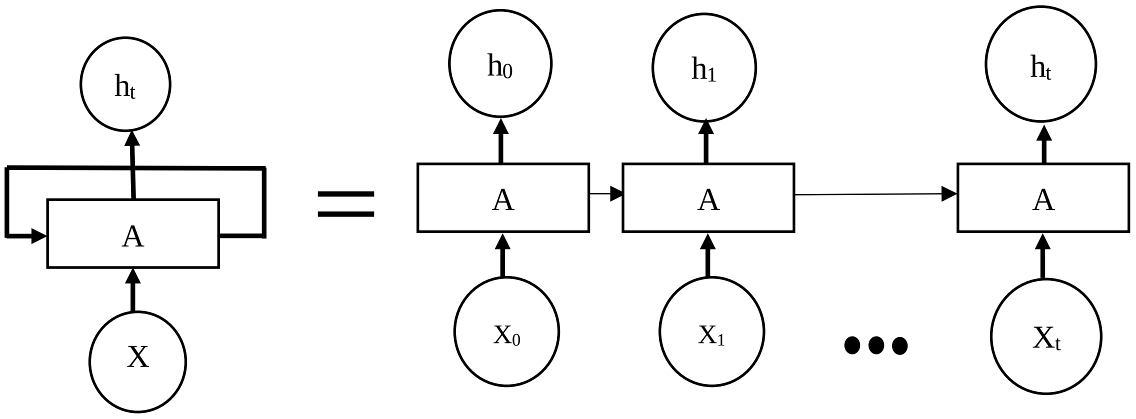


Figure 2 Unrolled recurrent neural network

This chain-like nature reveals that recurrent neural networks are intimately related to sequences and lists. They're the natural architecture of neural network to use for such data.

1.4. Long Short Term Memory

LSTM were invented by Hochreiter and Schmidhuber in 1997 [5]. They can capture context specific temporal dependencies for long periods of time [8]. LSTM are aimed for a better performance by tackling the vanishing gradient issue that recurrent networks would suffer when dealing with long sequence of data. LSTMs are explicitly designed to avoid the long- term dependency problem. Remembering information for long periods of time is practically their default behavior. An LSTM architecture consists of memory cells that can maintain its own state.

The repeating module in an LSTM contains four interacting layers. An LSTM memory cell, has the following three components.

1. **Forget Gate:** The forget gate decides 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 neglected.
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 propagated forward and cell state to the next node in the network [7].

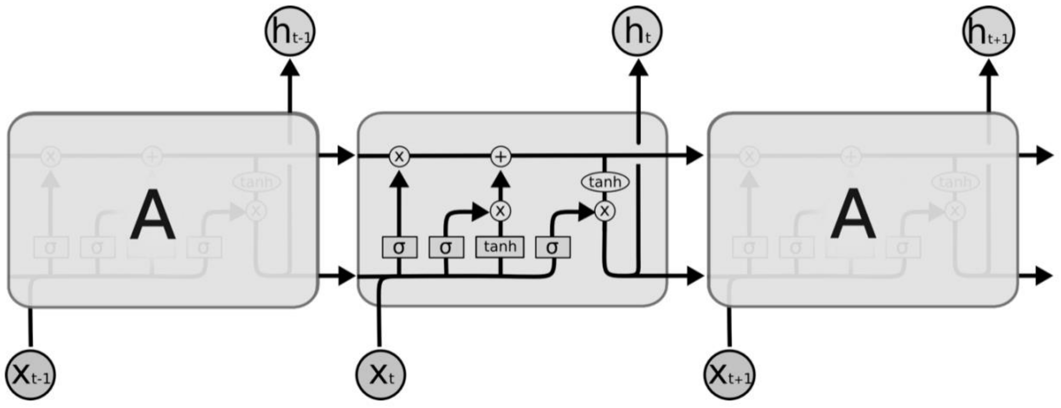


Figure 3 The repeating module in an LSTM contains four interacting layers.

1.5. Problem Statement

Stock price prediction and forecasting is an important topic in forecasting financial time series data. Once the forecasting is successful, we could determine a suitable trading strategy and could get a very high profit. Obtaining accurate prediction of the stock has been challenging and difficult task because of its nonlinear nature and volatile nature. Also, people get information from the financial news and social Medias about the company and their stocks. These news and social Medias are generated in large amount. So in order to carry out the technical analysis it is becoming impossible for an investor or even a group of investors to find out relevant news from the big chunk of news available.

1.6. Objective

The main objectives of this research are given below:

1. To preprocess and analyze stock and financial news data.
2. To study the co-relation between financial news and stock data.
3. To detect trend of stock market using LSTM with the sentiment scores of stock news as a feature input along with stock fundamental indices like opening price, closing price, high, and low.

1.7. Organization of Report

This report is organized in six chapters as follows:

- Chapter 1 introduces stock market, sentiment analysis and LSTM. It also contains the problem statement and objective of the thesis.
- Chapter 2 contains background of the stock market and sentiment analysis and explanation of previous studies and work under literature review.
- Chapter 3 describes the proposed approach and contains discussions about working of VADER and LSTM. Data collection and preprocessing is also described here.
- Chapter 4 explains about the tools and packages used in the experimental. It contains the algorithm for scrapping the data from the website and normalizing the data. It describes about the design of LSTM also.
- Chapter 5 is about the result and its analysis.
- Chapter 6 provides the conclusion and area of enhancement for future work.

Chapter 2: Background and Literature Review

2.1. Background

Stock market is non-linear in nature. In traditional approach there exists two important theories, efficient market hypothesis and random walk theory. According to EMH hypothesis, the future stock price is unpredictable based on the stock historical data. The EMH exits in three forms namely weak EMH, semi-strong EMH and strong EMH. In weak EMH, historical data are used to predict the stock price. In semi-strong EMH, besides historical data all the current public information are used to predict the stock price. In the strong EMH, all the data including historical, public and private information such as insider's information are used to predict the stock price. On the other hand, the random walk theory states that the stock prices do not depend on the past stocks and states that these are not patterns to be exploited since the historical data do not reflect the pattern of the current stock price [1].

According to [9], some of existing methods for stock price forecasting are fundamental analysis, technical analysis and time series analysis forecasting. Fundamental analysis is a type of investment analysis where the share value of a company is estimated by analyzing its sales, earnings, profits and other economic factors. Technical analysis uses the historical price of stocks for identifying the future price. Moving average is a commonly used algorithm for technical analysis. Time series analysis involves two classes of algorithms, linear models and non-linear models. Linear models uses some predefined equations to fit a mathematical models to a univariate time series. They do not account for the latent dynamics existing in the data and consider only univariate data. Non-linear models involves methods like ARCH, GARCH, TAR, Deep learning, The purpose work focuses on the application of deep learning algorithm for stock price forecasting.

Sentiment analysis, or opinion mining, is an active area of study in the field of natural language processing that analyzes people's opinions, sentiments, evaluations, attitudes, and emotions via the computational treatment of subjectivity in text. A gold-standard sentiment lexicon that is especially attuned to microblog-like contexts has been developed and validated. VADER lexicon performs exceptionally well in the social media domain. The correlation coefficient shows that VADER ($r = 0.881$)

performs as well as individual human raters (r = 0.888) at matching ground truth (aggregated group mean from 20 human raters for sentiment intensity of each tweet). Also, on further inspection of the classification accuracy, VADER (F1 = 0.96) outperforms individual human raters (F1 = 0.84) at correctly classifying the sentiment of tweets into positive, neutral, or negative classes [10].

2.2. Literature Review

An LSTM model had been presented in [11] that is used to predict China's stock market in Shanghai and Shenzhen. The data were collected from Yahoo Finance. Dataset contains 7767102 daily records of 3049 stocks from 1990/12/19 to 2015/09/10. The neural network model built had a single input layer, followed by multiple LSTM layers, a dense layer and a single output layers with seven neurons. In the experiment 900000 sequences of daily stock data was used for training purpose and 311361 sequences of data was used for testing and validation purpose. The model was trained by the stochastic gradient descent of mini batch size 64, using RMSprop with the learning rate of 0.001. The experiment was carried out in 6 different methods. In the first method closing price and volume of the stock was used as learning feature. In the second method the learning features were normalized and then fed into the model. In the third method high, low, open, close and volume of the stock were the input features. In the fourth method the input features with 5 extra features from Shanghai Securities Composite Index was used to train the model. In the fifth model only SSE features were used. From the experiment it was found that normalizing the data will increase the accuracy. Also, it can be concluded that different stock set will affect the accuracy of the prediction.

In [12] , had selected three companies from Colombo Stock Exchange that had highest annual share trading volume from twenty sector of CSE. Dataset was from 2002/01/01 to 2013/06/30. For the experiment closing, high and low prices of the past two days were selected as input variables for each company. MLP, SRNN, LSTM and GRU neural network architectures were selected for model building. For each neural network architecture 10 neural network models were developed by varying the number of hidden units from 2 to 11 which in total had 120 models. In each model there were 6 neurons in the input layer, 2 to 11 neurons in the hidden layers and 1 neuron in output layer. Data were preprocessed using min-max normalization. In order to measure the accuracy of each model MAD and MAPE were calculated. From the experiment carried out it was found that LSTM and SRNN networks generally produces lower errors compared to feedforward network in some occasions only. Feedforward network was producing lowest forecasting error most of the time Also, GRU network was producing comparatively higher forecasting errors in the experiment. According to authors LSTM and SRNN would have produced better results if the number of look back is more, wherein the number of look back is only two.

In [4], Bidirectional LSTM (BLSTM) and Stacked LSTM (SLSTM) is studied, evaluated and compared. In BLSTM preceding and succeeding input sequences can be used to exploit all input data in the learning process. In SLSTM several LSTM layers are stacked to perform deep learning. The experiment is focused on the usage of deep learning in the financial time series prediction. Data from Standard & Poor 500 Index (S&P500) for the period from Jan 2010 to Nov 2017 was used in this study. The closing price at the end of every trading day is used for the prediction and evaluating purpose. The data was splitted in the ratio of 4:1 for the experiment. The experiment was conducted in four different network structure. Both BLSTM and SLSTM were designed to experiment with 4, 8, 16, and 32 neurons/memory cells. Also, both architectures were run for short-term prediction and long-term prediction. According to the authors, both the architectures performed well. Authors extended their experiment by comparing previous two architectures with MLP-ANN and LSTM. The performance is evaluated on a benchmark dataset for short-and long term prediction using MAE, RMSE, and r^2.
BLSTM and SLSTM networks produced better performance for predicting short-term prices as opposed to long-term prediction results. Overall, BLSTM network performed well for both long and short-term prediction.

The effect of emotion classification of financial news to the prediction of stock market has been studied in [13]. To find the correlation between sentiment predicted from news and original stock price and to test efficient market hypothesis, the sentiment of two companies Infosys and Wipro over a period of 10 years were plotted. For emotion classification, Naïve Bayes, KNN, and SVM were evaluated.

The comparison between positive sentiment curve and stock price trends revealed co-relation in between them.

In regards to forecasting stock market using sentiment analysis, authors have observed how well the changes in stock prices, the increases and falls are associated with the public views expressed in the company's tweets [14]. The writers used two distinct textual representations to analyze the public semantics in tweets, Word2Vec and N-gram. Twitter API extracts a total of 2, 50,000 tweets over the period from August 31, 2015 to August 25, 2016 on Microsoft. On the basis of the sentiments present, tweets were categorized as positive, negative and neutral. The paper concludes that there is a powerful correlation between a company's stock price rise and fall to the public views or feelings about that firm expressed through tweets on twitter.

In the paper [15], authors analyzed influencing factors of stock market trend prediction and proposed an innovative neural network approach to achieve stock market trend prediction. The study uses two different kinds of datasets i.e. financial news data and stock data. The experiment goes along with feature engineering where Stacked Denoising Auto Encoder (SDA) to reduce the dimension of features which is not sensitive to the noise, Sentiment Analysis for financial news and Stock trend prediction where LSTM is used. The authors concluded that market emotion is a very important factor influencing stock market and can help prediction accuracy.

A method for stock market prediction has been proposed, which adopts LSTM model and incorporates investor sentiment and market factors to improve forecasting performance [16]. Authors investigated stock data of CSI300 index values in a model of 4 layers with 30 nodes at most. Naïve Bayes sentiment classifier was used to classify the sentiments into three classes: positive, negative and neutral. Upon training 90% of the entire data, the model gave prediction accuracy of 87.86% in the rest of 10% of testing data which outperformed the input transformation and SVM methods.

Three machine learning models for forecasting stock prices were designed and implemented in the paper [17]. The models were set in such a way that it forecast same day price for same date, forecast n days ahead with n days date as input and have n days input to predict price of n + 1 date. Social media API StockTwits was used to get the posts and tweets about a company and Quandl API was used to gather daily stock price. Every twits was converted to TF-IDF vector. The results were compared to manually labelled data for sentiment. Naïve Bayes and MLP classifier were used to classify twits where MLP Classifier performed better. In the experiment, the model with the look back performed better than the other twos and conclude that there exists co-relation between the tweets and the stock market.

Authors of [18], used time series model, neural networks, and a combination of neural networks and financial news articles to predict he stock prices. The time series models used in the paper were ARIMA, RNN and Facebook Prophet. Daily stock prices for S&P500 companies were collected which contains stock data for five years. Further, 265,000 financial news were collected that were related to those companies. With the dataset experiment was carried out in which the RNN performed better than ARIMA and Facebook Prophet. The authors concluded that there exists correlation between the textual information and stock price direction.

In the paper [19], the authors had collected Chinese stock data from 2014 to 2017 in daily frequency of major Chinese stocks. Similarly, 1271442 economic news from 2014 to 2017 were extracted. A total of 425,250 data were then co-related with the stock data which was used in the experiment. With the principle of sequential content dependency, diverse influence, and effective and efficient learning, HAN-SPL model was proposed. The proposed model was compared with Random forest, multi-layer perceptron, news-RNN, One-Direction GRU-based RNN, Temporal-Attention-RNN, News-Attention-RNN, and HAN. The purposed model HAN-SPL achieved the best accuracy

Chapter 3: Research Methodology

3.1. Proposed Approach

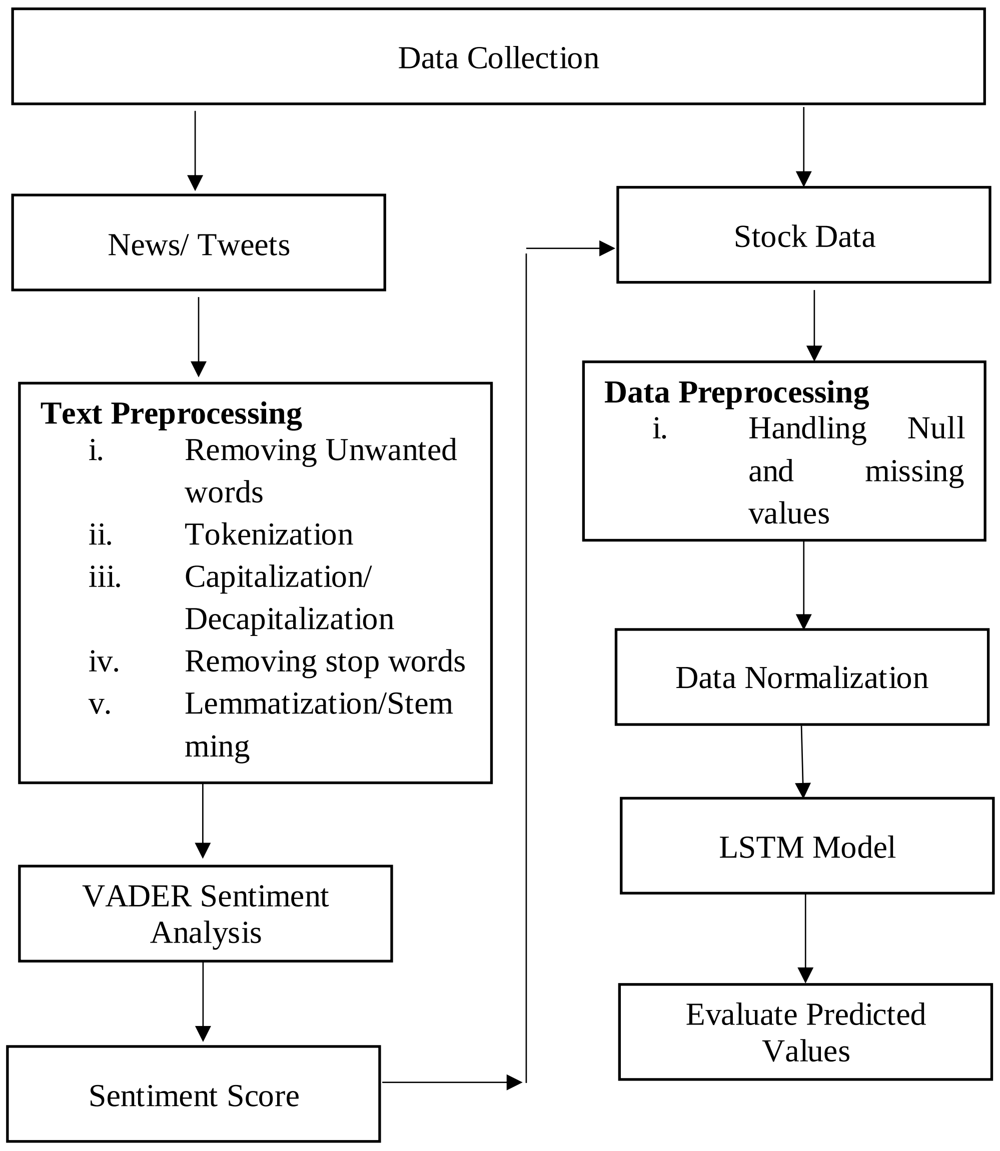


Figure 4 Proposed system design

3.2. Discussion

3.2.1. VADER Sentiment Scoring

In order to find the polarity of the sentence, sentiment scores is calculated. For this VADER sentiment analysis will be used. VADER sentiment analysis returns a sentiment score in the range of -1 to 1 for the most negative to most positive.

VADER is a lexicon and rule-based sentiment analysis tool that is specifically attuned to sentiments expressed in social media. VADER uses a combination of a sentiment lexicon is a list of lexical features (e.g., words) which are generally labeled according to their semantic orientation as either positive or negative. VADER not only tells about the Positivity and Negativity score but also tells us about how positive or negative a sentiment is.

VADER belongs to a type of sentiment analysis that is based on lexicons of sentiment-related words. In this approach, each of the words in the lexicon is rated as to whether it is positive or negative, and in many cases, how positive or negative.

VADER produces four sentiment metrics from word ratings. The first three, positive, neutral and negative, represent the proportion of the text that falls into those categories. The final metric, the compound score, is the sum of all of the lexicon ratings which have been standardized to range between -1 and 1. The sentiment score for a sentence is calculated by summing up the sentiment scores of each stemmed tokens. The scores for the stemmed tokens is obtained from the VADER-dictionary-listed word as VADER sentiment analysis relies on a dictionary which maps lexical features to emotion intensities i.e. sentiment scores. The creators of VADER sentiment have enlisted not just one, but a range of human raters, averaging their scores per each word realizing the concept of wisdom of the crowd: collective opinion is oftentimes more trustworthy than individual opinion [10] .

Sentiment scores are then normalized the get the value between -1 to 1. The normalization used by Hutto is

$$\frac{x}{\sqrt{x^2 + \alpha}}$$

where, where x is the sum of the sentiment scores of the constituent words of the sentence and α is a normalization parameter.

3.2.2. Working of LSTM

The role of forget gate begins the working of LSTM network. It decides what information is going to be thrown away from the cell state. The sigmoid layer helps in the decision process. The inputs to forget gate are h_{t-1} and x_t. The output from forget gate is a number between 0 and 1 for each number in the cell state C_{t-1}. The output 1 means to store the information while output 0 means to forget the information.

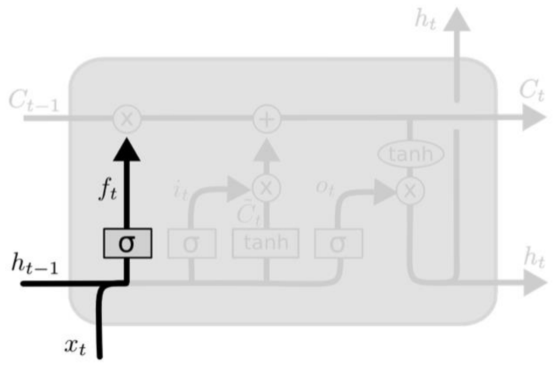


Figure 5 Forget gate in LSTM

$$f_t = \sigma\left(W_f . [h_{t-1}, x_t] + b_f\right)$$

In second step of working of LSTM, a decision is made about what new information are going to be stored in the cell state. Decision process is carried with the help of two layers input gate layer and tanh layer. Input gate layer is a sigmoid layer that decides which values will be updated. Tanh layer creates a vector of new candidate values $\tilde{C}_t$ that could be added to the state.

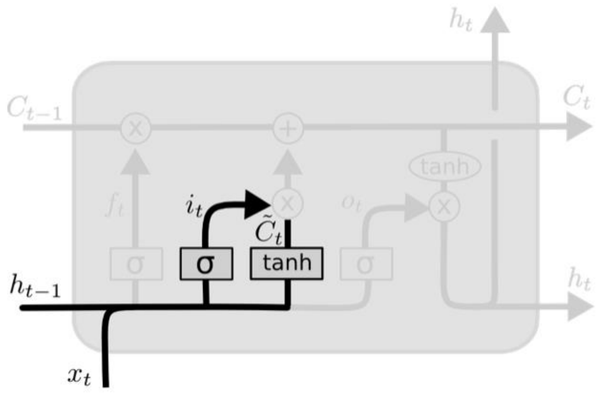


Figure 6 Decision made for Input in Input Gate

$$i_t = \sigma(W_i . [h_{t-1}, x_t] + b_i)$$
$$\tilde{C} = \tanh(W_C . [h_{t-1}, x_t] + b_C$$

The old cell state C_{t-1} is updated into new cell state $\tilde{C}_t$. Previous two steps have already decided what to do. In this step the implementation is carried out. The old state is multiplied by ft forgetting the things that is decided to forget in the earlier step. The result is then added with $i_t * \tilde{C}_t$ which gives the new candidate values that is being scaled.

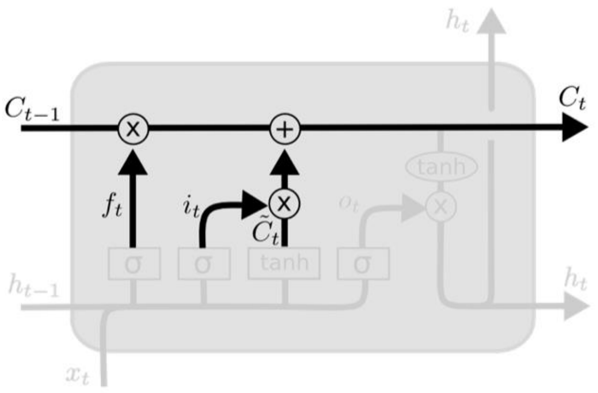


Figure 7 Implementation of Input gate

$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$$

This output will be based on our cell state, but will be a filtered version. First, we run a sigmoid layer which decides what parts of the cell state we're going to output. Then, we put the cell state through tanh (to push the values to be between −1 and 1) and multiply it by the output of the sigmoid gate, so that we only output the parts we decided to [7].

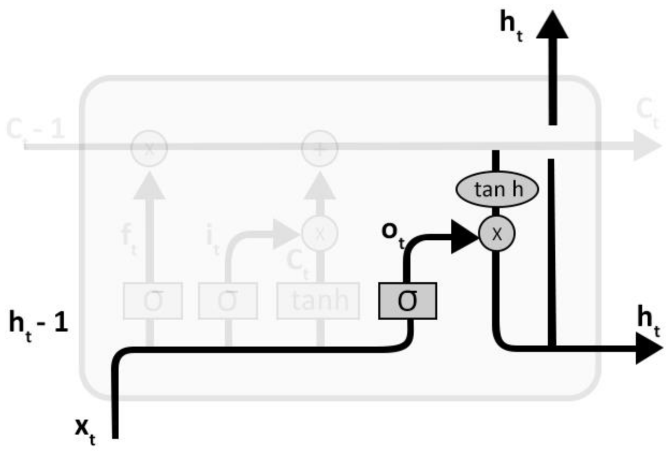


Figure 8 Output Gate

$$o_t = \sigma(W_0 . [h_{t-1}, x_t] + b_0)$$
$$h_t = o_t * \tanh(C_t)$$

3.3. Data Collection

For the purpose of this research, financial news data and stock data are needed. In order to collect the data a python script was created that crawls the website shareshansar.com.

From the script, a total of 42110 financial new headlines from 17th March 2011 to 15th November 2019. All news headlines was stored in a plain utf-8 csv file sorted by the date of publication.

Table 1 Financial news data set from BeautifulSoup

Date	**Title**
Friday, 15 November, 2019	Nepal Insurance Company posts net profit of Rs.4.91 Crore till Q1 of FY 76/77; insurance fund crosses Rs.30 Crore
Friday, 15 November, 2019	Look at the major highlights from the first quarter reports of three development banks: Green, Tinau Mission and Sahayogi Bikas Bank Limited
Friday, 15 November, 2019	Sanima Life Insurance and Mahalaxmi Life Insurance publish unaudited first quarter reports
Friday, 15 November, 2019	NMB Bank was traded the most this week; Sabaiko Laghubitta gains 24.83%; both total turnover and transaction rises.
Friday, 15 November, 2019	Ganapati Microfinance announces 3rd AGM on Poush 4, 2076; to endorse distribution of 50% right shares after bonus adjustment

Also, a total of 1996 stock data of ADBL from 20th March 2011 to 14th November 2019 was crawled from Shareshansar's website. The dataset contains the sensitive indices like Open, High, Low, LTP, Change, Quantity and Turnover. The data was stored in a plain utf-8 csv file.

Table 2 Sample of Stock dataset obtained from BeautifulSoup

Date	**Open**	**High**	**Low**	**LTP**	**Change**	**Quantity**	**Turnover**
11/14/19	417	417	415	417	0.24	18,775	7,779,457
11/13/19	414	417	413	416	-0.24	23,563	9,781,256
11/12/19	425	425	417	417	-0.95	12,550	5,243,574
11/10/19	414	419	410	419	0.96	11,958	1,494,243
11/07/19	417	417	414	415	0	18,305	7,596,806
11/06/19	415	417	415	415	-0.24	16,754	6,966,192
11/05/19	415	414	413	416	0.73	12,478	5,166,249
11/04/19	414	414	414	410	0.24	13,907	5,720,752

3.4. Data preparation

The news headlines obtained from scraping were clean text. It was free from html tags and escape sequences. As the punctuation marks like exclamation marks and multiple question marks represents the emotion and strengths in the sentence, punctuation marks were not removed. In order calculate, the sentiment scores, VADER sentiment scoring was applied to each news headlines of news dataset. The sentiment scores were placed into a new field named “Score” and saved into a csv file. The date field in the news dataset was converted to the format mm/dd/YYYY to match the date format with stock dataset.

The fields “Quantity “and “Turnover” in stock dataset had commas to represent the numeric positions. For the purpose of research, the comma was removed and represented the value as numeric value.

The news data was grouped by Date field and average sentiment score was calculated for the particular date. The stock data and sentiment scores were then combined in correspondence to the date.

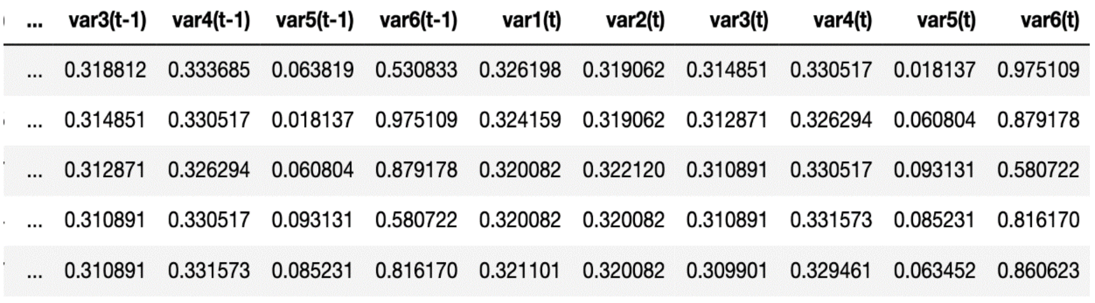
The final dataset for the research looks like as:

Table 3 Final dataset after merging the sentiment score with fundamental indices

Date	LTP	Open	High	Low	Quantity	Score
11/14/19	417	417	417	415	18775	0.41083667
11/13/19	416	414	417	413	23563	0.34688611
11/12/19	417	425	425	417	12550	0.1764
11/10/19	421	414	421	414	3581	0.52365455
11/7/19	419	414	419	410	11958	0.41083667

Time series forecasting problems need to be re-framed as supervised learning problems before machine learning can be used. They must be converted to pairs of input and output from a sequence type. The dataset is reframed using the shift function from the pandas library which creates a copies of columns that are pushed forward or pulled back. This behavior creates the columns of lag observations as well as columns of forecast observations for a time series dataset.

Re-framed dataset for the lag value of 3 can be visualized as:



...	var3(t-1)	var4(t-1)	var5(t-1)	var6(t-1)	var1(t)	var2(t)	var3(t)	var4(t)	var5(t)	var6(t)
...	0.318812	0.333685	0.063819	0.530833	0.326198	0.319062	0.314851	0.330517	0.018137	0.975109
...	0.314851	0.330517	0.018137	0.975109	0.324159	0.319062	0.312871	0.326294	0.060804	0.879178
...	0.312871	0.326294	0.060804	0.879178	0.320082	0.322120	0.310891	0.330517	0.093131	0.580722
...	0.310891	0.330517	0.093131	0.580722	0.320082	0.320082	0.310891	0.331573	0.085231	0.816170
...	0.310891	0.331573	0.085231	0.816170	0.321101	0.320082	0.309901	0.329461	0.063452	0.860623

Figure 9 Top 5 reframed dataset for the lag value of 3

Chapter 4: Implementation

4.1. Tools and packages used

The experiment is being implemented in Python 3.6 with the help of packages of deep learning. The scripts are executed in a laptop having 8GB of RAM and core i5 processor. The packages that are used during the experiment are as follows:

4.1.1. Python

Python is a high-level, interpreted scripting language developed by Guido van Rossum in the late 1980s at the Netherlands National Institute of Mathematics and Computer Science. There are two major Python versions available till date- Python 2 and Python 3. Python is the primary programming language used for much of the research and development in Machine Learning. Some features that made Python so popular and best suited for machine learning are as follows:

- Python is easy to learn.
- Python has multiple libraries and framework for machine learning.
- Python has community and corporate support.
- Python is portable and extensible.

4.1.2. Jupyter notebook

The Jupyter Notebook is an open-source web application that enables to create and share live code, calculations, visualizations, and narrative text documents extending the console-based approach to interactive computing in a qualitatively new direction. Jupyter notebook is widely used in data cleaning and transformation, numerical simulation, statistical modeling, data visualization, machine learning and much more.

It combines two components a web application and notebook document. Web application combines explanatory text, mathematics, computations and rich media output. Notebook documents on the other hand is a representation of all contents in the web application that includes computational inputs and outputs, mathematics and images.

4.1.3. Keras

Keras is a Python-based high-level neural network API that can run on top of TensorFlow, CNTK, or Theano. It has been designed to allow quick experimentation. According to Keras, it is necessary to do good research to be able to go from idea to outcome with the least possible delay. Keras allows easy and fast prototyping through user friendliness, modularity and extensibility. Also, Keras supports convolutional networks, recurrent networks and combination of both.

4.2. Algorithm

4.2.1. Scrapping financial news using BeautifulSoup

Input: A URL to financial news from Shareshansar and page index

Output: A list of data.

Step 1: Set page index = 1

Step 2: Send a HTTP request to the specified URL.

Step 2: Searching and navigating through the parse tree

Step 3: Save the response from server in a response object ‘r’.

Step 4: Zipping the list of response.

Step 5: Convert list to data frame.

Step 6: Convert data frame to csv file.

4.2.2. Normalization

Step1: Subtract min(x) from each value in order to Change the data in order to have a lower limit of 0.

$$x - \min(x)$$

Step 2: Change the data in order to have an upper limit of 1.

$$\frac{x}{\max(x) - \min(x)}$$

Step 3: Finally, combine these two steps we get:

$$\frac{x - \min(x)}{\max(x) - \min(x)}$$

4.2.3. LSTM stock prediction algorithm

Input: Historical stock price data with sentiment score.

Output: Predicted value of LTP

Step 1: Dataset with fundamental indices and sentiment scores is stored in a numpy array of dimension 3 as (N, W, and F) where, N is number of training sequences, W is sequence length and F is the number of features of each sequence.

Step 2: A network of structure is built with [1, a,1] dimensions where there is one input layer, a neuron in the next layer and a single layer with a tanh activation function.

Step 3: Train the constructed network on the data.

Step 4: Use the output of the last layer as prediction of the next time step.

Step 5: Repeat step 3 and 4 until optimal convergence is reached.

Step 6: Obtain predictions by providing test data as input to the network.

Step 7: Evaluate accuracy by comparing predictions made with actual data.

4.3. Data Exploration

A total of 1977 data was available after merging the scores to stock data. Exploring the dataset, it was found that there are 1786 positive, 190 negative news and 1 neutral news headline.

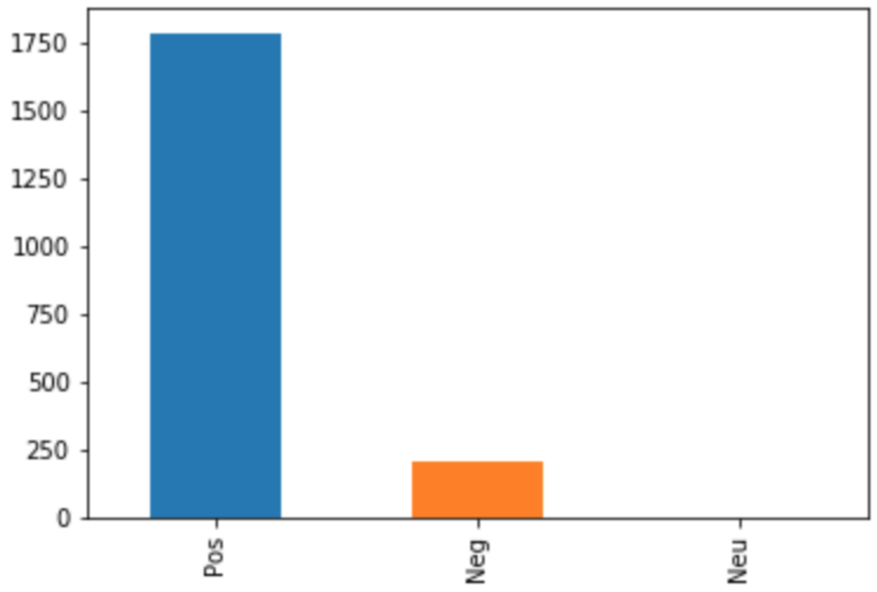


Figure 10 Bar graph of negative and positive news

In order to perform regression the variables must be co-related. So, the dataset was explored to find some correlation between the features. Upon exploring the correlation matrix presented below, it was found that there is correlation between the sentiment score and sensitive indices of the stock data which is good for further process.

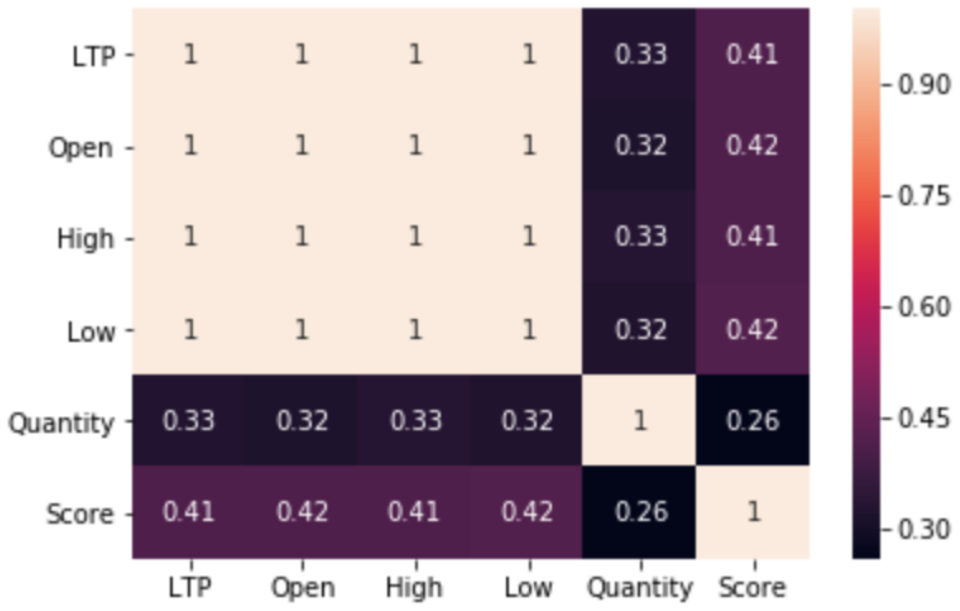


Figure 11 Heatmap representation of correlation matrix of dataset

4.4. Data Normalization

The main objective of data normalization is to change our observations so that they can be described as a normal distribution. Normal distribution also, known as Gaussian distribution of bell curve is a specific statistical distribution where a roughly equal observations fall above and below the mean, the mean and the median are the same, and there are more observations closer to the mean.

For data normalization min-max normalization technique is being used. Min-Max transforms features by scaling each feature to a given range. The features are rescaled within the range of [0, 1].

The Min-Max transformation can be achieved by the following formula

$$X_{Norm} = \frac{X - X_{min}}{X_{\max} - X_{min}}$$

4.5. Selection of Input and output variables.

The inputs or the primary variables of the dataset are “Open”, “High”, “Low”, “Quantity” and “Score”. The experiment was carried out with and without the sentiment score. Using these features variables closing price i.e. LTP is predicted.

Data has been divided into training set and test set. Out of 1977 records, 1095 data was used for training purpose and 882 data was used for testing purpose.

For model building LTP, High, Low, Open and Quantity of past t days were selected as input variables with the output variable LTP. Accordingly, the input variables are as follows where t is the current time:

Table 4 Input and Output Variables

Input Variable	Open (t-1), Open (t-2), …, Open(t-n)
	High (t-1), High (t-2), …, High(t-n)
	Low(t-1), Low(t-2), …, Low(t-n)
	Quantity(t-1), Quantity(t-2), …, Quantity(t-n)
	Score(t-1), Score(t-2),…,Score(t-n)
Output Variable	LTP(t-1), LTP(t-2), … ,LTP(t-n)

4.6. Modelling LSTM

In the experiment, an LSTM model is composed of one single input layer, followed by a LSTM layers, following a dropout layer and finally a dense output layer. The first layer i.e. input layer contains number of memory units that is equal to the number of input features. The LSTM layers consists of 120 memory units. The activation function used in each LSTM layer is hyperbolic tangent (tanh).

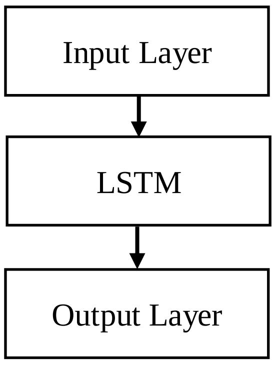


Figure 12 LSTM model used in the experiment.

In order to prevent overfit and underfit of the training dataset due to too many or too few epochs, Early stopping method is being implemented in the designed model. Early stopping is a method that allows to specify a large arbitrary number of training epochs and stop training once the performance of the model has stopped improving on a validation dataset.

The trained data is splitted into train and validation set, thus enabling the model to be validated during the training of each epoch. The loss function used in the model is MSE. MSE is calculated at the end of each epoch. The training process is stopped if there is no improvement or no changes in the loss function value, i.e. MSE in the designed model. The first sign of no further improvement may not be the best time to stop training as the model may coast into a plateau of no improvement or even get slightly worse before getting much better. In order to account this situation, a delay to the trigger in terms of the number of epochs on which there is no improvement was set. The delay was set to 10 in the designed model.

According to [20] , Adaptive Moment Estimation (Adam) is a method that computes adaptive learning rates for each parameter. Adam is a method for efficient stochastic optimization that only require first-order gradients with little memory requirement. Adam algorithms combines the heuristic of momentum and RMSProp. Adam keeps an exponentially decaying average of past gradients. Adam compute the decaying averages of past and past squared gradients v_t and s_t respectively as follows:

$$v_t = \beta_1 v_{t-1} + (1-\beta_1) g_t$$

$$s_t = \beta_2 s_{t-1} + (1-\beta_2) g_t^2$$

Where, v_t = exponential average of gradients along w_j

s_t = exponential average of squares of gradients along w_j

g_t = Gradient at time t along w_j

β_1, β_2 = Hyper parameters

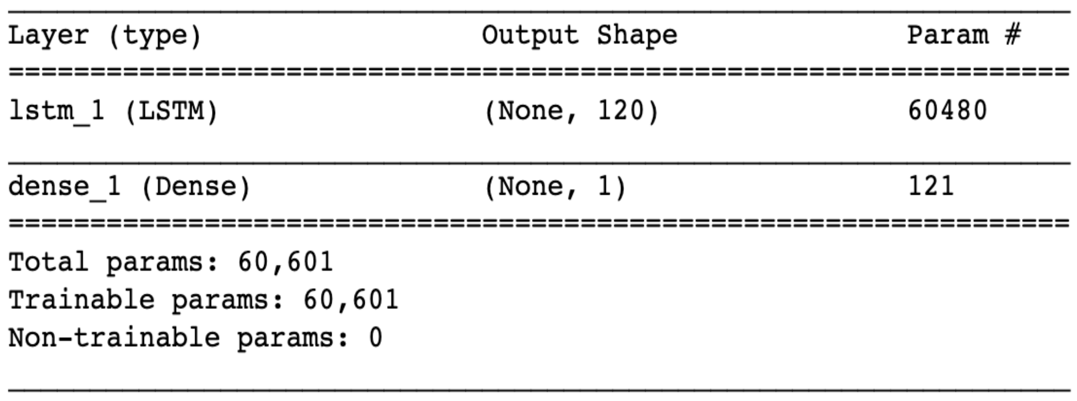
$$\Delta w_t = -\eta \frac{v_t}{\sqrt{s_t + \varepsilon}} * g_t$$

$$w_{t+1} = w_t + \Delta w_t$$

Where, η = Initial learning rate

According to Adam's author the hyper parameter β_1 is generally kept around 0.9 while β_2 is kept at 0.99. Epsilon is chosen to be e^{-10} generally.

The model summary of designed LSTM can be viewed as:



```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
lstm_1 (LSTM)                (None, 120)               60480
_________________________________________________________________
dense_1 (Dense)              (None, 1)                 121
=================================================================
Total params: 60,601
Trainable params: 60,601
Non-trainable params: 0
_________________________________________________________________
```

Figure 13 Model Summary of designed LSTM model.

4.7. Evaluation Metrics

4.7.1. Mean Absolute Error (MAE):

MAE is a loss function used for regression models. MAE is the sum of the absolute differences between the predicted and the target variables. Thus, it measures the average magnitude of errors in a set of predictions, without considering their directions. If the direction of the error is also considered then, it is called Mean Bias Error (MBE). The range of MAE is 0 to ∞ [21].

$$MAE = \frac{1}{n}\sum_{j=1}^{n}\frac{\sum_{1}^{n} |\,(\hat{y_i} - y_i)|}{n}$$

4.7.2. Root Mean Squared Error (RMSE):

RMSE is the square root of the mean of the squared error. It measures the average magnitude of the error. RMSE is thus the distance, on average, of a data point from the fitted line, measured along a vertical line [21].

$$RMSE = \sqrt{\frac{1}{n}\sum_{j=1}^{n}(y_i - \hat{y_j})^2}$$

4.7.3. Coefficient of determination (r^2):

Coefficient of determination gives an idea of how many data points fall within the results of the line formed by the regression equation. The higher the coefficient, the higher percentage of points the line passes through when the data points and line are plotted. A higher coefficient is an indicator of a better goodness of fit for the observations.

Coefficient of determination can be negative which means that the model is poor fit for the data. r^2 is able to find the likelihood of future events falling within the predicted outcomes [22].

$$R^2 = \frac{n(\sum xy) - \sum x \sum y}{\sqrt{[n\sum x^2 - (\sum x)^2][n\sum y^2 - (\sum y)^2]}}$$

Chapter 5: Result, Analysis and Discussion

The designed LSTM model was executed number of lag values with epoch value of 100 and batch size 30. The experiment had two types of input. In the first input set I, fundamental indices and sentiment scores were used while in the second set II only fundamental indices were used. These two set of input were used to train the LSTM model. During the course of experiment different lag values were given as input and root mean square error, mean absolute error and coefficient of determination were noted down, which is as follows:

Table 5 Comparison of RMSE, MAE and r^2 for the result with input set I and II in LSTM model

Lag value	**Input Set I**				**Input Set II**			
	RMSE	**MAE**	**r^2**	**Time (s)**	**RMSE**	**MAE**	**r^2**	**Time (s)**
10	20.984	16.856	0.983	17	59.186	55.07	0.868	6.73
12	20.93	15.699	0.983	17.64	57.731	53.73	0.875	8.53
14	24.563	19.334	0.977	17.32	54.092	49.4	0.889	8.47
16	21.927	16.575	0.982	29	58.051	53.25	0.872	8.75
18	22.793	16.386	0.98	23.03	51.153	46.39	0.9	9.16
20	27.225	21.289	0.97	24.05	37.387	32.95	0.947	10
Avg	**23.070**	**17.689**	**0.979**	**21.34**	**52.933**	**48.47**	**0.892**	**8.61**

From the above table, it can be seen that average value for coefficient of co-relation (r^2) for input set I is 9.79 and for input set II it is 0.9. Also, average values for RMSE and MAE for input set I is lesser than that of input set II. Also, with the lag value of 12 the model with input set I is performing with least value of MAE and RMSE and highest value of r^2. In an account to the lag value of 12 in the LSTM model the actual and predicted values can be viewed as:

Table 6 First 20 records from (6th Jan 2015 to 4th Feb 2015) representing error in the output of LSTM with lag value 12

S.N.	**Actual**	**Predicted**	**Error**	**Mean Relative Error**
1	480	480.7585	-0.75851	0%
2	483	483.0069	-0.0069	0%
3	487	480.1356	6.86441	1%
4	493	480.4436	12.55643	3%
5	500	481.987	18.013	4%
6	509	479.8465	29.15353	6%
7	503	493.1748	9.825195	2%
8	481	492.5785	-11.5785	2%
9	475	494.3567	-19.3567	4%
10	473	506.2545	-33.2545	7%
11	478	490.8952	-12.8951	3%
12	487	483.9038	3.096222	1%
13	475	483.1191	-8.11914	2%
14	484	479.4975	4.502533	1%
15	475	478.6971	-3.69711	1%
16	473	488.2065	-15.2065	3%
17	484	489.6842	-5.68414	1%
18	473	476.4268	-3.42676	1%
19	476	474.2032	1.796783	0%
20	475	474.0017	0.998322	0%

The mean error for the result obtained is -1.36. The above information can be pictorially represented as:

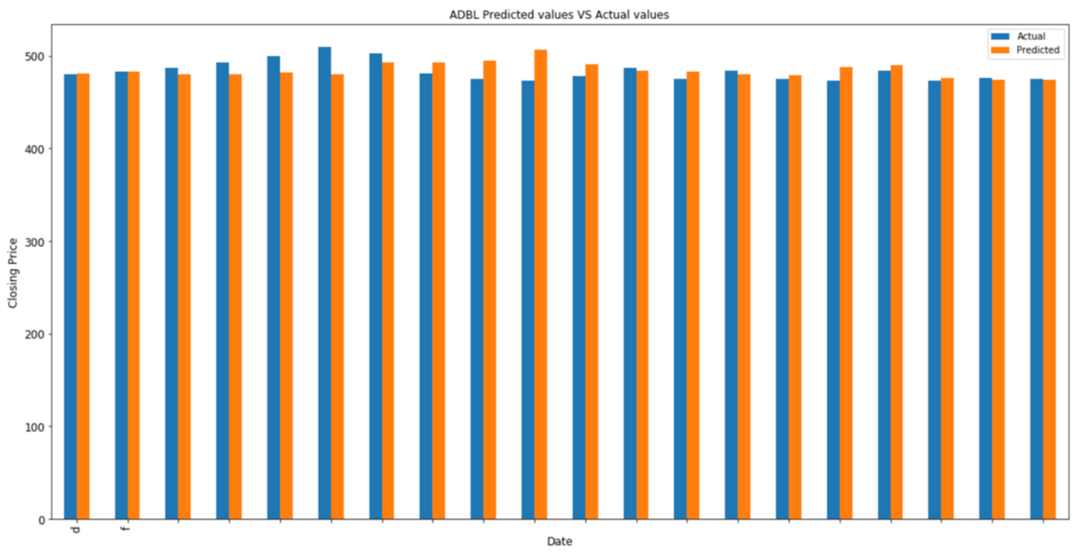


Figure 14 Bar graph showing actual and predicted value of first twenty records from (6th Jan 2015 to 4th Feb 2015)

The scatter plot for first 40 records can be viewed as

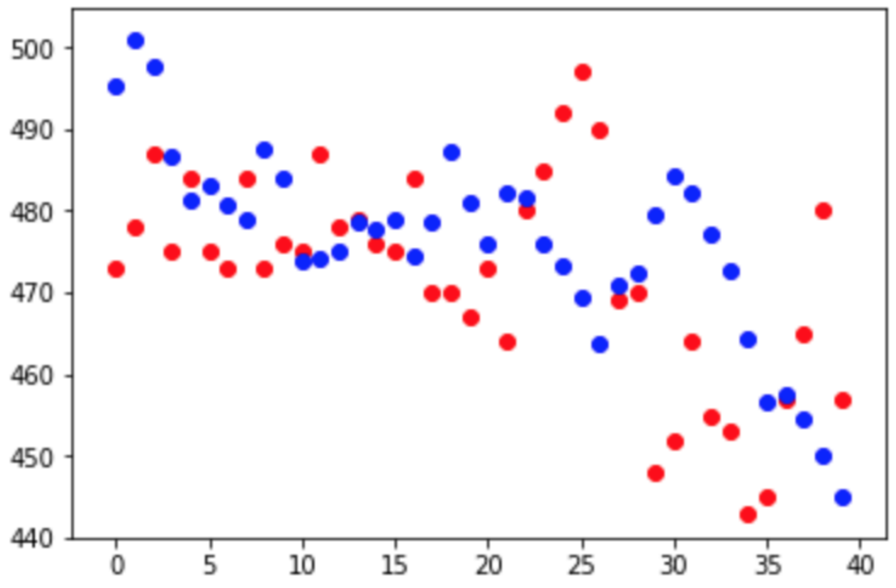


Figure 15. Scatter plot for first 40 records from result of LSTM model with look back value 12

In the above figure the red dots represents the actual LTP and the blue dots represents the predicted LTP for ADBL which is obtained from the trained LSTM model. From the above scatter plot it can be shown that the model is trying to learn and predict LTP for stock of ADBL.

Scatter plot for the result from the trained LSTM model can be viewed as:

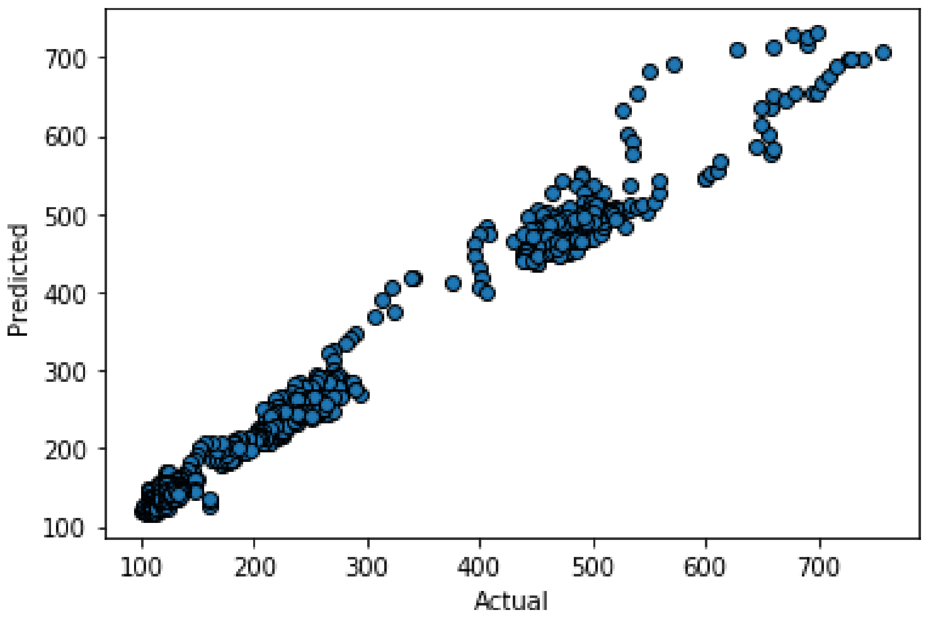


Figure 16 Scatter plot diagram for LTP of ADBL.

The distribution plot for the actual and predicted values can be viewed as:

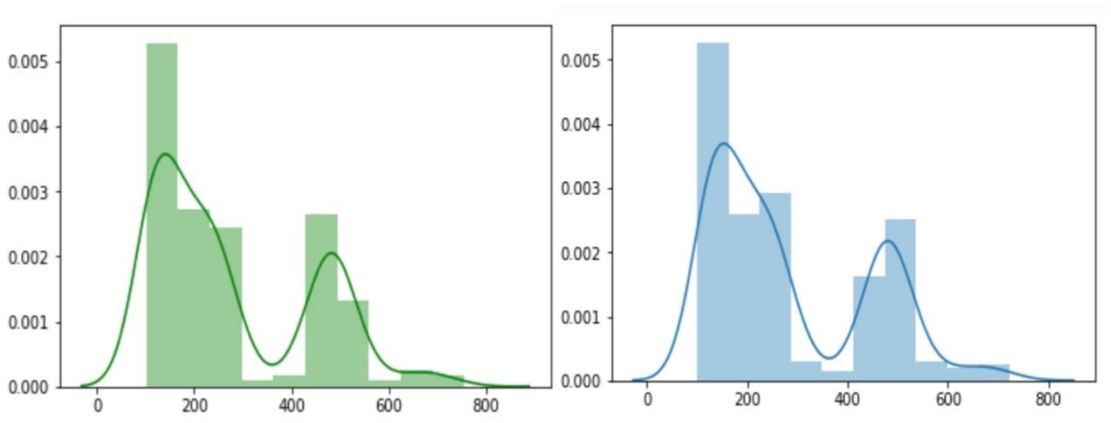


Figure 17 Distribution plot for actual (green) and predicted (blue) values of LTP of ADBL from LSTM model

Since GRU is a variation of LSTM, the experiment was extended comparing the results from GRU and LSTM model. GRU model was designed similar to that of LSTM model. Same input sets i.e. Input set I and II were fed into GRU with same lag values used in LSTM model. The results from GRU can be viewed as:

Table 7 Comparison of RMSE, MAE and r^2 for the result with input set I and II in GRU model

Lag value	**Input Set I**				**Input Set II**			
	RMSE	**MAE**	**r^2**	**Time (s)**	**RMSE**	**MAE**	**r^2**	**Time (s)**
10	26.222	22.233	0.974	12	44.234	39.52	0.926	10
12	26.562	22.088	0.973	12.84	53.645	49.61	0.891	6.61
14	34.346	29.797	0.955	10	45.216	40.45	0.923	14.2
16	27.907	23.102	0.97	13.97	43.877	39.35	0.927	15.4
18	27.666	22.491	0.971	15.16	53.103	48.8	0.893	8.28
20	32.216	27.124	0.96	17.85	43.784	39.16	0.927	16.6
Avg	29.153	24.473	0.967	13.64	47.31	42.81	0.915	11.9

Comparing the results of GRU with input set I and II, it can be seen that GRU model performed better with the input set I i.e. having sentiment score as one of its feature. The average value of r^2 is greater and the average value of RMSE and MAE is lower in case of GRU model trained with input set I which is similar to the results from the LSTM model.

The average value for RMSE and MAE from LSTM model is lesser than that of GRU model. Similarly, the average value of r^2 for LSTM model is greater than the average value of r^2 for GRU model. Comparing these results for input set I it can be concluded that LSTM model performed better than GRU. But if comparison is made in terms of training time between GRU and LSTM, GRU's average training time is minimum. For GRU and LSTM model with input set II, the average values for evaluation metrics are similar but the average training time is greater for GRU model.

From the co-relation matrix presented above, we can see that there is high correlation between fundamental indices and LTP and comparatively lower correlation between LTP and sentiment score. Analyzing the results, it can be concluded that, GRU's training time is least if the dataset is not highly co-related and in this condition the results from LSTM is better.

Chapter 6: Conclusion and Future Enhancements

6.1. Conclusion

In this research work, the closing price or LTP of ADBL stock is predicted using LSTM. Two variations of input feature set was used in the experiment. One of the input set contains the fundamental indices of stock market and in another set, compound sentiment scores of financial news is appended to the first feature set. The LSTM model was trained with the look back values of 10, 12, 14, 16, 18, and 20. In order to evaluate the performance of built model, RMSE, MAE and r^2 were calculated.

From the correlation matrix obtained during the data exploration it was found that there is some correlation between the sentiment score of the financial news and the fundamental index LTP of the stock price.

In the experiment, the average RMSE and MAE value were 23.0703 and 17.689 respectively for the input set with sentiment score whereas the average RMSE and MAE value were 52.933 and 48.47 respectively for the input set with fundamental indices only. Similarly, the average value for r^2 was greater for the input set with sentiment scores. In order to have, better prediction in the regression model, the values of RMSE and MAE must be less and closer to zero and r^2 must be greater and closer to 1. The curves on distribution plot for actual and predicted LTP, doesn't have vast difference and looks similar as the mean error is -1.36. Thus, it can be concluded that, the sentiment scores obtained from the financial news do affect the stock market. Sentiment scores from financial news in addition to the stock indices do make the prediction or forecasting process more predictable.

6.2. Future Enhancements

The experiment was conducted in general computing device with 8 GB RAM and core i5 CPU with a single layer of LSTM with 120 memory units. The hyper parameters like number of epochs, and batch-size were set to default values from keras and no optimization techniques have been implemented. So, there can be made some enhancement for further studies like:

1. GPU environment with large memory and strong processor can be used.
2. Tuning up the hyper parameters with regularization can be done.

3. Multiple hidden layers can be added.
4. Nepali news can be used for sentiment analysis.
5. For validation of the model, it can be compared with the gold standard set of data from Google, Tesla, and so on.
6. Statistical analysis can be carried out to test the normality of the dataset during dataset exploration.
7. Decision Support System can be integrated for the usage of the forecasting model.

References

[1] S. V. Chang, K. S. Gan, K. O. Chin, A. Rayner and A. Patricia, "A Review of Stock Market Prediction with Artificial Neural Network," International Conference on Control System, Computing and Engineering, Penang, Malaysia, 2013.

[2] S. Gupta, "Towards data Science," 07 01 2018. [Online]. Available: https://towardsdatascience.com/sentiment-analysis-concept-analysis-and-applications-6c94d6f58c17. [Accessed 29 08 2019].

[3] MonkeyLearn, "Sentiment Analysis Nearly Everything You Need to Know," MonkeyLearn, 2019. [Online]. Available: https://monkeylearn.com/sentiment-analysis/. [Accessed 29 08 2019].

[4] K. Althelaya, S. El and S. Mohammed, "Evaluation of Bidirectional LSTM for Short- and Long-Term Stock Market Prediction," IEEE, Saudi Arabia, 2018.

[5] J. Chawalit, C. Rujira and T. Laksmey, "Stock Price Prediction With Long Short-Term Memory Recurrent Neural Network," IEEE.

[6] D. M. Q. Nelson, A. C. M. Pereira and R. A. de Oliveira, "Stock Market's Prediction Price Movement With LSTM Neural Networks," IEEE, Brazil, 2017.

[7] C. Olah, "Understanding LSTM Networks," 15 08 2015. [Online]. Available: http://colah.github.io/posts/2015-08-Understanding-LSTMs/. [Accessed 29 08 2019].

[8] R. Nandakumar, U. K. R and Y. V. Lokeshowri, "Stock Price Prediction Using Long Short Term Memory," Interntational Research Journal of Engineering and Technology (IRJET), Chennai, Tamil, 2018.

[9] S. Selvin, V. R, G. E, V. K. Menon and S. K.P, "Stock price prediction using LSTM, RNN, and CNN-sliding window model," IEEE, India, 2017.

[10] C. J. Hutto and E. Gilbert, "VADER: A Parsimonious Rule-based Model for Sentiment Analysis of Social Media Text".

[11] C. Kai, Z. Yi and F. Dau, "A LSTM-based method for stock returns prediction: A case study of China stock market," IEEE International Conference on Big Data, 2015.

[12] A. Samarawickrama and T. Fernando, "A Recurrent Neural Network Approach in Predicting Daily Stock Prices," ICIIS, Nugegoda, Sri Lanka, 2017.

[13] D. K. Kirange and R. R. Deshmukh, "Sentiment Analysis of News Headlines for Stock Price Prediction," COMPUSOFT, An International Journal of Advance Computer Technology, 2016.

[14] S. P. Venkata, R. C. Kamal Nayan, P. Ganapati and B. Majhi, "Sentiment Analysis of Twitter Data for Predicting Stock Market Movements," IEEE, 2016.

[15] J. Xu and M. Tomohiro, "Stock Market Trend Prediction with Sentiment Analysis based on LSTM Neural Network," IMECS, Hong Kong, 2019.

[16] J. Li, H. Bu and J. Wu, "Sentiment-Aware Stock Market Prediction: A Deep Learning Method," National Natural Science Foundation of China, Beijing, China.

[17] J. Coelho, P. Madiraju and S. Coyne, "Forecasting Stock Prices using Social Media Analysis," IEEE, USA, 2017.

[18] S. Mohan, S. Mullapudi, S. Sammeta, P. Vijayvergia and D. C. Anastasiu, "Stock Price Prediction Using News Sentiment Analysis," IEEE, 2019.

[19] Z. Hu, W. Liu, J. Bian, X. Liu and T. Y. Liu, "Listening to Chaotic Whispers: A Deep Learning Framework for News-oriented Stock Trend Prediction," WSDM, USA, 2018.

[20] R. Sebastian, "An overview of gradient descent optimization," Cornell University, Dublin, 2017.

[21] G. Drakos, "Medium," Medium, 27 08 2018. [Online]. Available: https://medium.com/@george.drakos62/how-to-select-the-right-evaluation-metric-for-machine-learning-models-part-1-regrression-metrics-3606e25beae0?. [Accessed 11 10 2019].

[22] P. Grover, "Statistics How To," June 2018. [Online]. Available: https://heartbeat.fritz.ai/5-regression-loss-functions-all-machine-learners-should-know-4fb140e9d4b0. [Accessed 15 11 2019].

Appendix A

(Final Stock Dataset with Sentiment Scores)

Date	LTP	Open	High	Low	Score
11/14/19	417	417	417	415	0.41083667
11/13/19	416	414	417	413	0.34688611
11/12/19	417	425	425	417	0.1764
11/11/19	421	414	421	414	0.52365455
11/10/19	419	414	419	410	0.44867308
11/7/19	415	417	417	414	0.21539412
11/6/19	415	415	417	415	0.399425
11/5/19	416	415	416	413	0.43417059
11/4/19	413	414	414	410	0.44263333
11/3/19	412	412	413	410	0.2906875
10/31/19	413	411	417	410	0.3173
10/24/19	414	415	416	412	0.27229565
10/23/19	413	415	416	411	0.33468
10/22/19	411	411	413	409	0.2235
10/21/19	412	412	415	411	0.3053125
10/20/19	413	412	414	411	0.15123571
10/15/19	413	412	416	410	0.31481053
10/14/19	410	415	415	409	0.25498095
10/10/19	415	413	415	412	0.4017
10/3/19	416	415	418	415	0.2689
10/2/19	411	417	417	411	0.17870526
10/1/19	417	412	418	412	0.23831379
9/30/19	415	412	415	411	0.18758846
9/29/19	418	410	418	407	0.15967083
9/26/19	410	412	412	409	0.15056191
9/25/19	412	407	412	407	0.27925714
9/24/19	412	411	412	410	0.27236842
9/23/19	414	413	415	412	0.18768235
9/22/19	414	413	415	412	0.31579697
9/19/19	419	425	425	419	0.2483
9/18/19	423	420	423	419	0.12049167
9/17/19	418	410	418	410	0.49520625

Appendix B

(Source Code)

```
from pandas import read_csv
from matplotlib import pyplot
from matplotlib import pyplot as plt
import seaborn as sns
get_ipython().run_line_magic('matplotlib', 'inline')
import numpy as np
from pandas import DataFrame
from pandas import concat
from sklearn.preprocessing import MinMaxScaler
from keras.models import Sequential
from keras.layers import Dense
from keras.layers import LSTM
from keras.callbacks import Callback
from keras.callbacks import EarlyStopping
from keras import metrics
from keras import backend
import time
import pandas as pd
from sklearn.metrics import mean_squared_error
from sklearn.metrics import mean_absolute_error
from sklearn.metrics import r2_score
from math import sqrt
from numpy import concatenate

def rmse(y_true, y_pred):
    return backend.sqrt(backend.mean(backend.square(y_pred - y_true), axis=-1))

class TimeHistory(Callback):
    def on_train_begin(self, logs={}):
        self.times = []

    def on_epoch_begin(self, epoch, logs={}):
        self.epoch_time_start = time.time()

    def on_epoch_end(self, epoch, logs={}):
        self.times.append(time.time() - self.epoch_time_start)

# load dataset
df = read_csv('dataset.csv', header=0, index_col=0)
dataset = df[['LTP','Open','High','Low','Quantity','Score']]

values = dataset.values
values = values.astype('float32')
```

```
# convert series to supervised learning
def series_to_supervised(data, n_in=1, n_out=1, dropnan=True):
	n_vars = 1 if type(data) is list else data.shape[1]
	df = DataFrame(data)
	cols, names = list(), list()
	# input sequence (t-n, ... t-1)
	for i in range(n_in, 0, -1):
		cols.append(df.shift(i))
		names += [('var%d(t-%d)' % (j+1, i)) for j in range(n_vars)]
	# forecast sequence (t, t+1, ... t+n)
	for i in range(0, n_out):
		cols.append(df.shift(-i))
		if i == 0:
			names += [('var%d(t)' % (j+1)) for j in range(n_vars)]
		else:
			names += [('var%d(t+%d)' % (j+1, i)) for j in range(n_vars)]
	# put it all together
	agg = concat(cols, axis=1)
	agg.columns = names
	# drop rows with NaN values
	if dropnan:
		agg.dropna(inplace=True)
	return agg

scaler = MinMaxScaler(feature_range=(0, 1))
scaled = scaler.fit_transform(values)

# specify the number of lag hours
n_lag = 12
n_features = 5

# frame as supervised learning
reframed = series_to_supervised(scaled, n_lag, 1)
print(reframed.shape)

# split into train and test sets
values = reframed.values


n_train = 365 * 3
train = values[:n_train, :]
test = values[n_train:, :]

# split into input and outputs
n_obs = n_lag * n_features
train_X, train_y = train[:, :n_obs], train[:, -n_features]
test_X, test_y = test[:, :n_obs], test[:, -n_features]
print(train_X.shape, len(train_X), train_y.shape)

# reshape input to be 3D [samples, timesteps, features]
```

```
train_X = np.reshape(train_X, (train_X.shape[0], n_lag, n_features))
test_X = np.reshape(test_X, (test_X.shape[0], n_lag, n_features))
print(train_X.shape, train_y.shape, test_X.shape, test_y.shape)

time_callback = TimeHistory()

# design network
model = Sequential()
model.add(LSTM(120,input_shape=(train_X.shape[1], train_X.shape[2])))
model.add(Dense(1))
model.compile(loss='mse', optimizer='adam',metrics=[rmse, 'mse','mae'])
model.summary()

# fit network
history = model.fit(train_X, train_y, epochs=100, batch_size=30,
validation_data=(test_X,
test_y),callbacks=[EarlyStopping(monitor='val_loss',patience=10), time_callback],
verbose=2, shuffle=False)

total=0.0
list1 = time_callback.times
for ele in range(0, len(list1)):
    total = total + list1[ele]
print(total)

pyplot.plot(history.history['rmse'],label='rmse')
pyplot.plot(history.history['mean_squared_error'],label='mse')
pyplot.plot(history.history['mean_absolute_error'],label='mae')
pyplot.legend()
pyplot.show()

# make a prediction
yhat = model.predict(test_X)
test_X = test_X.reshape((test_X.shape[0], n_lag*n_features))

# invert scaling for forecast
inv_yhat = concatenate((yhat, test_X[:, -5:]), axis=1)
inv_yhat = scaler.inverse_transform(inv_yhat)
inv_yhat = inv_yhat[:,0]

# invert scaling for actual
test_y = test_y.reshape((len(test_y), 1))
inv_y = concatenate((test_y, test_X[:, -5:]), axis=1)
inv_y = scaler.inverse_transform(inv_y)
inv_y = inv_y[:,0]

# calculate RMSE
rmse = sqrt(mean_squared_error(inv_y, inv_yhat))
print('Test RMSE: %.3f' % rmse)
```

```
# calculate MAE
mae = mean_absolute_error(inv_y, inv_yhat)
print('Test MAE: %.3f' % mae)

# calculate r2 Score
r2 = r2_score(inv_y,inv_yhat)
print('Test r2: %.3f' % r2)

df = pd.DataFrame({'Actual':inv_y,'Predicted':inv_yhat,'Error': inv_y-inv_yhat,})
df["Error"].mean()
df = df[:20]

ax = df[['Actual','Predicted']].plot(kind='bar', title ="ADBL Predicted values VS
Actual values", figsize=(20, 10), legend=True, fontsize=12)
ax.set_xlabel("Date", fontsize=12)
ax.set_ylabel("Closing Price", fontsize=12)
ax.set_xticklabels('df')
plt.show()

sns.distplot(inv_yhat)
sns.distplot(inv_y, color='green')

fig, ax = plt.subplots()
ax.scatter(inv_y, inv_yhat, edgecolors=(0, 0, 0))
ax.set_xlabel('Actual')
ax.set_ylabel('Predicted')
plt.show()
```